

UC Berkeley

Research Reports

Title

Investigating Intelligent Transportation Systems Strategies On The Santa Monica Freeway Corridor: Technical Appendix

Permalink

<https://escholarship.org/uc/item/74q4w5fs>

Authors

Bacon, Jr., V.
Windover, J.
May, A.

Publication Date

1995

This paper has been mechanically scanned. Some errors may have been inadvertently introduced.

CALIFORNIA PATH PROGRAM
INSTITUTE OF TRANSPORTATION STUDIES
UNIVERSITY OF CALIFORNIA, BERKELEY

Investigating Intelligent Transportation Systems Strategies on the Santa Monica Freeway Corridor: Technical Appendix

Vinton W. Bacon Jr.

John R. Windover

Adolf D. May

University of California, Berkeley

California PATH Research Report

UCB-ITS-PRR-95-39

This work was performed as part of the California PATH Program of the University of California, in cooperation with the State of California Business, Transportation, and Housing Agency, Department of Transportation; and the United States Department of Transportation, Federal Highway Administration.

The contents of this report reflect the views of the authors who are responsible for the facts and the accuracy of the data presented herein. The contents do not necessarily reflect the official views or policies of the State of California. This report does not constitute a standard, specification, or regulation.

November 1995

ISSN 1055- 1425

**INVESTIGATING INTELLIGENT TRANSPORTATION SYSTEMS
STRATEGIES ON THE SANTA MONICA FREEWAY CORRIDOR**

TECHNICAL APPENDIX

**Vinton W. Bacon Jr.
John R. Windover
Adolf D. May**

June, 1995

**Research Report
California PATH Program
Institute of Transportation Studies
University of California, Berkeley**

TABLE OF CONTENTS - APPENDIX

	Page
Appendix A: The INTOMAN Program	A-1
Appendix B: The ODDATA Program	A-19
Appendix C: The MPT Program	A-21
Appendix D: The RED 12 Program	A-25
Appendix E: The INTMAP Program	A-32
Appendix F: Caltrans' Ramp Metering Plans	A-50
Appendix G: Freeway Demand Data	A-54
Appendix H: The FILE-DIV Program	A-71
Appendix I: The BY-DETEC Program	A-72
Appendix J: The BY-TIME Program	A-75
Appendix K: The MAKEFLOW Program	A-78
Appendix L: The SIGNAL Program	A-81
Appendix M: The SIG-REDC Program	A-83
Appendix N: Signal Control Data	A-85
Appendix O: The INTGEN Program	A-112
Appendix P: The Node File SCFINAL1.DAT	A-136
Appendix Q: The Link File SCFINAL2.DAT	A-151
Appendix R: The TOSEED Program	A-202
Appendix S: Final Freeway and Arterial Link Flows	A-206
Appendix T: The TO_QOD5 Program	A-243
Appendix U: The TO_INT3 Program	A-246

LIST OF FIGURES - APPENDIX

	Page
Figure A. 1: INTOMAN Main Menu	A-2
Figure A.2: INTOMAN Assign File Names Screen	A-2
Figure A.3: INTOMAN Contour Map Screen	A-3
Figure A.4: INTOMAN Link Statistics Screen	A-4
Figure A.5: Sample MTOMAN Link Statistics File	A-4
Figure A.6: Format of the INTEGRATION Node File	A-136
Figure A.7: Format of the INTEGRATION Link File	A-151

LIST OF TABLES - APPENDIX

	Page
Table A. 1: ATSAC Signal Timing Plan Implementation	A-85
Table A.2: Format of Signal Control Data	A-85

Appendix A: The INTOMAN Program

In order to efficiently interpret the output from INTEGRATION, it was deemed important to develop a utility program to manipulate the output files into a more easily readable format. This appendix describes that utility program (called INTOMAN) and provides some basic instructions for using it.

Motivation

INTEGRATION provides output in the form of ASCII files, using several different formats. The two most useful files (numbered 10 and 12) contain traffic performance data for network links by time slice. File type 10 is formatted, and can be read directly; file type 12 is provided in tabular form that is best suited for importing into a spreadsheet. Neither file, however, provides for a quick visual assessment of traffic performance. The lack of such a facility was the motivation for INTOMAN.

INTOMAN (short for INTeGration Output MANager), is custom utility software, developed by Bloomberg and May [9] and written in the Pascal programming language. Through menus, the program allows the user to create a number of useful charts and diagrams using the output from INTEGRATION.

Input to the Program

In addition to the INTOMAN executable, several additional files are needed to run the utility. They are described below:

- **INTEGRATION output files** type 10 and/or 12. Each is needed as the data source for creating specific types of output charts.
- **INTEGRATION link file** from the original simulation run.
- **Valid links file** describing the types of links in the network. Any number of groups of links can be described in this file, and the specific link numbers in each group must be listed. Examples of groups might include on-ramps, off-ramps, and mainline links. This file is needed by the program to provide output for specific groups of links.

Running the Program

The user starts the INTOMAN program from the DOS prompt simply by typing *intoman*. Then, the menu shown in Figure A. 1 appears on the screen.

The first step is to consider the files listed in the center of the screen. Since calibration often involved comparing several sets of data, it was useful to maintain several different sets of input files and INTOMAN-generated output files. The current set of files is shown on the main menu screen; this group can be changed by selected main menu option 4, which calls up the screen shown in Figure A.2.

```

                                INTEGRATION Output Manager
                                Institute for Transportation Studies - UC Berkeley
                                Main Menu

Current Files:  Eastbound Mainline Freeway (ver 1.5c)
                version:  INTEGRATION v1.5b or later

    link file:   eblinks.dat           valid file:   valideb.dat
output file 10: fway12eb.out         output file 12: fway12eb.out
    contour file: contour.eb         flow file:   flowdata.eb
    summary file: summdata.eb

                                1) Contour Maps
                                2) Statistics by Link or Link Type
                                3) Summary Statistics
                                4) Change File Names
                                0) Quit the program

                                Enter your choice:

                                Figure A. 1: INTOMAN Main Menu

```

```

                                INTEGRATION Output Manager
                                Institute for Transportation Studies - UC Berkeley
                                Assign File Names Screen

AVAILABLE FILESETS:
1) Eastbound Mainline Freeway (ver 1.5c)
2) Westbound Mainline Freeway (ver 1.5c)
3) Eastbound Mainline Freeway (ver 1.5b)
4) Westbound Mainline Freeway (ver 1.5b)
5) Eastbound Mainline Freeway (ver 1.5b)
6) Westbound Mainline Freeway (ver 1.5b)

Current set #1: Eastbound Mainline Freeway (ver 1.5c)
                version:  INTEGRATION 1.5b or later
    link file:   eblinks.dat           valid file:   valideb.dat
output file 10: fway12eb.out         output file 12: fway12eb.out
    contour file: contour.eb         flow file:   flowdata.eb
    summary file: summdata.eb

New fileset (or 0 to end):

                                Figure A.2: INTOMAN Assign File Names Screen

```

The user can select the appropriate dataset (the list of datasets and their files are stored in a file called *intoman.fil*, which can be modified by any ASCII editor). Once the correct set of files is selected, typing in 0 will return to the main menu.

Then, the user can select any of three general types of output from the program: contour maps, link statistics, or summary statistics. Each is described below.

Contour Maps

Typing in **1** from the main menu brings up the contour map menu. A sample dialogue from his menu is shown in Figure A.3.

```

                                INTEGRATION Output Manager
                                Institute for Transportation Studies - UC Berkeley
                                Contour Map Menu

                                1) Speed Contours
                                2) Density Contours
                                3) Queueing Diagram
                                0) Quit this Menu

                                Enter your choice: 1

Please select a group number from the following:

1) Eastbound Mainline Links
2) Eastbound On-Ramps
3) Eastbound Off-Ramps

Your choice: 1

Looking at output file 10...
Time slice 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
Output file was saved as contour.eb. Hit return to continue.
```

Figure A.3: INTOMAN Contour Map Screen

In this example, the user has selected a speed contour diagram, one of three **FREQ**-link contour diagrams that can be produced by **INTOMAN**. Then, the user can select from any of the link groups available for the current **dataset**; in this case, the mainline links were selected. Finally, the program reads output file 10 (looking at each of 24 time slices in this example) and produces an output *file* called *contour.eb*. The resulting file will look siiar to Figure 8.1 in the main text: a contour map of speed, graphed by link number on the horizontal axis and time on the vertical axis.

Link Statistics

Another **function** can be performed by typing in **2** from the main menu. A sample dialogue from the link statistics menu is shown in Figure A.4.

```

INTEGRATION Output Manager
Institute for Transportation Studies ~ UC Berkeley
Link Statistics Menu

1) Time-Space Series of Link Volumes
0) Quit this Menu

Enter your choice: 1

Please select a group number from the following

1) Eastbound Mainline Links
2) Eastbound On-Ramps
3) Eastbound Off-Ramps

Your choice: 1

Looking at output file 12...
Time slice 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
24
Output file was saved as flowdata.eb. Hit return to continue.

```

Figure A.4: INTOMAN Link Statistics Screen

For this function, only volume data output are available. Then, the user can again select which group of data for the output *file* (called *flowdata.eb*). An excerpt of the results from this sample run (the volume counts for 8 mainline links for the first 6 time slices) are shown in Figure A. 5

```

TRAFFIC STATISTICS SUMMARY
Time-Space Series of
FLOW data (in vehicles/hour/roadway)
Eastbound Mainline Freeway (ver 1.5c)
Eastbound Mainline Links

```

Link	Time Slice					
	1	2	3	4	5	6
1	2982	4558	4870	3006	3828	4276
2	2944	4532	4624	3048	3842	4328
3	5912	8036	7190	5208	6624	7496
4	5330	7296	6284	4726	5912	6620
5	6096	8358	6930	5528	7026	7998
6	6018	8320	6502	5660	7052	8094
7	6440	8860	6824	6198	7548	8662
8	5850	8054	5792	5664	6770	7792

Figure A. 5: Sample INTOMAN Link Statistics File

Summary Statistics

Summary data for the simulation run can be obtained by choosing the third main menu option. Currently, the only data provided are total vehicle-miles by time slice (i.e., for all links in the selected group) and for the entire simulation.

Source Code

The Pascal source code for *intoman.pas* is given below and on the following pages.

```
Program IntegrationOutputManager (input, output, infile, outfile);
{
    INTOMAN.pas utility program
    INSTITUTE FOR TRANSPORTATION STUDIES
    UNIVERSITY OF CALIFORNIA AT BERKELEY, 1994

    This program creates several types of summary charts and graphs using output from
    INTEGRATION simulation runs. The required input files are: INTEGRATION output
    files 10 and 12, the INTEGRATION link file (input), the valid link input file
    (describing the groups of links in the network), and intoman.fil (the list of
    dataset groups).
}

{$M 65520 0 655360}    (increase the stack size to accommodate the large
                      data structures)
uses Crt, Dos;

const
    MAXLINKS = 2000;
    MAXGROUPLINKS = 100;
    MAXTSLICES = 24;
    PAGEWIDTH = 80;
    PRINTWIDTH = 200;
    MAXLINKTYPES = 5;
    MAXFILETYPES = 10;
    FLOW = 1;
    SPEED = 2;
    TRAVELTIME = 3;
    DENSITY = 4;

type
    linkRec = record          {data for a specific links}
        linkNum: integer;
        linkLen: real;
        linkGroup: integer;
        orderNum: integer;   {number within linkGroup}
    end;

    bigLinkRec = record      {data for links used for output}
        linkNum: integer;
        linkLen: real;
        useLink: integer;    {1=true, 0=false, #=group}
    end;

    linkArray = array[-1..MAXLINKS] of linkRec;
    outputArray = array[1..MAXGROUPLINKS, 1..MAXTSLICES] of real;
    outputLinkNumArray = array[0..MAXGROUPLINKS] of integer;

    fileSetSubRec = record   {record of filenames for a single dataset}
        fileSetName: string[60];
        oldVersion: char;
        lf, vf, 010, 012, cf, ff, sf: string[12];
    end;
```

```

fileSetRec = record (data structure for datasets)
  totalNumber: integer;
  currentNumber: integer;
  fileSetData: array[1..MAXFILETYPES] of fileSetSubRec;
end;

var (global variables)
  linkData: linkArray;
  outputData,moreOutputData: outputArray;
  outputLinkNum: outputLinkNumArray;
  numTimeSlices, dataType, groupNum: integer;
  timeSliceSize: integer;
  menuItem, subMenuItem: integer;
  simulationName: string[80];
  groupNames: array[1..MAXLINKTYPES] of string[80];
  fileSet: fileSetRec;
  linkFile, validFile, outputFile10, outputFile12,
  contourFile, flowFile, summaryFile: string[12];
  eastbound, oldVersionData: char;

Function getValidInt(a,b: integer): integer;
{gets an integer in the range (a,b) from the user}

var
  i: integer;

begin
  repeat
    readln(i);
    if (i<a) or (i>b) then
      write('Please enter a number between ',a,' and ',b,':');
    until (i>=a) and (i<=b);
    getValidInt:=i;
end;
{*****}

Function validFileName(var fn: PathStr): boolean;
{checks to see if a file name is valid}
var
  i: integer;

begin
  if fSearch(fn,'e:')==fn then
    validFileName:=true
  else begin
    write('Sorry, no file named ',fn,' exists. Hit return to continue: ');
    readln;
    validFileName:=false;
  end;
end;
{*****}

Function getChar: char;
{gets one character (lower-case) from the user}

var
  s: string[80];
  c: char;

begin
  readln(s);
  c:=s[1];
  if (ord(c)>=ord('A')) and (ord(c)<=ord('Z'))
  then c:=chr(ord(c) t ord('a')-ord('A'));
  getChar:=c;
end;
{*****}

```

```

Function power(a,b: integer): integer;
{raises a to the bpower}
var
  i,j: integer;

begin
  j:=1;
  for i:=1 to b do j:=j*a;
  power:=j;
end;
{*****}

Function searchFileString (var infile: text; s: string; l: integer): boolean;
{reads a file until the first l chars of s are found}

var
  found, sameStr: boolean;
  inStr: string[80];
  i: integer;

begin
  found:=false;
  while (not(eof(infile)) and not(found)) do begin
    readln(infile,inStr);
    sameStr:=true;
    for i:=1 to l do
      if inStr[i] <> s[i] then sameStr:=false;
    found:=sameStr;
  end;
  searchFileString:=found;
end;
{*****}

Procedure centerString(var outfile: text; s: string);
[centers a string for an output file]

var
  i,l: integer;
begin
  l:=80-length(s);
  for i:=1 to trunc(l/2) do write(outfile,' ');
  writeln(outfile,s);
end;
{*****}

Procedure markValidLinks (var linkData: linkArray);
{reads the selected links from an input file}

var
  linkListCount, numLinkTypes, numLinks, i,j,k: integer;
  fn: PathStr;
  infile: text;

begin

{There should be a link input file that lists the links to be included in
the output. The file format is as follows:

line 1: Name of Simulation (text)
line 2: Number of groups
line 3: Name of Group #1 (text)
line 4: Number of links in group #1
line 5-n: Enumerated links in group #1
line n+1: Name of Group #2....
}

```

```

fn:=validFile;
if validFileName(fn) then begin
  assign(infile, fn);
  reset(infile);
  readln(infile);
  readln(infile,numLinkTypes);
  if numLinkTypes > MAXLINKTYPES then numLinkTypes:=MAXLINKTYPES;
  linkData[0].linkGroup:=numLinkTypes;
  linkListCount:=1;
  for i:=1 to numLinkTypes do begin
    readln(infile,groupNames[i]); {name of link type}
    readln(infile,numLinks);
    if (numLinks > MAXGROUPLINKS) then begin
      writeln;
      writeln('The number of links in group #',i,' exceeds the maximum (' ,
        MAXGROUPLINKS,') - excess links will be ignored.');
```

write('Hit return to continue ');

```

      readln;
    end;
    for k:=1 to numLinks do begin
      readln(infile,j);
      if k<=MAXGROUPLINKS then
        for linkListCount:=1 to linkData[0].linkNum do
          if linkData[linkListCount].linkNum = j
            then begin
              linkData[linkListCount].linkGroup:=i;
              linkData[linkListCount].orderNum:=k;
            end;
          end;
    end;
  end;
  close(infile);
end;
end;
{*****}

Procedure readLinkData(var linkData: linkArray);
[Reads in data about links from input file type 2]

var
  fn: pathStr;
  infile: text;
  numLinks,i,j: integer;

begin
  {write('Please enter the name of the INTEGRATION link file (#2):');}
  fn:=linkFile;
  if validFileName(fn) then begin

    assign(infile,fn);
    reset(infile);
    readln(infile); {skip first blank line of input file}
    readln(infile, numLinks);

    {read in link data to bigLinkArray}
    linkData[0].linkNum:= numLinks;
    for i:=1 to numLinks do if i<MAXLINKS then begin
      read(infile, linkData[i].LinkNum);
      read(infile,j,j); {skip over start/end node data}
      read(infile,linkData[i].linkLen);
      readln(infile);
      linkData[i].linkGroup:= 0;
      linkData[i].orderNum:=0;
    end;

    {determine which links are to be studied}
    markValidLinks(linkData);
  end
end

```

```

        else begin (no data was read)
            linkData[0].linkNum:=0;    {total links to study1
            linkData[0].linkGroup:=0; {number of link groups]
        end;
    end; {procedure}
}*****}

Procedure readOutputData(var outputData: outputArray; var numTimeSlices: integer;
    var linkData: linkArray; groupNum, dataType: integer];
{reads general data from INTEGRATION output file 10}

var
    infile: text;
    i,j,k,l,linkNum: integer;
    c: char;
    compareStr: string[15];
    whiteSpace: integer;    [how much space to skip in reading input]
    simTime,r: real;
    numLinks: integer;
    fn: pathStr;

begin
    numLinks:= linkData[0].linkNum;
    case dataType of
        SPEED: whiteSpace:= 86;    {average speed}
        FLOW: whiteSpace:= 52;    {flow}
        TRAVELTIME: whiteSpace:= 66;
    end;

    fn:=outputFile10;
    if validFileName(fn) then begin
        assign(infile,fn);
        reset(infile);
        if oldVersionData='y' then compareStr:= ' - File: 15'
            else compareStr:= ' - File: 19';
        if searchFileString(infile,compareStr,13) then begin
            readln(infile);
            for i:=1 to 24 do read(infile,c);
            readln(infile,simTime);
            for i:=1 to 26 do read(infile,c);
            readln(infile,timeSliceSize);
            if timeSliceSize < 0 then timeSliceSize:= -1*timeSliceSize;
            numTimeSlices:=trunc(simTime/timeSliceSize);
            if numTimeSlices > MAXTSLICES then numTimeSlices:=MAXTSLICES;

            writeln('Looking at output file 10...');
            write(' Time slice');
            for i:=1 to numTimeSlices do
                if searchFileString(infile,' LINK FLOW SUMMARIES',20) then begin
                    if i>9 then write(i:3)
                    else write(i:2);
                    for j:=1 to 5 do readln(infile);
                    k:=0;
                    for j:=1 to numLinks do begin
                        read(infile,linkNum);
                        {skip the next n characters, depending on whitespace}
                        for l:=1 to whiteSpace do read(infile,c);
                        readln(infile, r);    [get the relevant data]
                        if linkData[j].linkGroup = groupNum then begin
                            k:=k+1;
                            outputLinkNum[k]:=j;
                            outputData[k,i]:=r;
                            outputLinkNum[0]:=k;    {total in group}
                        end;
                    end;
                end;
            end;
            {if..then}
            writeln;
        end;    {if..then}
    end;
end;

```

```

        close(infile);
    end {if..then}
    else
        linkData[0].linkNum:=0;      (flag that no output data was read]
    end; {procedure}
{*****}

Procedure readLinkStatistics(var outputData: outputArray; var linkData: linkArray;
                             groupNum, dataType: integer);
[reads general data from INTEGRATION output file 12]

var
    infile: text;
    s: string;
    fn: pathStr;
    i,j,k,l,timePeriods: integer;
    r: array[1..12] of real;
    linkNum, numLinks, totalLinks: integer;
    c: char;

begin
    fn:=outputFile12;
    if validFileName(fn) then begin
        writeln('Looking at output file 12...');
        write(' Time slice');
        assign(infile,fn);
        reset(infile);
        readln(infile); (file title]
        read(infile, timePeriods);
        readln(infile, timeSliceSize);
        if timeSliceSize < 0 then timeSliceSize:=-1*timeSliceSize;

        numLinks:=linkData[0].linkNum;
        linkData[-1].linkNum:=timePeriods;

        for j:=1 to timePeriods do begin
            readln(infile); (time and period number]
            if j<10 then write(j:2)
            else write(j:3);
            l:=0;
            for i:=1 to numLinks do begin
                for k:=1 to 12 do read(infile,r[k]); (get some fields]
                readln(infile);
                if linkData[i].linkGroup = groupNum then begin
                    l:=l+1;
                    outputLinkNum[l]:=i;
                    outputLinkNum[0]:=l; (total number in group]
                    case dataType of
                        FLOW: outputData[l][j]:=trunc(r[2]);
                        DENSITY: outputData[l][j]:=trunc(r[11]);
                    end; {case]
                end; {if..then}
            end;
        end;
        writeln;
        close(infile);
    end
    else
        linkData[0].linkNum:=0;      (flag that no output data was read]
    end;
{*****}

```



```

Procedure findDensityData(var outputData: outputArray; linkData: linkArray);
[converts densities in cars/link to veh/mile]

var
  i,j: integer;

begin
  for j:=1 to linkData[-1].linkNum do    (loop through time slices)
    for i:=1 to outputLinkNum[0] do
      outputData[i,j]:= outputData[i,j] /
        linkData[outputLinkNum[i]].linkLen/ 1.609;
    end;
  {*****}

Procedure findSpeedData(var array1, array2: outputArray;
  linkData: linkArray; t: integer);
{separates aggregate travel times in array1 into time slice stats by using
cumulative flows in array2, and then determines speeds using link lengths}

var
  i,j: integer;

begin
  for i:=1 to outputLinkNum[0] do for j:=t downto 2 do begin
    if array2[i,j] - array2[i,j-1] = 0 then
      array1[i,j]:=-1.0
    else begin
      array1[i,j]:= (array1[i,j] * array1[i,j-1]) /
        (array2[i,j] - array2[i,j-1]) /60;
      if array1[i,j]> 0 then
        array1[i,j]:=linkData[outputLinkNum[i]].linkLen/array1[i,j]
      else array1[i,j]:=-1.0
    end;
  end;
  for i:=1 to outputLinkNum[0] do
    if (array1[i,1] <= 0) or (array2[i,1] <= 0)
    then array1[i,1]:= -1.0
    else
      array1[i,1]:= linkData[outputLinkNum[i]].linkLen/
        (array1[i,1]/array2[i,1]/60);
  end;
  {*****}

Procedure writeContourToFile(outputData: outputArray; numTimeSlices: integer;
  linkData: linkArray; groupNum, dataType: integer);
{creates a contour map from outputData[i,j]}

var
  i,j,k,p,map: integer;
  origStart: integer;
  linkNum,linkCol,offset: integer;
  origLinkNum, origLinkCol: integer;
  scale: real;      (scaling factor for contour map)
  l, lenUnit, adjust: real;
  linkSpacing: array[1..MAXGROUPLINKS] of integer;
  outfile: text;

begin
  assign(outfile,contourFile);
  rewrite(outfile);

  {determine scale factor}
  case dataType of
    1:  scale:= 16.09;      (1.609 km/mile divided by 10 for speed)
    2:  scale:= 10.0;      {vehicles/mile factored by 10 for density}
    3:  scale:= 1.0;      {use density, but don't scale for queues}
  end;
end;

```

```

writeln;
write('Adjusting total length...');
adjust:=1.0;
repeat
  l:=0.0;
  linkCol:=0;
  for i:=1 to outputLinkNum[0] do
    l:=l+linkData[outputLinkNum[i]].linkLen;
  lenUnit:= PRINTWIDTH/l;
  for i:=1 to outputLinkNum[0] do begin
    linkSpacing[i]:=trunc(linkData[outputLinkNum[i]].linkLen
      * lenUnit + adjust);
    if linkSpacing[i] < 1 then linkSpacing[i]:=1
  end;
  for i:=1 to outputLinkNum[0] do
    linkCol:=linkCol+linkSpacing[i];
  adjust:=adjust - 0.05;
  if linkCol < 200 then adjust:=adjust t 0.055;
until (linkCol=200);
writeln('Total length was ',linkCol);

{fill in output data}
case dataType of
  1: centerString(outfile,'SPEED CONTOUR MAP');
  2: centerString(outfile,'DENSITY CONTOUR MAP');
  3: centerString(outfile,'QUEUEING DIAGRAM');
end;
centerString(outfile,simulationName);
centerString(outfile,groupNames[groupNum]);

linkCol:=1;
linkNum:=1;
for map:=1 to 2 do begin
  if map=2 then offset:=linkCol-1; {save amount previously printed}
  origStart:=linkNum;
  origLinkNum:=linkNum;
  origLinkCol:=linkCol;
  writeln(outfile,' Time');
  write(outfile,'Slice ');
  j:=0;
  for i:=1 to 100 do write(outfile, '.');
  writeln(outfile);
  for i:=1 to numTimeSlices do begin
    linkCol:=origLinkCol;
    linkNum:=origLinkNum;
    write(outfile,i:5, '.');
    for j:=1 to 100 do begin
      if linkCol > linkSpacing[linkNum] then begin
        linkCol:=1;
        linkNum:=linkNum + 1;
      end;
      if linkNum <= outputLinkNum[0] then begin
        case dataType of 1,2:
          if (outputData[linkNum,i] > 0.0) and
            (outputData[linkNum,i]/scale < 10.0) then
            write(outfile,trunc(outputData[linkNum,i]/scale))
          else write(outfile,'*');
          3:
            if (outputData[linkNum,i] > 71*1.609 {Vc in veh/miles}
              then write(outfile,'*')
            else write(outfile,'');
          end; {case}
        end;
        linkCol:=linkCol t 1;
      end;
      writeln(outfile, '.');
    end;
  end;
end:

```

```

write(outfile,' ');
for i:=1 to 100 do write(outfile,'. ');
writeln(outfile);
(draw linear map of link numbers)
write(outfile,' ');
if map=2 then
  for i:=1 to linkSpacing[origStart]-offset do write(outfile,' ');
for i:=origStart + map - 1 to linkNum do begin
  write(outfile,'^ ');
  for j:=1 to linkSpacing[i] - 1 do write(outfile,' ');
end;
writeln(outfile);

write(outfile,'link #');
if map=2 then
  for i:=1 to linkSpacing[origStart]-offset do write(outfile,' ');
for i:=origStart + map - 1 to linkNum do begin
  write(outfile,trunc(linkData[outputLinkNum[i]].linkNum/1000));
  for j:=2 to linkSpacing[i] do write(outfile,' ');
end;
writeln(outfile);

write(outfile,' ');
if map=2 then
  for i:=1 to linkSpacing[origStart]-offset do write(outfile,' ');
for i:=origStart + map - 1 to linkNum do begin
  write(outfile,trunc((linkData[outputLinkNum[i]].linkNum -
  trunc(linkData[outputLinkNum[i]].linkNum/1000)*1000)/100));
  for j:=2 to linkSpacing[i] do write(outfile,' ');
end;
writeln(outfile);

write(outfile,' ');
if map=2 then
  for i:=1 to linkSpacing[origStart]-offset do write(outfile,' ');
for i:=origStart + map - 1 to linkNum do begin
  write(outfile,trunc((linkData[outputLinkNum[i]].linkNum
  - trunc(linkData[outputLinkNum[i]].linkNum/100)*100)/10));
  for j:=2 to linkSpacing[i] do write(outfile,' ');
end;
writeln(outfile);

write(outfile,' ');
if map=2 then
  for i:=1 to linkSpacing[origStart]-offset do write(outfile,' ');
for i:=origStart + map - 1 to linkNum do begin
  write(outfile,trunc(linkData[outputLinkNum[i]].linkNum
  - trunc(linkData[outputLinkNum[i]].linkNum/10)*10));
  for j:=2 to linkSpacing[i] do write(outfile,' ');
end;

writeln(outfile);
writeln(outfile);

end; {for..do}

writeln(outfile,' Institute of Transportation Studies');
writeln(outfile,' University of California, Berkeley');
writeln(outfile,' INTOMAN report (1994)');
close(outfile);
writeln;
write('Output file was successfully saved as ',contourFile)
write('. Hit return to continue. ');
readln;
end; {procedure}
{*****}

```

```

Procedure writeSummaryStats(outputData: outputArray; linkData: linkArray;
                           groupNum: integer);
{computes and writes total mainline vehicle miles}

var
  l,i: integer;
  numTimeSlices: integer;
  infile,outfile: text;
  totVehMiles,subVehMiles: real;
link:integer;

begin
  numTimeSlices:=linkData[-1].linkNum;
  assign(outfile,summaryFile);
  rewrite(outfile);
  centerString(outfile,'TRAFFIC STATISTICS SUMMARY');
  centerString(outfile,simulationName);
  centerString(outfile,groupNames[groupNum]);
  writeln(outfile);
  totVehMiles:=0.0;
  for i:=1 to numTimeSlices do begin
    subVehMiles:=0.0;
    link:=0;
    for l:=1 to linkData[0].linkNum do
      if linkData[l].linkGroup = groupNum then begin
        link:=link+1;
        totVehMiles:=totVehMiles +
          linkData[l].linkLen*outputData[outputLinkNum[link]][i]/
          (3600/timeSliceSize);
        subVehMiles:=subVehMiles +
          linkData[l].linkLen*outputData[outputLinkNum[link]][i]/
          (3600/timeSliceSize);
      end;
    writeln(outfile,'Time slice ',i:2,' ':',subVehMiles/1.609:7:0, ' vehicle-miles. ');
  end;
  writeln(outfile,'Total: ',totVehMiles/1.609:10:0,' vehicle-miles' );
  close(outfile);

  write('Output file was successfully saved as ',summaryFile);
  write('. Hit return to continue. ');
  readln;
end;
{*****}

Procedure writeLinkStats(outputData: outputArray; linkData: linkArray;
                        dataType: integer);
{provides summary statistics for a link type}

var
  i,j,k,l: integer;
  numTimeSlices: integer;
  outfile: text;

begin
  numTimeSlices:=linkData[-1].linkNum;
  assign(outfile,flowFile);
  rewrite(outfile);
  centerString(outfile,'TRAFFIC STATISTICS SUMMARY');
  centerString(outfile,'Time-Space Series of');
  centerString(outfile,'FLOW data (in vehicles/hour/roadway)');
  centerString(outfile,simulationName);
  centerString(outfile,groupNames[groupNum]);
  writeln(outfile);
  writeln(outfile,'          Time Slice');
  write(outfile,'Link  ');
  for i:=1 to numTimeSlices do write(outfile,i:6);
  writeln(outfile);
  write(outfile,'-----');

```

```

    for i:=1 to numTimeSlices do write(outfile,'-----');
    writeln(outfile);
    for j:=1 to outputLinkNum[0] do begin
        write(outfile,outputLinkNum[j]:4,' ');
        for k:=1 to numTimeSlices do write(outfile,
            outputData[j][k]:6:0);
        writeln(outfile);
    end;
    writeln(outfile);
    close(outfile);

    write('Output file was successfully saved as ',flowFile);
    write('. Hit return to continue. ');
    readln;
end;
{*****}

Function getGroupNum(linkData:linkArray): integer;
[gets a valid group number from the user]

var
    i: integer;
begin
    writeln;
    writeln('Please select a group number from the following: ');
    writeln;
    for i:=1 to linkData[0].linkGroup do
        writeln(i:2,') ',groupNames[i]);
    writeln;
    write('Your choice: ');
    getGroupNum:= getValidInt(1,linkData[0].linkGroup);
end;
{*****}

Procedure assignFileNames(var fileSet: fileSetRec; start: integer);

var
    i,j: integer;

begin
    if start<>1 then
        repeat
            i:=fileSet.currentNumber;
            ClrScr;
            writeln('
                INTEGRATION Output Manager');
            writeln('
                Institute for Transportation Studies - UC Berkeley');
            writeln('
                Assign File Names Screen');
            writeln;
            writeln('AVAILABLE FILESETS:');
            for j:=1 to fileSet.totalNumber do
                writeln(j:2,') ',fileSet.fileSetData[j].fileSetName);
            writeln;
            writeln(' Current set #',i,': ',fileSet.fileSetData[i].fileSetName);
            write ('
                version: ');
            if fileSet.fileSetData[i].oldVersion = 'y' then
                writeln(' INTEGRATION v1.5a or earlier')
            else writeln(' INTEGRATION v1.5b or later');
            writeln('
                link file: ',fileSet.fileSetData[i].lf:15,
                    valid file: ',fileSet.fileSetData[i].vf:15);
            writeln(' output file 10: ',fileSet.fileSetData[i].o12:15,
                output file 12: ',fileSet.fileSetData[i].o12:15);
            writeln('
                contour file: ',fileSet.fileSetData[i].cf:15,
                    flow file: ',fileSet.fileSetData[i].ff:15);
            writeln('
                summary file: ',fileSet.fileSetData[i].sf:15);
            writeln;
            write('New fileset (or 0 to end): ');
            j:=getValidInt(0,fileSet.totalNumber);
            if j>0 then fileSet.currentNumber:=j;

```

```

until (j=0);

i:=fileSet.currentNumber;
linkFile:=fileSet.fileSetData[i].lf;
validFile:=fileSet.fileSetData[i].vf;
outputFile10:=fileSet.fileSetData[i].ol0;
outputFile12:=fileSet.fileSetData[i].ol2;
contourFile:=fileSet.fileSetData[i].cf;
flowFile:=fileSet.fileSetData[i].ff;
summaryFile:=fileSet.fileSetData[i].sf;
oldVersionData:=fileSet.fileSetData[i].oldVersion;
simulationName:=fileSet.fileSetData[i].fileSetName;
end;
{*****}

Function mainMenu: integer;
{displays the main menu}

var
  i: integer;

begin
  ClrScr;
  writeln;
  writeln('                    INTEGRATION Output Manager');
  writeln('          Institute for Transportation Studies - UC Berkeley');
  writeln('                    Main Menu');
  i:=fileSet.currentNumber;

writeln('_____');
  writeln;
  writeln('  Current Files: ',fileSet.fileSetData[i].fileSetName);
  write ('          version: ');
  if fileSet.fileSetData[i].oldVersion = 'y' then
    writeln('  INTEGRATION v1.5a or earlier')
  else writeln('  INTEGRATION v1.5b or later');
  writeln;
  writeln('    link file: ',fileSet.fileSetData i).lf:15,
          valid file: ',fileSet fileSetData i).vf:15)
  writeln('  output file 10: ',fileSet.fileSetData i).ol2:15,
          output file 12: ',fileSet fileSetData i).ol2:15 ;
  writeln('    contour file: ',fileSet.fileSetData[i].cf:15,
          flow file: ',fileSet fileSetData i).ff:15)
  writeln('    summary file: ',fileSet.fileSetData[i].sf:15);

writeln('_____');
  writeln;
  writeln('          1) Contour Maps');
  writeln('          2) Statistics by Link or Link Type');
  writeln('          3) Summary statistics');
  writeln('          4) Change file names');
  writeln('          0) Quit the program');
  writeln;
  write('          Enter your choice: ');
  mainMenu:=getValidInt(0,4);
end;
{*****}

```

```

function contourMapMenu: integer;
{displays the contour map menu}

begin
  ClrScr;
  writeln;
  writeln('
                                INTEGRATION Output Manager');
  writeln('
                                Institute for Transportation Studies - UC Berkeley');
  writeln('
                                Contour Map Menu');
  writeln;
  writeln('
                                1) Speed Contours');
  writeln('
                                2) Density Contours');
  writeln('
                                3) Queueing Diagram');
  writeln('
                                0) Quit this menu');
  writeln;
  write('
                                Enter your choice: 0
contourMapMenu:=getValidInt(0,3);
end;
{*****}

function linkStatsMenu: integer;
{displays the link stats menu}

begin
  ClrScr;
  writeln;
  writeln('
                                INTEGRATION Output Manager');
  writeln('
                                Institute for Transportation Studies - UC Berkeley');
  writeln('
                                Link Statistics Menu');
  writeln;
  writeln('
                                1) Time-Space Series of Link Volumes');
  writeln('
                                0) Quit this menu');
  writeln;
  write('
                                Enter your choice: ');
  linkStatsMenu:=getValidInt(0,1);
end;
{*****}

Procedure readFileSet(var fileSet: fileSetRec);
[gets the current file set data from the disk]

var
  infile: text;
  i: integer;

begin
  assign(infile,'intoman.fil');
  reset(infile);
  readln(infile,fileSet.totalNumber);
  readln(infile,fileSet.currentNumber);
  for i:=1 to fileSet.totalNumber do begin
    readln(infile,fileSet.fileSetData[i].fileSetName);
    readln(infile,fileSet.fileSetData[i].oldVersion);
    readln(infile,fileSet.fileSetData[i].lf);
    readln(infile,fileSet.fileSetData[i].vf);
    readln(infile,fileSet.fileSetData[i].ol0);
    readln(infile,fileSet.fileSetData[i].ol2);
    readln(infile,fileSet.fileSetData[i].cf);
    readln(infile,fileSet.fileSetData[i].ff);
    readln(infile,fileSet.fileSetData[i].sf);
  end;
  close(infile);
end;
{*****}

```

```

begin {main}
  readFileSet(fileSet);
  assignFileNames(fileSet,1);
  repeat {main menu loop}
    menuItem:=mainMenu;

    if (menuItem = 1) then repeat
      subMenuItem:=contourMapMenu;
      if (subMenuItem>0) then begin
        readLinkData(linkData);
        if (linkData[0].linkNum > 0) then
          case subMenuItem of
            1: begin
              groupNum:= getGroupNum(linkData);
              readOutputData(outputData,numTimeSlices,linkData,
                groupNum,TRAVELTIME);
              readOutputData(moreOutputData,numTimeSlices,linkData,
                groupNum,FLOW);
              findSpeedData(outputData,moreOutputData,linkData,
                numTimeSlices);
            end;
            2,3: begin
              groupNum:=getGroupNum(linkData);
              readLinkStatistics(outputData,linkData,groupNum,DENSITY);
              findDensityData(outputData,linkData);
              numTimeSlices:=linkData[-1].linkNum;
            end;
          end;
        if (linkData[0].linkNum > 0) then
          writeContourToFile(outputData,numTimeSlices,linkData,
            groupNum,subMenuItem);
        end;
      until (subMenuItem=0);

      if (menuItem = 2) then repeat
        subMenuItem:=linkStatsMenu;
        if (subMenuItem > 0) then begin
          readLinkData(linkData);
          if (linkData[0].linkNum > 0) then begin
            case subMenuItem of
              1: begin
                groupNum:=getGroupNum(linkData);
                readLinkStatistics(outputData,linkData,groupNum,FLOW);
              end;
            end; (case)
            if (linkData[0].linkNum > 0) then
              writeLinkStats(outputData,linkData,dataType);
            end;
          end;
        until (subMenuItem=0);

        if (menuItem = 3) then begin
          readLinkData(linkData);
          groupNum:=getGroupNum(linkData);
          readLinkStatistics(outputData,linkData,groupNum,FLOW);
          writeSummaryStats(outputData,linkData,groupNum);
        end;

        if (menuItem = 4) then
          assignFileNames(fileSet,0);

      until (menuItem=0);
    end. {main program}

```


Appendix B: The ODDATA Program

The ODDATA program was written to process output from the **QUEENSOD** program in a timely manner. Given a network with over 100 zones, the possible number of origin/destination pairs is over 10,000. Processing these data on an individual basis is simply too time consuming. The ODDATA program works by reading output file number 9 from the **QUEENSOD** program and a file containing a list of origins and destinations. For each of the origins and destinations listed in the auxiliary input file, the program tabulates the total number of vehicles assigned out of (origins) or into (destinations) a zone. The program also sums the total number of vehicles that are assigned in the origin/destination file.

The C++ source code of the program is given below

```
#include <stdio.h>
#include <stdlib.h>

void getorigins(void);
void getoutfile(void);
void getinfile(void);
void printoutfile(void);

char odfile[12], newodfile[12], dummystring[121], outdatafile[12];
FILE *in, *out;
double totalod = 0;
int numoforigins, origins[200], seedod;
long int origintotal[200];
int numpairs = 0, odpair, origin, destination, numofseeds = 0, zeros = 0;
float ODrate, Krand, vt1, vt2, vt3, vt4, vt5, numofvehs;
float Starttime, Endtime;
main ()
{
    int j;
    for (j = 0; j < 200; j++)
        origintotal[j] = 0;
    getinfile();
    getoutfile();
    getorigins();
    in = fopen(odfile, "rt");
    fgets(dummystring, 80, in);
    //fscanf(in, "%d", &numpairs);
    fgets(dummystring, 80, in);
    while (!feof(in)) {
        fscanf(in, "%d %d %d %f %f %f %f %f %f %f %f", &odpair, &origin, \
            &destination, &ODrate, &Krand, &Starttime, &Endtime, &vt1, \
            &vt2, &vt3, &vt4, &vt5);
        fgets(dummystring, 80, in);
        numofvehs = (ODrate*((Endtime - Starttime)/3600));
        numpairs += t;
        totalod += numofvehs;
        for (j = 0; j < numoforigins; j++) {
            if (origin == origins[j]) {
                origintotal[j] += numofvehs;
                break;
            }
        }

        if (ODrate == seedod) numofseeds++;
        if (ODrate == 0) zeros++;

    }

    fclose(in);
    printoutfile();
    return (0);
}
```

```

void getinfile()

    printf("\n\nEnter the name of the file with the O/D data. ");
    printf("\nThat is the QUEENSOD output file:  ");
    scanf("%s", odfile);
    printf("\n\nEnter the amount of the demand for each o/d pair  ");
    printf("\nin the seed o/d file:  ");
    scanf("%d", &seedod);
    if ((in = fopen(odfile, "rt")) == NULL) {
        printf ("\n\nO/D data input file not opened!");
        exit(1);
    }

    fclose(in);

void getoutfile()
{
    printf("\n\nEnter the name of the output file to be appended: ");
    scanf("%s", outdatafile);
    if ((out = fopen(outdatafile, "at")) == NULL) {
        printf ("\n\nOutput file can not be opened!");
        exit(1);
    }

    fclose(out);

void getorigins(void)

    char originfile[12];
    FILE *ofile;
    int i;

    printf("\n\nEnter the name of the file with the origins to be counted: ");
    scanf("%s", originfile);
    if ((ofile = fopen(originfile, "rt")) == NULL) {
        printf ("Origin list file not opened!");
        exit(1);
    }
    fscanf(ofile, "%d", &numoforigins);
    if (numoforigins > 200) {
        printf("\n\nFATAL ERROR - The maximum number of origins is 200");
        exit(1);
    }
    fgets(dummystring, 80, ofile);
    for(i = 0; i < numoforigins; i++){
        fscanf(ofile, "%d", &origins[i]);
        fgets(dummystring, 80, ofile);
    }

    fclose(ofile);

void printoutfile(void)

    int l;
    char title[80];
    out = fopen(outdatafile, "at");
    printf("\n\n\nType the title of this o/d run for ID purposes:  \n");
    gets(title);
    gets(title);
    fprintf(out, "%s\n", title);
    fprintf(out, "\nTotal number of vehicles -- %8.0f", totalod);
    fprintf(out, "\nTotal number of o/d pairs -- %8d", numpairs - 1);
    fprintf(out, "\nNumber of zero o/d pairs -- %d", zeros);
    fprintf(out, "\nNumber of pairs with seeded demand (%d) -- %d\n", \
        seedod, numofseeds);
    fprintf(out, "\nThere were %d origins to be counted.", numoforigins);
    fprintf(out, "\nOrigin      Total demand from origin");
    for (l = 0; l < numoforigins; l++){
        fprintf(out, "\n%6d          %6d", origins[l], origintotal[l]);
    }
    fclose(out);

```

Appendix C: The MPT Program

The MPT program was designed to determine the path from a given origin to a given destination using INTEGRATION output file number 13, the minimum path tree file. This file is also used as an input file (number 6) to the QUEENSOD program. Section 3.6.3. discusses this file in great detail. In running the QUEENSOD program, it was discovered that the output was very sensitive to the path tree file that was used. For this reason, it was necessary to thoroughly analyze the minimum path trees that were used in any QUEENSOD program.

The analysis of minimum path trees is not an easy task given the manner in which the data is stored in the minimum path tree file. The path tree file works by giving the next link number that a vehicle should take from a given link or origin, in order to arrive at a given destination. To follow a path from an origin to a destination, one must trace the movement from link to link until the destination is reached. In the Santa Monica Freeway corridor network, a single set of path trees comprised a file over two megabytes in size. The effort involved in manually determining a path tree was simply too great. The MPT program was developed to automate this process and allow the analysis of many path trees.

The program first asks the user to specify an origin, a destination, the name of the minimum path tree file, and the name of the output file. The program first checks to determine if the origin and destination supplied by the user are actually an origin and a destination. Then, the program searches the minimum path tree file to determine what link should be taken from the origin to arrive at the destination. Then the program searches for this link and determines the next link to be taken. This process continues until the link number -1 is found. This indicates that this link will take the vehicle to its destination. At this point the program terminates. The output from the program is a list of the links that were used in the path from the origin to the destination.

The C++ source code for the program is given below;

```
// FILE = MPT.CPP

#include <stdio.h>
#include <stdlib.h>

void getOandD();
int checkOorD(int O1D2, int zone);
void openNodeFile();
void openMPTfile();
void openOutFile();
int findOriginLink(int origin);
int findNextLink(int currLink);

char MPTfile[12], outFile[12], nodeFile[12];
int slices, duration, numDests, maxDest, numOrigins, numLinks, numNodes;
int origin, destination;
int linkList[100];
FILE * MPT, * out, * node;

main()
```

```

{
    int i = 0, link;
    getOandD();
    openNodeFile();
    checkOorD(1, origin);
    checkOorD(2, destination);
    openMPTfile();
    openOutFile();

    link = findOriginLink(origin);
    linkList[i] = link; i ++;

    while (link != -1){
        if (link == 0){
            printf("It is not possible to get from origin %d to destination %d",
                origin, destination);
            exit (1);
        }
        link = findNextLink(link);
        linkList[i] = link; i ++;

        printf("\n\nThe minimum path from %d to %d is: ", origin, destination);
        fprintf(out, "\n\nThe minimum path from %d to %d is: ",
            origin, destination);

        for(i = 0; linkList[i] != -1; i ++){
            printf ("\n%d. Link %d", i + 1, linkList[i]);
            fprintf (out, "\n%d. Link %d", i + 1, linkList[i]);
        }

        fclose(out);
        return 0;
    }

void getOandD()

    printf("\n\nWhat is the origin number: ");
    scanf("%d", &origin);
    printf("\n\nWhat is the destination number: ");
    scanf("%d", &destination);

int checkOorD(int O1D2, int zone)

    char ch;
    int i, nodeNum, nodeType;
    float xCo, yCo;

    node = fopen(nodeFile, "rt");
    ch = 0; while(ch != '\n') ch = fgetc(node);
    ch = 0; while(ch != '\n') ch = fgetc(node);
    for (i = 0; i < numNodes; i++){
        fscanf(node, "%d %f %f %d", &nodeNum, &xCo, &yCo, &nodeType);
        ch = 0; while(ch != '\n') ch = fgetc(node);
        if (zone == nodeNum){
            if (O1D2 == 1) {
                if (nodeType != 1 && nodeType != 3) {
                    printf("\n\nERROR - node %d is not an origin.", nodeNum);
                    exit(0);
                }

                fclose(node); return 1;
            }

            if (O1D2 == 2) {
                if (nodeType != 1 && nodeType != 2) {
                    printf("\n\nERROR - node %d is not an destination.", nodeNum);
                    exit (0);
                }
            }
            fclose(node); return 1;
        }
    }
}

```

```

fclose(node);
printf("\n\nI cannot find a zone number %d", zone);
exit (0);
return 0;
}

void openNodeFile()
{
char ch;
printf("\n\nWhat is the name of the node file: ");
scanf("%s", nodeFile);

if ((node = fopen(nodeFile, "rt")) == NULL) {
printf ("\n\nNode file not opened!");
exit(0);
}
ch = 0; while(ch != '\n') ch = fgetc(node);
fscanf(node, "%d", &numNodes);
fclose(node);

void openMPTfile()

char ch;
printf("\n\nWhat is the name of the minimum path tree file: ");
printf("\n\nThis is output file number 13. ");
scanf("%s", MPTfile);

if ((MPT = fopen(MPTfile, "rt")) == NULL) {
printf ("Minimum path tree file not opened!");
exit(0);
}
ch = 0; while(ch != '\n') ch = fgetc(MPT);
fscanf(MPT, "%d %d %d %d %d", &sllices, &duration, &numDests,
&maxDest, &numOrigins, &numLinks);
fclose(MPT);
}

void openOutFile()
{
printf("\n\nWhat is the name of the output file: ");
scanf("%s", outFile);

if ((out = fopen(outFile, "wt")) == NULL) {
printf ("Output file not opened!");
exit(0);
}

int findOriginLink(int origin)

int j, k, aLink;
char ch;
MPT = fopen(MPTfile, "rt");
for (j = 0; j < (3 + origin); j ++){
ch = 0; while(ch != '\n') ch = fgetc(MPT);
}
for (k = 0; k < (destination + 1); k ++){
fscanf(MPT, "%d", &aLink);
return aLink;
}

int findNextLink(int currLink)

int j, k, l, aLink;
char ch;
MPT = fopen(MPTfile, "rt");
for (j = 0; j < (4 + numOrigins); j ++){
ch = 0; while(ch != '\n') ch = fgetc(MPT);
}
for (k = 0; k < (currLink - 1); k ++){
ch = 0; while(ch != '\n') ch = fgetc(MPT);
}
}

```

```
for (l = 0; l < (destination + 1); l++)  
    fscanf(MPT, "%d", &aLink);  
return aLink;}
```

Appendix D: The RED12 Program

Analyzing the output from the INTEGRATION simulation runs was difficult given that a single half hour time slice would produce an output file **12** (link data) roughly two and one half megabytes in size. The **RED12** program was written to extract certain information from the program. In some cases, the extracted data is converted into a specified format so that the data can be easily imported into a spreadsheet so that the link flow data can be easily displayed graphically.

The program requires two input files; an INTEGRATION output file number 12 and a file named "LINKS.DAT" that contains a list of links and other pertinent data. The program output consists of seven different files. The first line of the LINKS.DAT file contains four floating point numbers that **signify**, in order, a high value for the volume-to-capacity ratio, a low value for the volume-to-capacity ratio, a low value for speed (km/h), and a low **value** for delay (seconds). The next line contains the number of links that the program is to record data for. The remaining lines contain the link numbers (one per line) of the links that data is to be recorded for.

The seven output files created by the program all retain the same file name as the output file number 12. The extension for the file is changed and used to **identify** the type of output file created by the program. The seven extensions and description of the file are contained below.

- “**.VCL**” - This file contains a list of all of the links where the volume-to-capacity ratio is lower than the value specified in the LINKS.DAT file. The file contains data for all of the time slices except the first half hour which the warm-up time slice. This is used to **identify** links that are being underutilized.
- “**.VCH**” - This file contains a list of all of the links where the volume-to-capacity ratio is higher than the value specified in the LINKS.DAT file. The file contains data for all of the time slices except the first half hour.
- “**.SDL**” - This file contains a list of all of the links where the speed is lower than the value specified in the LINKSDAT file. The file contains data for all of the time slices except the first half hour.
- “**.DEL**” - This file contains a list of all of the links in the network that have a travel time delay greater than the value specified in the LINKSDAT file. The file contains data for all of the time slices except the first half hour. Delay is computed by subtracting the free flow travel time from the actual travel time on the link.

- “.TOT”** This file provides summary statistics concerning the number of links that are listed in the above four files for each of the time slices simulated. This allows the user to identify at a glance the general level of congestion in a network over the entire simulation period.
- “.RED”** - This file contains link data for the list of links that is provided in the LINKS.DAT file. Data is provided for each time slice with the exception of the first 30 minutes of simulation time since this is usually used as a warm-up time period. Each line of the file contains the link number, the link flow (vehs./hr.), the volume-to-capacity ratio, the free flow travel time, the actual travel time, and the speed on the link.
- “.QPR”** - This file is similar in format to the “.RED” file except that the file has been formatted so that it is easily imported into a Quattro Pro spreadsheet and graphed. This enables the user to quickly compare the simulated flow along the freeway or an arterial against the actual flow. The file only prepares data once for each 30 minutes of simulation time with the exception of the first 30 minutes.

In running INTEGRATION, certain runs were done outputting data every 30 minutes and others every 10 minutes. Since the RED12 program provides data on a 30 minute time slice basis, the user must tell the program which output interval is being used. If the output data interval is 10 minutes, the program will eliminate the first three time slices for both the “.RED” and “.QPR” files. Only the first time slice is eliminated for a 30 minute output interval. For the time slices after the first 30 minutes, the program lists all the time slices (10 or 30 minutes) in the “.RED” file. For the “.QPR” file, the program lists every subsequent time slice with the 30 minute output interval, and every third subsequent time slice with the 10 minute interval.

The C++ source code for the MPT program is provided below.

```
// FILE = RED12.CPP

#define MAX_NUM_LINKS 4000
#define MAX_SLICES 29
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

void getLinksDat();
int get10or30();
void getOutFile();
int makeFileNames();
int open12file();
int printHeader(char *, int slice, int returns);
int checkMarked(int);
int tally(int, int, int, int, int);
double calcTotLength();
float getLength(int);

struct link {
    int ID;
    float length;
};
```



```

}; link theLinks[MAX_NUM_LINKS];

FILE *in, *outRed, *outVcl, *outVch, *outSdl, *outDel, *linkIn;
FILE *outQPro;
int numLinks, numMarkedLinks, markedLinks[300], speedLow;
int VC_highCount[MAX_SLICES], VC_lowCount[MAX_SLICES];
int speedLowCount[MAX_SLICES], tTRatioCount[MAX_SLICES];
int flag10_30, dummies;
float VC_high, VC_low, delay;
char red12Text[9], orig12file[13], redFile[13], vclFile[13];
char vchFile[13], sdlFile[13], totFile[13], linkFile[13], delFile[13];
char redExt[] = ".red", vclExt[] = ".vcl", vchExt[] = ".vch";
char sdlExt[] = ".sdl", totExt[] = ".tot", delExt[] = ".del";
char QProExt[] = ".qpr", QProFile[13];

main ()
{
    int numMarkedLinks, numSlices, linkNum, freeftt;
    int usertt, systt, stdtt, queue, stops, maxDen, currDen, speed, occ;
    int slice, link;
    float VC_ratio, flow, capacity, length;
    FILE *in12;
    char ch;

    flag10_30 = get10or30();
    getLinksDat();
    numSlices = open12file();
    getOutFile();
    makeFileNames();
    in12 = fopen(orig12file, "rt");
    for (ch = 0; ch != 10; ch = fgetc(in12));
    for (ch = 0; ch != 10; ch = fgetc(in12));

    for (slice = 0; slice < numSlices; slice ++){
        printf("\n\nProcessing slice number %d.", slice);
        for (ch = 0; ch != 10; ch = fgetc(in12));

        if (slice >= dummies) {
            printHeader(redFile, slice, 0);
            printHeader(vclFile, slice, 1);
            printHeader(vchFile, slice, 1);
            printHeader(sdlFile, slice, 1);
            printHeader(delFile, slice, 0);
            if ((slice + 1)%3 == 0)
                printHeader(QProFile, slice, 0);
        }

        for (link = 0; link < numLinks; link ++){
            if (link % 250 == 0 && link != 0)
                printf("\n Processing link number %d.", link);
            fscanf(in12, "%d %f %f", &linkNum, &flow, &capacity);
            fscanf(in12, "%d %d %d", &freeftt, &usertt, &systt);
            fscanf(in12, "%d %d %d", &stdtt, &queue, &stops);
            fscanf(in12, "%d %d %d %d", &maxDen, &currDen, &speed, &occ);
            VC_ratio = flow / capacity;
            for (ch = 0; ch != 10; ch = fgetc(in12));

            if (checkMarked(linkNum) && slice >= dummies) {
                outRed = fopen(redFile, "at");
                fprintf(outRed, "\n%6d %6.0f %5.3f %5d %5d %5d", linkNum,
                    flow, VC_ratio, freeftt, usertt, speed);
                fclose(outRed);

                if (checkMarked(linkNum) && slice >= dummies) {
                    if ((slice + 1)%3 == 0 && flag10_30 == 10) {
                        outQPro = fopen(QProFile, "at");
                        fprintf(outQPro, "\n%6d %6.0f %5.3f %5d %5d %5d", linkNum,
                            flow, VC_ratio, freeftt, usertt, speed);
                        fclose(outQPro);
                    }
                }
            }
        }
    }
}

```

```

if(VC_ratio >= VC_high && slice >= dummies) {
    outVch = fopen(vchFile, "at");
    fprintf(outVch, "\n%6d %6.0f %5.3f %5d %5d %5d", linkNum,
        flow, VC_ratio, freeftt, usertt, speed);
    fclose(outVch);
    VC_highCount[slice] ++;

if(VC_ratio <= VC_low && slice >= dummies) {
    outVcl = fopen(vclFile, "at");
    fprintf(outVcl, "\n%6d %6.0f %5.3f %5d %5d %5d", linkNum,
        flow, VC_ratio, freeftt, usertt, speed);
    fclose(outVcl);
    VC_lowCount[slice] ++;
}

if(speed <= speedLow && slice >= dummies) {
    outSdl = fopen(sdlFile, "at");
    fprintf(outSdl, "\n%6d %6.0f %5.3f %5d %5d %5d", linkNum,
        flow, VC_ratio, freeftt, usertt, speed);
    fclose(outSdl);
    speedLowCount[slice] ++;

if((usertt - freeftt) >= delay) && slice >= dummies) {
    outDel = fopen(delFile, "at");
    fprintf(outDel, "\n%6d %6.0f %5.3f %5d %5d %5d", linkNum,
        flow, VC_ratio, freeftt, usertt, speed);
    fclose(outDel);
    ttRatioCount[slice] ++;

fclose(inl2);
for (slice = dummies; slice < numSlices; slice++){
    tally(VC_highCount[slice], VC_lowCount[slice], speedLowCount[slice],
        ttRatioCount[slice], slice);

printf("\nCumulative data by slice can be found in file %s.", totFile);
return (0);
}
// END OF MAIN

int get10or30()
{
    int theFlag = 0;
    while (theFlag != 10 && theFlag != 30) {
        printf("\n\nIs the output file contain 10 or 30 minute data: ");
        scanf("%d", &theFlag);
    }
    if (theFlag == 10) dummies = 3;
    if (theFlag == 30) dummies = 1;
    return theFlag;

void getLinksDat()
{
    char markedLinkFile[] = "LINKS.DAT", ch;
    int i;

    if ((in = fopen(markedLinkFile, "rt")) == NULL) {
        printf("\n\nInput file with the relevant links was not opened!");
        exit(1);

fscanf(in, "%f %f %d%f", &VC_high, &VC_low, &speedLow, &delay);
for (ch = 0; ch != 10; ch = fgetc(in));
fscanf (in, "%s", linkFile);
fscanf(in, "%d", &numMarkedLinks);
for (ch = 0; ch != 10; ch = fgetc(in));
for (i = 0; i < numMarkedLinks; i++){
    fscanf(in, "%d", &markedLinks[i]);
    for {ch = 0; ch != 10; ch = fgetc(in));

```

```

    }
    fclose(in);
}

void getOutFile()

    int i;
    for (i = 0; i < 13; i++){
        if (origl2file[i] == '.') {
            strncpy(redl2Text, origl2file, i);
        }
    }

int openl2file()
{
    FILE * inl2;
    int numSlices, duration, maxLink;
    char ch;

    printf("\n\nEnter the name of the original output file number 12. ");
    scanf("%s", origl2file);
    if ((inl2 = fopen(origl2file, "rt")) == NULL) {
        printf("\n\nOriginal file number 12 can not be opened!");
        exit(1);
    }

    for (ch = 0; ch != 10; ch = fgetc(inl2));
    fscanf(inl2, "%d %d %d %d", &numSlices, &duration, &numLinks, &maxLink);

    fclose(inl2);
    return numSlices;
}

int checkMarked(int linkNum) {
    int link;

    for (link = 0; link < numMarkedLinks; link++){
        if (linkNum == markedLinks[link])
            return 1;
    }

    return 0;
}

int makeFileNames()
{
    strcpy(redFile, redl2Text);
    strcpy(vclFile, redl2Text);
    strcpy(vchFile, redl2Text);
    strcpy(sdlFile, redl2Text);
    strcpy(totFile, redl2Text);
    strcpy(delFile, redl2Text);
    strcpy(QProFile, redl2Text);
    strcat(redFile, redExt);
    strcat(vclFile, vclExt);
    strcat(vchFile, vchExt);
    strcat(sdlFile, sdlExt);
    strcat(totFile, totExt);
    strcat(delFile, delExt);
    strcat(QProFile, QProExt);

    if ((in = fopen(redFile, "wt")) == NULL) {
        printf("\n\nCould not open the output file %s", redFile);
        exit(1);
    }

    fprintf(in, "Filename = %s\n\n", redFile);
    fprintf(in, "Data for selected links.\n");
    fclose(in);

    if (flag10 30 == 10) {
        if (fin = fopen(QProFile, "wt")) == NULL) {
            printf("\n\nCould not open the output file %s", QProFile);
            exit(1);
        }
    }
}

```

```

        fprintf(in, "Filename = %s\n\n", QProFile);
        fprintf(in, "Data for selected links.\n");
        fclose(in);
    }

    if ((in = fopen(vchFile, "wt")) == NULL) {
        printf ("\n\nCould not open the output file %s", vclFile);
        exit(1);

        fprintf(in, "Filename = %s\n\n", vchFile);
        fprintf(in, "Links with V/S ratio greater than %.2f.\n", VC_high);
        fclose(in);

        if ((in = fopen(vclFile, "wt")) == NULL) {
            printf ("\n\nCould not open the output file %s", vchFile);
            exit(1);
        }
        fprintf(in, "Filename = %s\n\n", vclFile);
        fprintf(in, "Links with V/S ratio less than %.2f.\n", VC_low);
        fclose(in);

        if ((in = fopen(sdlFile, "wt")) == NULL) {
            printf ("\n\nCould not open the output file %s", sdiFile);
            exit(1);

            fprintf(in, "Filename = %s\n\n", sdlFile);
            fprintf(in, "Links with speed less than %d kph.\n", speedLow);
            fclose(in);

            if ((in = fopen(delFile, "wt")) == NULL) {
                printf ("\n\nCould not open the output file %s", delFile);
                exit(1);

                fprintf(in, "Filename = %s\n\n", delFile);
                fprintf(in, "Links with delay greater than %.Of seconds.\n",
                    delay);
                fclose(in);

                if ((in = fopen(totFile, "wt")) == NULL) {
                    printf ("\n\nCould not open the output file %s", totFile);
                    exit(1);
                }
                fprintf(in, "Filename = %s\n\n", totFile);
                fclose(in);

                return 1;
            }

int printHeader(char * outFile, int slice, int returns)
{
    FILE *out;
    int i;

    out = fopen(outFile, "at");
    for (i = 0; i < returns; i++){
        fprintf(out, "\n");
    }
    fprintf(out, "\nSlice %d", slice);
    fprintf(out, "\n Link      Flow   V/S      FTT      UTT      Speed");
    fclose(out);
    return 1;
}

tally (int vcHighC, int vcLowC, int speedLowC, int ttRatioC, int slice)
{
    FILE *totOut;
    totOut = fopen(totFile, "at");
    fprintf(totOut, "Slice %d\n", slice);
    fprintf(totOut, " %d links with V/S greater than %.2f were counted.\n",
        vcHighC, VC_high);
    fprintf(totOut, " %d links with V/S lower than %.2f were counted.\n",
        vcLowC, vc_low);
    fprintf(totOut, " %d links with travel times greater than %.2f.\n",

```

```

        ttRatioC, delay);
fprintf(totOut, " %d links with speed lower than %d were counted.\n\n",
        speedLowC, speedLow);
fclose(totOut);

if (slice == 1) printf("\n\n");
printf ("\nSlice %d. %d links with V/S greater than %.2f were counted.",
        slice, vcHighC, VC_high);
printf ("\nSlice %d. %d links with V/S lower than %.2f were counted.",
        slice, vcLowC, vc_low);
printf ("\nSlice %d. %d links with speed lower than %d were counted.",
        slice, speedLowC, speedLow);
printf ("\nSlice %d. %d links with travel times greater than %.2f.\n",
        slice, ttRatioC, delay);
return 1;
}

double calcTotLength()

char ch;
int l, linkNum, upNode, downNode;
double totLength = 0;
float length;

printf ("\nGetting link lengths from %s", linkFile);
if ((linkIn = fopen(linkFile, "rt")) == NULL) {
    printf ("\n\nInput file with the link data was not opened!");
    exit(1);
}
for (ch = 0; ch != 10; ch = fgetc(linkIn));
for (ch = 0; ch != 10; ch = fgetc(linkIn));
for (l = 0; l < numLinks; l ++){
    fscanf(in, "%d %d %d %f", &linkNum, &upNode, &downNode, &length);
    theLinks[l].ID = linkNum;
    theLinks[l].length = length;
    for (ch = 0; ch != 10; ch = fgetc(in));
    totLength += length;
}
return totlength;
}

float getLength(int linkNum)

int l;
float length;
for (l = 0; l < numLinks; l ++){
    if (theLinks[l].ID == linkNum)
        return theLinks[l].length;
}
return 0;
}

```

Appendix E: The INTMAP Program

The INTMAP program creates plots of an INTEGRATION network that show the location of all the nodes and links as well as other data. The program was found to be an excellent tool in the creation of an INTEGRATION network. Certain coding errors, such as erroneous x-y coordinates or links assigned to incorrect nodes, are easily discernible with a plot of the network but very difficult to find using the ASCII files themselves.

The INTMAP program was originally written at Queen's University using the BASIC computer programming language. However, limitations to the program required that it be redeveloped to use the program with the Santa Monica Freeway corridor network. These limitations included the size of the network that could be simulated and the requirement that all links be numbered sequentially.

The program works by reading INTEGRATION input files number one and two (the node and link files) and an optional label file, and producing as output an AutoCAD LISP macro. This macro is read by the AutoCAD program to produce an AutoCAD drawing that can be manipulated, exported to other programs or plotted.

A number of special features were added to the program to allow for the creation of specialized plots that reveal additional data. The program can display the node numbers for each zone or other numbers such as signal number. Zones and other nodes are treated differently by the program, allowing the user to display only one or the other, or to display different data for each. The user may specify the size of the display numbers and have different display characteristics for zones and other nodes.

The display of links in the network may also be customized by the user. The link number or data such as link flow can be displayed for each of the links in the network or a user-specified subset. The user may also display some links as thin lines and others as thick lines to graphically display the location of a subset of links in the network.

The C++ source code of the INTMAP program is given below. Note that much of the code is contained in various header files, all of which are given below.

Main File "ucplot.cpp"

```
#include "ucplot.h"
#include "openfile.h"
#include "getcoord.h"
#include "params.h"
#include "plottype.h"
#include "buildspc.h"
#include "nodes.h"
#include "links.h"
#include "specialn.h"
#include "specialz.h"
#include "speciall.h"
#include "legends.h"
```

```

#include "labels.h"

void main(void)
{
    getTitleData();
    openInputFiles();
    calcParams();
    getMaxCoords();
    getUserParams();
    getPlotType();

    if (Nspecial == 0) plotRegNodes();
    if (Nspecial == 1) plotSpecialNodes();
        if (Nlabel == 1) plotNodeTitles(Nlabel);
    if (Zspecial == 0) plotRegZones();
    if (Zspecial == 1) plotSpecialZones();
        if (Zlabel == 1) plotZoneTitles(Zlabel);
    if (Lspecial == 0) plotRegLinks();
    if (Llabel == 1) plotLinkTitles(Llabel);
        if (islabelfile == 1) plotLabels();
    drawLegend();
    drawSpecialTypeLegend();
}

```

Header File "ucplot.h"

```

#include <math.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <dos.h>
#include <conio.h>

#define MAXNODES 1000
#define MAXLABELS 50

char  outfile[12], nodefile[12], linkfile[12], labelfile[12];
char  issubtitle, islabelfile;
FILE  *out, *nodein, *linkin, *labelin;
float  xcoord, ycoord, xmin, xmax, ymin, ymax;
short int Ntype, Nspecial, Nlabel, Ztype, Zspecial, Zlabel;
short int Ltype, Lspecial, Llabel;

char  ch;
char  dummystring[121], plottype, linktypename[3][41];

float  xScaleFactor, yScaleFactor, pageXspread = 31, pageYspread;
float  diameter, nodetextsize, linktextsize, zonetextsize, labeltextsize;
float  nodetextdist, linktextdist, zonetextdist, flow;
int  i, numofnodes, numoflinks, numofzones, upnode, downnode, maxflow;
int  nodenum, linknum, nodetype, other;
short int plotLinkLabels = 1, plotNodeLabels = 1;
short int plotZoneLabels = 1, plotLinkWidths = 0;
short int plotLinkSymbols = 1, plotNodeSymbols = 1, plotZoneSymbols = 1;
short int specialNodes = 0, specialZones = 0, specialLinks = 0;
short int numOfNodeTypes, numOfZoneTypes, numOfLinkTypes;

struct  aNodesData {
    int  aNodeNum;
    float  anXcoord;
    float  ayCoord;
    int  aNodeType;
    float  other;
    float  other2;
};
struct  aNodesData theNodes[MAXNODES];

struct  nodeTypeData {
    float  nodeDiameter;
    short int nodeTypeColor;
    char  nodeTypeName[21];
}

```

```

        float nodeTypeHigh;
        float nodeTypeLow;
        short int nodeFillType;
};
struct nodeTypeData nodeTypes[4];
struct nodeTypeData zoneTypes[4];

struct linkTypeData {
    short int lineOrSolid;
    short int linkTypeColor;
    short int linkFillType;
    float linkWidth;
    char linkTypeName[21];
    float linkTypeHigh;
    float linkTypeLow;
};
struct linkTypeData linkTypes[4];

```

Header File "openfile.h"

```

void getTitleData(void)
{
    char outname[8], lispext[] = ".lsp";
    char title[81], subtitle[81];
    int length, count = 0;

    clrscr();
    printf("\n\nWhat is the title of the plot? (80 characters max.)\n\n");
    gets(title);
    clrscr();
    printf("\n\nWhat is the subtitle of the plot? (enter for none)\n\n");
    gets(subtitle);
    clrscr();
    if ((strlen(subtitle)) == 0)
        issubtitle = '0';
    else issubtitle = '1';

    printf("\n\n\nWhat is the name of the lisp output file? ");
    printf("\n\nIf a file in the current directory has this name");
    printf(" it will be deleted!\n");
    gets(outname);

    length = strlen(outname);
    for(i = 0; i < length; i ++){
        if (outname[i] == '.')
            break;
        else count++;
    }
    strncpy(outfile, outname, count);
    strcat(outfile, lispext);

    out = fopen(outfile, "w");
    fprintf(out, "(graphscr)\n");
    fprintf(out, "(command \"layer\" \"m\" \"borders\" ");
    fprintf(out, "\"c\" \"4\" \"\" \"\")\n");
    fprintf(out, "(command \"LINE\" \"0, 0\" \"33, 0\" \"\")\n");
    fprintf(out, "(command \"LINE\" \"0, 0\" \"0, 23\" \"\")\n");
    fprintf(out, "(command \"LINE\" \"33, 0\" \"33, 23\" \"\")\n");
    fprintf(out, "(command \"LINE\" \"0, 23\" \"33, 23\" \"\")\n");
    fprintf(out, "(command \"LINE\" \"0, 2\" \"33, 2\" \"\")\n");
    fprintf(out, "(command \"LINE\" \"15.5, 2\" \"15.5, 0\" \"\")\n");
    if ((strlen(subtitle)) > 0)
        fprintf(out, "(command \"LINE\" \"0, 21\" \"33, 21\" \"\")\n");
    else
        fprintf(out, "(command \"LINE\" \"0, 21.75\" \"33, 21.75\" \"\")\n");

    fprintf(out, "(command \"TEXT\" \"C\" \"16.5, 22.3\" \"0.4\" ");
    fprintf(out, "\"0.0\" \"%s\")\n", title);
    if ((strlen(subtitle)) > 0) {
        fprintf(out, "(command \"TEXT\" \"C\" \"16.5, 21.3\" \"0.3\" ");
        fprintf(out, "\"0.0\" \"%s\" )\n", subtitle);
    }
}

```



```
fprintf(out, "\n");
fclose(out);
```

```
void openInputFiles(void)
```

```
printf("\n\nWhat is the name of the node input file? \n ");
gets(nodefile);

if ((nodein = fopen(nodefile, "rt")) == NULL) {
    clrscr(); printf("\n\n\nNode input file not opened!");
    printf("\n\nThis file must be in the current directory"); exit(1);
}
fgets(dummystring, 80, nodein);
fscanf(nodein, " %d", &numofnodes);
fgets(dummystring, 80, nodein);
if (numofnodes > MAXNODES) {
    clrscr();
    printf("\n\nThe maximum number of nodes is %d", MAXNODES);
    printf("\n\nThis number must be reduced!");
    exit(1);
}
for (i = 0; i < numofnodes; i++) {
    fscanf(nodein, "%d %f %f %d %f %f", &theNodes[i].aNodeNum, \
        &theNodes[i].anXcoord, &theNodes[i].ayYcoord, \
        &theNodes[i].aNodeType, &theNodes[i].other, \
        &theNodes[i].other2);
    fgets(dummystring, 80, nodein);
}

fclose(nodein);

printf("\n\n\nWhat is the name of the link input file? \n ");
gets(linkfile);

if ((linkin = fopen(linkfile, "rt")) == NULL) {
    clrscr(); printf("\n\n\nLink input file not opened!");
    printf("\n\nThis file must be in the current directory"); exit(1);
}
fgets(dummystring, 80, nodein);
fscanf(linkin, " %d", &numoflinks);
fclose(linkin);

printf("\n\nIs there a label file? (y/n)");
ch = getch();
if (ch == 'y' /I ch == 'Y') {
    islabelfile = 1;
    printf ("\n\nWhat is the name of the label file? \n ");
    gets(labelfile);
    if ((labelin = fopen(labelfile, "rt")) == NULL) {
        clrscr(); printf("\n\n\nLabel input file not opened!");
        printf("\n\nThis file must be in the current directory");
        exit(1);
    }
}
fclose(labelin);
```

Header File "getcoord.h"

```
void getMaxCoords(void)
```

```
struct aLabel {
    int labelnode;
    float angle;
    char labeltext[21];
};
struct aLabel theLabels[MAXLABELS];
float xcoord, ycoord;
int j, k, nodenum, numoflabels;

if (islabelfile == 1) {
```

```

labelin = fopen(labelfile, "rt");
fgets(dummystring, 80, labelin);
fscanf(labelin, "%d", &numoflabels);
fgets(dummystring, 80, labelin);
for (j = 0; j < numoflabels; j ++){
    fscanf(labelin, " %d %f ", &theLabels[j].labelnode, \
        &theLabels[j].angle);
    fgets(theLabels[j].labeltext, 21, labelin);
}
fclose (labelin);
}

xmax = theNodes[0].anXcoord;    xmin = theNodes[0].anXcoord;
ymax = theNodes[0].aYcoord;    ymin = theNodes[0].aYcoord;
for (j = 1; j < numofnodes; j ++){
    if (theNodes[j].anXcoord > xmax) xmax = theNodes[j].anXcoord;
    if (theNodes[j].anXcoord < xmin) xmin = theNodes[j].anXcoord;
    if (theNodes[j].aYcoord > ymax) ymax = theNodes[j].aYcoord;
    if (theNodes[j].aYcoord < ymin) ymin = theNodes[j].aYcoord;

    if (islabeledfile == 1){
        for(k = 0; k < numoflabels; k ++){
            if (theNodes[j].aNodeNum == theLabels[k].labelnode) {
                xcoord = theNodes[j].anXcoord + \
                    ((strlen(theLabels[k].labeltext)) *
                    (0.4*labeltextsize)*(cos(theLabels[k].angle/57.29)));
                ycoord = theNodes[j].aYcoord \
                    + ((strlen(theLabels[k].labeltext)) * \
                    (0.4*labeltextsize)*(sin(theLabels[k].angle/57.29)));
                if (xcoord > xmax) xmax = xcoord;
                if (xcoord < xmin) xmin = xcoord;
                if (ycoord > ymax) ymax = ycoord;
                if (ycoord < ymin) ymin = ycoord;

            }
        }
    }

linkin = fopen(linkfile, "rt");
fgets(dummystring, 80, linkin);
fgets(dummystring, 80, linkin);
fscanf(linkin, "%d %d %d %d", &linknum, &hupnode, &downnode, &flow);
fgets(dummystring, 81, linkin);
maxflow = flow;
while (!feof(linkin)){
    fscanf(linkin, "%d %d %d %d", &linknum, &upnode, &downnode, &flow);
    fgets(dummystring, 81, linkin);
    if (feof(linkin)) break;
    if (flow > maxflow) maxflow = flow;
}
fclose(linkin);

if (issubtitle == 0) pageYspread = 17.9;
else pageYspread = 17.15;
xScaleFactor = pageXspread / (xmax - xmin);
yScaleFactor = pageYspread / (ymax - ymin);

```

Header File "params.h"

```

void calcParams(void)

    if (numofnodes <= 50) {
        diameter = 0.250;
        nodetextsize = 0.220;
        linktextsize = 0.180;
        zonetextsize = 0.300;
        nodetextdist = 0.415;
        linktextdist = 0.300;
        zonetextdist = 0.460;
        labeltextsize = 0.300;
    }

```

```

    }
    else if (numofnodes <= 200){
        diameter = 0.175;
        nodetextsize = 0.120;
        linktextsize = 0.140;
        zonetextsize = 0.160;
        nodetextdist = 0.300;
        linktextdist = 0.240;
        zonetextdist = 0.320;
        labeltextsize = 0.160;
    }
    else if (numofnodes <= 500){
        diameter = 0.120;
        nodetextsize = 0.040;
        linktextsize = 0.050;
        zonetextsize = 0.075;
        nodetextdist = 0.240;
        linktextdist = 0.210;
        zonetextdist = 0.250;
        labeltextsize = 0.075;
    }
    else if (numofnodes <= 10001 {
        diameter = 0.080;
        nodetextsize = 0.045;
        linktextsize = 0.035;
        zonetextsize = 0.055;
        nodetextdist = 0.200;
        linktextdist = 0.180;
        zonetextdist = 0.220;
        labeltextsize = 0.055;
    }
    else if (numofnodes <= 2000){
        diameter = 0.065;
        nodetextsize = 0.040;
        linktextsize = 0.030;
        zonetextsize = 0.050;
        nodetextdist = 0.180;
        linktextdist = 0.160;
        zonetextdist = 0.190;
        labeltextsize = 0.040;
    }
    else if (numofnodes <= 4000) {
        diameter = 0.050;
        nodetextsize = 0.035;
        linktextsize = 0.025;
        zonetextsize = 0.045;
        nodetextdist = 0.155;
        linktextdist = 0.140;
        zonetextdist = 0.165;
        labeltextsize = 0.045;
    }
    else {
        clrscr();
        printf("\n\nProgram Error!!\n\nError 1");
        exit(1);
    }
}

void changeUserParams(void);
void printParams(void);

void getUserParams(void)
{
    char chl;

    clrscr();
    printf("\n\nIt has been determined that the network has %d nodes.",\
        numofnodes);
}

```

```

printf("\nWith this, the program will be using the following values:");
printf("\n\n\tDiameter of node circles -- %f.", diameter);
printf("\n\n\tSize of text for node titles -- %f.", nodetextsize);
printf("\n\n\tSize of text for link titles -- %f.", linktextsize);
printf("\n\n\tSize of text for zone titles -- %f.", zonetextsize);
printf("\n\n\tDistance from nodes to their titles --%f.", nodetextdist);
printf("\n\n\tDistance from links to their titles --%f.", linktextdist);
printf("\n\n\tDistance from zones to their titles --%f.", zonetextdist);
printf("\n\nWould you like to change any of these values? (y/n)\n\n");
chl = getche();
if (chl == 'y' || chl == 'Y') changeUserParams();
}

void changeUserParams(void)
{
    int v;
    printParams();
    scanf("%d", &v);
    while (v != 9) {
        while(v > 9 || v < 1) {
            printf ("\nThat is not a valid choice. Try again.\n\n");
            printParams();
            scanf("%d", &v);
        }

        switch (v) {
            case 1:
                printf ("\n\nEnter the diameter of the node circles: ");
                scanf ("%f", &diameter); break;
            case 2:
                printf ("\n\nEnter the size of text for the nodes: ");
                scanf ("%f", &nodetextsize); break;
            case 3:
                printf ("\n\nEnter the size of text for the links: ");
                scanf ("%f", &linktextsize); break;
            case 4:
                printf ("\n\nEnter the size of text for the zones:");
                scanf ("%f", &zonetextsize); break;
            case 5:
                printf ("\n\nEnter the distance from the nodes to their titles: ");
                scanf ("%f", &nodetextdist); break;
            case 6:
                printf ("\n\nEnter the distance from the links to their titles: ");
                scanf ("%f", &linktextdist); break;
            case 7:
                printf ("\n\nEnter the distance from the zones to their titles: ");
                scanf ("%f", &zonetextdist); break;
            case 8:
                printf ("\n\nEnter the size of the text labels: ");
                scanf ("%f", &labeltextsize); break;
        }
        printParams();
        scanf("%d", &v);
    }

void printParams(void)

printf("\n\n1. Diameter of node circles -- %f.", diameter);
printf("\n\n2. Size of text for node titles -- %f.", nodetextsize);
printf("\n\n3. Size of text for link titles -- %f.", linktextsize);
printf("\n\n4. Size of text for zone titles -- %f.", zonetextsize);
printf("\n\n5. Distance from nodes to their titles -- %f.", nodetextdist ;
printf("\n\n6. Distance from links to their titles -- %f.", linktextdist ;
printf("\n\n7. Distance from zones to their titles -- %f.", zonetextdist ;
printf("\n\n8. Size of text for labels -- %f.", labeltextsize);
if (islabelfile == 0) printf(" (Not applicable)");
printf("\n\n9. Use these parameters and continue.");
printf("\n\nEnter the number of the parameter that you wish to change.");
printf("\n\nOr press '9' to use these parameters.\n\n");
}

```

Header File "plotype.h"

```

void buildSpecialNodes(void);
void buildSpecialZones(void);
void buildSpecialLinks(void);
void buildLinksbyWidth(void);
void getSpecialPlotType(void);

void getPlotType(void)
{
    int i;
    clrscr();
    printf("\n\nA regular type of plot uses the node numbers,");
    printf("\n zone numbers, and link numbers as labels.");
    printf("\n\nA specialized plot allows you to use labels other than");
    printf("\n the node, zone or link numbers, specify certain shapes for");
    printf("\n classes of nodes, zones or links, and show link widths");
    printf("\n proportional to flow to illustrate flow on the network.");
    printf("\n\nWhich would you like to plot?");
    printf("\n\n\t1. Regular plot");
    printf("\n\t2. Specialized Plot");
    printf("\n\nEnter 1 or 2:   ");
    scanf("%d", &i);
    while (1 < i & i > 2) {
        clrscr();
        printf ("\n\nThat is not a valid choice. Enter again.");
        printf("\n\n\t1. Regular plot");
        printf("\n\t2. Specialized Plot");
        printf("\n\nEnter 1 or 2");
        scanf("%d", &i);
    }
    if (i == 1){
        Ntype = 1; Nspecial = 0; Nlabel = 1;
        Ztype = 1; Zspecial = 0; Zlabel = 1;
        Ltype = 1; Lspecial = 0; Llabel = 1;
        return;
    }
    else getSpecialPlotType();
}

void getSpecialPlotType()
{
    int n;

    clrscr();
    printf("\n\nFor the NODES what would you like to plot?\n");
    printf("\n\t1. Regular node types - node numbers as labels.");
    printf("\n\t2. Regular node types - special numbers as labels.");
    printf("\n\t3. Regular node types - no labels.");
    printf("\n\t4. Specialized node types - node numbers as labels.");
    printf("\n\t5. Specialized node types - special numbers as labels.");
    printf("\n\t6. Specialized node types - no labels.");
    printf("\n\nEnter your selection.\n\n");
    scanf("%d", &n);
    while (n < 1 & n > 6) {
        clrscr();
        printf ("\n\nThat is not a valid choice. Enter again.");
        printf("\n\t1. Regular node types - node numbers as labels.");
        printf("\n\t2. Regular node types - special numbers as labels.");
        printf("\n\t3. Regular node types - no labels.");
        printf("\n\t4. Specialized node types - node numbers as labels.");
        printf("\n\t5. Specialized node types - special numbers as labels.");
        printf("\n\t6. Specialized node types - no labels.");
        printf("\n\nWhich would you like to plot?");
        scanf("%d", &n);
    }
    if (n == 1) { Ntype = 1; Nspecial = 0; Nlabel = 1; }
    if (n == 2) { Ntype = 2; Nspecial = 0; Nlabel = 2; }
    if (n == 3) { Ntype = 3; Nspecial = 0; Nlabel = 0; }
    if (n == 4) { Ntype = 4; Nspecial = 1; Nlabel = 1; }
    if (n == 5) { Ntype = 5; Nspecial = 1; Nlabel = 2; }
    if (n == 6) { Ntype = 6; Nspecial = 1; Nlabel = 0; }
    clrscr();
}

```

```

if (Nspecial == 1) buildSpecialNodes();

clrscr();
printf("\n\nFor the ZONES what would you like to plot?\n");
printf("\n\t1. Regular zone types - zone numbers as labels.");
printf("\n\t2. Regular zone types - special numbers as labels.");
printf("\n\t3. Regular zone types - no labels.");
printf("\n\t4. Specialized zone types - zone numbers as labels.");
printf("\n\t5. Specialized zone types - special numbers as labels.");
printf("\n\t6. Specialized zone types - no labels.");
printf("\n\nEnter your selection.\n\n");
scanf("%d", &n);
while (n < 1 & n > 6) {
    clrscr();
    printf ("\n\nThat is not a valid choice. Enter again.");
    printf("\n\t1. Regular zone types - zone numbers as labels.");
    printf("\n\t2. Regular zone types - special numbers as labels.");
    printf("\n\t3. Regular zone types - no labels.");
    printf("\n\t4. Specialized zone types - node numbers as labels.");
    printf("\n\t5. Specialized zone types - special numbers as labels.");
    printf("\n\t6. Specialized zone types - no labels.");
    printf("\n\nWhich would you like to plot?");
    scanf("%d", &n);
}
if (n == 1) { ztype = 1; Zspecial = 0; Zlabel = 1; }
if (n == 2) { Ztype = 2; Zspecial = 0; Zlabel = 2; }
if (n == 3) { ztype = 3; Zspecial = 0; Zlabel = 0; }
if (n == 4) { Ztype = 4; Zspecial = 1; Zlabel = 1; }
if (n == 5) { Ztype = 5; Zspecial = 1; Zlabel = 2; }
if (n == 6) { Ztype = 6; Zspecial = 1; Zlabel = 0; }
if (Zspecial == 1) buildSpecialZones();

clrscr();
printf("\n\nFor the LINKS what would you like to plot?\n");
printf("\n\t1. Regular link types - link numbers as labels.");
printf("\n\t2. Regular link types - special numbers as labels.");
printf("\n\t3. Regular link types - no labels.");
printf("\n\t4. Specialized link types - link numbers as labels.");
printf("\n\t5. Specialized link types - special numbers as labels.");
printf("\n\t6. Specialized link types - no labels.");
printf("\n\t7. Links displayed with variable width - no labels.");
printf("\n\nEnter your selection.\n\n");
scanf("%d", &n);
while (n < 1 & n > 7) {
    clrscr ();
    printf ("\n\nThat is not a valid choice. Enter again.");
    printf("\n\t1. Regular link types - link numbers as labels.");
    printf("\n\t2. Regular link types - special numbers as labels.");
    printf("\n\t3. Regular link types - no labels.");
    printf("\n\t4. Specialized link types - link numbers as labels.");
    printf("\n\t5. Specialized link types - special numbers as labels.");
    printf("\n\t6. Specialized link types - no labels.");
    printf("\n\t7. Links displayed with variable width - no labels.");
    printf("\n\nWhich would you like to plot?");
    scanf("%d", &n);

    if (n == 1) { Ltype = 1; Lspecial = 0; Llabel = 1; }
    if (n == 2) { Ltype = 2; Lspecial = 0; Llabel = 2; }
    if (n == 3) { Ltype = 3; Lspecial = 0; Llabel = 0; }
    if (n == 4) { Ltype = 4; Lspecial = 1; Llabel = 1; }
    if (n == 5) { Ltype = 5; Lspecial = 1; Llabel = 2; }
    if (n == 6) { Ltype = 6; Lspecial = 1; Llabel = 0; }
    if (n == 7) { Ltype = 7; Lspecial = 2; Llabel = 0; }
    clrscr();

    if (Lspecial == 1) buildSpecialLinks();
    if (Lspecial == 2) buildLinksbyWidth();
}

```

Header File "buildspc.h"

```
void printColorNums();
void printHatchNums();

void buildSpecialNodes()
{
    int i;
    clrscr();
    printf("\n\nHow many types of node symbols would you like?");
    printf("\n\nThe maximum number of node types is 4.\n\n");
    scanf("%d", &numOfNodeTypes);
    while (numOfNodeTypes < 1 & numOfNodeTypes > 4) {
        clrscr();
        printf("\n\nThat is not a valid choice. Enter again.");
        printf("\n\nThe maximum number of choices is 4.");
        printf("\n\nHow many types of nodes are there?\n\n");
        scanf("%d", &numOfNodeTypes);
    }
    for (i = 0; i < numOfNodeTypes; i ++){
        clrscr();
        printf("\n\nFor node type number %d; \n\n", i + 1);
        printf("What is the diameter of the node symbol?\n");
        printf("The current size being used is %f.", diameter);
        scanf("%f", &nodeTypes[i].nodeDiameter);

        printf("\n\n\nFor node type number %d; \n\n", i + 1);
        printf("What is the color of the node symbol?\n");
        printColorNums();
        scanf("%d", &nodeTypes[i].nodeTypeColor);

        printf("\n\n\nFor node type number %d; \n\n", i + 1);
        printf("What is the name of the node type? (30 chars max)\n");
        scanf("%s", nodeTypes[i].nodeTypeName);

        printf("\n\n\nFor node type number %d; \n\n", i + 1);
        printf("What is the low number of the node type?\n");
        scanf("%f", &nodeTypes[i].nodeTypeLow);

        printf("\n\n\nFor node type number %d; \n\n", i + 1);
        printf("What is the high number of the node type?\n");
        scanf("%f", &nodeTypes[i].nodeTypeHigh);

        printf("\n\n\nFor node type number %d; \n\n", i + 1);
        printf("What is the fill pattern of the node symbol?\n");
        printf("\n\t1. Empty circles\n\t2. Filled circles");
        scanf("%d", &nodeTypes[i].nodeFillType);
    }
}

void buildSpecialZones(void)

    int i;
    clrscr();
    printf("\n\nHow many types of zone symbols would you like?");
    printf("\n\nThe maximum number of zone types is 4.\n\n");
    scanf("%d", &numOfNodeTypes);
    while (numOfNodeTypes < 1 & numOfNodeTypes > 4) {
        clrscr();
        printf("\n\nThat is not a valid choice. Enter again.");
        printf("\n\nThe maximum number of choices is 4.");
        printf("\n\nHow many types of zones are there?\n\n");
        scanf("%d", &numOfNodeTypes);
    }
    for (i = 0; i < numOfNodeTypes; i ++){
        clrscr();
        printf("\n\nFor zone type number %d; \n\n", i + 1);
        printf("What is the diameter of the zone symbol?\n");
        printf("The current size being used is %f.", diameter);
        scanf("%f", &zoneTypes[i].nodeDiameter);

        printf("\n\n\nFor zone type number %d; \n\n", i + 1);
```

```

printf("What is the color of the zone symbol?\n");
printColorNums();
scanf("%d", &zoneTypes[i].nodeTypeColor);

printf("\n\n\nFor zone type number %d; \n\n", i + 1);
printf("What is the name of the zone type? (30 chars max)\n");
scanf("%s", zoneTypes[i].nodeTypeName);

printf("\n\n\nFor zone type number %d; \n\n", i + 1);
printf("What is the low number of the zone type?\n");
scanf("%f", &zoneTypes[i].nodeTypeLow);

printf("\n\n\nFor zone type number %d; \n\n", i + 1);
printf("What is the high number of the zone type?\n");
scanf("%f", &zoneTypes[i].nodeTypeHigh);

printf("\n\n\nFor zone type number %d; \n\n", i + 1);
printf("What is the fill pattern of the zone symbol?\n");
printf("\n\t1. Empty circles\n\t2. Filled circles");
scanf("%d", &zoneTypes[i].nodeFillType);
}

}
void buildSpecialLinks(void)
{
clrscr();
printf("\n\nHow many types of link symbols would you like?");
printf("\n\nThe maximum number of link types is 4.\n\n");
scanf("%d", &numOfLinkTypes);
while (numOfLinkTypes < 1 & numOfLinkTypes > 4) {
clrscr();
printf("\n\nThat is not a valid choice. Enter again.");
printf("\n\nThe maximum number of choices is 4.");
printf("\n\n\nHow many types of links are there?\n\n");
scanf("%d", &numOfLinkTypes);
}
for (i = 0; i < numOfLinkTypes; i++) {
clrscr();
printf("\n\nFor link type number %d; \n\n", i + 1);
printf("What is the width of the link symbol?\n");
printf("The current size of node diameter is %f.", diameter);
scanf("%f", &linkTypes[i].linkWidth);

printf("\n\n\nFor link type number %d; \n\n", i + 1);
printf("What is the color of the link symbol?\n");
printColorNums();
scanf("%d", &linkTypes[i].linkTypeColor);

printf("\n\n\nFor link type number %d; \n\n", i + 1);
printf("What is the name of the link type? (30 chars max)\n");
scanf("%s", linkTypes[i].linkTypeName);

printf("\n\n\nFor link type number %d; \n\n", i + 1);
printf("What is the low number of the link type?\n");
scanf("%f", &linkTypes[i].linkTypeLow);

printf("\n\n\nFor link type number %d; \n\n", i + 1);
printf("What is the high number of the link type?\n");
scanf("%f", &linkTypes[i].linkTypeHigh);

printf("\n\n\nFor link type number %d; \n\n", i + 1);
printf("What is the fill pattern of the link symbol?\n");
printf("\n\t1. Straight line\n\t2. Filled rectangles");
scanf("%d", &linkTypes[i].linkFillType);
}
}
void buildLinksbyWidth(void)
{
}

void printColorNums(void)
{
printf("\n\t1. Red \t\t5. Blue");
}

```



```

printf("\n\t2. Yellow\t\t6. Magenta");
printf("\n\t3. Green \t\t7. White");
printf("\n\t4. Cyan\n\n");

```

Header File "nodes.h"

```

void plotRegNodes(void)
{
    int nodesplotted = 0;
    clrscr();
    printf("\n\nWriting node data.");

    out = fopen(outfile, "a");
    fprintf(out, "(command \"layer\" \"m\" \"nodes\" ");
    fprintf(out, "\"c\" \"5\" \"\" \"\")\n");
    for (i = 0; i < numofnodes; i++){
        // Data must be imported from node file

        if (theNodes[i].aNodeType == 4){
            xcoord = 1 + (pageXspread/2) + (xScaleFactor \
                * (theNodes[i].anXcoord - ((xmax + xmin)/2)));
            ycoord = 3.1 + (pageYspread/2) + (yScaleFactor \
                * (theNodes[i].aYcoord - ((ymax + ymin)/2)));
            fprintf(out, "(command \"CIRCLE\" \"%f, %f\" \"D\" \"%f\")\n", \
                xcoord, ycoord, diameter);
            nodesplotted++;
        }
    }
    fprintf(out, "\n");
    fclose(out);
    numofzones = numofnodes - nodesplotted;
    printf ("\n\nData completed for %d nodes.\n\n", nodesplotted);
}

void plotNodeTitles(int j)
{
    out = fopen(outfile, "a");
    fprintf(out, "(command \"layer\" \"m\" \"node-titles\" ");
    fprintf(out, "\"c\" \"5\" \"\" \"\")\n");
    for (i = 0; i < numofnodes; i++){
        if (theNodes[i].aNodeType == 4) {
            //Data must be imported from node file
            xcoord = 1 + ((pageXspread)/2) t (xScaleFactor \
                * (theNodes[i].anXcoord - ((xmax t xmin)/2)));
            ycoord = 3.1 + ((pageYspread)/2) t (yScaleFactor \
                * (theNodes[i].aYcoord ~ ((ymax + ymin)/2)));
            fprintf(out, "(command \"TEXT\" \"C\" ");

            if (j == 1)
                fprintf(out, "\"%f, %f\" \"%f\" \"0.0\" \"%d\" )\n", \
                    xcoord t nodetextdist/3, ycoord ~ nodetextdist, \
                    nodetextsize, theNodes[i].aNodeNum);

            if (j == 2)
                fprintf(out, "\"%f, %f\" \"%f\" \"0.0\" \"%d\" )\n", \
                    xcoord + nodetextdist/3, ycoord ~ nodetextdist, \
                    nodetextsize, theNodes[i].other2);

        }

        fprintf(out, "\n");
        fclose(out);
    }
}

void plotRegZones(void)
{
    printf("\n\nWriting zone data.");
    out = fopen(outfile, "a");
    fprintf(out, "(command \"layer\" \"m\" \"zones\" ");
    fprintf(out, "\"c\" \"4\" \"\" \"\")\n");
    for (i = 0; i < numofnodes; i++){
        if (theNodes[i].aNodeType != 4) {
            xcoord = 1 + (pageXspread/2) t (xScaleFactor \
                * (theNodes[i].anXcoord' - ((xmax t xmin)/2)));

```

```

        ycoord = 3.1 + (pageYspread/2) + (yScaleFactor \
            * (theNodes[i].aYcoord - ((ymax + ymin)/2)));
        fprintf(out, "(command \"CIRCLE\" \"%f, %f\" \"D\" \"%f\")\n\", \
            xcoord, ycoord, diameter);
    }
    fprintf(out, "\n");
    fclose(out);
    printf ("\n\nData completed for %d zones.\n\n", numofzones);
}

void plotZoneTitles(int l)

    out = fopen(outfile, "a");
    fprintf(out, "(command \"layer\" \"m\" \"zone-titles\" ");
    fprintf(out, "\"c\" \"5\" \"\" \"\")\n");
    numofzones = 0;
    for (i = 0; i < numofnodes; i ++){
        if (theNodes[i].aNodeType != 4) {
            xcoord = 1 t ((pageXspread)/2) t (xScaleFactor \
                * (theNodes[i].anXcoord - ((xmax t xmin)/2)));
            ycoord = 3 t ((pageYspread)/2) + (yScaleFactor \
                * (theNodes[i].aYcoord - ((ymax + ymin)/2)));
            fprintf(out, "(command \"TEXT\" \"C\" ");
            if (l == 1)
                fprintf(out, "\"%f, %f\" \"%f\" \"0.0\" \"%d\" )\n\", \
                    xcoord + zonetextdist/3, ycoord - zonetextdist, \
                    zonetextsize, theNodes[i].aNodeNum);
            if (l == 2)
                fprintf(out, "\"%f, %f\" \"%f\" \"0.0\" \"%d\" )\n\", \
                    xcoord + zonetextdist/3, ycoord - zonetextdist, \
                    zonetextsize, theNodes[i].other);
            numofzones tt;
        }
    }
    fprintf(out, "\n");
    fclose(out);

```

Header File "links.h"

```

#include <math.h>

void plotRegLinks(void)
{
    float  upXcoord, upYcoord, downXcoord, downYcoord, alphaRAD;
    int j, gotnodes;
    out = fopen(outfile, "a");
    fprintf(out, "(command \"layer\" \"m\" \"links\" \"c\" ");
    fprintf(out, "\"4\" \"\" \"\" \"\")\n");
    fclose(out);
    linkin = (fopen(linkfile, "r"));
    fgets(dummystring, 80, linkin);
    fgets(dummystring, 80, linkin);
    for (i = 0; i < numoflinks; i ++){
        fscanf(linkin, "%d %d %d %f", &linknum, &upnode, &downnode, &flow);
        fgets(dummystring, 121, linkin);
        gotnodes = 0;
        for(j = 0; j < numofnodes; j tt) {
            if (theNodes[j].aNodeNum == upnode) {
                gotnodes t= 1;
                upnode = j;
            }
            if (theNodes[j].aNodeNum == downnode) {
                gotnodes t= 1;
                downnode = j;
            }
        }
        if (gotnodes == 2) break;
    }
}

```

```

        if ((theNodes[downnode].anXcoord - theNodes[upnode].anXcoord) == 0) {
            if ((theNodes[downnode].aYcoord - theNodes[upnode].aYcoord) > 0)
                alphaRAD = 1.5707965;
            else alphaRAD = 4.7123889;

        else {
            alphaRAD = atan2((theNodes[downnode].aYcoord - \
                theNodes[upnode].aYcoord), (theNodes[downnode].anXcoord - \
                theNodes[upnode].anXcoord));

            upXcoord = 1 t (pageXspread/2) t (xScaleFactor \
                * (theNodes[upnode].anXcoord - ((xmax + xmin)/2))) \
                t ((diameter/2) * (sin(alphaRAD)));
            upYcoord = 3.1 + (pageYspread/2) + (yScaleFactor \
                * (theNodes[upnode].aYcoord - ((ymax + ymin)/2))) \
                - ((diameter/2) * (cos(alphaRAD)));
            downXcoord = 1 + (pageXspread/2) t (xScaleFactor \
                * (theNodes[downnode].anXcoord - ((xmax + xmin)/2))) \
                t ((diameter/2) * (sin(alphaRAD)));
            downYcoord = 3.1 t (pageYspread/2) t (yScaleFactor \
                * (theNodes[downnode].aYcoord - ((ymax + ymin)/2))) \
                - ((diameter/2) * (cos(alphaRAD)));

            out = fopen(outfile, "a");
            fprintf(out, "(command \"LINE\" \"%f, %f\" \"%f, %f\" \"\")\n\", \
                upXcoord, upYcoord, downXcoord, downYcoord);
            fclose(out);

            fclose(linkin);
            out = fopen(outfile, "a");
            fprintf(out, "\n");
            fclose(out);

void plotLinkTitles(int l)
{
    float upXcoord, upYcoord, downXcoord, downYcoord, alphaRAD, alpha;
    int j, gotnodes;
    out = fopen(outfile, "a");
    fprintf(out, "(command \"layer\" \"m\" \"link-titles\" \"c\" ");
    fprintf(out, "\"4\" \"\" \"\" \"\")\n");
    fclose(out);
    linkin = (fopen(linkfile, "r"));
    fgets(dummystring, 80, linkin);
    fgets(dummystring, 80, linkin);
    for (i = 0; i < numoflinks; i++){
        fscanf(linkin, "%d %d %d %f", &linknum, &upnode, &downnode, &flow);
        fgets(dummystring, 121, linkin);
        gotnodes = 0;
        for(j = 0; j < numofnodes; j++){
            if (theNodes[j].aNodeNum == upnode) {
                upnode = j;
                gotnodes t= 1;

            if (theNodes[j].aNodeNum == downnode) {
                downnode = j;
                gotnodes t= 1;
            }
        }
        if (gotnodes == 2) break;

        if ((theNodes[downnode].anXcoord-theNodes[upnode].anXcoord) == 0.0) {
            if ((theNodes[downnode].aYcoord - theNodes[upnode].aYcoord) >0)
                alphaRAD = 1.5707965;
            else alphaRAD = 4.7123889;

        else {
            alphaRAD = atan2((theNodes[downnode].aYcoord - \
                theNodes[upnode].aYcoord), (theNodes[downnode].anXcoord - \
                theNodes[upnode].anXcoord));

```

```

upXcoord = 1 + (pageXspread/2) + {xScaleFactor \
* (theNodes[upnode].anXcoord - ((xmax + xmin)/2)) \
+ ((diameter/2) * (sin(alphaRAD)))};
upYcoord = 3.1 + (pageYspread/2) + {yScaleFactor \
* (theNodes[upnode].aYcoord - ((ymax + ymin)/2)) \
- ((diameter/2) * (cos(alphaRAD)))};
downXcoord = 1 + (pageXspread/2) + {xScaleFactor \
* (theNodes[downnode].anXcoord - ((xmax + xmin)/2)) \
+ ((diameter/2) * (sin(alphaRAD)))};
downYcoord = 3.1 + (pageYspread/2) + {yScaleFactor \
* (theNodes[downnode].aYcoord - ((ymax + ymin)/2)) \
- ((diameter/2) * (cos(alphaRAD)))};
if ((downXcoord - upXcoord) == 0.0) {
    if ((theNodes[downnode].aYcoord - theNodes[upnode].aYcoord) > 0)
        alpha = 1.5707965;
    else alpha = 4.7123889;
}
else alpha = atan2((downYcoord - upYcoord), (downXcoord - upXcoord));

out = fopen(outfile, "a");
fprintf(out, "(command \"TEXT\" \"M\" \"%f, %f\" ", \
((upXcoord + downXcoord)/2) + (linktextdist * sin(alphaRAD)), \
((upYcoord + downYcoord)/2) - (linktextdist * cos(alphaRAD)));
fprintf(out, "\"%.4f\" \"%.5f\" ", linktextsize, alpha * 57.29578);
if (1 == 1)
    fprintf(out, "\"%d\" )\n", linknum);
if (1 == 2)
    fprintf(out, "\"%f\" )\n", flow);
fclose(out);
}
fclose(linkin);
}

```

Header File "special.h"

```

void plotSpecialNodes()
{
    int nodesplotted = 0;
    clrscr();
    printf ("\nWriting special nodes\n\n");

    out = fopen(outfile, "a");
    fprintf(out, "(command \"layer\" \"m\" \"Nodes\" ");
    fprintf(out, "\"c\" \"5\" \"\" \"\")\n");
    int m, t;
    for (t = 0; t < numofNodeTypes; t++) {
        fprintf(out, "(command \"setcolor\" \"%d\" )\n", \
            nodeTypes[t].nodeTypeColor);
        if (nodeTypes[t].nodeFillType == 0)
            fprintf(out, "(command \"fill\" \"off\" )\n");
        else
            fprintf(out, "(command \"fill\" \"on\" )\n");
        for (i = 0; i < numofnodes; i++) {
            if (theNodes[i].aNodeType == 4 && nodeTypes[t].nodeTypeLow \
                <= theNodes[i].other && nodeTypes[t].nodeTypeHigh > \
                theNodes[i].other) {
                nodesplotted++;
                if (nodeTypes[t].nodeFillType == 1) {
                    fprintf(out, "(command \"circle\" \"%f, %f\" ", \
                        (1 + (pageXspread/2) + (xScaleFactor * \
                            (theNodes[i].anXcoord - ((xmax + xmin)/2))), \
                        (3.1 + (pageYspread/2) + (yScaleFactor * \
                            (theNodes[i].aYcoord - ((ymax + min)/2))));
                    fprintf(out, "\"D\" \"%f\" )\n", \
                        nodeTypes[t].nodeDiameter);
                }
                if (nodeTypes[t].nodeFillType == 2) {
                    fprintf(out, "(command \"donut\" \"0\" \"%f\" ", \
                        nodeTypes[t].nodeDiameter);
                    fprintf(out, "\"%f, %f\" \"\" \"\", (1 + \
                        (pageXspread/2) + (xScaleFactor * (theNodes[i].anXcoord \
                            - ((xmax + xmin)/2))), (3.1 + (pageYspread/2) + \

```

```

        (yScaleFactor * (theNodes[i].aYcoord - \
        ((ymax + ymin)/2)))));

fclose(out);
numofzones = numofnodes - nodesplotted;
printf ("\n\nData completed for %d nodes.\n\n", nodesplotted);
}

```

Header File "specialz.h"

```

void plotSpecialZones()
{
    out = fopen(outfile, "a");
    fprintf(out, "(command \"layer\" \"m\" \"Zones\" ");
    fprintf(out, "\"c\" \"5\" \"\" \"\")\n");
    int m, t;
    for (t = 0; t < numofZoneTypes; t++) {
        fprintf(out, "(command \"setcolor\" \"%d\" )\n", \
                zoneTypes[t].nodeTypeColor);
        if (zoneTypes[t].nodeFillType == 0)
            fprintf(out, "(command \"fill\" \"off\" )\n");
        else
            fprintf(out, "(command \"fill\" \"on\" )\n");
        nodein = fopen(nodefile, "r");
        fgets(dummystring, 80, nodein);
        fgets(dummystring, 80, nodein);
        for (i = 0; i < numofnodes; i++) {
            fscanf(nodein, "%d %f %f %d ", nodenum, xcoord, \
                    ycoord, nodetype, other);
            fgets(dummystring, 80, nodein);
            if (nodetype != 4 && zoneTypes[t].nodeTypeLow < other && \
                zoneTypes[t].nodeTypeHigh > other) {
                if (zoneTypes[t].nodeFillType == 0) {
                    fprintf(out, "(command \"circle\" \"%f,%f\" ", \
                            (1 t (pageXspread/2) t (xScaleFactor * xcoord
                            - ((xmax t xmin)/2))), (3.1 t (pageYspread/2) + \
                            (yScaleFactor * (ycoord - ((ymax + ymin)/2))),
                    fprintf(out, "D\" \"%f\" )\n", zoneTypes[t].nodeDiameter);
                }
                if (zoneTypes[t].nodeFillType == 1) {
                    fprintf(out, "(command \"donut\" \"0\" \"%f\" ", \
                            zoneTypes[t].nodeDiameter);
                    fprintf(out, "\"%f,%f\" \"\" )\n", (1 + \
                            (pageXspread/2) t (xScaleFactor * (xcoord - \
                            ((xmax t xmin)/2))), (3.1 t (pageYspread/2) + \
                            (yScaleFactor * (ycoord - ((ymax + ymin)/2)))));
                }
            }
        }
        fclose (nodein);
    }
    fclose(out);
}

```

Header File "special.h"

```

void plotSpecialLinks()
{
    out = fopen(outfile, "a");
    fprintf(out, "(command \"layer\" \"m\" \"Links\" ");
    fprintf(out, "\"c\" \"5\" \"\" \"\")\n");
    int m, t;
    for (t = 0; t < numofLinkTypes; t++) {
        fprintf(out, "(command \"setcolor\" \"%d\" )\n", \
                nodeTypes[t].nodeTypeColor);
        if (nodeTypes[t].nodeFillType == 0)

```

```

        fprintf(out, "(command \"fill\" \"off\")\n");
    else
        fprintf(out, "(command \"fill\" \"on\")\n");
    nodein = fopen(nodefile, "r");
    fgets(dummystring, 80, nodein);
    fgets(dummystring, 80, nodein);
    for (i = 0; i < numofnodes; i ++){
    fscanf(nodein, "%d %f %f %d ", nodenum, xcoord, \
        ycoord, nodetype, other);
        fgets(dummystring, 80, nodein);
        if (nodetype == 4 && nodeTypes[t].nodeTypeLow < other && \
            nodeTypes[t].nodeTypeHigh > other) {
            if (nodeTypes[t].nodeFillType == 0){
                fprintf(out, "(command \"circle\" \"%f,%f\" ", \
                    (1 t (pageXspread/2) t (xScaleFactor * xcoord
\
                    - ((xmax + xmin)/2))), (3.1 +
\
                    (pageYspread/2) + \
                    (yScaleFactor * (ycoord - ((ymax +
ymin)/2))))),
                fprintf(out, "D\" \"%f\")\n", \
                    nodeTypes[t].nodeDiameter);
            }
            if (nodeTypes[t].nodeFillType == 1) {
                fprintf(out, "(command \"donut\" \"%0\" \"%f\" ", \
                    nodeTypes[t].nodeDiameter);
                fprintf(out, "\"%f, %f\" \"\" )\n", (1 + \
                    (pageXspread/2) t (xScaleFactor * (xcoord - \
                    ((xmax t xmin)/2))))), (3.1 t (pageYspread/2) + \
                    (yScaleFactor * (ycoord - ((ymax + ymin)/2)))));
            }
        }
    }
}
fclose (nodein);
fclose(out);

```

Header File "legends.h"

```

void drawLegend(void)
{
    out = fopen(outfile, "a");
    fprintf(out, "(command \"layer\" \"m\" \"legends\" ");
    fprintf(out, "\"c\" \"1\" \"\" \"\")\n");

    fprintf(out, "(command \"TEXT\" \"ML\" \"17.5, 1.50\" ");
    fprintf(out, "\"0.350\" \"0.00\" \"Nodes\" )\n");
    fprintf(out, "(command \"TEXT\" \"ML\" \"17.5, 0.50\" ");
    fprintf(out, "\"0.350\" \"0.00\" \"Zones\" )\n");
    fprintf(out, "(command \"TEXT\" \"ML\" \"25.25, 1.50\" ");
    fprintf(out, "\"0.350\" \"0.00\" \"Links\" )\n");

    fprintf(out, "(command \"TEXT\" \"MR\" \"21.25, 1.50\" ");
    fprintf(out, "\"%f\" \"0.00\" \"%d\" )\n", \
        nodetextsize, numofnodes - numofzones);
    fprintf(out, "(command \"TEXT\" \"MR\" \"21.25, 0.50\" ");
    fprintf(out, "\"%f\" \"0.00\" \"%d\" )\n", zonetextsize, numofzones);
    fprintf(out, "(command \"TEXT\" \"MR\" \"31.00, 1.50\" ");
    fprintf(out, "\"%f\" \"0.00\" \"%d\" )\n", linktextsize, numoflinks);

    fclose(out);
}

void drawSpecialTypeLegend()

```

Header File "labels.h"

```

void plotLabels ()
{
    printf ("\n\nPrinting Labels");

    struct aLabel {
        int labelnode;
        float angle;
        char labeltext[21];
    };
    struct aLabel theLabels[MAXLABELS];
    int j, k, l, m;
    int nodenum, numoflabels;
    float xcoord, ycoord;
    labelin = fopen(labelfile, "rt");
    fgets(dummystring, 80, labelin);
    fscanf(labelin, "%d", &numoflabels);
    fgets(dummystring, 80, labelin);
    for (j = 0; j < numoflabels; j++){
        fscanf(labelin, " %d %f ", &theLabels[j].labelnode,
            &theLabels[j].angle);
        fgets(theLabels[j].labeltext, 20, labelin);
        m = strlen(theLabels[j].labeltext);
        theLabels[j].labeltext[m - 1] = '\0';
    }
    fclose (labelin);
    out = fopen(outfile, "a");
    fprintf(out, "(command \"layer\" \"%m\" \"labels\" ");
    fprintf(out, " \"c\" \"2\" \"\" \"\" \"\" \"\" \n");
    for (k = 0; k < numoflabels; k++){
        for (l = 0; l < numofnodes; l++){
            if (theLabels[k].labelnode == theNodes[l].aNodeNum){
                break;
            }
        }
        xcoord = 1 + ((pageXspread)/2) + (xScaleFactor \
            * (theNodes[l].anXcoord - ((xmax + xmin)/2)));
        ycoord = 3.1 + ((pageYspread)/2) + (yScaleFactor \
            * (theNodes[l].aYcoord - ((ymax + ymin)/2)));
        fprintf(out, "(command \"TEXT\" \"ML\" ");
        fprintf(out, "\"%f, %f\" \"%f\" \"%f\" \"%s\" \n", \
            xcoord + ((cos((theLabels[k].angle)/57.29)) * 3 * diameter], \
            ycoord + ((sin((theLabels[k].angle)/57.29)) * 3 * diameter], \
            labeltextsize, theLabels[k].angle, \
            theLabels[k].labeltext);
    }
    fclose(out);
}

```

Appendix F: Caltrans' Ramp Metering Plans

Listed below are the ramp metering plans from Caltrans. Note that these plans are from 1993 (before the Northridge Earthquake), so they are not necessarily still valid. However, these were the plans that were coded into the baseline simulations of the Santa Monica Freeway. The eastbound and westbound ramps are given separately. In each case, the morning and afternoon timing plans are given; the signal time listed for each period is the phase length for the ramp meter. The metering rate can be determined by dividing 3600 by the phase length. For example, a phase length of 6 seconds will yield a metering rate of 600 vehicles/hour.

Eastbound Freeway Ramps

	time	phase length	time	phase length
Overland	6:30 to 9:30 A.M.	6.6 seconds	2:00 to 3:30 P.M.	4 seconds
	9:30 to 10:00 A.M.	4 seconds	3:30 to 6:30 P.M.	5.5 seconds
			6:30 to 7:00 P.M.	4 seconds
Manning	6:30 to 7:00 A.M.	6 seconds	2:00 to 3:30 P.M.	4 seconds
	7:00 to 8:00 A.M.	12 seconds	3:30 to 4:00 P.M.	6 seconds
	8:00 to 8:15 A.M.	10 seconds	4:00 to 6:30 P.M.	8.6 seconds
	8:15 to 9:00 A.M.	8.6 seconds	6:30 to 7:00 P.M.	4 seconds
	9:00 to 9:30 A.M.	6 seconds		
	9:30 to 10:00 A.M.	4 seconds		
Nat'l/Roberts	6:30 to 7:00 A.M.	7.5 seconds	2:00 to 3:30 P.M.	4 seconds
	7:00 to 9:00 A.M.	12 seconds	3:30 to 5:30 P.M.	4.6 seconds
	9:00 to 9:30 A.M.	7.6 seconds	5:30 to 6:30 P.M.	6 seconds
	9:30 to 10:00 A.M.	4 seconds	6:30 to 7:00 P.M.	4 seconds
La Cienega	6:30 to 7:00 A.M.	10 seconds	2:00 to 3:30 P.M.	4 seconds
	7:00 to 8:45 A.M.	15 seconds	3:30 to 4:00 P.M.	10 seconds
	8:45 to 9:00 A.M.	10 seconds	4:00 to 5:00 P.M.	15 seconds
	9:00 to 9:30 A.M.	7.5 seconds	5:00 to 6:30 P.M.	12 seconds
	9:30 to 10:00 A.M.	4 seconds	6:30 to 7:00 P.M.	4 seconds
Venice	6:30 to 7:00 A.M.	10 seconds	2:00 to 3:30 P.M.	4 seconds
	7:00 to 8:45 A.M.	15 seconds	3:30 to 4:30 P.M.	10 seconds
	8:45 to 9:15 A.M.	10 seconds	4:30 to 5:15 P.M.	8.6 seconds
	9:15 to 9:30 A.M.	7.5 seconds	5:15 to 6:30 P.M.	10 seconds
	9:30 to 10:00 A.M.	4 seconds	6:30 to 7:00 P.M.	4 seconds
Washington	6:30 to 7:00 A.M.	4 seconds	2:00 to 7:00 P.M.	4 seconds
	7:00 to 7:45 A.M.	5 seconds		
	7:45 to 9:00 A.M.	6 seconds		
	9:00 to 10:00 A.M.	4 seconds		

	time	phase length	time	phase length
SB La Brea	6:30 to 7:00 A.M.	5 seconds	2:00 to 3:30 P.M.	4 seconds
	7:00 to 8:30 A.M.	10 seconds	3:30 to 4:00 P.M.	6 seconds
	8:30 to 9:00 A.M.	8.6 seconds	4:00 to 5:00 P.M.	8.6 seconds
	9:00 to 9:30 A.M.	5 seconds	5:00 to 6:30 P.M.	6 seconds
	9:30 to 10:00 A.M.	4 seconds	6:30 to 7:00 P.M.	4 seconds
NB La Brea	6:30 to 7:00 A.M.	6 seconds	2:00 to 3:30 P.M.	4 seconds
	7:00 to 9:00 A.M.	8.6 seconds	3:30 to 4: 15 P.M.	7.5 seconds
	9:00 to 9:30 A.M.	6 seconds	4:15 to 6:30 P.M.	8.6 seconds
	9:30 to 10:00 A.M.	4 seconds	6:30 to 7:00 P.M.	4 seconds
Crenshaw	6:30 to 7:00 A.M.	6 seconds	2:00 to 3:30 P.M.	4 seconds
	7:00 to 7: 15 A.M.	6.5 seconds	3:30 to 4:15 P.M.	6 seconds
	7: 15 to 8: 15 A.M.	7 seconds	4: 15 to 6:30 P.M.	7.5 seconds
	8: 15 to 9:00 A.M.	8.6 seconds	6:30 to 7:00 P.M.	4 seconds
	9:00 to 9:30 A.M.	6 seconds		
	9:30 to 10:00 A.M.	4 seconds		
Arlington	6:30 to 8:00 A.M.	6 seconds	2:00 to 3:30 P.M.	4 seconds
	8:00 to 9:00 A.M.	8.6 seconds	3:30 to 6:30 P.M.	17 seconds
	9:00 to 9:30 A.M.	6 seconds	6:30 to 7:00 P.M.	4 seconds
	9:30 to 10:00 A.M.	4 seconds		
Western	6:30 to 7:00 A.M.	5 seconds	2:00 to 3:30 P.M.	4 seconds
	7:00 to 8:30 A.M.	7.5 seconds	3:30 to 5:45 P.M.	10 seconds
	8:30 to 9: 15 A.M.	10 seconds	5:45 to 6:30 P.M.	8.6 seconds
	9: 15 to 9:30 A.M.	5 seconds	6:30 to 7:00 P.M.	4 seconds
	9:30 to 10:00 A.M.	4 seconds		
Normandie	6:30 to 7:00 A.M.	7.5 seconds	2:00 to 3:30 P.M.	4 seconds
	7:00 to 9: 15 A.M.	8.6 seconds	3:30 to 7:00 P.M.	8.6 seconds
	9:15 to 9:30 A.M.	7.5 seconds		
	9:30 to 10:00 A.M.	4 seconds		

	time	phase length	time	phase length
Vermont	6:30 to 9:30 A.M.	6 seconds	2:00 to 3:30 P.M.	4 seconds
	9:30 to 10:00 A.M.	4 seconds	3:30 to 6:30 P.M.	8.6 seconds
			6:30 to 7:00 P.M.	4 seconds
Hoover	6:30 to 9:00 A.M.	10 seconds	2:00 to 3:30 P.M.	4 seconds
	9:00 to 9:30 A.M.	6 seconds	3:30 to 6:30 P.M.	7.5 seconds
	9:30 to 10:00 A.M.	4 seconds	6:30 to 7:00 P.M.	4 seconds

Westbound Freeway Ramps

	time	phase length	time	phase length
Hoover	6:30 to 8:15 A.M.	6.6 seconds	2:30 to 3:30 P.M.	4 seconds
	8: 15 to 9:00 A.M.	12 seconds	3:30 to 6:15 P.M.	10 seconds
	9:00 to 10:00 A.M.	4 seconds	6: 15 to 6:30 P.M.	4 seconds
Vermont	6:30 to 9:00 A.M.	10 seconds	2:30 to 3:30 P.M.	10 seconds
	9:00 to 10:00 A.M.	8.6 seconds	3:30 to 6:30 P.M.	15 seconds
Normandie	6:30 to 8:00 A.M.	6 seconds	2:30 to 3:30 P.M.	4 seconds
	9:00 to 9:00 A.M.	8.6 seconds	3:30 to 5:00 P.M.	12 seconds
	9:00 to 10:00 A.M.	4 seconds	5:00 to 6:30 P.M.	15 seconds
Western	6:30 to 8:00 A.M.	7.5 seconds	2:30 to 3:30 P.M.	4 seconds
	8:00 to 8:45 A.M.	10 seconds	3:30 to 4:00 P.M.	8.6 seconds
	8:45 to 9:00 A.M.	5 seconds	4:00 to 4:30 P.M.	12 seconds
	9:00 to 10:00 A.M.	4 seconds	4:30 to 6:00 P.M.	15 seconds
			6:00 to 6:30 P.M.	8.6 seconds
Arlington	6:30 to 9:00 A.M.	12 seconds	2:30 to 4:45 P.M.	10 seconds
	9:00 to 10:00 AM.	4 seconds	4:45 to 5:45 P.M.	12 seconds
			4:45 to 6:30 P.M.	10 seconds
Crenshaw	6:30 to 7:15 A.M.	10 seconds	2:30 to 3:30 P.M.	4 seconds
	7: 15 to 9:00 A.M.	7.5 seconds	3:30 to 3:45 P.M.	10 seconds
	9:00 to 10:00 A.M.	4 seconds	3:45 to 4:30 P.M.	12 seconds
			4:30 to 6: 15 P.M.	15 seconds
			6: 15 to 6:30 P.M.	10 seconds

	time	phase length	time	phase length
NB La Brea	6:30 to 8:30 A.M.	8.6 seconds	2:30 to 3:30 P.M.	4 seconds
	8:30 to 9:00 A.M.	6.6 seconds	3:30 to 3:45 P.M.	8.6 seconds
	9:00 to 10:00 A.M.	4 seconds	3:45 to 5:15 P.M.	10 seconds
			5:15 to 5:45 P.M.	8.6 seconds
			5:45 to 6:30 P.M.	12 seconds
SB La Brea	6:30 to 9:00 A.M.	7.5 seconds	2:30 to 3:30 P.M.	4 seconds
	9:00 to 10:00 A.M.	4 seconds	3:30 to 5:00 P.M.	10 seconds
			5:00 to 6:30 P.M.	12 seconds
Fairfax	6:30 to 7:15 A.M.	10 seconds	2:30 to 3:30 P.M.	4 seconds
	7:15 to 8:15 A.M.	6.6 seconds	3:30 to 4:00 P.M.	7.5 seconds
	9:15 to 9:00 A.M.	10 seconds	4:00 to 6:30 P.M.	10 seconds
	9:00 to 10:00 A.M.	4 seconds		
La Cienega	6:30 to 10:00 A.M.	4 seconds	2:30 to 6:30 P.M.	4 seconds
Robertson	6:30 to 7:15 A.M.	12 seconds	2:30 to 3:30 P.M.	4 seconds
	7:15 to 9:00 A.M.	6.6 seconds	3:30 to 6:30 P.M.	6 seconds
	9:00 to 10:00 A.M.	4 seconds		
Overland	6:30 to 7:15 A.M.	8.6 seconds	2:30 to 3:30 P.M.	4 seconds
	7:15 to 9:00 A.M.	5.4 seconds	3:30 to 6:30 P.M.	6 seconds
	9:00 to 10:00 A.M.	4 seconds		

Appendix G: Freeway Demand Data

The charts on the following pages contain the complete demand data collected for the Santa Monica Freeway. Section 4.2 in the main report describes these data and the process by which they were obtained. As mentioned previously, this may be the best available single source of these data, so they are provided here for reference.

To make these data readable, the charts were split into two sets of pages. Each line on pages **A55-A62** is a single source of demand data for a single location on the **freeway** (e.g., the Overland on-ramp to the eastbound **freeway** as captured by the MODCOMP system on Tuesday May 25, 1993). Each subsequent column is the hourly flow observed (measured in vehicles/hour) for half hour time slices **from 6:00 A.M. to 11:30 A.M.** Pages **A63-A70** contain similar information for the remainder of the typical day. Each line contains the observed flows at the same locations from 11:30 A.M. to 8:00 P.M.

Location	Dir	Source	Date	6:00	6:30	7:00	7:30	8:00	8:30	9:00	9:30	10:00	10:30	11:00	11:30	
West of I405	EB	(SIMULATION DATA)	10/04/93	1810	3005	3776	4806	5924	5934	5019	4199	3625	3928	4326	4497	
		calculated		.818	3018	4394	5228	5648	5658	5040	4618	4042	3944	4344	4516	
		Previous UCB studies	1980s	1499	2854	3648	3856	4271	4409	3814	3657					
NBI-405 on ramp	EB	(SIMULATION DATA)		1553	1474	1418	1291	966	1048	1611	1834	1846	1878	1878	1806	
		smart corridor data	11/21/90	1520	1520	1360	1360	1010	1010	1730	1730	1870	1870	1850	1850	
		(converted)		1560	1480	1424	1296	970	1052	1618	1842	1854	1886	1886	1814	
SBI-405 on ramp	EB	(SIMULATION DATA)		1269	1679	2161	2420	2480	2460	2259	2163	2129	2053	1900	1926	
		Smart Corridor data	11/21/90	1480	1480	2300	2300	2480	2480	2220	2220	2100	2100	1920	1920	
		(converted)		1274	1686	2170	2430	2490	2470	2268	2172	2138	2062	1908	1934	
I-405 on ramps (after merge)	EB	(SIMULATION DATA)		3193	3356	3587	3615	3336	3434	3916	4099	4067	4135	4280	4284	
		Caltrans Hourly Count	WED 05/16/90	3288	3288	3616	3616	3399	3399	4023	4023	4118	4118	4300	4300	
		(converted)		3206	3370	3602	3630	3350	3448	3932	4116	4084	4152	4298	4302	
		Smart Corridor data	11/21/90	3000	3000	3660	3660	3590	3590	3950	3950	3970	3970	3770	3770	
National off ramp	EB	(SIMULATION DATA)		388	540	683	842	1052	1159	1209	1161	898	944	1297	1460	
		Caltrans Hourly Count	TUE 11/14/89	466	466	766	766	1109	1109	1190	1190	925	925	1384	1384	
		(converted)		390	542	686	846	1056	1164	1214	1166	902	948	1302	1466	
		Previous UCB studies	1980s	467	529	709	872	947	1016	994	871					
Overland on ramp	EB	(SIMULATION DATA)		665	852	1093	1181	1113	1066	970	930	950	906	783	787	
		Caltrans Hourly Count	TUE 11/14/89	162	762	1141	1141	1094	1094	954	954	932	932	788	788	
		(converted)		668	856	1098	1186	1118	1070	974	934	954	910	786	790	
		MODCOMP data	TUE 05/25/93	400	820	650	730	780	720	700	700	920	910	900	850	
		MODCOMP data	WED 05/26/93	450	810	700	790	930	750	690	750	860	870	850	890	
		MODCOMP data	THU 05/27/93	480	840	740	750	1030	870	690	760	850	840	940	890	
Mainline 6.745 (Motor)	EB	Caltrans Hourly Count	WED 09/19/90	6001	6001	8803	8803	9036	9036	8617	8617	8119	8119	8135	8135	
		(converted)		5302	6702	8408	9198	9060	9012	8732	8502	8178	8058	8126	8142	
		Caltrans Hourly Count	TUE 06/16/92	6078	6078	8742	8742	7302	7302	9450	9450	8099	8099	7885	7885	
		(converted)		5412	6744	8578	8904	7218	7386	9346	9554	8290	7990	7810	7960	
		MODCOMP data	TUE 05/25/93	4140	7040	9190	8440	8370	8310	8260	8400	7800	7590	7280	7810	
		MODCOMP data	WED 05/26/93	4440	7200	8890	8490	8160	7860	8260	7880	7200	7430	7620	8090	
		MODCOMP data	THU 05/27/93	4340	7310	9130	8460	8060	8120	8300	8170	7710	7770	7600	7710	
Manning on ramp	EB	(SIMULATION DATA)		190	490	570	520	470	490	530	470	460	400	400	430	
		Caltrans Hourly Count	TUE 11/28/89	369	369	543	543	513	513	484	484	432	432	391	391	
		(converted)		326	414	524	560	520	506	494	474	444	420	394	386	
		MODCOMP data	TUE 05/25/93	190	490	570	520	470	490	530	470	460	400	400	430	
		MODCOMP data	WED 05/26/93	220	490	590	420	520	460	580	480	460	380	430	430	
		MODCOMP data	THU 05/27/93	190	480	590	520	SW	490	480	460	440	380	450	460	
Robertson off ramp	EB	(SIMULATION DATA)		323	528	752	926	1091	1101	900	859	920	929	916	906	
		Caltrans Hourly Count	TUE 11/28/89	419	419	826	826	1079	1079	866	866	909	909	897	897	
		(converted)		318	520	740	912	1074	1084	886	846	906	914	902	892	
		Previous UCB studies	1980s	447	529	579	864	927	1012	1003	931					
Mainline 7.99 (National)	EB	MODCOMP data	TUE 05/25/93	3850	6920	8980	7290	7550	7780	8070	7880	7550	7010	7420	7280	
		MODCOMP data	WED 05/26/93	4050	7010	8610	7330	6860	7330	8350	7600	7170	7120	7210	7240	
		MODCOMP data	THU 05/27/93	3960	7080	8900	7900	7030	7950	7780	7550	7330	7260	7500	7360	
		MODCOMP data	TUE 12/08/92	3890	6100	7894	7220	6882	6876	6922	7044	6932	6894	6800	6968	

Location	Dir	Source	Date	6:00	6:30	7:00	7:30	8:00	8:30	9:00	9:30	10:00	10:30	11:00	11:30
National on ramp	EB	(SIMULATION DATA)		260	470	440	390	380	390	540	630	690	800	730	670
		Caltrans Hourly Count	TUE 11/28/89	389	389	509	509	486	486	614	614	664	664	708	708
		(converted)		360	420	496	522	474	500	592	638	652	676	698	718
		MODCOMP data	TUE 05/25/93	260	470	440	390	380	390	540	630	690	800	730	670
		MODCOMP data	WED 05/26/93	270	470	420	390	370	390	530	580	630	650	580	720
		MODCOMP data	THU 05/27/93	280	470	400	460	360	470	480	580	640	660	690	660
		Previous UCB studies	1980s	351	374	332	336	322	339	400	404				
SB La Cienega off ramp	EB	(SIMULATION DATA)		59	100	148	173	173	181	197	203	199	209	242	250
		Caltrans Hourly Count	TUE 11/28/89	78	78	159	159	174	174	198	198	201	201	241	241
		(converted)		58	98	146	170	170	178	194	200	196	206	7.38	246
SB La Cienega on ramp	EB	(SIMULATION DATA)		390	540	660	560	600	620	580	500	600	590	m 1	680
		Caltrans Hourly Count	TUE 11/28/89	505	505	598	598	598	598	640	640	661	661	617	617
		(converted)		482	528	588	608	592	604	632	648	664	658	612	622
		MODCOMP data	TUE 05/25/93	390	540	660	560	600	620	580	400	0	0	420	680
		MODCOMP data	WED 05/26/93	360	500	550	510	580	580	500	600	590	600	560	560
		MODCOMP data	THU 05/27/93	310	430	560	490	410	500	520	540	520	540	520	520
		Previous UCB studies	1980s	61	279	312	385	380	500	452	507				
NB La Cienega off ramp	EB	(SIMULATION DATA)		179	297	410	538	725	788	725	707	699	681	634	636
		Caltrans Hourly Count	TUE 11/28/89	234	234	468	468	745	745	705	705	679	679	625	625
		(converted)		176	292	404	530	714	776	714	6%	688	670	624	626
Fairfax off ramp	EB	(SIMULATION DATA)		170	270	320	450	440	620	620	530	570	480	560	490
		Caltrans Hourly Count	TUE 11/28/89	221	221	436	436	604	604	593	593	486	486	525	523
		(converted)		168	274	388	484	582	624	608	578	494	478	526	524
		MODCOMP data	TUE 05/25/93	170	270	320	450	440	620	620	530	570	480	560	490
		MODCOMP' data	WED 05/26/93	170	270	330	460	410	530	620	600	460	470	440	490
		MODCOMP data	THU 05/27/93	170	270	360	480	450	630	560	550	490	460	510	510
		Previous UCB studies	1980s	151	179	252	251	300	318	372	359				
Mainline 9.21 (Fairfax)	EB	MODCOMP data	TUE 05/25/93	4130	7000	8870	7310	7260	7720	7790	7970	7700	7540	7440	7470
		MODCOMP data	WED 05/26/93	4210	7350	8680	6790	7110	7230	8160	7830	7260	7120	7320	7430
		MODCOMP data	THU 05/27/93	4200	7270	8960	7500	6740	8320	7660	7580	7420	7360	7570	7650
		MODCOMP data	TUE 12/08/92	4502	7733	9223	8100	6965	7680	7466	8064	7442	7750	7570	8150
		Previous UCB studies	1980s	339	381	607	636	515	473	385	346				
Venice on ramp	EB	(SIMULATION DATA)		258	289	317	349	386	400	392	376	331	339	417	439
		Caltrans Hourly Count	TUE 11/28/89	269	269	328	328	387	387	378	378	330	330	421	421
		(converted)		254	284	312	344	380	394	386	370	326	334	410	432
		MODCOMP data	TUE 05/25/93	120	210	340	270	270	320	300	290	370	410	280	210
		MODCOMP data	WED 05/26/93	120	230	290	290	270	320	270	270	270	230	230	270
Washington on ramp	EB	(SIMULATION DATA)		944	956	980	966	890	892	968	992	980	996	1032	1064
		Caltrans Hourly Count	TUE 11/28/89	954	954	977	977	895	895	984	984	992	992	1052	1052
		(converted)		948	960	984	970	894	8%	972	996	984	1000	1036	1068
		Previous UCB studies	1980s	587	1029	975	894	796	766	858	815				
		ATSAAC data	TUE 10/19/93	134	126	176	150	172	172	222	234	274	274	284	280
SB La Brea off ramp	EB	(SIMULATION DATA)		69	95	132	138	108	116	163	181	185	189	187	195
		Caltrans Hourly Count	TUE 12/05/89	81	81	133	133	110	110	168	168	183	183	188	188
		(converted)		68	94	130	136	106	114	160	178	182	186	184	192
		Previous UCB studies	1980s	80	86	104	110	146	77	110	131				

Location	Dir	Source	Date	6:00	6:30	7:00	7:30	8:00	8:30	9:00	9:30	10:00	10:30	11:00	11:30
SB La Brea on ramp	EB	(SIMULATION DATA)		440	530	440	480	400	450	420	490	510	520	570	540
		Caltrans Hourly Count	TUE 12/05/89	566	566	674	674	528	528	552	552	520	520	598	598
		(converted)		538	594	678	668	542	514	552	552	516	526	582	612
		MODCOMP data	TUE 05/25/93	440	530	440	480	400	450	420	490	510	520	570	540
		MODCOMP data	WED 05/26/93	360	550	420	460	420	450	440	450	520	500	510	550
		MODCOMP data	THU 05/27/93	380	520	430	390	390	430	470	510	470	490	550	610
Mainline 10.53 (La Brea)	EB	Previous UCB studies	1980s	422	478	424	356	318	393	413	450				
		MODCOMP data	TUE 05/25/93	4520	7530	8530	6740	7320	7480	7460	7920	7550	7180	7440	7600
		MODCOMP data	WED 05/26/93	4570	7610	8050	6670	6760	7130	7680	7700	7270	7270	7360	7590
		MODCOMP data	THU 05/27/93	4570	7560	8240	6820	6930	7600	7750	7670	7320	7600	7530	7750
		MODCOMP data	TUE 12/08/92	4584	7542	8686	7718	7130	7666	7446	7938	7190	7494	7338	7732
NB La Brea off ramp	EB	(SIMULATION DATA)		180	250	350	260	210	390	470	590	870	560	600	550
		Caltrans Hourly Count	TUE 12/05/89	215	215	242	242	323	323	430	430	417	417	474	474
		(converted)		208	222	228	254	298	346	418	442	412	422	468	480
		MODCOMP data	TUE 05/25/93	180	250	350	260	210	390	470	590	870	560	600	550
		MODCOMP data	WED 05/26/93	180	270	350	240	130	270	460	260	280	490	500	480
		MODCOMP data	THU 05/27/93	150	300	330	280	240	300	190	470	280	500	540	460
NB La Brea on ramp	EB	(SIMULATION DATA)		590	940	1040	840	730	790	720	630	680	450	660	570
		Caltrans Hourly Count	TUE 12/05/89	830	830	852	852	738	738	660	660	605	605	601	601
		(converted)		824	836	864	840	762	714	678	644	612	598	5 %	608
		MODCOMP data	TUE 05/25/93	590	940	1040	840	730	790	720	630	680	450	660	570
		MODCOMP data	WED 05/26/93	550	1030	1050	800	560	620	760	380	410	630	680	530
		MODCOMP data	THU 05/27/93	590	960	1020	790	750	570	210	620	360	660	620	650
Mainline 10.710 (La Brea)	EB	Previous UCB studies	1980s	799	827	844	694	626	664	594	768				
		Caltrans Hourly Count	WED 12/13/89	7850	7850	8686	8686	10262	10262	9880	9880	8988	8988	9208	9208
		(converted)		7640	8060	8388	8984	10110	10414	10040	9720	9070	8906	9204	9214
		Caltrans Hourly Count	WED 09/13/92	7751	7751	10819	10819	9840	9840	9941	9941	9815	9815	9224	9214
		(converted)		6984	8518	10546	11092	9948	9731	9944	9938	9904	9726	9292	9158
		Previous UCB studies	1980s	799	827	844	694	626	664	594	768				
Crenshaw off ramp	EB	(SIMULATION DATA)		258	280	299	325	354	396	471	504	502	4%	469	473
		Caltrans Hourly Count	TUE 12/05/89	265	265	306	306	368	368	480	480	491	491	465	465
		(converted)		254	276	294	320	348	390	464	496	494	488	462	466
		Previous UCB studies	1980s	149	165	264	262	251	236	312	353				
Mainline 11.53 (Crenshaw)	EB	MODCOMP data	TUE 05/25/93	5260	8940	9910	8610	9420	9240	9510	9190	8400	8110	8400	8450
		MODCOMP data	WED 05/26/93	5390	9080	9560	8350	8750	9130	9460	8720	8290	8400	8390	8590
		MODCOMP data	THU 05/27/93	5320	9000	10060	8450	8820	9420	9260	8830	8400	8580	8470	8750
		MODCOMP data	TUE 12/08/92	5220	8563	9848	9180	8513	9164	8828	8621	7993	8043	8035	8431
		Previous UCB studies	1980s	854	939	1004	1028	943	891	797	768				
Crenshaw on ramp	EB	(SIMULATION DATA)		770	1030	1100	1060	890	830	860	840	840	800	790	880
		Caltrans Hourly Count	TUE 12/05/89	958	958	1069	1069	921	921	877	877	848	848	842	842
		(converted)		930	986	1074	1064	946	898	886	868	852	844	836	846
		MODCOMP data	TUE 05/25/93	770	1030	1100	1060	890	830	860	840	840	800	790	880
		MODCOMP data	WED 05/26/93	790	990	1090	1050	910	900	850	820	770	910	880	900
		MODCOMP data	THU 05/27/93	800	990	1070	1080	940	860	900	760	870	810	890	930
Arlington off ramp	EB	Previous UCB studies	1980s	854	939	1004	1028	943	891	797	768				
		ATSAC data	TUE 10/19/93	1290	1512	1564	1460	1244	1158	998	976	1054	986	1086	1100
		(SIMULATION DATA)		89	183	299	354	335	345	376	358	276	260	301	305
		Caltrans Hourly Count	TUE 12/05/89	134	134	321	321	335	335	361	361	264	264	298	298
		(converted)		88	180	294	348	330	340	370	352	272	256	2 %	300
		Previous UCB studies	1980s	49	99	188	228	283	298	242	206				
ATSAC data	TUE 10/19/93	140	194	258	274	248	252	192	264	226	262	274	286		

Location	Dir	Source	Date	6:00	6:30	7:00	7:30	8:00	8:30	9:00	9:30	10:00	10:30	11:00	11:30
Arlington on ramp	EB	(SIMULATION DATA)		681	829	1036	1056	894	769	599	492	412	382	422	426
		Caltrans Hourly Count	TUE 12/10/89	758	758	1051	1051	835	835	548	548	398	398	426	426
		(converted)		684	832	1040	1060	898	772	602	494	414	384	424	428
		Previous UCB studies	1980s	712	746	794	1060	762	681	593	433				
Western off ramp	EB	ATSAC data	TUE 10/19/93	742	8 %	850	878	702	628	534	532	476	458	514	482
		(SIMULATION DATA)		203	191	173	173	175	226	370	423	400	404	425	414
		Caltrans Hourly Count	THU 12/14/89	194	194	170	170	198	198	390	390	3%	3%	412	412
		(converted)		7.00	188	170	170	172	222	364	416	394	398	418	408
Western on ramp	EB	Previous UCB studies	1980s	157	172	180	177	162	143	189	239				
		ATSAC data	TUE 10/19/93	128	144	148	162	164	166	224	230	266	288	304	2 %
		(SIMULATION DATA)		520	740	780	710	560	480	490	560	500	550	600	660
		Caltrans Hourly Count	THU 12/14/89	632	632	606	606	574	574	490	490	571	571	609	609
Normandie off ramp	EB	(converted)		638	626	614	600	588	558	490	490	556	586	598	620
		MODCOMP data	TUE 05/25/93	520	740	780	710	560	480	490	560	500	530	600	660
		MODCOMP data	WED 05/26/93	500	700	730	730	610	530	540	500	530	590	600	600
		MODCOMP data	THU 05/27/93	470	790	690	690	560	570	520	480	500	600	590	600
		Previous UCB studies	1980s	457	605	679	698	543	522	499	433				
		ATSAC data	TUE 10/19/93	570	686	612	578	576	400	4 %	402	462	514	512	5 %
		(SIMULATION DATA)		148	238	315	431	679	658	339	262	313	311	E-3	293
Normandie on ramp	EB	Caltrans Hourly Count	THU 12/14/89	190	190	367	367	658	658	2 %	2%	307	307	290	290
		(converted)		146	234	310	424	668	648	334	258	308	306	294	288
		Previous UCB studies	1980s	120	189	322	342	344	373	309	275				
Vermont off ramp	EB	(SIMULATION DATA)		508	512	524	502	445	427	439	427	380	384	443	459
		Caltrans Hourly Count	THU 12/14/89	502	502	505	505	428	428	427	427	376	376	444	444
		(converted)		500	504	516	494	438	420	432	420	374	378	436	452
Mainline 13.95 (Vermont)	EB	Previous UCB studies	1980s	377	409	411	397	377	289	283	230				
		(SIMULATION DATA)		180	330	380	410	430	530	680	500	470	490	500	480
		Caltrans Hourly Count	THU 01/18/90	280	280	431	431	522	522	548	548	567	567	594	594
		(converted)		242	318	400	462	508	536	542	554	562	572	592	3 %
		MODCOMP' data	TUE 05/25/93	180	330	380	410	430	530	680	500	470	490	500	480
		MODCOMP data	WED 05/26/93	170	300	390	380	370	460	540	610	440	530	500	510
Vermont on ramp	EB	MODCOMP data	THU 05/27/93	210	300	430	420	480	470	580	490	630	440	490	470
		Previous UCB studies	1980s	256	349	524	543	605	624	595	5 %				
		MODCOMP data	TUE 05/25/93	5560	9020	8780	8510	8460	8060	8280	8210	7790	7390	7730	8040
		MODCOMP' data	WED 05/26/93	5620	9090	8510	8200	7970	8090	8080	7690	7420	7820	7880	7970
Hoover off ramp	EB	MODCOMP data	THU 05/27/93	5650	9170	8780	8280	8440	8150	8000	8050	7510	7810	7830	8210
		MODCOMP data	TUE 12/08/93	4979	7941	8692	7356	7212	7507	7194	7246	7177	7060	7052	7670
		(SIMULATION DATA)		460	680	900	720	640	500	450	540	570	560	580	550
		Caltrans Hourly Count	THU 01/18/90	563	563	638	638	547	547	484	484	504	504	605	605
		(converted)		544	582	640	636	566	528	488	478	490	520	592	618
		MODCOMP data	TUE 05/25/93	460	680	900	720	640	500	450	540	570	560	580	550
Hoover on ramp	EB	MODCOMP data	WED 05/26/93	450	690	680	710	530	460	510	530	500	530	600	680
		MODCOMP' data	THU 05/27/93	420	680	680	720	580	470	500	590	540	550	510	630
		Previous UCB studies	1980s	276	388	367	621	344	286	276	313				
		(SIMULATION DATA)		7.85	478	699	842	914	958	974	910	729	653	633	619
Hoover off ramp	EB	Caltrans Hourly Count	THU 01/18/90	383	383	773	773	940	940	946	946	693	693	628	628
		(converted)		286	480	702	846	918	962	978	914	732	656	636	622
		Previous UCB studies	1980s	357	431	585	633	598	654	743	572				
Hoover on ramp	EB	(SIMULATION DATA)		414	431	449	451	441	419	366	351	364	388	449	479
		Caltrans Hourly Count	THU 01/18/90	416	416	443	443	423	423	353	353	369	369	456	456
		(converted)		408	424	442	444	434	412	360	346	358	382	442	472
Previous UCB studies	1980s	354	362	368	358	359	280	263	256						

Location	Dir	Source	Date	6:00	6:30	7:00	7:30	8:00	8:30	9:00	9:30	10:00	10:30	11:00	11:30
SB110 off ramp	EB	(SIMULATION DATA)		1295	1346	1446	1390	1141	1077	1125	1185	1314	1398	1470	1494
		Caltrans Hourly Count	TUE 01/23/90	1326	1326	1425	1425	1114	1114	1159	1159	1361	1361	1488	1488
		(converted)		1300	1352	1452	1396	1146	1082	1130	1190	1320	1404	476	1500
		Previous UCB studies	1980s	1300	1396	1445	1325	1153	1084	1020	1034				
NB110 off ramp	EB	(SIMULATION DATA)		2888	2968	3089	3081	2870	2920	3294	3286	2924	2796	2760	2782
		Caltrans Hourly Count	WED 05/16/90	2939	2939	3098	3098	2907	2907	3304	3304	2872	2872	2783	2783
		(converted)		2900	2980	3102	3094	2882	2932	3308	3300	1936	2808	2772	2794
		Previous UCB studies	1980s	2349	3594	3323	2891	2638	2633	2625	2620				
East of SR110	EB	(SIMULATION DATA)	10/04/93	1476	7081	7862	7730	7835	7344	5929	5818	5831	5439	5027	5498
		calculated		1482	4632	3966	3618	3948	3496	3224	3152	3160	2904	3288	35%
		calculated from previous studies		3901	5694	6264	6279	5587	5505	4490	4345				
East of SR110	WB	(SIMULATION DATA)	10/04/93	2810	4639	5067	5610	6395	6288	6406	6329	5850	5640	5754	6116
		calculated		2822	3818	3382	2214	2802	3424	3942	38%	3192	3664	3732	3650
		Previous UCB studies	1980s	4144	6026	6301	6159	6207	6191	6290	6564				
SB110 on ramp	WB	(SIMULATION DATA)		2832	2725	2605	2513	2404	2416	1567	2651	2735	2762	1725	2713
		Smart Corridor data	11/21/90	2790	2790	2570	2570	2420	2420	2620	2620	2760	2760	2730	2730
		(converted)		2844	2736	2616	2524	2414	2426	2578	2662	2746	2774	2736	2724
		Previous UCB studies	1980s	2625	2937	4488	2694	2532	2676	2888	2998				
NB110 on ramp	WB	(SIMULATION DATA)		1255	1083	868	783	787	831	948	1050	1195	1227	1133	1111
		Caltrans Hourly Count	WED 01/24/90	1174	1174	830	830	813	813	1004	1004	1216	1216	1127	1127
		(converted)		1260	1088	872	786	790	834	952	1054	1200	1232	1138	1116
		Smart Corridor data	11/21/90	830	830	590	590	710	710	970	970	1050	1050	920	920
		(converted)		890	770	604	578	664	756	926	1014	1058	1044	912	926
Previous UCB studies	1980s	1047	1054	959	803	769	802	996	1090						
20th Street off ramp	WB	(SIMULATION DATA)		152	262	3%	447	392	402	455	469	449	441	429	431
		Caltrans Hourly Count	THU 01/18/90	204	204	415	415	390	390	455	455	437	437	423	423
		(converted)		150	258	390	440	386	3%	448	462	442	434	422	424
		Previous UCB studies	1980s	216	273	484	503	504	524	473	413				
20th Street on ramp	WB	(SIMULATION DATA)		414	490	597	614	524	492	467	484	563	597	605	616
		Caltrans Hourly Count	THU 01/18/90	445	445	596	596	500	500	468	468	570	570	601	601
		(converted)		408	482	588	604	516	484	460	476	554	588	5%	606
		Previous UCB studies	1980s	340	462	675	603	541	446	521	519				
Vermont off ramp	WB	(SIMULATION DATA)		264	299	329	372	441	482	502	510	504	494	465	453
		Caltrans Hourly Count	THU 01/18/90	277	277	344	344	453	453	498	498	491	491	452	452
		(converted)		260	294	324	366	434	474	494	502	496	486	458	446
		MODCOMP data	TUE 05/25/93	420	530	590	640	650	680	660	710	640	700	610	660
		MODCOMP data	WED 05/26/93	440	490	460	540	520	600	650	620	710	620	670	630
		MODCOMP data	THU 05/27/93	410	500	460	620	520	630	640	660	690	660	670	620
Vermont 0" ramp	WB	(SIMULATION DATA)		433	451	482	475	429	414	423	429	439	465	514	540
		Caltrans Hourly Count	THU 01/18/90	435	435	471	471	416	416	419	419	445	445	518	518
		(converted)		426	444	474	468	422	408	416	422	432	458	506	532
		MODCOMP data	TUE 05/25/93	580	790	850	670	600	620	600	730	740	700	790	780
		MODCOMP data	WED 05/26/93	530	850	890	720	620	630	650	700	740	720	770	790
		MODCOMP data	THU 05/27/93	600	780	800	640	630	580	620	660	750	770	800	830
		Previous UCB studies	1980s	381	418	487	528	441	410	441	413				
Normandie off ramp	WB	(SIMULATION DATA)		234	295	370	400	368	392	465	504	508	542	614	640
		Caltrans Hourly Count	WED 12/13/89	260	260	379	379	374	374	477	477	517	517	617	617
		(converted)		230	290	364	394	362	386	458	4%	500	534	604	630
		Previous UCB studies	1980s	242	337	513	547	517	573	518	498				

Location	Dir	Source	Date	6:00	6:30	7:00	7:30	8:00	8:30	9:00	9:30	10:00	10:30	11:00	11:30
Normandie on ramp	WB	(SIMULATION DATA)		380	550	560	430	390	380	330	380	330	410	350	440
		Caltrans Hourly Count	THU 01/18/90	502	502	505	505	428	428	427	427	376	376	444	444
		(converted)		500	504	516	494	438	420	432	420	374	378	436	452
		MODCOMP data	TUE 05/25/93	380	550	560	430	390	380	560	280	230	200	160	150
		MODCOMP data	WED 05126193	380	530	500	400	420	360	330	380	330	410	350	440
		MODCOMP data	THU 05/27/93	350	530	520	430	370	370	330	390	350	360	390	450
	Previous UCB studies	1980s	336	413	447	426	384	339	292	284					
Mainline 13.21 (Normandie)	WB	MODCOMP data	TUE 05/25/93	7620	8230	7370	5890	6200	6740	7380	7 4 7 0	7060	7640	7660	7580
		MODCOMP data	WED 05/26/93	7600	7530	6810	5200	5350	6560	7190	7680	7590	7850	7840	7860
		MODCOMP data	THU 05/27/93	7840	7880	7310	5170	5270	5860	6670	7480	7590	7880	8040	8000
		MODCOMP data	TUE 12/08/92	4494	7802	6630	4422	3576	4702	4756	4714	6204	7374	7556	7420
Western off ramp	WB	(SIMULATION DATA)		637	177	944	1034	1042	1095	1209	1205	1070	1056	1143	1167
		Caltrans Hourly Count	TUE 12/12/89	710	710	993	993	1073	1073	1212	1212	1066	1066	1160	1160
		(converted)		640	780	948	1038	1046	1100	1214	1210	1 0 7 4	1060	1148	1172
		Previous UCB studies	1980s	317	560	1091	1096	1005	1000	932	712				
		ATSAC data	TUE 10/19/93	622	808	852	1006	924	934	866	958	936	924	970	928
Western on ramp	WB	(SIMULATION DATA)		290	350	360	300	220	270	240	310	340	390	320	370
		Caltrans Hourly Count	TUE 12/12/89	591	591	604	604	494	494	453	453	533	533	543	543
		(converted)		588	594	618	590	514	476	448	458	522	544	542	544
		MODCOMP data	TUE 05125193	290	350	360	300	220	270	240	310	340	390	320	370
		MODCOMP data	WED 05/26/93	270	350	300	260	220	200	270	330	300	330	400	360
		MODCOMP data	THU 05/27/93	260	390	320	280	240	260	280	300	300	350	340	340
		Previous UCB studies	1980s	350	312	292	320	243	267	241	246				
	ATSAC data	TUE 10/19/93	364	482	444	376	388	216	284	218	250	312	284	392	
Arlington off ramp	WB	(SIMULATION DATA)		241	442	669	821	854	1002	1398	1305	675	516	611	617
		Caltrans Hourly Count	TUE 12/12/89	343	343	749	749	933	933	1357	1357	598	598	617	617
		(converted)		242	444	672	824	858	1006	1404	1310	678	518	614	620
		Previous UCB studies	1980s	306	482	799	868	901	620	606	584				
	ATSAC data	TUE 10/19/93	446	654	862	870	820	686	534	560	572	562	590	594	
Arlington on ramp	WB	(SIMULATION DATA)		280	510	530	610	440	400	380	340	270	320	340	370
		Caltrans Hourly Count	TUE 12/12/89	400	400	553	553	424	424	393	393	310	310	321	321
		(converted)		362	438	550	556	444	404	408	378	318	302	320	324
		MODCOMP data	TUE 05/25/93	280	510	530	610	440	400	380	340	270	320	340	370
		MODCOMP data	WED 05/26/93	310	500	580	600	450	330	360	380	360	340	340	340
		MODCOMP data	THU 05/27/93	330	520	550	640	530	560	360	330	320	290	330	370
		Previous UCB studies	1980s	339	497	605	628	501	502	450	359				
	ATSAC data	TUE 10/19/93	254	246	292	220	212	186	176	168	152	174	146	154	
Mainline 11.53 (Crenshaw)	WB	MODCOMP data	TUE 05125193	9120	10850	9860	7660	7340	8290	8810	9000	8 1 7 0	9510	8890	9050
		MODCOMP data	WED 05/26/93	9160	10210	9190	7270	7140	7760	8740	9320	8930	9370	9230	9240
		MODCOMP data	THU 05/27/93	9380	10160	9740	6760	6720	7330	8340	9160	9170	9510	9060	9410
		MODCOMP data	TUE 12/08/92	4365	7856	6518	4770	3928	5042	5461	5354	7226	7595	7506	7181
Crenshaw off ramp	WB	(SIMULATION DATA)		572	749	920	1107	1340	1476	1617	1480	978	817	854	852
		Caltrans Hourly Count	TUE 12/12/89	663	663	1019	1019	1415	1415	1555	1555	901	901	857	857
		(converted)		574	752	924	1112	1346	1482	1624	1486	982	820	858	856
		Previous UCB studies	1980s	662	776	931	818	911	935	996	1042				
	ATSAC data	TUE 10/19/93	708	886	984	348	380	1416	1216	1086	1012	1070	1214	1164	
Crenshaw on ramp	WB	(SIMULATION DATA)		443	626	857	939	861	798	689	614	542	512	528	520
		Caltrans Hourly Count	TUE 12/12/89	526	526	884	884	818	818	642	642	518	518	516	516
		(converted)		436	616	844	924	848	786	678	604	534	504	520	512
		Previous UCB studies	1980s	416	743	827	839	701	630	551	261				
	ATSAC data	TUE 10/19/93	616	710	778	692	600	512	504	514	492	592	544	620	

Location	Dir	Source	Date	6:00	6:30	7:00	7:30	8:00	8:30	9:00	9:30	10:00	10:30	11:00	11:30
Mainline 10.710 (La Brea)	WB	Caltrans Hourly Count	WED 12/13/89	8120	8120	7295	7295	8405	8405	9465	9465	9165	9165	9983	9983
		(converted)		8326	7914	7260	7330	8134	8676	9368	9562	9102	9 2 2 8	9928	10040
		Caltrans Hourly Count	TUE 06/23/92	10498	10498	8245	8245	7613	7613	Pool	Pool	9613	9613	10137	10137
NB La Brea off ramp	WB	(SIMULATION DATA)		420	460	310	310	370	370	510	580	510	600	600	550
		Caltrans Hourly Count	WED 12/06/89	375	375	237	237	313	313	408	408	576	576	535	535
		(converted)		410	340	244	230	292	334	376	440	558	592	532	536
		MODCOMP data	TUE 05/25/93	420	460	310	310	370	370	510	580	510	600	600	550
		MODCOMP data	WED 05/26/93	420	390	310	310	290	410	560	590	520	570	620	550
		MODCOMP' data	THU 05/27/93	420	400	330	300	320	410	420	600	570	630	610	600
		Previous UCB studies	1980s	264	303	332	274	315	406	479	525				
		ATSAC data	TUE 10/19/93	406	394	354	268	306	394	478	536	608	564	630	590
NB La Brea on ramp	WB	(SIMULATION DATA)		170	380	390	390	340	330	270	210	230	210	170	130
		Caltrans Hourly Count	WED 12/06/89	418	418	802	802	642	642	357	357	248	248	209	209
		(converted)		324	516	772	834	700	584	402	310	266	230	206	212
		MODCOMP data	TUE 05/25/93	170	380	390	390	340	330	270	210	230	210	170	130
		MODCOMP data	WED 05/26/93	140	370	400	400	360	310	270	180	170	200	170	150
		MODCOMP data	THU 05/27/93	160	340	440	400	320	280	250	220	130	120	130	150
Mainline 10.53 (La Brea)	WB	MODCOMP data	TUE 05/25/93	7920	9490	8330	6640	6450	6780	7650	8020	7220	8390	7770	8100
		MODCOMP data	WED 05/26/93	7890	8670	7920	6440	5940	6740	7530	8090	8040	8290	7960	8250
		MODCOMP data	THU 05/27/93	7980	8920	7890	6130	5730	6630	7200	8150	7940	8500	8330	8380
		MODCOMP data	TUE 12/08/92	4672	8272	6810	5196	4836	5712	6344	6036	7852	8288	8164	7 8 %
SB La Brea off ramp	WB	(SIMULATION DATA)		748	589	382	315	323	417	723	794	658	610	561	563
		Caltrans Hourly Count	WED 12/06/89	658	658	343	343	364	364	747	747	623	623	553	553
		(converted)		736	580	376	310	318	410	712	782	648	600	552	554
SB La Brea on ramp	WB	(SIMULATION DATA)		280	382	502	561	571	545	451	410	392	390	423	427
		Caltrans Hourly Count	WED 12/06/89	326	326	524	524	549	549	425	425	385	385	417	417
		(converted)		276	376	494	552	562	536	444	404	386	384	416	420
Washington off ramp	WB	(SIMULATION DATA)		1249	1101	928	827	739	769	964	1038	1044	1042	1006	990
		Caltrans Hourly Count	WED 11/29/89	1180	1180	881	881	758	758	1005	1005	1048	1048	1002	1002
		(converted)		1254	1106	932	830	742	772	968	1042	1048	1046	1010	994
		Previous UCB studies	1980s	1031	1065	886	704	559	608	673	533				
Venice off ramp	WB	(SIMULATION DATA)		928	916	882	908	1002	1052	1083	1091	1068	1058	1066	1032
		Caltrans Hourly Count	WED 11/29/89	926	926	899	899	1030	1030	1092	1092	1066	1066	1052	1052
		(converted)		932	920	886	912	1006	1056	1088	1096	1072	1062	1070	1036
		Previous UCB studies	1980s	531	706	1060	1085	989	974	969	508				
Fairfax on ramp	WB	(SIMULATION DATA)		270	330	430	540	450	360	360	380	410	350	390	500
		Caltrans Hourly Count	WED 11/29/89	488	488	879	879	781	781	605	605	432	432	475	475
		(converted)		390	586	840	918	816	746	650	562	448	416	460	490
		MODCOMP data	TUE 05/25/93	270	330	430	540	450	360	360	380	410	350	390	500
		MODCOMP data	WED 05/26/93	320	340	370	520	460	370	370	370	480	390	450	500
		MODCOMP data	THU 05/27/93	300	350	430	530	460	360	390	380	430	410	750	530
Mainline 9.21 (Fairfax)	WB	MODCOMP data	TUE 05/25/93	6800	8560	8120	6750	6860	6730	7460	7240	6500	7240	7070	7210
		MODCOMP data	WED 05/26/93	6700	8080	7800	6720	6260	6710	7310	7350	7240	7430	7340	7440
		MODCOMP data	THU 05/27/93	6750	8270	7740	6090	5940	7050	7060	7560	7170	7480	11720	7590
		Previous UCB studies	1980s	7135	8680	9659	8259	7380	8305	7262	7741				
		MODCOMP data	TUE 12/08/92	3974	7446	6154	5314	5626	6284	6860	5870	6946	7406	7170	7074

Location	Dir	Source	Date	6:00	6:30	7:00	7:30	8:00	8:30	9:00	9:30	10:00	10:30	11:00	11:30
La Cienega on ramp	WB	(SIMULATION DATA)		286	557	886	1044	1020	1032	1042	985	819	786	868	906
		Caltrans Hourly Count	WED 11/29/89	415	415	950	950	1010	1010	999	999	790	790	873	873
		(converted)		282	548	872	1028	1004	1016	1026	970	806	774	854	892
Robertson off ramp	WB	(SIMULATION DATA)		733	657	548	526	564	653	880	948	858	860	920	924
		Caltrans Hourly Count	TUE 11/21/89	698	698	539	539	611	611	918	918	862	862	926	926
		(converted)		736	660	550	528	56.5	656	884	952	862	864	924	928
Robertson on ramp		ATSAC data	TUE 10/19/93	560	710	764	664	862	964	1024	1166	1134	1132	1106	1160
	WB	(SIMULATION DATA)		333	670	1065	1292	1382	1333	1048	933	880	874	955	979
		Caltrans Hourly Count	TUE 11/21/89	494	494	1160	1160	1336	1336	975	975	863	863	952	952
Mainline 6.73 (Motor)		(converted)		328	660	1048	1272	1360	1312	1032	918	866	860	940	964
	WB	MODCOMP data	TUE 05/25/93	6690	8920	9240	9000	8780	8690	8990	8310	7170	7890	7560	8350
		MODCOMP data	WED 05/26/93	6650	8750	8820	8890	8460	8680	8790	8530	8030	8260	8184	8360
National off ramp		MODCOMP data	THU 05/27/93	6680	8880	8670	8400	8380	8710	8360	8110	8120	8270	8380	8730
	WB	(SIMULATION DATA)		504	500	477	506	589	636	664	677	656	656	677	668
		Caltrans Hourly Count	TUE 11/21/89	494	494	484	484	603	603	659	659	645	645	662	662
Overland off ramp		(converted)		4 %	492	470	498	580	626	654	666	646	646	666	658
		ATSAC data	TUE 10/19/93	550	580	630	630	738	810	798	808	718	736	684	686
	WB	(SIMULATION DATA)		651	705	761	807	838	878	946	930	809	791	854	862
Overland on ramp		Caltrans Hourly Count	TUE 11/21/89	681	681	786	786	862	862	943	943	803	803	862	862
		(converted)		654	708	764	810	842	882	950	934	812	794	858	866
	WB	(SIMULATION DATA)		440	815	1263	.486	1504	1428	1141	1024	990	970	990	1012
SB I405 off ramp		Caltrans Hourly Count	TUE 11/21/89	630	630	1379	1379	1472	1472	1088	1088	984	984	1004	1004
		(converted)		442	818	1268	1492	1510	1434	1146	.028	994	974	994	1016
	WB	(SIMULATION DATA)		1924	1665	1348	1203	1119	1247	1707	1950	2089	2171	2141	2155
NBI405 off ramp		Caltrans Hourly Count	TUE 11/21/89	1802	1802	1281	1281	1188	1188	1836	1836	2139	2139	2157	2157
		(converted)		1932	1672	1354	.208	1124	1252	1714	1958	2098	2180	2150	2164
	WB	(SIMULATION DATA)		2794	3045	3404	3428	3109	2910	1677	2561	2561	2539	2549	2513
West of I405		Caltrans Hourly Count	TUE 11/21/89	2932	2932	3430	3430	3023	3023	2630	2630	2560	2560	2542	2542
		(converted)		2806	3058	3418	3442	3122	2922	2688	2572	2572	2550	2560	2524
	WB	(SIMULATION DATA)		1239	2808	3802	4434	4222	4242	2330	1908	2036	2194	2320	2654
	calculated		1244	3808	4502	4534	4622	4442	4130	3208	2036	2694	2320	3154	

Location	12:00	12:30	1:00	1:30	2:00	2:30	3:00	3:30	4:00	4:30	5:00	5:30	6:00	6:30	7:00	7:30
West of I405	4708	4603	4053	3997	4104	4319	5132	5637	6154	6020	5290	5033	4203	3617	3356	3049
	4728	4622	4070	4014	4322	4538	4852	4958	4874	4840	4910	4652	4020	3632	3370	3062
NB I-405 on ramp	1591	1575	1749	17%	1816	1731	1504	1382	1303	1269	1305	1307	1480	1109	0	0
	1590	1590	1780	1780	1780	1780	1450	1450	1290	1290	1310	1310	1300	1300	0	0
	1598	1582	1756	1804	1824	1738	1510	1388	1308	1274	1310	1312	1486	1114	0	0
SBI-405 on ramp	2163	2219	2131	2131	2243	2159	1798	1808	2147	2332	2597	2482	2155	1468	0	0
	2200	2200	2140	2140	2210	2210	1810	1810	2250	2250	2550	2550	1820	1820	0	0
	2172	1228	2140	2140	2251	2168	1806	1816	2156	2342	2608	2492	2164	1474	0	0
I-405 on ramps (after merge)	4119	4109	4206	4274	4412	4350	4077	3924	3806	3728	3728	3659	3485	3499	3679	3804
	4131	4131	4259	4259	4398	4398	4017	4017	3783	3783	3709	3709	3507	3507	3757	3757
	4136	4126	4224	4192	4430	4368	4094	3940	3822	3744	3744	3674	3500	3514	3694	3820
	3790	3790	3920	3920	3990	3990	3260	3260	3540	3540	3860	3860	3120	3120	0	0
	3770	3808	38%	3944	4074	3906	3316	3206	3466	3616	3914	3804	3654	2588	0	0
National off ramp	1615	1506	1032	900	988	1008	1044	1060	1042	1083	1221	1239	1171	1091	974	866
	1567	1567	970	970	1001	1001	1056	1056	1067	1067	1236	1236	1136	1136	924	924
	1622	1512	1036	904	992	1012	1048	1064	1046	1088	1226	1244	1176	1096	978	870
Overland on ramp	1518	1566	1532	1638	1628	1572	1604	1478	1502	1464	1488	15%	1512	1560	1430	1236
	912	966	992	1000	956	978	1070	1087	1034	1044	1113	1115	1087	1010	874	757
	944	944	1000	1000	971	971	1082	1082	1043	1043	1120	1120	1053	1053	819	819
	916	970	996	1004	960	982	1074	1092	1038	1048	1118	1120	1092	1014	878	760
	860	1000	970	900	640	700	720	650	690	650	720	690	690	760	870	500
	920	890	1030	990	700	740	750	670	640	780	700	760	740	780	990	520
	920	970	1000	1030	810	880	810	680	840	750	910	840	940	1000	1020	760
Mainline 6.745 (Motor)	8181	8181	8331	8331	8798	8798	8950	8950	8616	8616	8374	8374	7250	7250	6868	6868
	8158	8206	8254	8406	8720	8876	8972	8926	8688	8544	8546	8202	7436	7064	6964	6772
	8697	8697	7928	7928	8619	8619	8871	8871	8653	8653	8841	8841	7716	7716	6835	6835
	8692	8702	7938	7920	8500	8736	8868	8876	8656	8648	8960	8711	7964	7466	7054	6614
	7840	7900	8020	8430	8690	8970	9180	9210	8940	8810	9310	8950	7940	8310	7220	4200
	7090	7950	8110	8310	8760	8880	9320	8950	8800	8640	8340	7400	7810	7450	7810	4310
	7010	7460	7980	7930	8390	8450	8980	8680	7680	7440	7310	7370	8240	7830	7590	6630
Manning on ramp	380	440	380	420	520	540	530	460	440	460	550	580	540	520	420	250
	402	402	434	434	547	547	589	589	455	455	586	586	500	500	384	384
	398	408	416	450	526	566	602	578	456	454	580	592	526	474	414	354
	380	440	380	420	520	540	530	460	440	460	550	580	540	520	420	250
	420	410	460	380	500	490	540	400	440	470	510	500	530	460	470	260
	470	400	490	460	550	540	480	440	460	430	460	490	570	500	450	380
Robertson off ramp	878	876	902	910	882	929	1087	1097	959	931	991	961	872	798	719	640
	863	863	891	891	892	892	1075	1075	930	930	961	961	822	822	669	669
	864	862	888	896	868	914	1070	1080	944	916	976	946	858	786	708	630
Mainline 7.99 (National)	7270	7370	7460	7840	8290	8540	8490	8660	8270	8180	8780	8540	8360	7970	6860	3930
	7490	7490	7580	6230	8180	8580	8370	8150	8050	8430	8160	8360	8570	7810	7590	4050
	7500	7390	7450	5920	8040	8120	7850	8140	7630	7870	7710	8070	8300	7400	7330	6360

Location	12:00	12:30	1:00	1:30	2:00	2:30	3:00	3:30	4:00	4:30	5:00	5:30	6:00	6:30	7:00	7:30
National on ramp	660	670	640	720	720	740	690	610	620	660	650	610	680	640	550	320
	753	753	699	699	850	850	687	687	612	612	705	705	592	592	501	501
	754	752	686	710	852	848	716	658	610	614	708	702	618	566	524	478
	660	670	640	720	720	740	690	610	620	660	650	610	680	640	550	320
	680	700	630	680	710	640	720	580	590	640	670	540	670	670	610	320
	760	720	730	680	750	720	700	630	590	620	680	590	600	620	590	440
SB La Cienega off ramp	66	68	62	78	66	90	66	92	66	92	84	92	88	76	54	54
	242	232	199	205	248	262	258	256	262	238	173	154	177	163	122	91
	232	232	199	199	251	251	253	253	245	245	161	161	166	166	105	105
	238	228	196	202	244	258	254	252	258	234	170	152	174	160	120	90
SB La Cienega on ramp	610	580	580	650	640	610	630	500	580	550	660	570	680	630	520	300
	705	705	635	635	677	677	621	627	527	527	616	616	576	576	539	539
	704	708	638	632	678	676	646	608	528	526	610	622	586	566	548	530
	610	580	580	650	640	610	630	500	580	550	660	570	680	630	520	300
	590	590	610	600	600	660	600	490	520	510	600	580	630	540	540	360
	580	580	540	570	550	490	530	500	600	530	560	480	530	520	580	430
NB La Cienega off ramp	687	709	723	717	683	670	683	670	624	618	654	652	628	597	551	510
	687	687	710	710	666	666	666	666	610	610	642	642	603	603	522	522
	676	698	712	706	672	660	672	660	614	608	644	642	618	588	542	502
Fairfax off ramp	510	550	550	560	589	591	709	642	650	630	820	770	770	650	560	320
	481	481	477	477	565	565	617	617	705	705	763	763	585	585	439	439
	486	476	468	486	548	582	600	634	686	724	780	748	626	546	474	402
	510	550	550	560	589	591	709	642	650	630	820	770	770	650	560	320
	570	530	510	580	580	600	690	650	660	630	730	680	760	610	550	330
	490	510	450	510	560	570	660	550	600	670	620	710	720	580	540	450
Mainline 9.21 (Fairfax)	780	866	808	884	922	942	980	936	1018	1062	1074	1022	1006	870	832	730
	7290	7410	7450	7900	8238	8438	8214	8129	8180	8280	8620	8580	8520	8220	6790	3940
	7350	7580	7550	7650	8210	8460	8140	8130	8120	8240	8220	8530	8340	7990	7520	4120
7640	7630	7870	7610	82W	8020	7770	8060	7950	7880	8090	8120	8520	7380	7280	6300	
Venice on ramp	425	417	394	402	449	461	441	449	467	504	591	579	419	410	343	276
	414	414	392	392	448	448	438	438	478	478	576	576	439	439	305	305
	418	410	388	396	442	454	434	442	460	496	582	570	472	404	338	272
	220	190	250	240	248	331	306	284	300	350	370	370	290	260	160	90
	200	210	240	220	250	330	310	280	300	340	370	380	290	220	220	100
	220	250	220	280	310	340	280	290	300	360	370	330	300	270	160	180
Washington on ramp	1113	1125	1099	1097	1117	1113	1081	1079	1101	1113	1129	1117	1074	1034	988	946
	1124	1124	1103	1103	1120	1120	1085	1085	1112	1112	1129	1129	1058	1058	971	971
	1118	1130	1104	1102	1122	1118	1086	1084	1106	1118	1134	1122	1078	1038	992	950
SB La Brea off ramp	209	217	226	234	246	254	250	274	327	360	404	394	325	276	223	173
	211	211	225	225	245	245	258	258	338	338	392	392	297	297	195	195
	206	214	222	230	242	250	246	270	322	354	398	388	320	272	220	170
	294	274	314	336	374	376	406	372	422	400	374	402	338	312	286	216

Location	12:00	12:30	1:00	1:30	2:00	2:30	3:00	3:30	4:00	4:30	5:00	5:30	6:00	6:30	7:00	7:30
SB La Brea on ramp	560	620	580	560	669	694	539	557	480	450	510	580	510	450	470	230
	633	633	669	669	640	640	571	571	488	488	523	523	517	517	426	426
	624	642	668	670	652	628	590	552	494	482	520	526	530	504	448	404
	560	620	580	560	669	694	539	557	480	450	510	580	510	450	470	230
	530	620	630	630	670	700	540	550	440	440	580	550	500	410	470	260
	530	570	600	570	570	660	550	550	480	480	540	510	530	510	450	440
Mainline 10.53 (La Brea)	7390	7500	7620	7800	8207	7832	7861	7729	7610	7710	8150	7850	7890	8200	6930	3900
	7310	7700	7570	7790	8170	7850	7810	7680	7710	7730	7870	8000	7740	7610	7580	4210
	7740	7700	7900	7750	8150	7500	7490	7610	7630	7270	7720	7680	7970	7560	6970	6250
NB La Brea off ramp	530	510	430	570	513	558	542	527	490	460	500	560	610	630	490	370
	468	468	416	416	421	421	442	442	427	427	414	414	501	501	408	408
	476	462	422	410	418	424	442	442	430	422	406	422	502	500	432	384
	530	510	430	570	513	558	542	527	490	460	500	560	610	630	490	370
	500	530	480	510	520	550	540	530	400	460	490	460	560	610	580	280
	530	520	510	480	530	520	490	480	360	500	420	420	600	570	430	490
NB La Brea on ramp	560	590	660	630	709	725	817	825	760	770	750	710	620	660	610	350
	657	657	634	634	724	724	675	675	689	689	580	580	572	572	521	521
	652	662	626	642	720	730	678	670	702	678	594	566	580	566	534	508
	560	590	660	630	709	725	817	825	760	770	750	710	620	660	610	350
	710	690	610	620	710	720	820	830	820	820	750	640	680	660	700	380
	600	660	690	700	810	810	860	840	740	780	680	680	660	660	840	550
Mainline 10.710 (La Brea)	9027	9027		9382	9810	9810	9658	9658	9744	9744	9569	9569	8771	8771	8037	8037
	9006	9048	9284	9480	9774	9844	9666	9650	9754	9734	9690	9446	8962	8580	8220	7852
	9270	9270	9411	9411	9927	9927	10259	10259	10068	10068	10063	10063	9793	9793	9176	9176
	9246	9294	9330	9492	9820	10034	10242	10276	10092	10044	10098	10030	9906	9682	9330	9022
Crenshaw off ramp	510	530	563	555	488	482	514	557	630	699	807	807	697	630	569	504
	511	511	550	550	477	477	526	526	655	655	794	794	654	654	528	528
	502	522	554	546	480	474	506	548	620	688	794	794	686	620	560	496
Mainline 11.53 (Crenshaw)	8420	8540	8340	8950	9294	9388	9322	9249	9260	9120	9550	9440	9280	9390	7820	4410
	8580	8730	8690	9000	9340	9370	9280	9390	9360	9260	9450	9510	9320	8740	8830	4750
	8570	8780	9000	8880	9070	9290	9210	9360	9210	8930	9240	9230	9450	8720	8230	7000
Crenshaw on ramp	870	810	950	980	954	848	860	1001	910	860	880	860	800	850	770	430
	887	887	965	965	961	961	856	856	720	720	677	677	737	737	731	731
	872	902	954	974	976	946	886	826	742	698	674	680	730	744	732	730
	870	810	950	980	954	848	860	1001	910	860	880	860	800	850	770	430
	870	860		890	950	860	870	970	910	830	910	860	780	810	790	490
	930	1000	870	1010	930	930	890	870	820	850	900	860	840	800	800	700
	1098	1204	1176	1200	1158	1092	1118	1088	1186	1108	1082	1082	984	926	880	694
Arlington off ramp	268	276	335	354	333	356	412	449	469	492	524	516	475	423	349	280
	268	268	340	340	339	339	424	424	473	473	511	511	442	442	310	310
	264	272	330	348	328	350	406	442	462	484	516	508	468	416	344	276
	272	308	288	318	344	412	414	468	494	450	570	472	504	422	368	266

Location	12:00	12:30	1:00	1:30	2:00	2:30	3:00	3:30	4:00	4:30	5:00	5:30	6:00	6:30	7:00	7:30
Arlington on ramp	410	414	430	458	532	524	428	402	414	414	418	416	412	398	378	360
	415	415	446	446	530	530	417	417	415	415	420	420	407	407	371	371
	412	416	432	460	534	526	430	404	416	416	420	418	414	400	380	362
Western off ramp	586	566	5 %	542	584	486	520	446	458	486	506	548	570	498	492	416
	368	368	417	425	410	392	347	351	414	437	439	431	402	390	390	388
	363	363	415	415	395	395	344	344	419	419	428	428	390	390	383	383
	362	362	410	418	404	386	342	346	408	430	432	424	3 %	384	384	382
Western on ramp	352	300	344	324	380	374	406	386	418	374	424	404	402	394	334	262
	580	610	640	640	630	660	610	650	630	540	550	540	530	510	580	370
	660	660	669	669	718	718	557	557	489	489	498	498	557	557	5 %	5 %
	654	668	662	678	734	704	584	528	496	482	490	506	544	570	586	606
	580	610	640	640	630	660	610	650	630	540	550	540	530	510	580	370
	590	540	630	630	640	640	410	0	0	0	0	0	0	0	0	0
	670	730	620	680	670	590	610	590	570	490	520	510	570	520	610	560
Normandie off ramp	618	588	702	528	654	524	574	540	492	476	510	474	584	514	500	418
	297	291	268	274	299	335	404	439	451	463	471	461	425	404	394	378
	289	289	267	267	312	312	415	415	450	450	458	458	408	408	380	380
	292	286	264	270	294	330	398	432	444	456	464	454	418	398	388	372
Normandie on ramp	447	463	508	526	534	520	475	451	431	429	441	449	457	465	467	473
	447	447	509	509	518	518	455	455	423	423	439	439	454	454	463	463
	440	456	500	518	526	512	468	444	424	422	434	442	450	458	460	466
Vermont off ramp	580	510	520	530	500	540	460	560	560	590	620	500	570	650	520	230
	586	586	5 %	5 %	477	477	411	411	546	546	660	660	647	647	549	549
	586	586	610	582	500	456	404	420	514	578	646	674	662	632	572	524
	580	510	520	530	500	540	460	560	560	590	620	500	570	650	520	230
	480	500	550	450	490	470	560	580	620	530	560	560	470	520	530	660
	490	540	480	500	450	450	440	470	560	580	500	540	590	470	1030	370
Mainline 13.95 (Vermont)	7320	7910	7820	8120	8790	8550	8250	8140	7540	8060	7790	7880	7500	6990	6970	4150
	7780	7800	8050	8210	8650	8060	4850	0	0	0	0	0	0	0	0	0
	8050	8250	8300	8120	8260	8360	8270	8010	7850	7570	7460	7030	7940	7620	7470	6510
Vermont on ramp	630	640	690	740	740	700	640	580	520	560	610	500	670	700	730	420
	607	607	686	686	699	699	607	607	475	475	521	521	533	533	605	605
	596	618	674	698	710	688	636	580	484	466	514	528	522	544	588	624
	630	640	690	740	740	700	640	580	520	560	610	500	670	700	730	420
	730	630	680	690	790	670	420	0	0	0	0	0	0	0	0	0
	640	670	710	690	780	660	610	590	550	620	670	620	640	680	740	650
Hoover off ramp	639	645	647	663	691	713	717	757	827	896	1006	1024	992	868	641	446
	644	644	658	658	706	706	740	740	865	865	1019	1019	934	934	546	546
	642	648	650	666	694	716	720	760	830	900	1010	1028	996	872	644	448
Hoover on ramp	479	506	559	587	599	589	545	512	459	459	520	536	524	526	540	551
	484	484	564	564	585	585	521	521	452	452	519	519	517	517	537	537
	472	498	550	578	590	580	536	504	452	452	512	528	516	518	532	542

Location	12:00	12:30	1:00	1:30	2:00	2:30	3:00	3:30	4:00	4:30	5:00	5:30	6:00	6:30	7:00	7:30
SB 110 off ramp	1434	1460	1555	1635	1755	1781	1735	1665	1516	1468	1508	1492	1432	1412	1416	1412
	1453	1453	1601	1601	1775	1775	1707	1707	1498	1498	1506	1506	1428	1428	1420	1420
	1440	1466	1562	1642	1762	1788	1742	1672	1522	1474	1514	1498	1438	1418	1422	1418
NB 110 off ramp	2954	2954	2798	2745	2776	2699	2504	2418	2436	2340	2085	2077	2296	2420	2515	2623
	2966	2966	2782	2782	2749	2749	2471	2471	2398	2398	2091	2091	2368	2368	2580	2580
	2966		2810	2756	2788	2710	2514	2428	2446	2350	2094	2086	2306	2430	2526	2634
East of SR110	6194	6087	6117	6555	6622	6786	6624	6684	7383	7290	6032		5908	2776	2898	191
	2744	3328	3348	3634	4136	3916	3810	3784	3194	3788	3684	3796	3276	2788	2910	192
East of SR110	5641	5789	5745	5204	5004	5213	5246	6123	6593	6576	6480	5643	5160	3027	4800	2376
	1576	3752	3130	3106	2986	2812	2534	2042	2264	2492	2256	2114	990	3040	4820	2386
SB 110 on ramp	2695	2705	2745	2772	2754	2882	3254	3298	3045	2952	2928	2868	3171	2348	0	0
	2710	3790	2770	2770	2830	2830	3290	3290	3010	3010	2910	2910	2770	2770	0	0
	2706	2716	2756	2784	2766	2894	3268	3312	3058	2964	2940	2880	3184	2358	0	0
NB 110 on ramp	1127	1105	1018	1050	1227	1261	1177	1139	1115	1066	974	932	924	916	920	920
	1121	3790	1037	1037	1249	1249	1163	1163	1095	1095	957	957	923	923	925	925
	1132	1110	1022	1054	1232	1266	1182	1144	1120	1070	978	936	928	920	924	924
	1110	3790	1070	1070	910	910	1050	1050	1030	1030	970	970	790	790	0	0
	1092	1130	1096	1044	912	908	1034	1066	1040	1020		940	924	656	0	0
20th Street off ramp	463	467	451	427	386	370	384	360	307	268	228	211	211	234	286	327
	458	458	432	432	372	372	366	36.5	284	284	216	216	219	219	302	302
	456	460	444	420	380	364	378	354	302	264	224	208	208	230	282	322
20th Street on ramp	620	628	616	670	819	868	831	819	794	784	792	782	762	717	640	573
	613	613	633	633	830	830	813	813	777	777	775	775	728	728	597	597
	610	618	606	660	806	854	818	806	782	772	780	770	750	706	630	564
Vermont off ramp	461	445	400	392	417	417	404	382	354	313	236	221	262	291	339	380
	446	446	389	389	410	410	387	387	328	328	224	224	272	272	353	353
	454	438	394	386	410	410	398	376	348	308	232	218	258	286	334	374
	620	610	660	600	550	530	650	550	410	400	400	380	370	470	540	330
	690	370	480	600	630	590	360	0	0	0	0	0	0	0	0	0
	600	650	640	580	600	660	680	530	490	400	420	400	370	490	610	490
Vermont on ramp	549	567	595	607	614	620	630	622	595	579	563	567	599	610	610	612
	549	549	592	592	607	607	616	616	578	578	555	555	595	595	601	601
	540	558	586	598	604	610	620	612	586	570	554	558	5%	600	600	602
	800	800	910	810	870	920	840	860	860	780	930	800	840	860	820	510
	780	810	830	780	880	930	880	790	880	850	960	910	800	810	710	570
	890	770	810	860	960	870	950	840	970	860	1030	770	810	830	810	740
Normandie off ramp	644	622	561	522	488	482	512	508	471	465	473	492	545	545	4%	463
	624	624	533	533	477	477	502	502	461	461	475	475	535	535	473	473
	634	612	552	514	480	474	504	500	464	458	466	484	536	536	488	456

Location	12:00	12:30	1:00	1:30	2:00	2:30	3:00	3:30	4:00	4:30	5:00	5:30	6:00	6:30	7:00	7:30
Normandie on ramp	440	440	410	480	700	550	460	390	410	420	290	360	330	360	270	190
	447	447	509	509	518	518	455	455	423	423	439	439	454	454	463	463
	440	456	500	518	526	512	468	444	424	422	434	442	450	458	460	466
	140	440	410	480	700	550	460	390	410	420	290	360	330	360	270	190
	440	460	470	440	650	550	460	360	400	400	320	340	300	350	340	230
	400	380	440	440	680	530	470	340	420	370	360	370	570	430	320	280
Mainline 13.21 (Normandie)	5460	7700	7210	7400	7650	7700	7610	7130	7120	7260	7020	6790	5890	7030	6330	3790
	7360	7410	7620	7790	7760	6890	7560	7000	6440	6140	5450	5760	6070	6220	6450	4120
	7540	7690	7650	7840	7880	7890	7490	7350	7250	7560	7550	6690	6790	6820	6700	5920
Western off ramp	1175	1149	1074	1030	994	974	1004	934	769	657	524	500	607	611	548	506
	1167	1167	1057	1057	987	987	973	973	716	716	514	514	612	612	529	529
	1180	1154	1078	1034	998	978	1008	938	772	660	526	502	610	614	550	508
Western on ramp	910	912	842	874	840	880	844	782	824	770	886	874	932	818	758	728
	440	450	340	340	400	370	400	350	330	280	270	260	280	320	310	230
	545	545	588	588	641	641	567	567	480	480	441	441	387	387	392	392
	540	550	576	600	644	638	586	546	4 %	464	452	430	394	380	392	392
	440	450	340	340	400	370	400	350	330	280	270	260	280	320	310	230
	380	390	340	370	410	380	380	310	260	290	270	290	330	310	340	200
	400	420	370	380	410	360	350	340	310	260	270	240	300	290	300	280
Arlington off ramp	334	350	438	308	382	342	322	340	342	284	324	258	334	278	284	246
	621	597	526	506	504	540	635	699	781	783	691	661	695	631	490	376
	612	612	518	518	524	524	670	670	785	785	679	679	666	666	435	435
	624	600	528	508	506	542	638	702	784	786	694	664	698	634	492	378
Arlington on ramp	580	610	662	616	676	684	688	796	754	842	744	862	816	740	700	490
	300	330	360	370	330	290	80	460	460	450	530	500	510	380	350	200
	322	322	325	325	453	453	398	398	426	426	450	450	435	435	275	275
	320	324	308	340	444	462	402	3 %	420	432	448	450	458	412	316	236
	300	330	360	370	330	290	80	460	460	450	530	500	510	380	350	200
	320	330	390	370	420	230	400	350	490	480	550	590	600	480	340	170
	290	370	90	390	410	390	490	420	430	380	530	470	440	400	330	270
Mainline 11.53 (Crenshaw)	154	142	170	202	202	222	202	222	192	244	200	230	230	188	180	134
	7810	9430	8740	8860	9270	9690	9210	8490	9420	9800	9420	8680	7260	8660	7880	5820
	8810	8870	8950	9490	9060	9020	9600	8100	8400	7900	7970	8570	8590	8 2 5 0	8390	4920
	9040	8990	9050	9460	9160	9680	8980	9610	9430	9960	10100	8760	8990	8970	8400	7140
Crenshaw off ramp	906	878	763	739	759	835	1044	1085	978	936	900	888	938	890	761	659
	896	8 %	754	754	800	800	1069	1069	% 1	% 1	897	897	917	917	713	713
	910	882	766	742	762	838	1048	1090	982	940	904	892	942	894	764	662
Crenshaw on ramp	1126	1080	1092	1052	1162	1224	1116	1200	1084	1106	970	846	766	1122	1052	840
	492	506	571	601	622	601	530	486	447	425	406	414	465	469	437	414
	492	492	577	577	602	602	500	500	429	429	404	404	460	460	419	419
	484	498	562	592	612	592	522	478	440	418	400	408	458	462	430	408
	658	620	650	662	596	628	526	516	438	478	430	472	476	492	488	432

Location	12:00	12:30	1:00	1:30	2:00	2:30	3:00	3:30	4:00	4:30	5:00	5:30	6:00	6:30	7:00	7:30
Mainline 10.710 (La Brea)	9611	9611	9509	9509	10668	10668	10439	10439	10216	10216	8969	8969	9457	9457	8521	8521
	9670	9552	9380	9638	10550	10786	10496	10384	10402	10030	9060	8876	9514	9400	8756	8288
	9978	9978	10167	10167	10116	10116	10034	10034	10407	10407	10129	10129	9317	9317	8435	
	9976	9982	10150		10134	10098	9998	10070	10396	10418	10266	9992	9530	9106	8654	8214
NB La Brea off ramp	540	660	560	540	550	610	540	550	520	500	450	420	380	540	46a	360
	597	597	595	595	511	511	483	483	439	439	413	413	413	413	433	433
	588	604	606	584	526	496	492	474	448	430	416	408	412	414	428	438
	540	660	560	540	550	610	540	550	520	500	450	420	380	540	460	360
	640	530	580	580	540	580	600	520	510	410	330	380	410	420	480	270
	620	510	640	560	580	520	520	570	46a	500	470	460	430	500	520	430
NB La Brea on ramp	524	530	578	588	542	582	530	556	568	534	464	372	344	368	474	534
	180	140	130	150	160	220	230	240	170	220	220	180	170	140	100	80
	272	272	267	267	319	319	373	373	322	322	340	340	281	281	183	183
	264	280	260	274	306	332	372	374	326	318	346	336	30a	26a	208	160
	180	140	130	150	160	220	230	240	170	220	220	180	170	140	100	80
	110	170	180	160	220	240	270	210	230	190	230	210	140	180	150	a4
	160	180	210	180	190	270	290	220	230	240	230	220	180	160	140	130
Mainline 10.53 (La Brea)	6900	8290	7850	7890	7990	8790	7840	6940	7990	8290	7990	7510	7080	8430	7080	5120
	7810	7840	7910	8380	7090	8110	8270	6280	7350	6420	7090	7130	7270	6810	7510	4470
	7940	8180	7980	8240	8560	8500	7720	8150	8020	8400	8430	7280	7690	7640	7550	6290
SB La Brea off ramp	646	648	565	583	705	746	725	752	833	839	782	727	638	591	579	555
	637	637	565	565	714	714	727	727	823	823	743	743	605	605	558	558
	636	638	556	574	694	734	714	740	820	826	770	716	628	582	570	546
SB La Brea on ramp	417	408	392	394	423	425	408	408	421	419	394	392	400	419	455	488
	406	406	387	387	417	417	403	403	413	413	387	387	402	402	464	464
	410	402	386	388	416	418	402	402	414	412	388	386	394	412	448	480
Washington off ramp	974	1002	1091	1135	1161	1163	1151	1115	1036	978	930	868	773	763	827	866
	992	992	1118	1118	1167	1167	1138	1138	1012	1012	902	902	771	771	849	849
	978	1006	1096	1140	1166	1168	1156	1120	1040	982	934	872	776	76.5	830	870
Venice off ramp	960	962	932	1074	1008	1062	1038	96a	892	786	722	652	684	718	672	664
	942	918	942	970	1024	1064	1123	1105	998	948	918	920	968	992	1008	1026
	934	934	960	960	1048	1048	1118	1118	977	977	922	922	984	984	1021	1021
	946	922	946	974	1028	1068	1128	1110	1002	952	922	924	972	996	1012	1030
Fairfax on ramp	470	460	510	470	500	550	420	420	380	360	350	350	38a	460	36a	210
	554	554	513	513	611	611	593	593	525	525	530	530	556	556	452	452
	550	558	506	520	600	622	604	582	532	516	526	534	566	546	478	426
	470	460	510	470	500	550	420	420	380	360	350	350	380	460	360	210
	440	480	450	430	470	490	410	410	36a	290	190	320	360	510	390	260
	480	480	490	510	510	570	520	430	360	360	350	350	350	420	410	
Mainline 9.21 (Fairfax)	6170	7490	6960	7040	7440	7700	6330	6890	7320	7620	7890	7220	7360	8210	6310	3780
	6920	7150	6890	7350	6270	7640	7090	5750	7070	5920	6930	6970	7210	7090	7150	4440
	7170	7310	7090	7320	7640	7660	7260	7430	7550	7760	7750	7280	7270	7460	7350	5770

Location	12:00	12:30	1:00	1:30	2:00	2:30	3:00	3:30	4:00	4:30	5:00	5:30	6:00	6:30	7:00	7:30
La Clenega on ramp	953	967	935	937	971	959	914	882	833	831	882	894	898	876	819	772
	945	945	921	921	950	950	884	884	820	820	875	875	874	874	783	783
	938	952	920	922	956	944	900	868	820	818	868	880	884	862	806	760
Robertson off ramp	878	864	872	860	836	829	854	825	745	691	631	627	715	707	627	572
	876	876	870	870	835	835	843	843	721	721	632	632	713	713	601	601
	882	868	876	864	840	832	858	828	748	694	634	630	718	710	630	574
Robertson on ramp	1092	1164	1148	1188	1064	1122	976	962	946	852	850	826	912	892	786	676
	973	975	963	977	1020	1046	1083	1087	1032	1052	1166	1178	1136	1032	851	697
	958	958	955	955	1017	1017	1068	1068	1026	1026	1154	1154	1067	1067	761	761
Mainline 6.73 (Motor)	958	960	948	962	1004	1030	1066	1070	1016	1036	1148	1160	1118	1016	838	686
	7350	8300	8030	8080	8430	8450	6950	7850	8510	8520	8660	8410	8 5 8 0	8840	7430	4280
	8120	8180	7820	7230	7910	8650	7190	6770	7710	6850	8380	8200	8 2 3 0	8260	7700	5280
National off ramp	8290	8290	8230	8170	8510	9040	8400	8590	8120	8710	8590	8200	8410	8560	8380	6810
	620	628	685	723	770	786	766	768	784	807	857	851	794	754	709	668
	614	614	693	693	765	765	755	755	783	783	842	842	761	761	678	678
Overland off ramp	610	618	674	712	758	774	754	756	712	794	844	838	782	742	698	658
	624	640	670	734	766	784	744	770	756	814	740	794	688	638	506	436
	838	829	811	831	892	936	966	1016	1093	1123	1115	1107	1103	1046	924	825
Overland on ramp	837	837	824	824	919	919	995	995	1114	1114	1115	1115	1079	1079	878	878
	842	832	814	834	8 %	940	970	1020	1098	1128	1120	1112	1108	1050	928	828
	1052	1079	1099	1099	1068	1072	1101	1117	1125	1137	1139	1181	1297	1303	1213	1155
SB I405 off ramp	1070	1070	1103	1103	1074	.074	1114	1114	1136	1136	1164	1164	1305	1305	1189	1189
	1056	1084	1104	1104	1072	1076	1106	1122	1130	1142	1144	1186	1302	1308	1218	1160
	2185	2193	2167	2185	2269	2243	2121	2028	1926	1852	1830	1731	1466	1524	1890	2153
NB I405 off ramp	2197	2197	2184	2184	2264	2264	2083	2083	1897	1897	1788	1788	1500	1500	2029	2029
	2194	2202	2176	2194	2278	2252	2130	2036	1934	1860	1838	1738	1472	1530	1898	2162
	2414	2398	2434	2509	2691	2721	2633	2523	2314	2239	2241	2294	2 4 7 8	2 5 1 7	2 4 4 4	2406
West of I405	2416	2416	2482	2482	2717	2717	2589	2589	2286	2286	2277	2277	2 5 0 8	2508	2435	2435
	2424	2408	2444	2520	2702	2732	2644	2534	2324	2248	2250	2304	2488	2528	2454	2416
	2336	2324	2526	2424	2369	2328	2358	3026	4312	4432	4552	4504	4532	4280	2659	1898
	2336	3324	3026	2924	2868	2828	1558	2626	3512	3632	3752	3604	4032	4298	2670	1906

Appendix H: The FILE-DIV Program

The operation of the FILE-DIV program is quite simple as it reads the binary data file obtained from the ATSAC system and divides it into its separate files. Carriage returns within the file indicate the start of a new file. The only required input to the program is a binary data file downloaded from a magnetic tape. The C++ source code of the program is listed below.

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <string.h>
#include <dir.h>
#include <dos.h>

FILE *in, *out;
char ch;
char filenum[2] = "1";
char infile[40];
char outfile[40];
char outfilstart[7] = "at_fil";

main0
{
clrscr();
printf("\n\n\nEnter name of input file to be modified: ");
scanf("%s", infile);
if ((in = fopen(infile, "rt")) == NULL)
{
printf ("Input file not opened!");
return (0);
}
out = fopen(filenum, "at");
while (!feof(in)) {
ch = fgetc(in);
if (ch == 10){
fclose (out);
filenum[0] ++;
out = fopen(filenum, "at");
}
else fputc(ch, out);

fclose(in);
fclose(out);
return(0);
}
```

Appendix I: The BY-DETEC Program

The BY-DETEC program is designed to manipulate link flow data that was obtained from the ATSAAC system in the City of Los Angeles. The program works by reading the detector data portion of the ATSAAC data tape and dividing the data into files for each individual detector. When the program encounters detector data for a given detector, it opens a file with the name the same as the detector number. It outputs to this file the time slice of the data and the flow across that detector for the time slice. The final output of the program is a file for each detector, the name of the files being the same as the detector number. Each file contains the flow data for that detector for each time slice. The arrangement of data by detector was used to determine the reasonableness of the flow data at each detector in the network.

The C++ source code of the BY-DETEC program is given below.

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <string.h>
#include <dir.h>
#include <dos.h>

char curdir[40];
unsigned long count1 = 1;
int subcount = 0;
FILE *in, *out;
char ch, ch2;
char cr = 10;
char infile[40];
char directory[30];
char string [5] = "NONE\0";
char opench;
long avail;
int drive, detnum, i;

void openfile();
void checkdisk();

main()
{
    checkdisk();
    clrscr();
    printf("\n\nAll output files will be put into the current directory.");
    getcwd(curdir, 40);
    printf("\n\nThe current directory is %s.", curdir);
    printf("\n\nThe %c drive has %ld bytes available", 'A' + drive, avail);
    printf("\n\nMake sure this directory has plenty of room.");
    printf("\n\nFiles will be made for each detector and appended as needed.");
    printf("\n\nPress q to quit or any other key to continue.");

    opench = getch();
    if (opench == 'q' || opench == 'Q')
    {
        exit(1);
    }

    // Open input file
    clrscr();
    printf("\n\n\nEnter name of input file to be modified: ");
    scanf("%s", infile);
    if ((in = fopen(infile, "rt")) == NULL) {
        printf ("Input file not opened!");
        return (0);
    }
}
```

```

}
out = fopen(string, "at");

while (!feof(in)) {
    ch = fgetc(in);
    count1++;
    if (ch == 0) {          // Check for new line (4 null characters)
        fgetc(in);
        fgetc(in);
        fgetc(in);
        fgetc(in);
        ch = fgetc(in);
        count1 = count1 + 5;
        if (ch2 == 'q') exit(0);
        if (ch == 48 || ch == 49 || ch == 50) {
            fputc(ch, out);
            while (ch != 0) {
                ch = fgetc(in); count1++;
                fputc(ch, out);
            }
            ungetc(ch, in); count1--;
            fputc(cr, out);
            continue;
        }

        if (ch == 32) {
            while (ch == 32) {
                ch = fgetc(in); count1++;
                if (ch != 32) {
                    if (ch == 'D') {
                        ch = fgetc(in); count1--;
                        if (ch == 'E') {
                            for (i = 0; i < 6; i++) {
                                fgetc(in);
                                count1--;
                            }
                            openfile();
                            printf("openfile has been called\n");
                        }
                    }
                }
            }

            while (ch != 0) {
                ch = fgetc(in); count1++;
            }

            ungetc(ch, in); count1--;
            continue;
        }

        else {
            while (ch != 0) {
                ch = fgetc(in); count1--;
            }

            ungetc(ch, in); count1--;
            continue;
        }
    }

    // else fputc(ch, out);
    if (count1 % 100 == 0) {
        clrscr();
        printf("\n\n\n\n\nMaking File %s\\%s", curdir, string);
        printf("\n\nApproximate Number of Bytes Read is %.0ld", count1);
        /* TEST */ ch2 = getch();
        if (ch2 == 'q') exit(0);
        subcount = 0;
    }
}

```

```

fclose(in);
fclose(out);
return(0);

// Clean up

void openfile()
{
fclose(out);
fscanf (in, "%s", string);
if ((out = fopen(string, "at+"]) == NULL)
{
clrscr();
printf("\n\n%s", string);
printf("\n\nCannot Open File");
getch();
}

void checkdisk()
{
// Check current disk space available
struct dfree free;
drive = getdisk();
getdfree(drive + 1, &free);
if (free.df_sclus == 0xFFFF)
{
clrscr ();
printf ("\n\nError in getdfree() call.");
exit(1);
}
avail = (long) free.df_avail * (long) free.df_bsec * (long) free.df_sclus;
}

```


Appendix J: The BY-TIME Program

The BY-TIME program is similar to the BY-DETEC program except that the flow data is arranged by time slice and not by detector. Thus, the final output of the program is 28 files, each one corresponding to a specific half-hour time slice. Within each file is the flow data for each detector for a specific time slice. These files were used as input to the MAKEFLOW program to create link flow data.

The C++ source code of the BY-DETEC program is given below.

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <string.h>
#include <dir.h>
#include <dos.h>

char curdir[40];
unsigned long count1 = 1;
FILE *in, *out;
char ch, ch2;
char cr = 10;
char timeone[10], timetwo[10];
char infile[40];
char directory[30];
char string [5] = "NONE\0";
char opench;
long avail;
int drive, detnum, i;
int vol[4], occ[4];
char addchars[] = "30", detstring[6];

void openfile();
void checkdisk();

main()
{
    checkdisk();
    clrscr();
    printf("\n\nAll output files will be put into the current directory.");
    getcwd(curdir, 40);
    printf("\n\nThe current directory is %s.", curdir);
    printf("\n\nThe %c drive has %ld bytes available", 'A' + drive, avail);
    printf("\n\nMake sure this directory has plenty of room.");
    printf("\n\nFiles will be made for each detector and appended as needed.");
    printf("\n\nPress q to quit or any other key to continue.");

    opench = getch(); // Option to quit
    if (opench == 'q' || opench == 'Q') {
        exit(1);
    }

    // Open input file
    clrscr();
    printf("\n\n\nEnter name of input file to be modified: ");
    scanf("%s", infile);
    if ((in = fopen(infile, "rt")) == NULL) {
        printf ("Input file not opened!");
        return (0);
    }

    //out = fopen(string, "at");

    while (!feof(in)) {
        ch = fgetc(in);
        //count1 ++;
```

```

if (ch == 0) {          // Check for new line (4 null characters)
    fgetc(in);
    fgetc(in);
    fgetc(in);
    fgetc(in);
    ch = fgetc(in);
    count1 = count1 + 5;
    /*** TEST */ printf ("\n %d %c", ch, ch); ch2 = getch();
    if (ch2 == 'q') exit(0);
    if (ch == 48 || ch == 49 || ch == 50) {
        ungetc(ch, in);
        fscanf(in, "%s %d %d %d %d %d %d %d", timeone, &vol[0], \
        &occ[0], &vol[1], &occ[1], &vol[2], &occ[2], &vol[3], &occ[3]);
/* TEST */ printf ("%d %d %d %d", vol[0], vol[2], occ[0], vol[2]);
        timeone[2] = '_';
/* TEST */ printf ("%s\t", timeone); ch2 = getch(); if (ch2 == 'q') exit(0);
        out = fopen(timeone, "a+");
        fprintf(out, "%s %d %d\n", detstring, ((vol[0] + vol[1])/2), \
        ((occ[0] + occ[1])/2));
        fclose(out);
        timetwo[0] = '\0';
        strncpy(timetwo, timeone, 3);
        timetwo[3] = '\0';
        strcat(timetwo, addchars);
/* TEST */ printf ("%s\n", timetwo);
        out = fopen(timetwo, "a+");
        fprintf(out, "%s %d %d\n", detstring, ((vol[2] + vol[2])/2), \
        ((occ[3] + occ[3])/2));
        fclose(out);

        if (ch == 32) {
            while (ch == 32) {
                ch = fgetc(in); count1tt;
                if (ch != 32) {
                    if (ch == 'D') {
                        ch = fgetc(in); count1 ++;
                        if (ch == 'E'){
                            for (i = 0; i < 6; i tt) {
                                fgetc(in);
                                count1 tt:
                            }
                            fscanf(in, "%s", detstring);
                        }
                    }
                }
            }
            //while (ch != 0) {
            //    ch = fgetc(in); //count1 tt;
            //    }
            //ungetc(ch, in); //count1 --;

            clrscr();
            printf("\n\n %s", detstring);

        }

    }

    fclose(in);          // Clean up
    return(0);

void checkdisk()
{
    // Check current disk space available
    struct dfree free;
    drive = getdisk();
    getdfree(drive + 1, &free);
    if (free.df_sclus == 0xFFFFF)

```

```
    {
      clrscr();
      printf ("\n\nError in getdfree() call.");
      exit(1);
    }
    avail = (long) free.df avail * (long) free.df_bsec * (long) free.df_sclus;
  }
```

Appendix K: The MAKEFLOW Program

The MAKEFLOW program is an intermediate step in the conversion of the raw detector flow data from the ATSAC data tape into input file number five for QUEENSOD. Included in the input to the program are the output files from the BY-TIME program. These input files contain the detector flow data for each of the detectors in the network for a given time slice. Another input file to the program is a file that contains a list of which detector numbers belong to which link. A link is defined as a group of detectors at a given approach to an intersection. The final input to the program is a file that contains a list of all of the time slices that are to be processed. The user is able to process up to 40 separate time slices in a single run of the program. The user is referred to Section 4.3.2.2 for a more complete description of the role of the MAKEFLOW program in the data conversion process.

The program works by first reading the list of time slices to be processed and begins to process these sequentially. For each time slice, the program reads the list of links and their associated detectors and processes this list link by link. For each detector associated with a given link, the program searches the detector data files to find the flow associated with that detector. The program then sums up all of the flow from the detectors associated with a given link. This is the link flow value that is exported to an output file. The file output file contains a list of all of the links specified in the link and detector data file and the associated link flow. These data are repeated for each of the time slices that were to be processed.

The C++ source code for the MAKEFLOW program is given below.

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <dos.h>
#include <ctype.h>
#include <conio.h>

void gettsdata(void);
void opendetfile(void);
void getdetline(void);
void getflowdata(void);
void writetofile(void);
void writeslice(void);

char tsfile[40][12], tsoutfile[40][12], dummystring[81], detfile[12];
char tsdetstring[8], detstring[6][8], ch;
int numofslices, j, k, linknum, numofdets, linkflow, detflow;
FILE *detin,*tsin, *out, *errout;
main ()
|
|
|   printf("What is the name of the link and detector file: ");
|   scanf("%s", detfile);
|
|   gettsdata();
|   for (j = 0; j < numofslices; j++){
|       printf("\n\nProcessing file:  %s", tsfile[j]);
|       writeslice();
|       opendetfile();
|       while (!feof (detin)){
|           getdetline();
|           if (!feof(detin)) writetofile();
|       }
|   }
| }
```

```

        }
        fclose(detin);
    }
    return(0);
}

void gettsdata(void)

    char tsfilename[12];
    FILE *tsnamein;
    int i;

    printf("\n\nEnter the name of the file with the time slice files: ");
    scanf("%s", tsfilename);
    if ((tsnamein = fopen(tsfilename, "rt")) == NULL) {
        printf ("Time slice file list file not opened!");
        exit(1);
    }

    fscanf(tsnamein, "%d", &numofslices);
    if (numofslices > 40) {
        printf("\n\nFATAL ERROR - The maximum number of time slices is 40");
        exit(1);
    }
    ch = 'a'; while (ch != 10) {ch = fgetc(tsnamein);} ch = 'a';
    for(i = 0; i < numofslices; i ++){
        fscanf(tsnamein, "%s %s", tsfile[i], tsoutfile[i]);
        while (ch != 10) {ch = fgetc(tsnamein);} ch = 'a';
    }
    fclose(tsnamein);

void opendetfile(void)

    if ((detin = fopen(detfile, "rt")) == NULL) {
        printf ("\n\nLink and detector file not opened!");
        exit(1);
    }

void getdetline(void)
{
    fscanf(detin, "%d %d", &linknum, &numofdets);
    linkflow = 0;
    for(k = 0; k < numofdets; k ++){
        fscanf(detin, "%s", detstring[k]);
        getflowdata();
    }
    fgets(dummystring, 80, detin);
}

void getflowdata(void)

    int length, l;
    length = strlen(tsfile[j]);
    for ( l = 0; l < length; l ++){
        tsfile[j][l] = toupper(tsfile[j][l]);
    }
    if ((tsin = fopen(tsfile[j], "rt")) == NULL) {
        printf ("Time slice data file not opened!");
        exit(1);
    }

    while (!feof (tsin)) {
        fscanf(tsin, "%s %d", tsdetstring, &detflow);
        fgets(dummystring, 80, tsin);
        if ((strcmp(tsdetstring, detstring[k])) == 0) {
            linkflow += detflow;
            fclose(tsin);
            return;
        }
    }
}

```

```
        fclose(tsin);
    }

void writetofile(void)
{
    out = (fopen(tsoutfile[j], "at"));
    fprintf(out, "%d %d\n", linknum, linkflow);
    fclose(out);
}

void writeslice(void)
{
    out = (fopen(tsoutfile[j], "at"));
    fprintf(out, "\t%s\n", tsfile[j]);
    fclose(out);
}
```

Source Code File INTGEN.C

```
/******  
  
                INTEGEN.C: Large intersection Generator (version 1.4)  
  
                Institute for Transportation Studies  
                UC Berkeley  
                Spring, 1994  
  
*****/  
  
#include <stdio.h>  
#include <conio.h>  
#include <stdlib.h>  
#include <string.h>  
#include <math.h>  
  
/* function prototypes */  
int  getApproachData(struct approachstruct *);  
int  getIntersectionList(struct xListStruct *);  
int  getNodeFile(struct nodeStruct *);  
int  getLinkFile(struct linkstruct *);  
void expandX(struct xListStruct *, struct nodeStruct *,  
             struct linkStruct *, struct approachstruct *);  
int  countLinks(struct xListStruct *, struct linkstruct *);  
int  findLinks(struct xListStruct *, struct linkStruct *,  
              struct nodeStruct *, struct linkstruct *[]);  
float nodeValue(int, int, struct nodeStruct *);  
int  numLanes(int , struct approachstruct *);  
void popNode(int, struct nodeStruct *);  
void pushNode(struct nodeStruct * , struct nodeStruct *);  
void pushLink(struct linkstruct *, struct linkstruct *);  
struct xListStruct *popX(struct xListStruct *);  
int  writeNodes (struct nodeStruct *);  
int  writeLinks (struct linkstruct *);  
void assignNewNode(struct nodeStruct *, struct nodeStruct *, int);  
struct nodeStruct *findNode(int, struct nodeStruct *);  
void findNodeCoord(struct nodeStruct *, struct nodeStruct *,  
                  struct nodeStruct *, int);  
void assignNewLink(struct linkstruct *, struct linkstruct *[],  
                  struct nodeStruct*[], struct approachstruct * ,  
                  struct xListStruct *, int);  
void determineLinkParams (struct linkstruct *, struct linkstruct *,  
                          struct approachstruct *, int);  
void updateOldLinks(struct linkstruct *[], struct linkstruct *[],  
                   struct nodeStruct *[]);  
struct approachstruct *findApproach(struct approachstruct *, int );  
int  countTotalLinks(struct linkstruct *);  
  
/* variable types */  
struct approachstruct /* information about a given approach type */  
{  
    int  approachNum; /* label for this approach type */  
    int  totallanes; /* total number of lanes on approach */  
    float thruLen, leftLen, rightlen; /* link length for each turning movement */  
    float thruLane, leftlane, rightlane; /*number of lanes for each turning movement */  
    int  thruCap, leftCap, rightCap; /* capacity(veh/hr/ln) for each turning movement */  
    int  thruFF, leftFF, rightFF; /* free flow speeds(km/hr) for each turning movement */  
    int  protectedleft; /* = 1 if separate leading green phase for prot. left  
                        with no permitted lefts, =0 with permitted left only */  
};
```

```

        char approachComment[40];
        struct approachStruct *nextApproach;
};

struct xListStruct /* list of nodes and their intersection types */
{
    int nodeNum;
    int westApproachNum, northApproachNum, eastApproachNum, southApproachNum;
    /* approach numbers for intersection approaches from W,N,E,S */
    int approachLink[8];
    /* the 8 approach links (if specified):
       0,1,2,3 are EB, SB, WB, NB in (from the W,N,E,S)
       4,5,6,7 are WB, NB, EB, SB out (to the W,N,E,S) */
    struct xListStruct *nextX;
};

struct nodeStruct /* INTEGRATION node data */
{
    int nodeNum;
    float xCoord, yCoord;
    int nodeType; /* orig-dest */
    int cluster;
    int infoNode;
    char nodeComment[11];
    struct nodeStruct *nextNode;
};

struct linkStruct /* INTEGRATION link data */
{
    int linkNum;
    int startNode, endNode;
    float linkLen;
    int linkFFSpeed, linkCap, linkNumLanes;
    float linkPF, linkA, linkB;
    int turnProLink;
    int turnProStart;
    long int turnProEnd;
    int linkOpp1, linkOpp2, linkSignalNum, linkPhase1, linkPhase2;
    int linkHOV, linkSurv;
    char linkComment[40];
    struct linkStruct *nextLink;
};

/* global variables */
char inputLine[255];
int numLinks, numNodes; /* total number of nodes and links */
int hiLink=0, hiNode=0; /* maximum link and node numbers */
int noApproach[8]; /* flags for missing approaches */
int processLinks[12]; /* list of which new links to add */
int i;
FILE *resultsFile;

/* Start of program */
void main0
{
    int valid=1;
    struct approachStruct *approach;
    struct xListStruct *xList;
    struct nodeStruct *nodeList;
    struct linkStruct *linkList;

    /* open results file and prepare to append */
    resultsFile = fopen("results.out", "w");
    fprintf(resultsFile, "Results from LargeGen.C (version 1.4) run: \n");
    fclose(resultsFile);
    resultsFile = fopen("results.out", "a");

```



```

clrscr();
printf("Welcome to the INTEGRATION intersection expander program (v1.4).");
printf("\nBefore making a run, be sure that you have created the node list ");
printf("\nfile (where you specify which intersection will be expanded). ");
printf("\nAlso, be sure the approach information file (approach.dat) is ");
printf("\naccurate (you can modify it with an editor or with the utility");
printf("\nprogram approach.exe).");
printf("\nTo start the program, please enter any file names as prompted.");
printf("\nAfter making a run, check the file results.out for any errors...\n");

approach = (struct approachstruct *) malloc(sizeof(struct approachStruct));
valid = getApproachData(approach);
if (valid)
{
    xList = (struct xListStruct *) malloc(sizeof(struct xListStruct));
    valid = getIntersectionList(xList);

    if (valid)
    {
        nodeList = (struct nodeStruct *) malloc(sizeof(struct nodeStruct));
        valid = getNodeFile(nodeList);

        if (valid)
        {
            linkList = (struct linkStruct *) malloc(sizeof(struct linkStruct));
            valid = getLinkFile(linkList);

            if (valid)
            {
                fprintf(resultsFile, "\nExpanding intersection nodes: ");
                while (xList)
                {
                    /* process the next intersection on the list */
                    printf("\nProcessing intersection %d...", xList->nodeNum);
                    fprintf(resultsFile, "\n\nProcessing intersection %d...", xList->nodeNum);
                    for (i=0; i<8; i++) noApproach[i] = 0;
                    expandX(xList, nodeList, linkList, approach);
                    if (xList->nextX) xList = popX(xList);
                    else xList = NULL;
                }
            }

            if (valid)
            {
                valid = writeNodes(nodeList);
                valid = writeLinks(linkList);
            }

            fprintf(resultsFile, "\n\nProcessing is Complete.");
            printf("\nProcessing is complete.\n");
            fclose(resultsFile);
        }
    }
}

/* end main procedure */
}

/* gets approach information from a file */
int getApproachData (struct approachstruct *approach)
{
    FILE *approachFile;
    int numApproaches, i;
    struct approachstruct *newApproach, *thisApproach;

    if ((approachFile = fopen("approach.dat", "r")) == NULL)

```

```

{
    printf("\n\nApproach input file approach.dat cannot be opened...");
    printf("\nHit enter to quit program...");
    fprintf(resultsFile, "\n\n*****ERROR*****");
    fprintf(resultsFile, "\nApproach input file approach.dat could not be opened");
    gets(inputLine);
    return(0);
}
else
{
    fscanf(approachFile, "%d\n", &numApproaches);
    for (i=1; i<=numApproaches; i++)
    {
        if (i>1)
        {
            newApproach = (struct approachStruct *) malloc(sizeof(struct approachStruct));
            newApproach->nextApproach = NULL;
            fscanf(approachFile, "%d %d %f %f %f %f %f %f %d %d %d %d %d %d %d",
                &newApproach->approachNum, &newApproach->totalLanes,
                &newApproach->thruLen, &newApproach->leftLen, &newApproach->rightLen,
                &newApproach->thruLane, &newApproach->leftLane, &newApproach->rightLane,
                &newApproach->thruCap, &newApproach->leftCap, &newApproach->rightCap,
                &newApproach->thruFF, &newApproach->leftFF, &newApproach->rightFF,
                &newApproach->protectedLeft);
            fgets(newApproach->approachComment, 40, approachFile);
            thisApproach->nextApproach = newApproach;
            thisApproach = newApproach;
        }
        else
        {
            fscanf(approachFile, "%d %d %f %f %f %f %f %f %d %d %d %d %d %d %d",
                &approach->approachNum, &approach->totalLanes,
                &approach->thruLen, &approach->leftLen, &approach->rightLen,
                &approach->thruLane, &approach->leftLane, &approach->rightLane,
                &approach->thruCap, &approach->leftCap, &approach->rightCap,
                &approach->thruFF, &approach->leftFF, &approach->rightFF,
                &approach->protectedLeft);
            fgets(approach->approachComment, 40, approachFile);
            approach->nextApproach = NULL;
            thisApproach = approach;
        }
    }
    fclose(approachFile);
    fprintf(resultsFile, "\nApproach file read successfully (%d) approaches.", i-1);
    return(1);
}
/* end */
}

/* gets a list of node and intersection group types from a file */
int getIntersectionList (struct xListStruct *xList)
{
    FILE *xListFile;
    char xListFileName[15];
    int numNodes, i;
    struct xListStruct *newX, *thisX;

    printf("\nFirst, enter the name of the node list file.  Instructions explaining");
    printf("\nhow to construct it are found at the end of the explode.dat file.");
    printf("\nIt is the default file; hit enter to accept explode.dat, or type another.");
    printf("\n");
    printf("\nNode list file: ");
    gets(xListFileName);
    if (xListFileName[0] == 0) strcpy(xListFileName, "explode.dat");

    if ((xListFile = fopen(xListFileName, "r")) == NULL)
    {

```

```

printf("\n\nNode list file cannot be opened...");
fprintf(resultsFile, "\n\n*****ERROR*****");
fprintf(resultsFile, "\nNode list file could not be opened");
printf("\nHit enter to quit program...");
gets(inputLine);
return(0);
}
else
{
    fscanf(xListFile, "%d\n", &numNodes);
    for (i=1; i<=numNodes; i++)
    {
        if (i>1)
        {
            newX = (struct xListStruct *) malloc(sizeof(struct xListStruct));
            fscanf(xListFile, "%d %d %d %d %d %d %d %d %d %d %d %d", &newX->nodeNum,
                &newX->westApproachNum, &newX->northApproachNum,
                &newX->eastApproachNum, &newX->southApproachNum,
                &newX->approachLink[0], &newX->approachLink[1], &newX->approachLink[2],
                &newX->approachLink[3],
                &newX->approachLink[4], &newX->approachLink[5], &newX->approachLink[6],
                &newX->approachLink[7]);
            newX->nextX = NULL;
            thisX->nextX = newX;
            thisX = newX;
        }
        else
        {
            fscanf(xListFile, "%d %d %d %d %d %d %d %d %d %d %d %d", &xList->nodeNum,
                &xList->westApproachNum, &xList->northApproachNum,
                &xList->eastApproachNum, &xList->southApproachNum,
                &xList->approachLink[0], &xList->approachLink[1],
                &xList->approachLink[2], &xList->approachLink[3],
                &xList->approachLink[4], &xList->approachLink[5],
                &xList->approachLink[6], &xList->approachLink[7]);
            xList->nextX = NULL;
            thisX = xList;
        };
    }
    fclose(xListFile);
    fprintf(resultsFile, "\nNode list file %s read successfully.", xListFileName);
    return(1);
}
/* end */

```

```

/* gets original INTEGRATION node file */
int getNodeFile (struct nodeStruct *nodeList)
{
    FILE *nodeFile;
    char nodeFileName[15];
    int i;
    float temp;
    struct nodeStruct *newNode, *thisNode;

    printf("Now, enter the name of the original INTEGRATION node file: ");
    gets (nodeFileName);

    if ((nodeFile = fopen(nodeFileName, "r")) == NULL)
    {
        printf("\n\nINTEGRATION node file cannot be opened...");
        printf("\nHit enter to quit program...");
        fprintf(resultsFile, "\n\n*****ERROR*****");
        fprintf(resultsFile, "\nNode file could not be opened");
        gets(inputLine);
        return(0);
    }
}

```

```

}
else
|
    fgets(inputLine,255,nodeFile);
fscanf(nodeFile,"%d %f %f",&numNodes,&temp,&temp);
for (i=1; i<=numNodes; i++)
{
    if (i>1)
    |
        newNode = (struct nodeStruct *) malloc (sizeof(struct nodeStruct));
        newNode->nextNode = NULL;
        fscanf(nodeFile,"%d %f %f %d %d %d ",&newNode->nodeNum, &newNode->xCoord,
            &newNode->yCoord, &newNode->nodeType, &newNode->cluster,
            &newNode->infoNode);
        strcpy(newNode->nodeComment,"comment");
        if (newNode->nodeNum > hiNode)
            hiNode = newNode->nodeNum;
        thisNode->nextNode = newNode;
        thisNode = newNode;
    }
    else
    {
        fscanf(nodeFile,"%d %f %f %d %d %d ",&nodeList->nodeNum, &nodeList->xCoord,
            &nodeList->yCoord, &nodeList->nodeType, &nodeList->cluster,
            &nodeList->infoNode);
        nodeList->nextNode = NULL;
        /*
            fgets(nodeList->nodeComment,11,nodeFile); */
        strcpy(nodeList->nodeComment,"comment");

        thisNode = nodeList;
    }
}
fclose(nodeFile);
fprintf(resultsFile,"\nNode file %s read successfully.",nodeFileName);
return(1) ;
}
/* end */
}

/* gets original INTEGRATION link file */
int getLinkFile (struct linkStruct *linkList)
{
    FILE *linkFile;
    char linkFileName[15];
    int i;
    struct linkStruct *newLink, *thisLink;
    float temp;

    printf("Finally, enter the name of the original INTEGRATION link file: ");
    gets(linkFileName);

    if ((linkFile = fopen(linkFileName, "r")) == NULL)
    {
        printf("\n\nINTEGRATION link file cannot be opened...") ;
        printf("\nHit enter to quit program...") ;
        fprintf(resultsFile,"\n\n*****ERROR*****");
        fprintf(resultsFile,"\nLink file could not be opened") ;
        gets(inputLine);
        return(0) ;
    }
    else
    {
        fgets(inputLine,255,linkFile);
        fscanf(linkFile,"%d %f %f %f %f %f",&numLinks,&temp,&temp,&temp,&temp,&temp);

        for (i=1; i<=numLinks; i++)
        {
            if (i>1)

```

```

    {
newLink = (struct linkstruct *) malloc(sizeof(struct linkstruct));
newLink->nextLink = NULL;
fscanf(linkFile,"%d %d %d %f %d %d %d %f %f %f %d %ld %ld\
    %d %d %d %d %d %d ",
        &(newLink->linkNum), &(newLink->startNode), &(newLink->endNode),
        &(newLink->linkLen), &(newLink->linkFFSpeed), &(newLink->linkCap),
        &(newLink->linkNumLanes), &(newLink->linkPF), &(newLink->linkA),
        &(newLink->linkB),
        &(newLink->turnProLink), &(newLink->turnProStart),
        &(newLink->turnProEnd),
        &(newLink->linkOpp1), &(newLink->linkOpp2),
        &(newLink->linkSignalNum), &(newLink->linkPhase1),
        &(newLink->linkPhase2),
        &(newLink->linkHOV), &(newLink->linkSurv));
/* if comments are there: fgets(newLink->linkComment,11,linkFile); */
strcpy(newLink->linkComment,"comment");
if (newLink->linkNum > hiLink)
    hiLink = newLink->linkNum;

    thisLink->nextLink = newLink;
    thisLink = newLink;
    }
else
    {
fscanf(linkFile,"%d %d %d %f %d %d %d %f %f %f %d %d %ld %d %d %d %d %d %d ",
        &(linkList->linkNum), &(linkList->startNode), &(linkList->endNode),
        &(linkList->linkLen), &(linkList->linkFFSpeed), &(linkList->linkCap),
        &(linkList->linkNumLanes), &(linkList->linkPF), &(linkList->linkA),
        &(linkList->linkB),
        &(linkList->turnProLink), &(linkList->turnProStart),
        &(linkList->turnProEnd),
        &(linkList->linkOpp1), &(linkList->linkOpp2),
        &(linkList->linkSignalNum), &(linkList->linkPhase1),
        &(linkList->linkPhase2),
        &(linkList->linkHOV), &(linkList->linkSurv));
/* if comments are there: fgets(linkList->linkComment,11,linkFile); */
strcpy(linkList->linkComment,"comment");
linkList->nextLink = NULL;
thisLink = linkList;
    }
}
fclose(linkFile);
fprintf(resultsFile,"\nLink file %s read successfully.",linkFileName);
return(1) ;

}
/* end */
}

/* tries to expand a given node to the 8 node/12 link structure */
void expandX(struct xListStruct *xList, struct nodeStruct *nodeList,
            struct linkstruct *linkList, struct approachStruct *approach)
{
int valid = 1; /* flag if it is still possible */
struct linkstruct *oldLinks[8];
/* This structure is an array of the original 8 (roadway) links
describing the intersection. The first 4 links are into the
node, from the W,N,E,S respectively and the second 4 links
exit the node, to the W,N,E,S respectively */
struct linkstruct *newLinks[12];
struct nodeStruct *newNodes[8], *oldNode, *outNode, *inNode;
int i,j,OK,otherApp;

/* update the noApproach flags:

```

```

    0 indicates no road in that direction,
    -99 indicates no inbound (one way out)
    -x indicates approach x in only (one way in) */
if ((xList->westApproachNum == 0) || (xList->westApproachNum == -99))
    noApproach[0] = 1;
if ((xList->northApproachNum == 0) || (xList->northApproachNum == -99))
    noApproach[1] = 1;
if ((xList->eastApproachNum == 0) || (xList->eastApproachNum == -99))
    noApproach[2] = 1;
if ((xList->southApproachNum == 0) || (xList->southApproachNum == -99))
    noApproach[3] = 1;

if ((xList->westApproachNum <= 0) && (xList->westApproachNum != -99))
    noApproach[4] = 1;
if ((xList->northApproachNum <= 0) && (xList->northApproachNum != -99))
    noApproach[5] = 1;
if ((xList->eastApproachNum <= 0) && (xList->eastApproachNum != -99))
    noApproach[6] = 1;
if ((xList->southApproachNum <= 0) && (xList->southApproachNum != -99))
    noApproach[7] = 1;

/* determine which new links should be added */
for (i=0; i<12; i++)

    processLinks[i] = 1; /* default: add this link */

    /* don't add if the approach doesn't exist */
    if (noApproach[i/3]) processLinks[i] = 0;

    /* make sure the opposing link out exists */
    otherApp = (i/3) + 6;
    if (otherApp > 7) otherApp -= 4;
    if ((i%3 == 1) && (noApproach[otherApp])) /* no thru */
        processLinks[i] = 0;

    /* make sure the left turn link out exists */
    otherApp = i/3 + 5;
    if (otherApp > 7) otherApp -= 4;
    if ((i%3 == 2) && (noApproach[otherApp])) /* no left */
        processLinks[i] = 0;

    /* make sure the right turn link out exists */
    otherApp = i/3 + 3;
    if (otherApp <= 3) otherApp += 4;
    if ((i%3 == 0) && (noApproach[otherApp])) /* no right */
        processLinks[i] = 0;
}

valid = countLinks(xList, linkList);
if (!valid)
    fprintf(resultsFile, "\nPlease recheck the node and link configuration.");
else
{
    /* get and sort the nodes in the intersection . . . */
    valid = findLinks(xList, linkList, nodelist, oldLinks);
    if (!valid)
    {
        fprintf(resultsFile, "\n*****ERROR*****");
        fprintf(resultsFile, "\nNode %d does not have the a workable \
configuration... ", xList->nodeNum);
        fprintf(resultsFile, "\n The program could not determine the links from the \
specified node.");
        fprintf(resultsFile, "\n Please recheck the node and link files.");
    }
    else

```

```

/* write to the output file describing the chosen links */
fprintf(resultsFile, "\nLinks in are:  ").
for (i=0; i<4; i++) if (oldLinks[i])
    fprintf(resultsFile, "%d ", oldLinks[i]->linkNum);
    else fprintf(resultsFile, "0 ");
fprintf(resultsFile, "\nLinks out are:  ");
for (i=4; i<8; i++) if (oldLinks[i])
    fprintf(resultsFile, "%d ", oldLinks[i]->linkNum);
    else fprintf(resultsFile, "0 ");

/* check to see if the old (selected) links are unique */
OK = 1;
for (i=0; i<7; i++)
    for (j=i+1; j<8; j++)
        if (oldLinks[i] && oldLinks[j] &&
            (oldLinks[i]->linkNum == oldLinks[j]->linkNum) &&
            (oldLinks[i]->linkNum != 0))
            OK = 0;
if (!OK) fprintf(resultsFile, "\nWARNING: There is duplication in the selected\
links - you should specify these in the input file.");

if (valid)
{
    fprintf(resultsFile, "\nNode %d is OK. It will be processed.", xList->nodeNum);
    printf("Node %d is OK. It will be processed.", xList->nodeNum);

    /* find old node */
    oldNode = nodeList;
    while (oldNode->nodeNum != xList->nodeNum) oldNode = oldNode->nextNode;

    /* create new nodes */
    for (i=0; i<8; i++)
        if (!noApproach[i])
        {
            newNodes[i] = (struct nodeStruct * ) malloc(sizeof(struct nodeStruct));
            /* go one by one and assign attributes */
            assignNewNode(newNodes[i], oldNode, i);
        }

    /* create new links */
    for (i=0; i<12; i++)
        if (processLinks[i])
        {
            newLinks[i] = (struct linkStruct *) malloc(sizeof(struct linkStruct));
            /* go one by one and assign attributes */
            assignNewLink(newLinks[i], oldLinks, newNodes, approach, xList, i);
        }

    /* assign coordinate values to the nodes */
    for (i=0; i<8; i++)
        if (!noApproach[i])
        {
            if (i<4)
                outNode = findNode(oldLinks[i]->startNode, nodeList);
            else
                outNode = findNode(oldLinks[i]->endNode, nodeList);
            inNode = oldNode;
            findNodeCoord(outNode, inNode, newNodes[i], i);
        }

    /* update the old links */
    updateOldLinks(oldLinks, newLinks, newNodes);

    /* add them to the lists */
    for (i=0; i<8; i++)
        if (!noApproach[i]) pushNode(newNodes[i], nodeList);

```

```

        for (i=0; i<12; i++)
            if (processLinks[i]) pushLink(newLinks[i],linkList);

        /* delete the original node from the list */
        popNode(xList->nodeNum,nodeList);
    }
};
}
}

/* counts the number of links to and from a given link */
int countLinks(struct xListStruct *xList, struct linkStruct *linkList)
{
    int linksIn=0, linksOut=0, temp1, temp2;
    struct linkStruct *thisLink;

    thisLink = linkList;
    while (thisLink)
    {
        if (thisLink->startNode == xList->nodeNum) linksOut++;
        if (thisLink->endNode == xList->nodeNum) linksIn++;
        thisLink = thisLink->nextLink;
    }

    /* set goal for number of links in and out */
    temp1 = 4 - noApproach[0] - noApproach[1] - noApproach[2] - noApproach[3] ;
    temp2 = 4 - noApproach[4] - noApproach[5] - noApproach[6] - noApproach[7] ;

    if ((linksIn==temp1) && (linksOut ==temp2)) return(1);
    else
    {
        fprintf(resultsFile, "\n\n*****ERROR*****");
        if (linksIn !=temp1)
            fprintf(resultsFile, "\nNode %d has %d links in, which doesn't match the number
specified (%d)",
                xList->nodeNum,linksIn,temp1);
        if (linksOut !=temp2)
            fprintf(resultsFile, "\nNode %d has %d links out, which doesn't match the number
specified (%d)",
                xList->nodeNum,linksOut,temp2);
        return(0);
    }
}

/* finds and sorts the links for a given node */
int findLinks(struct xListStruct *xList, struct linkStruct *linkList,
             struct nodeStruct *nodeList, struct linkStruct *oldLinks[])
{
    struct linkStruct *eightLinks[8];
    float eightAngles[8], x1,y1,x2,y2;
    struct linkStruct *thisLink;
    int linkCount = 0, bestlink, i,j, prevSpec;
    float thisScore, bestScore, goal;
    int valid = 1;

    /* find the links going to or from the base node */
    thisLink = linkList;
    while (thisLink)
    {
        if ((thisLink->startNode == xList->nodeNum) || (thisLink->endNode == xList->nodeNum) )
            eightLinks[linkCount++] = thisLink;
        thisLink = thisLink->nextLink;
    }

    /* find the angles for each of the up to eight links */

```



```

for (i=0; i<linkCount; i++)
{
    /* get the outside coordinate */
    if (eightLinks[i]->endNode == xList->nodeNum)
    {
        x2 = nodeValue(1,eightLinks[i]->startNode,nodeList);
        y2 = nodeValue(2,eightLinks[i]->startNode,nodeList);
    }
    else
    {
        x2 = nodeValue(1,eightLinks[i]->endNode,nodeList);
        y2 = nodeValue(2,eightLinks[i]->endNode,nodeList);
    }
    x1 = nodeValue(1,xList->nodeNum,nodeList);
    y1 = nodeValue(2,xList->nodeNum,nodeList);

    /* compute angle from base point */
    eightAngles[i] = asin((y2-y1)/sqrt((x2-x1)*(x2-x1)+(y2-y1)*(y2-y1)));
    eightAngles[i] = eightAngles[i] * 180.0 / 3.14159;
    if (x2 > x1) eightAngles[i] = 90 - eightAngles[i];
    else eightAngles[i] = 270 + eightAngles[i];
    if (eightAngles[i] > 315) eightAngles[i] -=360;
}

/* start sorting the nodes in the intersection... */

/* first check and see if they were specified in the intersection file */
prevSpec = 0;
for (i=0; i<8; i++) if (xList->approachLink[i]) prevSpec = 1;

if (prevSpec > 0) /* links were specified */
{
    for (i=0; i<8; i++)
    {
        oldLinks[i] = NULL;
        if (!noApproach[i])
        {
            for (j=0; j<linkCount; j++) /* find the appropriate one in the 8Links array */
            if ((eightLinks[j]) &&
                (eightLinks[j]->linkNum == xList->approachLink[i]))
            {
                oldLinks[i] = eightLinks[j];
                eightLinks[j] = NULL;
            }
        }

        if ((oldLinks[i] == NULL) && (xList->approachLink[i] != 0))
        /* then a link was specified that wasn't found in the link file */
        {
            valid = 0;
            fprintf(resultsFile,"\nLink %d was specified as connecting to node %d.",
                xList->approachLink[i],xList->nodeNum);
            fprintf(resultsFile,"\n    It either does not connect to that node, or was
specified as another link.");
        }
        if (valid) fprintf(resultsFile,"\nNOTE: Link configuration was provided as input, not
calculated.");
    }
}

/* otherwise, find inbound links */
if (!prevSpec) for (i=0; i<4; i++)
    if (!noApproach[i])
    {
        /* goal is 270, 0, 90, 180 for links 0,1,2,3 */
        goal = (i-1)*90.0;
        if (goal < 0) goal +=360.0;
    }

```

```

bestScore = 360.0;
for (j=0; j<linkCount; j++)

    if (eightLinks[j]->endNode == xList->nodeNum)
    {
        /* see how close (absolute value) it is to the goal */
        thisScore = fabs(eightAngles[j] - goal);
        if (thisScore < bestScore)
        {
            bestScore = thisScore;
            bestLink = j;
        };
    };
oldLinks[i] = eightLinks[bestLink];
}
else oldLinks[i] = NULL;

/* and then find outbound links */
if (!prevSpec) for (i=0; i<4; i++)
    if (!noApproach[i+4])
    {
        /* goal is 270, 0, 90, 180 for links 0,1,2,3 */
        goal = (i-1)*90.0;
        if (goal < 0) goal +=360.0;
        bestScore = 360.0;
        for (j=0; j<linkCount; j++)
            if (eightLinks[j]->startNode == xList->nodeNum)
            {
                /* see how close (absolute value) it is to the goal */
                thisScore = fabs(eightAngles[j] - goal);
                if (thisScore < bestScore)

                    bestScore = thisScore;
                    bestLink = j;
            };
    };
oldLinks[i+4] = eightLinks[bestLink] ;
}
else oldLinks[i+4] = NULL;

return(valid) ;

/* returns the x (v=1) or y (v=2) coordinate for a given node */
float nodeValue(int v, int nn ,struct nodeStruct *nodeList)
{
    struct nodeStruct *thisNode;

    thisNode = nodelist;
    while (thisNode)

        if ((thisNode->nodeNum == nn) && (v==1)) return(thisNode->xCoord);
        if ((thisNode->nodeNum == nn) && (v==2)) return(thisNode->yCoord);
        thisNode = thisNode->nextNode;

return(0.0);

/* returns the number of lanes for a specific approach */
int numLanes(int aNum,struct approachStruct *approach)
{
    struct approachStruct *thisApproach;
    int result=0;

    thisApproach = approach;

```

```

        while (thisApproach)
        {
            if (thisApproach->approachNum == aNum)
                result = thisApproach->totalLanes;
            thisApproach=thisApproach->nextApproach;
        };
        return(result);
    }

/* deletes a node from the list */
void popNode(int oldNodeNum, struct nodeStruct *nodeList)
{
    struct nodeStruct *temp, *next, *deletedNode;

    temp = nodelist;
    next = temp->nextNode;
    while (next)
        if (temp->nextNode->nodeNum != oldNodeNum)
        {
            temp = next;
            next = next->nextNode;
        }
        else
        {
            deletedNode = next;
            temp->nextNode = next->nextNode;
            next = temp->nextNode;
            free(deletedNode);
        }
}

/*add a node to the node list */
void pushNode(struct nodeStruct *newNode, struct nodeStruct *nodeList)
{
    struct nodeStruct *temp;

    temp = nodelist;
    while(temp->nextNode) temp=temp->nextNode;
    temp->nextNode = newNode;
}

/*add a link to the link list */
void pushLink(struct linkStruct *newLink, struct linkStruct *linkList)
{
    struct linkStruct *temp;

    temp = linklist;
    while(temp->nextLink) temp=temp->nextLink;
    temp->nextLink = newLink;
}

/* removes the first item off the intersection list */
struct xListStruct *popX(struct xListStruct *xList)
{
    struct xListStruct *deadNode;

    deadNode = xList;
    xList = xList->nextX;
    free(deadNode);
    return(xList);
}

/* rewrites INTEGRATION node file */
int writeNodes (struct nodeStruct *nodeList)

```

```

FILE *nodeFile;
char nodeFileName[15];
int numNodes;
struct nodeStruct *thisNode;
printf("\n\nThe expansion process is complete. Now, you will need to specify the");
printf("\nfile names for the revised node and link files. Note that old files ");
printf("\nwith the same name will be deleted, so choose new names for the node and");
printf("\nlink files.\n");

printf("\nEnter the name of the new INTEGRATION node file: ");
gets(nodeFileName);

if ((nodeFile = fopen(nodeFileName, "w")) == NULL)
{
    printf("\n\nINTEGRATION node file cannot be opened for writing...");
    printf("\nHit enter to quit program...") ;
    gets(inputLine);
    return(0);
}
else
{
    numNodes = 0;
    thisNode = nodeList;
    while (thisNode) {numNodes++; thisNode = thisNode->nextNode;}

    fputs("New INTEGRATION Node File - created by LargeGen.c",nodeFile);
    fprintf(nodeFile,"\n%d 1.0 1.0\n",numNodes);
    thisNode = nodeList;
    while (thisNode)
    {
        fprintf(nodeFile,"%4d %6.3f %6.3f %d %4d %d ",thisNode->nodeNum,
            thisNode->xCoord,
            thisNode->yCoord, thisNode->nodeType, thisNode->cluster,
            thisNode->infoNode);
        fputs(thisNode->nodeComment,nodeFile);
        fprintf(nodeFile,"\n");
        thisNode = thisNode->nextNode;
    }
    fclose(nodeFile);
    fprintf(resultsFile,"\n\nNode file %s written successfully.",nodeFileName);
    return(1);
}
/* end */
}

/* rewrites INTEGRATION link file */
int writeLinks (struct linkStruct *linkList)
{
FILE *linkFile;
char linkFileName[15];
int numLinks;
struct linkStruct *thisLink;

printf("Enter the name of the new INTEGRATION link file: ");
gets(linkFileName);

if ((linkFile = fopen(linkFileName, "w")) == NULL)
{
    printf("\n\nINTEGRATION link file cannot be opened for writing...");
    printf("\nHit enter to quit program...");
    gets(inputLine);
    return(0);
}
else
{
    numLinks = 0;
    thisLink = linkList;
    while (thisLink) {numLinks++; thisLink = thisLink->nextLink;}
}

```

```

fputs("New INTEGRATION Link File - created by LargeGen.c",linkFile);
fprintf(linkFile,"\n%d 1.0 1.0 1.0 1.0 1.0\n",numLinks);
thisLink = linkList;
while (thisLink)
{
    fprintf(linkFile,"%4d %4d %4d",
        (thisLink->linkNum), (thisLink->startNode), (thisLink->endNode));
    fprintf(linkFile, " %5.3f %3d %5d",
        (thisLink->linkLen), (thisLink->linkFFSpeed), (thisLink->linkCap));
    fprintf(linkFile, " %d %4.1f %4.1f",
        (thisLink->linkNumLanes), (thisLink->linkPF), (thisLink->linkA));
    fprintf(linkFile, " %5.1f %5d %2d", (thisLink->linkB),
        (thisLink->turnProLink), (thisLink->turnProStart));
    fprintf(linkFile, " %5ld %4d %4d",
        (thisLink->turnProEnd), (thisLink->linkOpp1), (thisLink->linkOpp2));
    fprintf(linkFile, " %3d %ld %ld %ld %ld ",
        (thisLink->linkSignalNum), (thisLink->linkPhase1),
        (thisLink->linkPhase2), (thisLink->linkHOV), (thisLink->linkSurv));
    fputs(thisLink->linkComment,linkFile);
    fprintf(linkFile,"\n");
    thisLink = thisLink->nextLink;
}
fclose(linkFile);
fprintf(resultsFile,"\nLink file %s written successfully.",linkFileName);
return(1);
}
/* end */
}

/* assigns attribute values to a new node...*/
void assignNewNode(struct nodeStruct *newNode, struct nodeStruct *oldNode, int n)
{
    char temp[10];

    newNode->nodeNum = ++hiNode;
    newNode->nodeType = oldNode->nodeType;
    newNode->cluster = oldNode->cluster;
    newNode->infoNode = oldNode->infoNode;
    newNode->xCoord = oldNode->xCoord;
    newNode->yCoord = oldNode->yCoord;
    strcpy(newNode->nodeComment,"e");
    itoa(oldNode->nodeNum, temp, 10);
    strcat(newNode->nodeComment,temp);
    strcat(newNode->nodeComment,":");
    itoa(n, temp, 10);
    strcat(newNode->nodeComment,temp);
    newNode->nextNode=NULL;
}

/* returns the occurrence of a given node */
struct nodeStruct *findNode(int sourceNode, struct nodeStruct *nodeList)
{
    struct nodeStruct *thisNode;
    thisNode = nodelist;
    while (thisNode->nodeNum != sourceNode)
        thisNode = thisNode->nextNode;
    return(thisNode);
}

/* calculates the x and y coordinates for a new node */
void findNodeCoord(struct nodeStruct *outNode, struct nodeStruct *inNode,
    struct nodeStruct *newNode, int type)
{
    float x1, x2, y1, y2, root, dist, ratio, appDist;

    x1 = inNode->xCoord;

```

```

x2 = outNode->xCoord;
y1 = inNode->yCoord;
y2 = outNode->yCoord;

/* calculate distance between points */
dist = sqrt((x1-x2)*(x1-x2) + (y1-y2)*(y1-y2));

appDist = 0.2; /* default value for screen length of approaches */

/* see if approach distance is less than link distance */
if (appDist < dist) ratio = appDist/dist;
else ratio = 0.95;

/* adjust new node coordinates by ratio ● /
newNode->xCoord = x1 + ratio*(x2-x1);
newNode->yCoord = y1 + ratio*(y2-y1);

/* adjust them slightly based on type */
appDist = 0.05; /* adjustment distance */
if ((type < 2) || (type > 5)) appDist *= -1.0;
if (type%2 == 1) newNode->xCoord += appDist;
else newNode->yCoord += appDist;
}

/* assigns attribute values to a new link...*/
void assignNewLink(struct linkStruct *newLink, struct linkStruct *oldLinks[],
                 struct nodeStruct *newNodes[], struct approachStruct *approach,
                 struct xListStruct *thisX, int linkNumber)

struct linkStruct *refLink;
struct approachStruct *refApproach;
int nodeStartEnd[12][2] = {{0,7},{0,6},{0,5},
                          {1,4},{1,7},{1,6},
                          {2,5},{2,4},{2,7},
                          {3,6},{3,5},{3,4}};
/* start and end nodes of the 12 new links */

newLink->startNode = newNodes[nodeStartEnd[linkNumber][0]]->nodeNum;
newLink->endNode = newNodes[nodeStartEnd[linkNumber][1]]->nodeNum;

/* get the reference old link */
if (linkNumber<3) refLink = oldLinks[0]; /* westbound in */
else if (linkNumber<6) refLink = oldLinks[1]; /* southbound in */
else if (linkNumber<9) refLink = oldLinks[2]; /* eastbound in */
else refLink = oldLinks[3]; /* northbound in */

/* get the reference approach data */
if (linkNumber<3) /* westbound in */
    refApproach = findApproach(approach,thisX->westApproachNum);
else if (linkNumber<6) /* southbound in */
    refApproach = findApproach(approach,thisX->northApproachNum);
else if (linkNumber<9) /* eastbound in */
    refApproach = findApproach(approach,thisX->eastApproachNum);
else /* northbound in */
    refApproach = findApproach(approach,thisX->southApproachNum);

/* calculate other parameters */
determineLinkParams (newLink, refLink, refApproach, linkNumber);
}

/* performs calculations for parameters of created links ● /
void determineLinkParams (struct linkStruct *newLink, struct linkStruct *refLink,
                        struct approachStruct *refApproach, int linkNumber)

```

```

{
char temp[10];
int linkOpp[12] = (0,0,0,0,0,0,0,0,0,0,0,0);
int oppApp,i;

/* Note the opposing link for each movement (in terms of a
relative reference of link numbers). For example, the
opposition to link 2 (LT from the west to the north)
is the link numbered 5 higher (thru from the east to
the west */
linkOpp[2] = 5;
linkOpp[5] = 5;
linkOpp[8] = -7;
linkOpp[11] = -7;

/* Adjust positional data based on missing links */
if (noApproach[0])
{linkOpp[5] = 4; linkOpp[8]=0; linkOpp[11] =0;}
if (noApproach[1])
{linkOpp[2] = 0; linkOpp[8]=-2; linkOpp[11] =0;}
if (noApproach[2])
{linkOpp[2] = 0; linkOpp[5]=0; linkOpp[11] =-2;}
if (noApproach[3])
{linkOpp[2] = 4; linkOpp[5]=0; linkOpp[8] =0;}

/* Assign positional data */
newLink->linkNum = ++hiLink;
newLink->nextLink=NULL;
newLink->linkOpp1 = newLink->linkOpp2 = 0;

/* copy some data from relevant old link...*/
newLink->linkHOV = refLink->linkHOV;
newLink->linkSurv = refLink->linkSurv;
newLink->turnProLink = refLink->turnProLink;
newLink->turnProStart = refLink->turnProStart;
newLink->turnProEnd = refLink->turnProEnd;
newLink->linkOpp2 = 0;
newLink->linkPF = refLink->linkPF;
newLink->linkA = refLink->linkA;
newLink->linkB = refLink->linkB;

/* modify parameters of specific turning movements */
if (linkNumber%3 == 0) /* right turn */
{
newLink->linkLen = refApproach->rightLen;
newLink->linkFFSpeed = refApproach->rightFF;
newLink->linkNumLanes = ceil(refApproach->rightLane);
newLink->linkCap = (refApproach->rightCap) * refApproach->rightLane;

/* signal phasing, assuming right on red is legal */
newLink->linkSignalNum = 0;
newLink->linkPhase1 = 0;
newLink->linkPhase2 = 0;
}
else if (linkNumber%3 == 1) /* thru link */
{
newLink->linkLen = refApproach->thruLen;
newLink->linkFFSpeed = refApproach->thruFF;
newLink->linkNumLanes = ceil(refApproach->thruLane);
newLink->linkCap = (refApproach->thruCap) * refApproach->thruLane
/ newLink->linkNumLanes;

/* signal phasing */
newLink->linkSignalNum = refLink->linkSignalNum;
if (refApproach->protectedLeft)
{
newLink->linkPhase1 = refLink->linkPhase2;

```

```

        newLink->linkPhase2 = 0;
    }
    else
    {
        newLink->linkPhase1 = refLink->linkPhase1;
        newLink->linkPhase2 = refLink->linkPhase2;
    }
}
else if (linkNumber%3 == 2) /* left turn */

    newLink->linkLen = refApproach->leftLen;
    newLink->linkFFSpeed = refApproach->leftFF;
    newLink->linkNumLanes = ceil(refApproach->leftLane);
    newLink->linkCap = (refApproach->leftCap) * refApproach->leftLane
        / newLink->linkNumLanes;

    /* signal phasing */
    newLink->linkSignalNum = refLink->linkSignalNum;
    if (refApproach->protectedLeft==2) /* leading protected, permitted */
    {
        newLink->linkPhase1 = refLink->linkPhase1;
        newLink->linkPhase2 = refLink->linkPhase2;
        newLink->linkOpp1 = newLink->linkNum*(linkOpp[linkNumber] != 0) +
            linkOpp[linkNumber] ;
    }
    else if (refApproach->protectedLeft==1) /* leading only*/
    {
        newLink->linkPhase1 = refLink->linkPhase1;
        newLink->linkPhase2 = 0;
    }
    else /* permissive only */
    {
        newLink->linkPhase1 = refLink->linkPhase1;
        newLink->linkPhase2 = refLink->linkPhase2;
        newLink->linkOpp1 = newLink->linkNum*(linkOpp[linkNumber] != 0) +
            linkOpp[linkNumber] ;
    }
}
strcpy(newLink->linkComment,"Expansion of link ");
itoa(refLink->linkNum, temp, 10);
strcat(newLink->linkComment,temp) ;
strcat(newLink->linkComment,":");
if (linkNumber%3 == 0)
    strcat(newLink->linkComment,"RT");
if (linkNumber%3 == 1)
    strcat(newLink->linkComment,"T ");
if (linkNumber%3 == 2)
    strcat(newLink->linkComment,"LT");
/*    strcat(newLink->linkComment,"\n");    */
}

/* updates attribute values for exisiting links...*/
void updateOldLinks(struct linkStruct *oldLinks[], struct linkStruct *newLinks[],
    struct nodeStruct *newNodes[])
{
    struct linkStruct *refLink;
    struct approachStruct *refApproach;
    int i, oppApp;

    /* number of the new link number that took some length from old link*/
    int newLinkDeltaLen[4] = {1,4,7,10};

    for (i=0; i<4; i++)
        if (!noApproach[i])

```



```

    {
        /* update link numbers */
        oldLinks[i] ->endNode = newNodes[i] ->nodeNum;

        /* remove signal number from inbound oldLinks */
        oldLinks[i] ->linkSignalNum = 0;
        oldLinks[i] ->linkPhase1 = 0;
        oldLinks[i] ->linkPhase2 = 0;

        /* change the inbound link lengths (but not the outbound) */
        oppApp = i + 6;
        if (oppApp > 8) oppApp -=4;
        if (!noApproach[oppApp])
            oldLinks[i] ->linkLen -= newLinks[newLinkDeltaLen[i]] ->linkLen;

        /* enforce INTEGRATION minimum of 0.005 link length */
        if (oldLinks[i] ->linkLen < 0.005)
        {
            fprintf(resultsFile, "\nWARNING: original link #%d had length = %6.3f km
which was",
                    oldLinks[i] ->linkNum, oldLinks[i] ->linkLen +
newLinks[newLinkDeltaLen[i]] ->linkLen);
            fprintf(resultsFile, "\n too low for the thru link reduction factor of
%6.3f km",
                    newLinks[newLinkDeltaLen[i]] ->linkLen);
            oldLinks[i] ->linkLen = 0.005;
        }
    }

    for (i=4; i<8; i++)
        if (!noApproach[i])
            /* update link numbers for outbound links*/
            oldLinks[i] ->startNode = newNodes[i] ->nodeNum;
}

/* finds an approach in the list */
struct approachstruct *findApproach(struct approachstruct *approach, int a)
{
    struct approachstruct *thisApproach;

    thisApproach = approach;
    while ((thisApproach ->approachNum != abs(a)) && (thisApproach))
        thisApproach = thisApproach ->nextApproach;

    return(thisApproach);
}

/* counts the number of links in the list */
int countTotalLinks(struct linkStruct *links)
{
    struct linkStruct *thisLink;
    int i=0;

    thisLink = links;
    while (thisLink)

        i++;
        thisLink = thisLink ->nextLink;
    }
    return(i);
}

```

Appendix P: The Node File SCFINAL1.DAT

This is the complete node file, SCFINAL1.DAT that was used in the simulation runs. Figure A.6 shows the format of the this file.

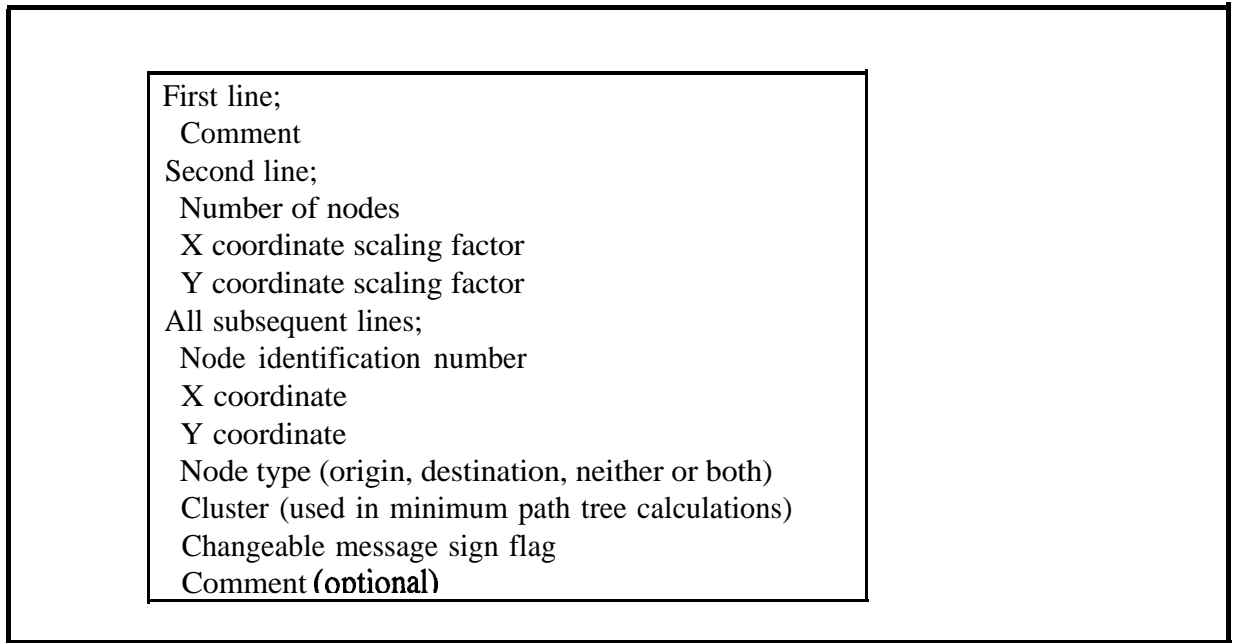


Figure A.6: Format of the INTEGRATION Node File

New INTEGRATION Node File -							
1747	1.0	1.0		64	31.946	13.009	1 -25 0 comment
	1	9.774	13.896	65	33.475	12.761	1 25 0 comment
	2	10.160	13.310	66	34.928	13.500	1 -38 0 comment
	3	11.226	11.680	67	35.142	13.918	1 -32 0 comment
	4	12.130	10.299	68	33.404	14.324	1 32 0 comment
	5	12.997	9.036	69	31.940	15.015	1 31 0 comment
	6	13.650	8.700	70	29.958	15.126	1 23 0 comment
	7	10.274	14.396	71	29.958	14.239	1 -23 0 comment
	8	10.742	14.771	72	28.028	15.129	1 24 0 comment
	9	11.623	15.478	73	25.490	14.766	1 17 0 comment
10	12.147	15.897	1	74	23.070	14.504	1 16 0 comment
11	12.654	16.303	1	75	20.369	14.302	1 13 0 comment
12	14.431	17.727	1	76	19.314	14.557	1 -13 0 comment
13	16.631	18.327	1	77	18.085	14.694	1 13 0 comment
14	19.008	18.212	1	78	18.380	13.354	1 14 0 comment
15	20.114	18.214	1	79	16.790	12.810	1 5 0 comment
16	21.959	17.936	1	80	15.546	14.017	1 -5 0 comment
17	23.610	17.769	1	81	14.592	13.998	1 5 0 comment
18	24.680	17.769	1	82	13.698	11.601	1 2 0 comment
19	26.530	17.769	1	83	13.151	11.273	1 -2 0 comment
20	28.156	17.263	1	84	12.186	13.183	1 4 0 comment
21	29.382	16.891	1	85	11.216	14.169	1 4 0 comment
22	30.273	16.886	1	86	13.342	15.741	1 8 0 comment
23	31.644	16.885	1	87	17.785	16.690	1 9 0 comment
24	32.912	16.858	1	88	19.843	15.899	1 12 0 comment
25	34.070	16.750	1	89	21.069	15.704	1 19 0 comment
26	34.690	16.691	1	90	21.957	15.704	1 -19 0 comment
27	35.540	16.266	1	91	23.425	16.304	1 21 0 comment
28	37.112	15.482	2	92	24.680	15.904	1 -21 0 comment
29	13.457	8.736	1	93	19.551	17.171	1 -12 0 comment
30	15.598	9.480	1	94	21.303	16.912	1 -39 0 comment
31	18.013	11.019	1	95	23.779	16.922	1 20 0 comment
32	18.680	11.461	1	96	26.055	16.769	1 -22 0 comment
33	20.681	12.301	1	97	27.180	16.445	1 22 0 comment
34	22.110	12.301	1	98	28.890	15.603	1 24 0 comment
35	23.080	12.285	1	99	29.723	15.822	1 -24 0 comment
36	23.886	12.283	1	100	30.794	15.573	1 24 0 comment
37	25.680	12.283	1	101	32.106	15.740	1 31 0 comment
38	26.284	12.277	1	102	33.601	15.818	1 30 0 comment
39	28.941	12.270	1	103	34.720	15.618	1 30 0 comment
40	30.256	12.270	1	104	35.604	15.052	1 -35 0 comment
41	31.572	12.270	1	105	34.360	14.168	1 32 0 comment
42	32.899	12.270	1	301	10.547	14.115	4 0 0 comment
43	34.050	11.510	1	303	11.249	14.677	4 0 0 comment
44	35.257	11.417	1	305	11.913	15.209	4 0 0 comment
45	37.612	14.982	1	307	12.420	15.615	4 0 0 comment
46	36.975	14.163	1	310	13.512	16.491	4 0 0 comment
47	36.658	13.604	1	311	14.254	17.085	4 0 0 comment
48	36.419	13.036	1	312	14.431	17.227	4 0 0 comment
49	35.757	11.917	1	313	14.681	17.577	4 0 0 comment
112	37.451	14.729	3	314	15.131	17.827	4 0 0 comment
50	13.876	10.121	1	315	15.631	17.827	4 0 0 comment
51	14.744	10.721	1	317	18.032	17.712	4 0 0 comment
52	15.120	11.285	1	319	19.508	17.712	4 0 0 comment
53	15.770	11.547	1	321	20.764	17.614	4 0 0 comment
54	17.191	11.961	1	325	22.309	17.350	4 0 0 comment
55	18.980	13.019	1	326	22.609	17.269	4 0 0 comment
56	23.081	13.650	1	327	22.909	17.269	4 0 0 comment
57	23.690	13.650	1	329	24.260	17.269	4 0 0 comment
58	25.305	13.570	1	331	25.180	17.269	4 0 0 comment
59	26.200	13.770	1	332	25.580	17.269	4 0 0 comment
60	28.294	13.891	1	334	26.930	17.070	4 0 0 comment
61	28.430	13.009	1	335	27.430	16.820	4 0 0 comment
62	29.650	13.770	1	337	28.450	16.560	4 0 0 comment
63	30.970	13.772	1	338	28.790	16.450	4 0 0 comment
				340	29.980	16.391	4 0 0 comment

342	31.000	16.386	4	0	0	comment	505	27.330	12.777	4	0	0	comment
344	31.790	16.380	4	0	0	comment	508	29.540	12.770	4	0	0	comment
345	31.940	16.370	4	0	0	comment	509	29.800	12.770	4	0	0	comment
346	32.700	16.360	4	0	0	comment	511	30.859	12.766	4	0	0	comment
348	33.312	16.300	4	0	0	comment	512	31.159	12.766	4	0	0	comment
349	33.690	16.300	4	0	0	comment	514	32.320	12.763	4	0	0	comment
352	35.182	15.945	4	0	0	comment	518	22.823	16.934	4	0	0	comment
354	36.211	15.431	4	0	0	comment	520	23.919	16.440	4	0	0	comment
355	36.721	15.177	4	0	0	comment	527	12.765	12.094	4	0	0	comment
359	11.710	14.052	4	0	0	comment	532	16.370	12.975	4	0	0	comment
361	12.428	14.550	4	0	0	comment	533	16.990	13.070	4	0	0	comment
363	12.976	14.922	4	0	0	comment	537	13.898	11.001	4	0	0	comment
365	13.567	15.328	4	0	0	comment	540	13.458	14.092	4	0	0	comment
367	14.157	15.734	4	0	0	comment	541	14.907	13.110	4	0	0	comment
370	16.220	16.740	4	0	0	comment	543	16.670	15.790	4	0	0	comment
373	17.570	16.890	4	0	0	comment	545	18.625	13.940	4	0	0	comment
375	18.600	16.890	4	0	0	comment	547	18.864	15.730	4	0	0	comment
377	19.264	16.770	4	0	0	comment	548	20.144	15.423	4	0	0	comment
378	19.564	16.678	4	0	0	comment	549	21.320	15.166	4	0	0	comment
380	20.621	16.470	4	0	0	comment	552	11.369	12.220	4	0	0	comment
382	21.712	16.199	4	0	0	comment	553	12.586	11.296	4	0	0	comment
383	22.201	16.094	4	0	0	comment	554	13.376	10.021	4	0	0	comment
384	22.592	16.010	4	0	0	comment	555	12.952	13.020	4	0	0	comment
386	23.423	15.830	4	0	0	comment	556	13.452	13.370	4	0	0	comment
389	25.050	15.600	4	0	0	comment	559	13.158	13.992	4	0	0	comment
392	26.270	15.660	4	0	0	comment	560	15.096	15.680	4	0	0	comment
396	28.028	15.480	4	0	0	comment	561	15.696	10.930	4	0	0	comment
397	28.428	15.440	4	0	0	comment	562	16.570	16.490	4	0	0	comment
399	29.405	15.403	4	0	0	comment	563	18.900	16.460	4	0	0	comment
403	31.790	15.394	4	0	0	comment	564	17.570	16.490	4	0	0	comment
404	32.426	15.394	4	0	0	comment	565	18.893	16.067	4	0	0	comment
406	33.260	15.390	4	0	0	comment	566	18.710	15.370	4	0	0	comment
407	33.640	15.390	4	0	0	comment	567	18.210	13.578	4	0	0	comment
410	34.428	15.295	4	0	0	comment	569	20.144	17.260	4	0	0	comment
412	35.216	14.856	4	0	0	comment	570	20.144	16.080	4	0	0	comment
413	35.650	14.617	4	0	0	comment	571	19.995	14.970	4	0	0	comment
414	35.960	14.448	4	0	0	comment	572	20.094	14.620	4	0	0	comment
418	14.826	9.957	4	0	0	comment	573	19.990	14.220	4	0	0	comment
426	19.141	12.859	4	0	0	comment	575	21.860	17.060	4	0	0	comment
430	21.797	14.644	4	0	0	comment	576	21.160	14.750	4	0	0	comment
431	22.437	14.757	4	0	0	comment	578	23.500	14.060	4	0	0	comment
436	25.490	15.297	4	0	0	comment	579	25.030	16.670	4	0	0	comment
439	26.873	15.126	4	0	0	comment	581	24.020	13.530	4	0	0	comment
442	28.294	14.766	4	0	0	comment	582	23.970	13.330	4	0	0	comment
444	29.450	14.739	4	0	0	comment	583	23.920	13.000	4	0	0	comment
445	29.650	14.739	4	0	0	comment	584	23.910	12.900	4	0	0	comment
450	33.307	14.840	4	0	0	comment	585	24.780	17.169	4	0	0	comment
452	34.507	14.506	4	0	0	comment	586	28.050	16.460	4	0	0	comment
454	35.380	14.026	4	0	0	comment	591	30.270	16.090	4	0	0	comment
457	15.000	9.500	4	0	0	comment	592	30.259	14.390	4	0	0	comment
460	16.415	10.507	4	0	0	comment	595	31.550	16.085	4	0	0	comment
464	17.520	11.290	4	0	0	comment	596	31.596	15.015	4	0	0	comment
467	19.018	12.206	4	0	0	comment	599	32.902	16.090	4	0	0	comment
468	19.520	12.500	4	0	0	comment	602	32.902	13.060	4	0	0	comment
472	21.970	13.650	4	0	0	comment	603	34.060	15.650	4	0	0	comment
476	25.005	14.090	4	0	0	comment	609	34.570	15.914	4	0	0	comment
477	25.600	14.090	4	0	0	comment	610	34.460	15.637	4	0	0	comment
480	27.550	14.091	4	0	0	comment	611	35.500	15.491	4	0	0	comment
482	28.430	14.091	4	0	0	comment	612	35.250	15.240	4	0	0	comment
484	29.650	14.078	4	0	0	comment	613	34.679	14.183	4	0	0	comment
488	32.171	14.075	4	0	0	comment	615	31.001	16.090	4	0	0	comment
490	33.404	14.074	4	0	0	comment	616	33.290	16.090	4	0	0	comment
492	34.244	13.918	4	0	0	comment	617	34.070	16.090	4	0	0	comment
494	35.142	13.500	4	0	0	comment	620	36.951	14.729	4	0	0	comment
499	23.380	12.786	4	0	0	comment	621	36.789	14.476	4	0	0	comment
501	24.636	12.783	4	0	0	comment	622	36.035	13.346	4	0	0	comment
502	25.080	12.783	4	0	0	comment	623	35.554	12.369	4	0	0	comment

624	33.710	13.680	4	0	0	comment	217	24.300	13.120	4	0	0	comment
625	21.260	13.029	4	0	0	comment	218	23.720	13.120	4	0	0	comment
629	17.550	12.060	4	0	0	comment	219	23.300	13.120	4	0	0	comment
630	17.600	12.250	4	0	0	comment	220	21.950	13.220	4	0	0	comment
632	15.470	12.900	4	0	0	comment	221	20.870	13.430	4	0	0	comment
633	20.046	13.301	4	0	0	comment	222	20.710	13.460	4	0	0	comment
634	10.912	13.476	4	0	0	comment	223	20.380	13.570	4	0	0	comment
635	15.546	16.399	4	0	0	comment	224	18.980	13.660	4	0	0	comment
636	15.820	16.540	4	0	0	comment	225	18.380	12.780	4	0	0	comment
638	19.860	13.390	4	0	0	comment	226	16.990	12.820	4	0	0	comment
639	19.830	13.290	4	0	0	comment	227	16.630	12.690	4	0	0	comment
640	10.716	13.843	4	0	0	comment	228	15.940	12.380	4	0	0	comment
643	20.881	16.985	4	0	0	comment	229	14.760	12.430	4	0	0	comment
644	21.910	17.248	4	0	0	comment	230	14.180	12.470	4	0	0	comment
645	29.299	16.193	4	0	0	comment	231	12.930	12.530	4	0	0	comment
646	36.226	15.110	4	0	0	comment	232	12.230	12.530	4	0	0	comment
647	17.200	16.490	4	0	0	comment	233	10.930	12.530	4	0	0	comment
648	21.959	17.686	4	0	0	comment	234	10.880	12.490	4	0	0	comment
649	22.190	17.235	4	0	0	comment	235	12.630	12.490	4	0	0	comment
651	16.631	18.077	4	0	0	comment	236	13.980	12.420	4	0	0	comment
652	24.145	13.650	4	0	0	comment	237	14.400	12.400	4	0	0	comment
653	13.537	12.655	4	0	0	comment	238	15.100	12.360	4	0	0	comment
654	14.589	17.029	4	0	0	comment	239	15.940	12.340	4	0	0	comment
655	17.745	12.050	4	0	0	comment	240	16.950	12.690	4	0	0	comment
656	17.770	12.145	4	0	0	comment	241	17.870	12.660	4	0	0	comment
657	24.140	13.530	4	0	0	comment	242	18.400	12.650	4	0	0	comment
658	24.090	13.330	4	0	0	comment	243	19.060	13.530	4	0	0	comment
659	24.040	13.000	4	0	0	comment	244	19.380	13.550	4	0	0	comment
660	24.030	12.900	4	0	0	comment	245	20.050	13.550	4	0	0	comment
661	26.730	17.170	4	0	0	comment	246	20.380	13.480	4	0	0	comment
106	10.580	12.530	2	-3	C	comment	247	20.480	13.460	4	0	0	comment
107	36.530	12.790	2	-34		comment	248	20.600	13.430	4	0	0	comment
186	11.660	12.200	4	0	0	comment	249	21.910	13.170	4	0	0	comment
187	11.350	12.930	4	0	0	comment	250	23.500	13.080	4	0	0	comment
108	34.880	12.640	2	34		comment	251	23.660	13.080	4	0	0	comment
188	35.980	13.600	4	0	0	comment	252	24.190	13.080	4	0	0	comment
113	36.610	12.930	3	0	0	comment	253	24.570	13.080	4	0	0	comment
114	10.530	12.490	3	0	0	comment	254	24.800	13.080	4	0	0	comment
189	35.780	13.600	4	0	0	comment	255	25.990	13.190	4	0	0	comment
115	35.250	12.630	3	0	0	comment	256	26.630	13.190	4	0	0	comment
190	11.800	12.080	4	0	0	comment	257	26.990	13.190	4	0	0	comment
191	11.110	12.980	4	0	0	comment	258	28.500	13.470	4	0	0	comment
192	31.470	13.570	4	0	0	comment	259	29.400	13.470	4	0	0	comment
193	31.270	13.570	4	0	0	comment	260	29.580	13.470	4	0	0	comment
194	30.130	13.590	4	0	0	comment	261	29.980	13.550	4	0	0	comment
195	29.930	13.590	4	0	0	comment	262	30.670	13.540	4	0	0	comment
196	36.035	12.980	4	0	0	comment	263	31.160	13.530	4	0	0	comment
197	35.460	13.030	4	0	0	comment	264	31.500	13.530	4	0	0	comment
198	35.060	13.040	4	0	0	comment	265	31.980	13.530	4	0	0	comment
199	34.750	13.020	4	0	0	comment	266	32.260	13.500	4	0	0	comment
200	33.910	13.210	4	0	0	comment	267	32.460	13.500	4	0	0	comment
201	33.620	13.310	4	0	0	comment	268	33.130	13.500	4	0	0	comment
202	33.370	13.540	4	0	0	comment	269	33.300	13.500	4	0	0	comment
203	32.440	13.540	4	0	0	comment	270	33.530	13.220	4	0	0	comment
204	32.140	13.570	4	0	0	comment	271	34.370	12.970	4	0	0	comment
205	32.080	13.570	4	0	0	comment	272	34.850	12.930	4	0	0	comment
206	31.980	13.570	4	0	0	comment	273	35.570	12.890	4	0	0	comment
207	31.130	13.570	4	0	0	comment	274	36.050	12.840	4	0	0	comment
208	30.860	13.590	4	0	0	comment	275	33.810	13.445	4	0	0	comment
209	29.810	13.590	4	0	0	comment	276	33.086	13.631	4	0	0	comment
210	29.400	13.510	4	0	0	comment	277	31.782	13.660	4	0	0	comment
211	28.500	13.510	4	0	0	comment	278	30.560	13.681	4	0	0	comment
212	26.990	13.230	4	0	0	comment	279	29.173	13.602	4	0	0	comment
213	26.840	13.230	4	0	0	comment	280	26.741	13.320	4	0	0	comment
214	25.990	13.230	4	0	0	comment	281	24.440	13.325	4	0	0	comment
215	24.800	13.120	4	0	0	comment	282	21.610	13.345	4	0	0	comment
216	24.630	13.120	4	0	0	comment	283	20.536	13.719	4	0	0	comment

284	18.165	12.955	4	0	0	comment	135	28.140	14.091	4	0	0	comment
285	16.300	12.740	4	0	0	comment	136	28.486	12.773	4	0	0	comment
286	14.352	12.558	4	0	0	comment	137	29.670	12.770	4	0	0	comment
287	13.945	12.687	4	0	0	comment	138	29.956	14.076	4	0	0	comment
288	12.230	12.730	4	0	0	comment	139	31.321	14.074	4	0	0	comment
289	14.382	12.245	4	0	0	comment	140	31.009	12.766	4	0	0	comment
290	20.637	13.577	4	0	0	comment	141	31.946	12.763	4	0	0	comment
291	23.540	12.920	4	0	0	comment	142	33.475	12.385	4	0	0	comment
292	26.241	13.100	4	0	0	comment	143	34.654	11.963	4	0	0	comment
293	28.723	13.380	4	0	0	comment	144	34.928	13.590	4	0	0	comment
294	29.920	13.396	4	0	0	comment	145	34.696	13.932	4	0	0	comment
295	31.372	13.440	4	0	0	comment	146	34.024	14.404	4	0	0	comment
296	32.681	13.410	4	0	0	comment	147	32.906	14.457	4	0	0	comment
297	33.795	13.030	4	0	0	comment	148	29.958	14,738	4	0	0	comment
298	35.420	13.180	4	0	0	comment	149	25.780	15.349	4	0	0	comment
299	35.120	12.728	4	0	0	comment	150	23.102	14.875	4	0	0	comment
662	33.660	13.495	4	0	0	comment	151	20.717	14.083	4	0	0	comment
663	32.671	13.631	4	0	0	comment	152	20.042	14.420	4	0	0	comment
664	31.357	13.660	4	0	0	comment	153	18.585	14.694	4	0	0	comment
665	30.035	13.681	4	0	0	comment	154	14.183	13.601	4	0	0	comment
666	28.723	13.602	4	0	0	comment	155	12.715	13.385	4	0	0	comment
667	26.241	13.320	4	0	0	comment	156	11.656	12.981	4	0	0	comment
668	24.165	13.225	4	0	0	comment	157	11.436	13.864	4	0	0	comment
669	23.690	13.325	4	0	0	comment	158	10.996	14.474	4	0	0	comment
670	20.893	13.703	4	0	0	comment	159	13.736	15.444	4	0	0	comment
671	19.415	13.715	4	0	0	comment	160	12.947	16.038	4	0	0	comment
672	17.295	12.535	4	0	0	comment	161	15.001	14.395	4	0	0	comment
673	13.859	12.640	4	0	0	comment	162	17.785	16.890	4	0	0	comment
674	12.030	12.230	4	0	0	comment	163	19.258	17.712	4	0	0	comment
675	11.950	12.330	4	0	0	comment	164	19.843	16.630	4	0	0	comment
676	12.330	12.350	4	0	0	comment	165	18.879	15.899	4	0	0	comment
677	14.059	12.313	4	0	0	comment	166	21.537	17.468	4	0	0	comment
678	15.705	12.620	4	0	0	comment	167	21.069	16.356	4	0	0	comment
679	17.870	12.550	4	0	0	comment	168	21.418	15.704	4	0	0	comment
680	19.605	13.420	4	0	0	comment	169	21.957	16.147	4	0	0	comment
681	20.276	13.353	4	0	0	comment	170	23.935	17.269	4	0	0	comment
682	21.585	13.100	4	0	0	comment	171	23.622	16.575	4	0	0	comment
683	23.760	12.970	4	0	0	comment	172	24.267	15.916	4	0	0	comment
684	24.390	12.990	4	0	0	comment	173	26.055	17.269	4	0	0	comment
685	26.741	13.100	4	0	0	comment	174	27.180	16.945	4	0	0	comment
686	29.173	13.380	4	0	0	comment	175	29.132	15.798	4	0	0	comment
687	30.465	13.431	4	0	0	comment	176	30.272	15.747	4	0	0	comment
688	31.782	13.440	4	0	0	comment	177	32.906	15.740	4	0	0	comment
689	33.016	13.411	4	0	0	comment	178	34.065	15.870	4	0	0	comment
690	34.215	12.905	4	0	0	comment	179	35.375	15.366	4	0	0	comment
116	10.501	13.861	3	0	0	comment	180	31.315	15.399	4	0	0	comment
109	10.768	13.999	2	-10	0	comment	181	35.433	14.737	4	0	0	comment
110	12.266	11.276	2	-33	0	comment	182	18.080	13.354	4	0	0	comment
117	12.416	11.326	3	0	0	comment	183	12.963	12.153	4	0	0	comment
118	36.061	15.637	3	0	0	comment	184	12.819	10.886	4	0	0	comment
111	36.361	15.637	2	-36	0	comment	185	28.228	15.460	4	0	0	comment
119	13.603	9.654	4	0	0	comment	691	17.250	12.375	4	0	0	comment
120	14.279	11.243	4	0	0	comment	692	19.445	13.410	4	0	0	comment
121	15.024	10.906	4	0	0	comment	693	19.955	13.470	4	0	0	comment
122	14.891	11.631	4	0	0	comment	694	24.115	13.040	4	0	0	comment
123	15.445	11.936	4	0	0	comment	695	23.845	13.225	4	0	0	comment
124	16.132	10.831	4	0	0	comment	696	12.580	12.765	4	0	0	comment
125	17.566	11.794	4	0	0	comment	641	35.730	13.245	4	0	0	comment
126	16.170	12.883	4	0	0	comment	642	11.790	12.730	4	0	0	comment
127	16.680	13.023	4	0	0	comment	697	14.492	11.328	4	0	0	e538:0
128	18.685	12.552	4	0	0	comment	698	14.454	11.609	4	0	0	e538:1
129	23.038	14.133	4	0	0	comment	699	14.830	11.642	4	0	0	e538:2
130	23.290	13.422	4	0	0	comment	700	14.817	11.316	4	0	0	e538:3
131	23.855	13.915	4	0	0	comment	701	14.492	11.428	4	0	0	e538:4
132	25.303	14.090	4	0	0	comment	702	14.554	11.609	4	0	0	e538:5
133	25.840	13.437	4	0	0	comment	703	14.830	11.542	4	0	0	e538:6
134	26.698	13.750	4	0	0	comment	704	14.717	11.316	4	0	0	e538:7

705	17.773	11.880	4	0	0	e424:0	772	26.545	13.580	4	0	0	e587:1
706	17.705	12.050	4	0	0	e424:1	773	26.679	13.392	4	0	0	e587:2
707	18.103	12.206	4	0	0	e424:2	774	26.541	13.209	4	0	0	e587:3
708	18.018	11.842	4	0	0	e424:3	775	26.303	13.392	4	0	0	e587:4
709	17.773	11.980	4	0	0	e424:4	776	26.645	13.580	4	0	0	e587:5
710	17.828	12.140	4	0	0	e424:5	777	26.441	13.209	4	0	0	e587:7
711	18.103	12.106	4	0	0	e424:6	778	12.554	14.507	4	0	0	e362:0
712	17.918	11.842	4	0	0	e424:7	779	12.586	14.730	4	0	0	e362:1
713	32.712	16.310	4	0	0	e347:0	780	12.860	14.788	4	0	0	e362:2
714	32.862	16.558	4	0	0	e347:1	781	12.919	14.405	4	0	0	e362:3
715	33.110	16.379	4	0	0	e347:2	782	12.554	14.607	4	0	0	e362:4
716	32.955	16.158	4	0	0	e347:3	783	12.686	14.730	4	0	0	e362:5
717	32.712	16.410	4	0	0	e347:4	784	12.860	14.688	4	0	0	e362:6
718	32.962	16.558	4	0	0	e347:5	785	12.819	14.405	4	0	0	e362:7
719	33.110	16.279	4	0	0	e347:6	786	13.663	12.416	4	0	0	e631:1
720	32.855	16.158	4	0	0	e347:7	787	13.932	12.247	4	0	0	e631:3
721	21.760	17.401	4	0	0	e323:0	788	13.763	12.416	4	0	0	e631:5
722	21.909	17.636	4	0	0	e323:1	789	14.026	12.255	4	0	0	e631:6
723	22.157	17.455	4	0	0	e323:2	790	12.969	12.105	4	0	0	e528:0
724	21.962	17.257	4	0	0	e323:3	791	13.061	12.406	4	0	0	e528:1
725	21.760	17.501	4	0	0	e323:4	792	13.259	12.437	4	0	0	e528:2
726	22.009	17.636	4	0	0	e323:5	793	13.411	12.199	4	0	0	e528:3
727	22.157	17.355	4	0	0	e323:6	794	12.969	12.205	4	0	0	e528:4
728	21.862	17.257	4	0	0	e323:7	795	13.161	12.406	4	0	0	e528:5
729	33.882	12.069	4	0	0	e516:0	796	13.276	12.325	4	0	0	e528:6
730	34.007	12.210	4	0	0	e516:1	797	13.305	12.162	4	0	0	e528:7
731	34.249	12.044	4	0	0	e516:2	798	13.664	9.126	4	0	0	e417:0
732	34.100	11.810	4	0	0	e516:3	799	13.675	9.458	4	0	0	e417:1
733	33.882	12.169	4	0	0	e516:4	800	13.996	9.450	4	0	0	e417:2
734	34.107	12.210	4	0	0	e516:5	801	13.936	9.096	4	0	0	e417:3
735	34.249	11.944	4	0	0	e516:6	802	13.664	9.226	4	0	0	e417:4
736	34.000	11.810	4	0	0	e516:7	803	13.775	9.458	4	0	0	e417:5
737	11.960	11.718	4	0	0	e525:0	804	13.996	9.350	4	0	0	e417:6
738	12.333	11.962	4	0	0	e525:2	805	13.836	9.096	4	0	0	e417:7
739	12.290	11.661	4	0	0	e525:3	806	17.913	12.670	4	0	0	e628:1
740	11.960	11.818	4	0	0	e525:4	807	18.225	12.330	4	0	0	e628:3
741	12.333	11.862	4	0	0	e525:6	808	17.875	12.599	4	0	0	e628:4
742	13.431	12.595	4	0	0	e557:0	809	18.013	12.670	4	0	0	e628:5
743	13.457	13.013	4	0	0	e557:1	810	18.125	12.330	4	0	0	e628:7
744	13.728	12.805	4	0	0	e557:2	811	17.551	12.976	4	0	0	e534:0
745	13.587	12.663	4	0	0	e557:3	812	17.872	13.113	4	0	0	e534:1
746	13.431	12.695	4	0	0	e557:4	813	17.872	12.902	4	0	0	e534:2
747	13.557	13.013	4	0	0	e557:5	814	17.761	12.814	4	0	0	e534:3
748	13.713	12.669	4	0	0	e557:6	815	17.551	13.076	4	0	0	e534:4
749	13.487	12.663	4	0	0	e557:7	816	17.971	13.113	4	0	0	e534:5
750	13.739	15.396	4	0	0	e366:0	817	17.836	12.780	4	0	0	e534:6
751	13.718	15.706	4	0	0	e366:1	818	17.661	12.814	4	0	0	e534:7
752	14.069	15.723	4	0	0	e366:2	819	18.318	14.108	4	0	0	e544:0
753	13.739	15.496	4	0	0	e366:4	820	18.446	14.214	4	0	0	e544:1
754	13.818	15.706	4	0	0	e366:5	821	18.617	13.994	4	0	0	e544:2
755	14.069	15.623	4	0	0	e366:6	822	18.411	13.843	4	0	0	e544:3
756	13.690	12.077	4	0	0	e558:0	823	18.318	14.208	4	0	0	e544:4
757	13.731	12.265	4	0	0	e558:1	824	18.546	14.214	4	0	0	e544:5
758	14.087	12.145	4	0	0	e558:2	825	18.617	13.894	4	0	0	e544:6
759	14.099	11.977	4	0	0	e558:3	826	18.311	13.843	4	0	0	e544:7
760	13.690	12.177	4	0	0	e558:4	827	25.467	15.589	4	0	0	e390:0
761	13.885	12.247	4	0	0	e558:5	828	25.434	15.740	4	0	0	e390:1
762	14.087	12.045	4	0	0	e558:6	829	25.865	15.728	4	0	0	e390:2
763	13.989	11.965	4	0	0	e558:7	830	25.893	15.564	4	0	0	e390:3
764	25.873	15.315	4	0	0	e437:0	831	25.467	15.689	4	0	0	e390:4
765	25.851	15.507	4	0	0	e437:1	832	25.534	15.740	4	0	0	e390:5
766	26.259	15.385	4	0	0	e437:2	833	25.865	15.628	4	0	0	e390:6
767	26.052	15.212	4	0	0	e437:3	834	25.772	15.531	4	0	0	e390:7
768	25.873	15.415	4	0	0	e437:4	835	11.733	11.633	4	0	0	e524:0
769	25.951	15.507	4	0	0	e437:5	836	11.955	11.864	4	0	0	e524:2
770	26.259	15.285	4	0	0	e437:6	837	11.733	11.733	4	0	0	e524:4
771	25.952	15.212	4	0	0	e437:7	838	11.955	11.669	4	0	0	e524:6

839	11.948	11.583	4	0	0	e524:7	906	15.775	12.883	4	0	0	e542:4
840	16.480	16.777	4	0	0	e371:0	907	16.201	12.874	4	0	0	e542:5
841	16.612	17.090	4	0	0	e371:1	908	15.865	12.598	4	0	0	e542:7
842	16.870	16.940	4	0	0	e371:2	909	35.982	13.649	4	0	0	e455:0
843	16.671	16.696	4	0	0	e371:3	910	36.207	13.778	4	0	0	e455:1
844	16.480	16.877	4	0	0	e371:4	911	36.358	13.654	4	0	0	e455:2
845	16.712	17.090	4	0	0	e371:5	912	36.122	13.423	4	0	0	e455:3
846	16.870	16.840	4	0	0	e371:6	913	35.982	13.749	4	0	0	e455:4
847	16.571	16.696	4	0	0	e371:7	914	36.307	13.778	4	0	0	e455:5
848	29.183	16.361	4	0	0	e339:0	915	36.358	13.554	4	0	0	e455:6
849	29.332	16.591	4	0	0	e339:1	916	36.022	13.423	4	0	0	e455:7
850	29.582	16.441	4	0	0	e339:2	917	36.300	14.210	4	0	0	e415:0
851	29.355	16.207	4	0	0	e339:3	918	36.567	14.304	4	0	0	e415:1
852	29.183	16.461	4	0	0	e339:4	919	36.675	14.213	4	0	0	e415:2
853	29.432	16.591	4	0	0	e339:5	920	36.445	13.980	4	0	0	e415:3
854	29.582	16.341	4	0	0	e339:6	921	36.300	14.310	4	0	0	e415:4
855	29.255	16.207	4	0	C	e339:7	922	36.667	14.304	4	0	0	e415:5
856	20.924	17.512	4	0	0	e322:0	923	36.675	14.113	4	0	0	e415:6
857	21.313	17.535	4	0	0	e322:2	924	36.311	13.999	4	0	0	e415:7
858	21.082	17.318	4	0	0	e322:3	925	36.933	15.021	4	0	0	e356:0
859	20.924	17.612	4	0	0	e322:4	926	37.312	15.032	4	0	0	e356:2
860	21.313	17.435	4	0	0	e322:6	927	37.055	14.813	4	0	0	e356:3
861	20.982	17.318	4	0	0	e322:7	928	36.933	15.121	4	0	0	e356:4
862	35.686	15.344	4	0	0	e618:0	929	37.162	15.182	4	0	0	e356:5
863	36.040	15.258	4	0	0	e618:2	930	37.312	14.932	4	0	0	e356:6
864	35.686	15.444	4	0	0	e618:4	931	36.955	14.813	4	0	0	e356:7
865	36.040	15.158	4	0	0	e618:6	932	18.000	13.303	4	0	0	e568:1
866	35.803	15.101	4	0	0	e618:7	933	18.105	13.054	4	0	0	e568:2
867	36.412	14.963	4	0	0	e619:0	934	18.020	13.114	4	0	0	e568:3
868	36.766	14.877	4	0	0	e619:2	935	18.100	13.303	4	0	0	e568:5
869	36.412	15.063	4	0	0	e619:4	936	17.825	13.114	4	0	0	e568:7
870	36.578	15.111	4	0	0	e619:5	937	15.590	11.957	4	0	0	e531:0
871	36.766	14.777	4	0	0	e619:6	938	15.520	12.095	4	0	0	e531:1
872	33.897	12.907	4	0	0	e608:0	939	15.807	12.342	4	0	0	e531:2
873	34.010	13.040	4	0	0	e608:1	940	15.590	12.057	4	0	0	e531:4
874	34.110	12.650	4	0	0	e608:3	941	15.620	12.095	4	0	0	e531:5
875	34.110	13.040	4	0	0	e608:5	942	15.842	12.232	4	0	0	e531:6
876	34.207	12.852	4	0	0	e608:6	943	13.800	8.650	4	0	0	e456:0
877	34.010	12.650	4	0	0	e608:7	944	13.873	8.885	4	0	0	e456:1
878	32.716	13.346	4	0	0	e601:0	945	14.156	8.875	4	0	0	e456:2
879	32.852	13.521	4	0	0	e601:1	946	13.800	8.750	4	0	0	e456:4
880	32.952	13.121	4	0	0	e601:3	947	14.018	8.897	4	0	0	e456:5
881	32.952	13.521	4	0	0	e601:5	948	14.156	8.775	4	0	0	e456:6
882	33.010	13.356	4	0	0	e601:6	949	15.443	9.808	4	0	0	e458:0
883	32.852	13.121	4	0	C	e601:7	950	15.443	10.154	4	0	0	e458:1
884	32.853	13.921	4	0	0	e600:1	951	15.750	10.164	4	0	0	e458:2
885	33.082	13.683	4	0	0	e600:2	952	15.648	9.784	4	0	0	e458:3
886	33.001	13.527	4	0	0	e600:3	953	15.443	9.908	4	0	0	e458:4
887	32.716	13.698	4	0	0	e600:4	954	15.543	10.154	4	0	0	e458:5
888	32.953	13.921	4	0	0	e600:5	955	15.750	10.064	4	0	0	e458:6
889	32.803	13.527	4	0	0	e600:7	956	15.548	9.784	4	0	0	e458:7
890	31.400	13.378	4	0	0	e598:0	957	15.695	10.483	4	0	0	e420:0
891	31.534	13.550	4	0	0	e598:1	958	15.711	10.818	4	0	0	e420:1
892	31.630	13.150	4	0	0	e598:3	959	16.026	10.808	4	0	0	e420:2
893	31.634	13.550	4	0	0	e598:5	960	16.003	10.468	4	0	0	e420:3
894	31.766	13.383	4	0	0	e598:6	961	15.695	10.583	4	0	0	e420:4
895	31.530	13.150	4	0	0	e598:7	962	15.811	10.818	4	0	0	e420:5
896	31.526	13.950	4	0	0	e597:1	963	16.026	10.708	4	0	0	e420:6
897	31.766	13.717	4	0	0	e597:2	964	15.903	10.468	4	0	0	e420:7
898	31.683	13.556	4	0	0	e597:3	965	15.880	10.150	4	0	0	e459:0
899	31.398	13.726	4	0	0	e597:4	966	15.859	10.461	4	0	0	e459:1
900	31.626	13.950	4	0	0	e597:5	967	16.203	10.470	4	0	0	e459:2
901	31.485	13.556	4	0	0	e597:7	968	15.852	10.283	4	0	0	e459:4
902	15.775	12.783	4	0	0	e542:0	969	16.054	10.461	4	0	0	e459:5
903	16.101	12.874	4	0	0	e542:1	970	16.203	10.370	4	0	0	e459:6
904	16.168	12.810	4	0	0	e542:2	971	16.642	10.653	4	0	0	e462:0
905	15.965	12.598	4	0	0	e542:3	972	16.659	11.015	4	0	0	e462:1

973	16.944	11.015	4	0	0	e462:2	1040	31.522	12.563	4	0	0	e513:7
974	16.642	10.753	4	0	0	e462:4	1041	32.699	12.712	4	0	0	e515:0
975	16.759	11.015	4	0	0	e462:5	1042	32.851	12.961	4	0	0	e515:1
976	16.944	10.915	4	0	0	e462:6	1043	33.066	12.702	4	0	0	e515:2
977	16.453	10.490	4	0	0	e461:0	1044	32.949	12.561	4	0	0	e515:3
978	16.453	10.844	4	0	0	e461:1	1045	32.699	12.812	4	0	0	e515:4
979	16.640	10.799	4	0	0	e461:2	1046	32.951	12.961	4	0	0	e515:5
980	16.453	10.590	4	0	0	e461:4	1047	33.066	12.602	4	0	0	e515:6
981	16.553	10.844	4	0	0	e461:5	1048	32.849	12.561	4	0	0	e515:7
982	16.640	10.604	4	0	0	e461:6	1049	35.058	11.882	4	0	0	e517:0
983	16.949	10.873	4	0	0	e463:0	1050	35.317	12.084	4	0	0	e517:1
984	17.060	11.257	4	0	0	e463:1	1051	35.457	11.967	4	0	0	e517:2
985	17.237	11.202	4	0	0	e463:2	1052	35.307	11.717	4	0	0	e517:3
986	16.949	11.068	4	0	0	e463:4	1053	35.058	11.982	4	0	0	e517:4
987	17.160	11.257	4	0	0	e463:5	1054	35.417	12.084	4	0	0	e517:5
988	17.237	11.102	4	0	0	e463:6	1055	35.457	11.867	4	0	0	e517:6
989	17.832	11.385	4	0	0	e465:0	1056	35.207	11.717	4	0	0	e517:7
990	17.906	11.711	4	0	0	e465:1	1057	33.963	12.650	4	0	0	e606:1
991	18.180	11.679	4	0	0	e465:2	1058	34.259	12.709	4	0	0	e606:2
992	18.063	11.319	4	0	0	e465:3	1059	34.110	12.440	4	0	0	e606:3
993	17.832	11.485	4	0	0	e465:4	1060	34.158	12.650	4	0	0	e606:5
994	18.066	11.719	4	0	0	e465:5	1061	34.259	12.609	4	0	0	e606:6
995	18.180	11.579	4	0	0	e465:6	1062	34.010	12.440	4	0	0	e606:7
996	17.963	11.319	4	0	0	e465:7	1063	33.963	12.433	4	0	0	e607:1
997	20.156	12.599	4	0	0	e469:0	1064	34.193	12.489	4	0	0	e607:2
998	20.132	12.801	4	0	0	e469:1	1065	34.155	12.214	4	0	0	e607:3
999	20.528	12.838	4	0	0	e469:2	1066	34.158	12.433	4	0	0	e607:5
1000	20.529	12.542	4	0	0	e469:3	1067	34.193	12.389	4	0	0	e607:6
1001	20.156	12.699	4	0	0	e469:4	1068	33.960	12.214	4	0	0	e607:7
1002	20.232	12.801	4	0	0	e469:5	1069	34.266	12.562	4	0	0	e614:0
1003	20.528	12.738	4	0	0	e469:6	1070	34.409	12.863	4	0	0	e614:1
1004	20.429	12.542	4	0	0	e469:7	1071	34.332	12.511	4	0	0	e614:3
1005	20.529	12.691	4	0	0	e470:0	1072	34.266	12.757	4	0	0	e614:4
1006	20.729	12.939	4	0	0	e470:2	1073	34.509	12.863	4	0	0	e614:5
1007	20.800	12.801	4	0	0	e470:3	1074	34.200	12.540	4	0	0	e614:7
1008	20.529	12.886	4	0	0	e470:4	1075	11.978	14.111	4	0	0	e360:0
1009	20.729	12.839	4	0	0	e470:6	1076	11.995	14.399	4	0	0	e360:1
1010	20.700	12.801	4	0	0	e470:7	1077	12.292	14.440	4	0	0	e360:2
1011	22.880	12.739	4	0	0	e498:0	1078	12.322	14.069	4	0	0	e360:3
1012	23.093	12.976	4	0	0	e498:1	1079	11.978	14.211	4	0	0	e360:4
1013	23.280	12.836	4	0	0	e498:2	1080	12.095	14.399	4	0	0	e360:5
1014	23.130	12.586	4	0	0	e498:3	1081	12.292	14.340	4	0	0	e360:6
1015	22.880	12.839	4	0	0	e498:4	1082	12.222	14.069	4	0	0	e360:7
1016	23.193	12.976	4	0	0	e498:5	1083	13.084	16.097	4	0	0	e309:0
1017	23.280	12.736	4	0	0	e498:6	1084	13.396	16.447	4	0	0	e309:2
1018	23.030	12.586	4	0	0	e498:7	1085	13.433	16.132	4	0	0	e309:3
1019	25.480	12.733	4	0	0	e503:0	1086	13.084	16.197	4	0	0	e309:4
1020	25.678	12.977	4	0	0	e503:1	1087	13.396	16.347	4	0	0	e309:6
1021	25.880	12.831	4	0	0	e503:2	1088	13.319	16.119	4	0	0	e309:7
1022	25.730	12.583	4	0	0	e503:3	1089	15.051	16.079	4	0	0	e369:0
1023	25.480	12.833	4	0	0	e503:4	1090	15.044	16.394	4	0	0	e369:1
1024	25.778	12.977	4	0	0	e503:5	1091	15.398	16.376	4	0	0	e369:2
1025	25.880	12.731	4	0	0	e503:6	1092	15.051	16.179	4	0	0	e369:4
1026	25.630	12.583	4	0	0	e503:7	1093	15.144	16.394	4	0	0	e369:5
1027	27.830	12.727	4	0	0	e506:0	1094	15.398	16.276	4	0	0	e369:6
1028	27.953	12.975	4	0	0	e506:1	1095	14.710	15.905	4	0	0	e368:0
1029	28.230	12.825	4	0	0	e506:2	1096	15.040	16.218	4	0	0	e368:2
1030	27.830	12.827	4	0	0	e506:4	1097	15.045	15.855	4	0	0	e368:3
1031	28.053	12.975	4	0	0	e506:5	1098	14.710	16.005	4	0	0	e368:4
1032	28.230	12.725	4	0	0	e506:6	1099	15.043	16.026	4	0	0	e368:6
1033	31.372	12.714	4	0	0	e513:0	1100	14.945	15.855	4	0	0	e368:7
1034	31.500	12.962	4	0	0	e513:1	1101	14.821	12.045	4	0	0	e530:0
1035	31.772	12.813	4	0	0	e513:2	1102	14.949	12.294	4	0	0	e530:1
1036	31.622	12.563	4	0	0	e513:3	1103	15.221	12.145	4	0	0	e530:2
1037	31.372	12.814	4	0	0	e513:4	1104	15.130	11.904	4	0	0	e530:3
1038	31.652	12.961	4	0	0	e513:5	1105	14.821	12.145	4	0	0	e530:4
1039	31.772	12.713	4	0	0	e513:6	1106	15.049	12.294	4	0	0	e530:5

1107	15.221	12.045	4	0	0	e530:6	1174	23.764	14.531	4	0	0	e474:5
1108	15.030	11.904	4	0	0	e530:7	1175	23.806	14.274	4	0	0	e474:6
1109	14.951	11.619	4	0	0	e539:0	1176	23.491	14.173	4	0	0	e474:7
1110	14.985	11.898	4	0	0	e539:1	1177	24.025	13.796	4	0	0	e580:0
1111	15.300	11.915	4	0	0	e539:2	1178	24.241	13.953	4	0	0	e580:1
1112	15.282	11.612	4	0	0	e539:3	1179	24.198	13.656	4	0	0	e580:3
1113	14.951	11.719	4	0	0	e539:4	1180	24.025	13.896	4	0	0	e580:4
1114	15.180	11.898	4	0	0	e539:5	1181	24.341	13.953	4	0	0	e580:5
1115	15.300	11.815	4	0	0	e539:6	1182	24.098	13.656	4	0	0	e580:7
1116	15.182	11.612	4	0	0	e539:7	1183	26.330	17.219	4	0	0	e333:0
1117	17.002	16.810	4	0	0	e372:0	1184	26.480	17.469	4	0	0	e333:1
1118	17.068	17.073	4	0	0	e372:1	1185	26.709	17.230	4	0	0	e333:2
1119	17.400	16.940	4	0	0	e372:2	1186	26.524	17.077	4	0	0	e333:3
1120	17.250	16.690	4	0	0	e372:3	1187	26.330	17.319	4	0	0	e333:4
1121	17.002	16.970	4	0	0	e372:4	1188	26.580	17.469	4	0	0	e333:5
1122	17.168	17.073	4	0	0	e372:5	1189	26.709	17.130	4	0	0	e333:6
1123	17.400	16.840	4	0	0	e372:6	1190	26.424	17.077	4	0	0	e333:7
1124	17.150	16.690	4	0	0	e372:7	1191	16.431	17.777	4	0	0	e316:0
1125	17.800	16.840	4	0	0	e374:0	1192	16.581	18.027	4	0	0	e316:1
1126	17.958	17.090	4	0	0	e374:1	1193	16.774	17.877	4	0	0	e316:2
1127	18.200	16.940	4	0	0	e374:2	1194	16.703	17.628	4	0	0	e316:3
1128	17.800	16.940	4	0	0	e374:4	1195	16.431	17.877	4	0	0	e316:4
1129	18.058	17.090	4	0	0	e374:5	1196	16.681	18.027	4	0	0	e316:5
1130	18.200	16.840	4	0	0	e374:6	1197	16.774	17.777	4	0	0	e316:6
1131	18.808	17.662	4	0	0	e318:0	1198	16.576	17.627	4	0	0	e316:7
1132	18.958	17.912	4	0	0	e318:1	1199	16.774	17.730	4	0	0	e650:0
1133	19.208	17.762	4	0	0	e318:2	1200	16.628	17.998	4	0	0	e650:1
1134	19.048	17.512	4	0	0	e318:3	1201	16.980	17.859	4	0	0	e650:2
1135	18.808	17.762	4	0	0	e318:4	1202	16.922	17.649	4	0	0	e650:3
1136	19.058	17.912	4	0	0	e318:5	1203	16.774	17.924	4	0	0	e650:4
1137	19.208	17.662	4	0	0	e318:6	1204	16.728	17.998	4	0	0	e650:5
1138	18.948	17.512	4	0	0	e318:7	1205	16.980	17.759	4	0	0	e650:6
1139	23.274	14.855	4	0	0	e433:0	1206	16.802	17.640	4	0	0	e650:7
1140	23.410	15.140	4	0	0	e433:1	1207	22.157	17.308	4	0	0	e324:0
1141	23.662	15.048	4	0	0	e433:2	1208	21.993	17.568	4	0	0	e324:1
1142	23.274	14.955	4	0	0	e433:4	1209	22.302	17.403	4	0	0	e324:2
1143	23.510	15.140	4	0	0	e433:5	1210	22.238	17.243	4	0	0	e324:3
1144	23.662	14.948	4	0	0	e433:6	1211	22.157	17.503	4	0	0	e324:4
1145	24.192	16.270	4	0	0	e521:0	1212	22.093	17.568	4	0	0	e324:5
1146	24.405	16.423	4	0	0	e521:1	1213	22.302	17.303	4	0	0	e324:6
1147	24.558	16.209	4	0	0	e521:2	1214	22.138	17.243	4	0	0	e324:7
1148	24.362	16.050	4	0	0	e521:3	1215	25.881	15.584	4	0	0	e391:0
1149	24.192	16.370	4	0	0	e521:4	1216	26.070	15.894	4	0	0	e391:1
1150	24.505	16.423	4	0	0	e521:5	1217	26.266	15.711	4	0	0	e391:2
1151	24.558	16.109	4	0	0	e521:6	1218	25.872	15.777	4	0	0	e391:4
1152	24.262	16.050	4	0	0	e521:7	1219	26.183	15.890	4	0	0	e391:5
1153	23.968	15.603	4	0	0	e387:0	1220	26.266	15.611	4	0	0	e391:6
1154	24.172	15.781	4	0	0	e387:1	1221	26.517	15.577	4	0	0	e393:0
1155	24.358	15.632	4	0	0	e387:2	1222	26.694	15.798	4	0	0	e393:1
1156	24.146	15.401	4	0	0	e387:3	1223	26.914	15.634	4	0	0	e393:2
1157	23.968	15.703	4	0	0	e387:4	1224	26.679	15.419	4	0	0	e393:3
1158	24.272	15.781	4	0	0	e387:5	1225	26.517	15.677	4	0	0	e393:4
1159	24.358	15.532	4	0	0	e387:6	1226	26.794	15.798	4	0	0	e393:5
1160	24.046	15.401	4	0	0	e387:7	1227	26.914	15.534	4	0	0	e393:6
1161	23.816	14.967	4	0	0	e434:0	1228	26.579	15.419	4	0	0	e393:7
1162	23.980	15.297	4	0	0	e434:1	1229	27.725	14.781	4	0	0	e441:0
1163	24.197	15.166	4	0	0	e434:2	1230	28.114	14.816	4	0	0	e441:2
1164	23.955	14.922	4	0	0	e434:3	1231	27.945	14.567	4	0	0	e441:3
1165	23.800	15.119	4	0	0	e434:4	1232	27.725	14.881	4	0	0	e441:4
1166	24.136	15.279	4	0	0	e434:5	1233	28.114	14.716	4	0	0	e441:6
1167	24.197	15.066	4	0	0	e434:6	1234	27.845	14.567	4	0	0	e441:7
1168	23.855	14.922	4	0	0	e434:7	1235	27.650	14.041	4	0	0	e481:0
1169	23.423	14.237	4	0	0	e474:0	1236	27.798	14.291	4	0	0	e481:1
1170	23.639	14.544	4	0	0	e474:1	1237	28.050	14.141	4	0	0	e481:2
1171	23.806	14.374	4	0	0	e474:2	1238	27.936	13.894	4	0	0	e481:3
1172	23.591	14.173	4	0	0	e474:3	1239	27.650	14.141	4	0	0	e481:4
1173	23.423	14.337	4	0	0	e474:4	1240	27.939	14.287	4	0	0	e481:5

1241	28.050	14.041	4	0	0	e481:6	1308	35.783	12.955	4	0	0	e495:7
1242	27.818	13.892	4	0	0	e481:7	1309	11.467	14.803	4	0	0	e304:0
1243	30.803	15.353	4	0	0	e401:0	1310	11.573	15.178	4	0	0	e304:1
1244	30.952	15.603	4	0	0	e401:1	1311	11.779	15.153	4	0	0	e304:2
1245	31.203	15.450	4	0	0	e401:2	1312	11.799	14.823	4	0	0	e304:3
1246	31.042	15.203	4	0	0	e401:3	1313	11.467	14.903	4	0	0	e304:4
1247	30.803	15.453	4	0	0	e401:4	1314	11.673	15.178	4	0	0	e304:5
1248	31.052	15.603	4	0	0	e401:5	1315	11.779	15.053	4	0	0	e304:6
1249	31.203	15.350	4	0	0	e401:6	1316	11.681	14.810	4	0	0	e304:7
1250	30.942	15.203	4	0	0	e401:7	1317	30.211	13.972	4	0	0	e593:1
1251	30.765	14.687	4	0	0	e447:0	1318	30.450	13.764	4	0	0	e593:2
1252	30.905	14.937	4	0	0	e447:1	1319	30.309	13.572	4	0	0	e593:3
1253	31.165	14.795	4	0	0	e447:2	1320	30.074	13.747	4	0	0	e593:4
1254	31.046	14.539	4	0	0	e447:3	1321	30.311	13.972	4	0	0	e593:5
1255	30.765	14.787	4	0	0	e447:4	1322	30.209	13.572	4	0	0	e593:7
1256	31.047	14.934	4	0	0	e447:5	1323	14.451	12.027	4	0	0	e627:0
1257	31.165	14.695	4	0	0	e447:6	1324	14.425	12.193	4	0	0	e627:1
1258	30.946	14.539	4	0	0	e447:7	1325	14.812	12.193	4	0	0	e627:2
1259	30.870	14.023	4	0	0	e486:0	1326	14.451	12.163	4	0	0	e627:4
1260	30.969	14.266	4	0	0	e486:1	1327	14.812	11.997	4	0	0	e627:6
1261	31.270	14.124	4	0	0	e486:2	1328	10.312	13.758	4	0	0	e551:1
1262	30.870	14.123	4	0	0	e486:4	1329	10.493	13.839	4	0	0	e551:2
1263	31.110	14.273	4	0	0	e486:5	1330	10.632	13.425	4	0	0	e551:3
1264	31.270	14.024	4	0	0	e486:6	1331	10.412	13.758	4	0	0	e551:5
1265	33.860	13.639	4	0	0	e605:0	1332	10.532	13.425	4	0	0	e551:7
1266	33.989	13.899	4	0	0	e605:1	1333	30.064	13.315	4	0	0	e594:0
1267	34.125	13.501	4	0	0	e605:3	1334	30.170	13.518	4	0	0	e594:1
1268	33.860	13.739	4	0	0	e605:4	1335	30.308	13.122	4	0	0	e594:3
1269	34.089	13.899	4	0	0	e605:5	1336	30.348	13.518	4	0	0	e594:5
1270	33.995	13.501	4	0	0	e605:7	1337	30.436	13.366	4	0	0	e594:6
1271	33.969	15.046	4	0	0	e604:1	1338	30.208	13.122	4	0	0	e594:7
1272	34.101	15.080	4	0	0	e604:2	1339	28.760	13.316	4	0	0	e590:0
1273	34.074	14.773	4	0	0	e604:3	1340	28.896	13.493	4	0	0	e590:1
1274	34.069	15.046	4	0	0	e604:5	1341	28.994	13.093	4	0	0	e590:3
1275	34.101	14.980	4	0	0	e604:6	1342	28.996	13.493	4	0	0	e590:5
1276	33.974	14.773	4	0	0	e604:7	1343	29.133	13.315	4	0	0	e590:6
1277	33.824	14.000	4	0	0	e491:0	1344	28.894	13.093	4	0	0	e590:7
1278	33.974	14.238	4	0	0	e491:1	1345	28.898	13.893	4	0	0	e589:1
1279	34.200	13.992	4	0	0	e491:2	1346	29.132	13.669	4	0	0	e589:2
1280	34.136	13.906	4	0	0	e491:3	1347	29.045	13.499	4	0	0	e589:3
1281	33.824	14.100	4	0	0	e491:4	1348	28.761	13.667	4	0	0	e589:4
1282	34.074	14.238	4	0	0	e491:5	1349	28.998	13.893	4	0	0	e589:5
1283	34.200	13.892	4	0	0	e491:6	1350	28.847	13.499	4	0	0	e589:7
1284	33.941	13.906	4	0	0	e491:7	1351	26.303	13.027	4	0	0	e588:0
1285	34.535	13.721	4	0	0	e493:0	1352	26.392	13.203	4	0	0	e588:1
1286	34.650	13.880	4	0	0	e493:1	1353	26.408	12.860	4	0	0	e588:3
1287	34.897	13.653	4	0	0	e493:2	1354	26.590	13.203	4	0	0	e588:5
1288	34.715	13.486	4	0	0	e493:3	1355	26.679	13.027	4	0	0	e588:6
1289	34.535	13.821	4	0	0	e493:4	1356	26.308	12.860	4	0	0	e588:7
1290	34.750	13.880	4	0	0	e493:5	1357	21.442	13.529	4	0	0	e626:0
1291	34.897	13.553	4	0	0	e493:6	1358	21.739	13.559	4	0	0	e626:1
1292	34.593	13.493	4	0	0	e493:7	1359	21.610	13.401	4	0	0	e626:2
1293	35.361	15.805	4	0	0	e353:0	1360	21.536	13.313	4	0	0	e626:3
1294	35.490	15.966	4	0	0	e353:1	1361	21.442	13.629	4	0	0	e626:4
1295	35.719	15.727	4	0	0	e353:2	1362	21.839	13.559	4	0	0	e626:5
1296	35.561	15.568	4	0	0	e353:3	1363	21.436	13.313	4	0	0	e626:7
1297	35.361	15.905	4	0	0	e353:4	1364	20.963	13.650	4	0	0	e577:0
1298	35.590	15.966	4	0	0	e353:5	1365	21.068	13.967	4	0	0	e577:1
1299	35.719	15.627	4	0	0	e353:6	1366	21.330	13.733	4	0	0	e577:2
1300	35.461	15.568	4	0	0	e353:7	1367	21.163	13.573	4	0	0	e577:3
1301	35.738	13.171	4	0	0	e495:0	1368	20.956	13.770	4	0	0	e577:4
1302	35.966	13.311	4	0	0	e495:1	1369	21.168	13.967	4	0	0	e577:5
1303	36.115	13.147	4	0	0	e495:2	1370	21.304	13.593	4	0	0	e577:6
1304	35.883	12.955	4	0	0	e495:3	1371	21.063	13.573	4	0	0	e577:7
1305	35.738	13.271	4	0	0	e495:4	1372	19.761	13.966	4	0	0	e574:1
1306	36.066	13.311	4	0	0	e495:5	1373	19.905	13.570	4	0	0	e574:3
1307	36.115	13.047	4	0	0	e495:6	1374	19.652	13.795	4	0	0	e574:4

1375	19.861	13.966	4	0	0	e574:5	1442	34.748	14.560	4	0	0	e453:1
1376	19.805	13.570	4	0	0	e574:7	1443	34.945	14.316	4	0	0	e453:2
1377	10.460	13.260	4	0	0	e357:0	1444	34.729	14.184	4	0	0	e453:3
1378	10.489	13.419	4	0	0	e357:1	1445	34.595	14.508	4	0	0	e453:4
1379	10.827	13.470	4	0	0	e357:2	1446	34.848	14.560	4	0	0	e453:5
1380	10.819	13.142	4	0	0	e357:3	1447	34.945	14.216	4	0	0	e453:6
1381	10.460	13.360	4	0	0	e357:4	1448	34.629	14.184	4	0	0	e453:7
1382	10.684	13.419	4	0	0	e357:5	1449	30.065	14.688	4	0	0	e446:0
1383	10.827	13.370	4	0	0	e357:6	1450	30.217	14.937	4	0	0	e446:1
1384	10.719	13.142	4	0	0	e357:7	1451	30.464	14.807	4	0	0	e446:2
1385	11.991	15.222	4	0	0	e306:0	1452	30.312	14.537	4	0	0	e446:3
1386	12.097	15.597	4	0	0	e306:1	1453	30.065	14.788	4	0	0	e446:4
1387	12.303	15.572	4	0	0	e306:2	1454	30.317	14.937	4	0	0	e446:5
1388	12.323	15.241	4	0	0	e306:3	1455	30.464	14.667	4	0	0	e446:6
1389	11.991	15.322	4	0	0	e306:4	1456	30.212	14.537	4	0	0	e446:7
1390	12.197	15.597	4	0	0	e306:5	1457	20.896	14.145	4	0	0	e429:0
1391	12.303	15.472	4	0	0	e306:6	1458	21.056	14.498	4	0	0	e429:1
1392	12.207	15.230	4	0	0	e306:7	1459	21.246	14.437	4	0	0	e429:2
1393	12.498	15.628	4	0	0	e308:0	1460	21.174	14.111	4	0	0	e429:3
1394	12.604	16.003	4	0	0	e308:1	1461	20.896	14.245	4	0	0	e429:4
1395	12.810	15.978	4	0	0	e308:2	1462	21.156	14.498	4	0	0	e429:5
1396	12.839	15.655	4	0	0	e308:3	1463	21.246	14.337	4	0	0	e429:6
1397	12.498	15.728	4	0	0	e308:4	1464	21.017	14.102	4	0	0	e429:7
1398	12.704	16.003	4	0	0	e308:5	1465	16.237	10.854	4	0	0	e421:0
1399	12.810	15.878	4	0	0	e308:6	1466	16.568	11.178	4	0	0	e421:1
1400	12.739	15.655	4	0	0	e308:7	1467	16.583	10.866	4	0	0	e421:3
1401	10.074	13.846	4	0	0	e300:0	1468	16.237	10.954	4	0	0	e421:4
1402	10.224	14.096	4	0	0	e300:1	1469	16.568	11.078	4	0	0	e421:6
1403	10.430	14.071	4	0	0	e300:2	1470	16.401	10.852	4	0	0	e421:7
1404	10.455	13.765	4	0	0	e300:3	1471	16.572	11.033	4	0	0	e422:0
1405	10.074	13.946	4	0	0	e300:4	1472	16.801	11.335	4	0	0	e422:2
1406	10.324	14.096	4	0	0	e300:5	1473	16.808	11.016	4	0	0	e422:3
1407	10.430	13.971	4	0	0	e300:6	1474	16.572	11.228	4	0	0	e422:4
1408	10.260	13.765	4	0	0	e300:7	1475	16.801	11.235	4	0	0	e422:6
1409	21.910	12.751	4	0	0	e497:0	1476	16.607	11.023	4	0	0	e422:7
1410	22.121	12.991	4	0	0	e497:1	1477	17.035	11.372	4	0	0	e423:0
1411	22.310	12.861	4	0	0	e497:2	1478	17.358	11.707	4	0	0	e423:2
1412	22.160	12.601	4	0	0	e497:3	1479	17.220	11.348	4	0	0	e423:3
1413	21.910	12.851	4	0	0	e497:4	1480	17.015	11.501	4	0	0	e423:4
1414	22.221	12.991	4	0	0	e497:5	1481	17.358	11.607	4	0	0	e423:6
1415	22.309	12.735	4	0	0	e497:6	1482	17.059	11.365	4	0	0	e423:7
1416	22.060	12.601	4	0	0	e497:7	1483	13.257	8.986	4	0	0	e416:0
1417	25.800	14.040	4	0	0	e478:0	1484	13.297	9.203	4	0	0	e416:1
1418	26.030	14.273	4	0	0	e478:1	1485	13.604	9.221	4	0	0	e416:2
1419	26.200	14.140	4	0	0	e478:2	1486	13.507	8.836	4	0	0	e416:3
1420	26.002	13.896	4	0	0	e478:3	1487	13.257	9.086	4	0	0	e416:4
1421	25.800	14.140	4	0	0	e478:4	1488	13.397	9.203	4	0	0	e416:5
1422	26.141	14.268	4	0	0	e478:5	1489	13.640	9.066	4	0	0	e416:6
1423	26.200	14.040	4	0	0	e478:6	1490	13.407	8.836	4	0	0	e416:7
1424	25.902	13.896	4	0	0	e478:7	1491	19.606	13.932	4	0	0	e546:0
1425	22.288	13.764	4	0	0	e473:0	1492	19.881	14.146	4	0	0	e546:1
1426	22.472	14.097	4	0	0	e473:1	1493	20.001	13.996	4	0	0	e546:2
1427	22.652	14.029	4	0	0	e473:2	1494	19.908	13.967	4	0	0	e546:3
1428	22.464	13.712	4	0	0	e473:3	1495	19.606	14.032	4	0	0	e546:4
1429	22.288	13.864	4	0	0	e473:4	1496	19.981	14.146	4	0	0	e546:5
1430	22.572	14.097	4	0	0	e473:5	1497	20.001	13.896	4	0	0	e546:6
1431	22.652	13.929	4	0	0	e473:6	1498	19.713	13.967	4	0	0	e546:7
1432	22.345	13.718	4	0	0	e473:7	1499	12.430	10.249	4	0	0	e535:0
1433	20.844	12.867	4	0	0	e471:0	1500	12.470	10.466	4	0	0	e535:1
1434	20.979	13.228	4	0	0	e471:1	1501	12.815	10.426	4	0	0	e535:2
1435	21.210	13.079	4	0	0	e471:2	1502	12.795	10.135	4	0	0	e535:3
1436	20.962	12.855	4	0	0	e471:3	1503	12.430	10.349	4	0	0	e535:4
1437	20.820	13.018	4	0	0	e471:4	1504	12.570	10.466	4	0	0	e535:5
1438	21.114	13.222	4	0	0	e471:5	1505	12.815	10.326	4	0	0	e535:6
1439	21.210	12.979	4	0	0	e471:6	1506	12.684	10.128	4	0	0	e535:7
1440	20.862	12.855	4	0	0	e471:7	1507	12.336	11.816	4	0	0	e526:0
1441	34.595	14.408	4	0	0	e453:0	1508	12.231	12.101	4	0	0	e526:1

1509	12.581	12.066	4	0	0	e526:2	1576	20.164	17.914	4	0	0	e320:5
1510	12.503	11.746	4	0	0	e526:3	1577	20.312	17.634	4	0	0	e320:6
1511	12.336	12.011	4	0	0	e526:4	1578	20.077	17.514	4	0	0	e320:7
1512	12.331	12.101	4	0	0	e526:5	1579	30.073	16.339	4	0	0	e341:0
1513	12.581	11.966	4	0	0	e526:6	1580	30.223	16.586	4	0	0	e341:1
1514	12.403	11.746	4	0	0	e526:7	1581	30.473	16.436	4	0	0	e341:2
1515	34.692	15.100	4	0	0	e411:0	1582	30.321	16.186	4	0	0	e341:3
1516	34.997	15.141	4	0	0	e411:1	1583	30.073	16.439	4	0	0	e341:4
1517	35.041	15.005	4	0	0	e411:2	1584	30.323	16.586	4	0	0	e341:5
1518	34.909	14.853	4	0	0	e411:3	1585	30.473	16.336	4	0	0	e341:6
1519	34.692	15.200	4	0	0	e411:4	1586	30.221	16.186	4	0	0	e341:7
1520	35.097	15.141	4	0	0	e411:5	1587	27.378	15.481	4	0	0	e395:0
1521	35.041	14.905	4	0	0	e411:6	1588	27.619	15.708	4	0	0	e395:1
1522	34.770	14.859	4	0	0	e411:7	1589	27.777	15.558	4	0	0	e395:2
1523	34.053	15.340	4	0	0	e409:0	1590	27.543	15.349	4	0	0	e395:3
1524	34.332	15.545	4	0	0	e409:1	1591	27.380	15.611	4	0	0	e395:4
1525	34.428	15.345	4	0	0	e409:2	1592	27.719	15.708	4	0	0	e395:5
1526	34.234	15.204	4	0	0	e409:3	1593	27.777	15.458	4	0	0	e395:6
1527	34.053	15.440	4	0	0	e409:4	1594	27.443	15.349	4	0	0	e395:7
1528	34.432	15.545	4	0	0	e409:5	1595	18.765	16.828	4	0	0	e376:0
1529	34.428	15.245	4	0	0	e409:6	1596	18.909	17.064	4	0	0	e376:1
1530	34.115	15.212	4	0	0	e409:7	1597	19.155	16.854	4	0	0	e376:2
1531	28.765	15.367	4	0	0	e398:0	1598	18.983	16.666	4	0	0	e376:3
1532	28.993	15.587	4	0	0	e398:1	1599	18.765	16.928	4	0	0	e376:4
1533	29.165	15.453	4	0	0	e398:2	1600	19.040	17.062	4	0	0	e376:5
1534	29.009	15.203	4	0	0	e398:3	1601	19.155	16.754	4	0	0	e376:6
1535	28.765	15.467	4	0	0	e398:4	1602	18.883	16.666	4	0	0	e376:7
1536	29.093	15.587	4	0	0	e398:5	1603	19.924	16.566	4	0	0	e379:0
1537	29.165	15.353	4	0	0	e398:6	1604	20.078	16.782	4	0	0	e379:1
1538	28.909	15.203	4	0	0	e398:7	1605	20.316	16.588	4	0	0	e379:2
1539	24.749	15.998	4	0	0	e522:0	1606	20.180	16.382	4	0	0	e379:3
1540	24.916	16.186	4	0	0	e522:1	1607	19.924	16.666	4	0	0	e379:4
1541	25.125	15.961	4	0	0	e522:2	1608	20.178	16.782	4	0	0	e379:5
1542	24.949	15.792	4	0	0	e522:3	1609	20.316	16.488	4	0	0	e379:6
1543	24.767	16.138	4	0	0	e522:4	1610	20.080	16.382	4	0	0	e379:7
1544	25.016	16.186	4	0	0	e522:5	1611	24.651	15.498	4	0	0	e388:0
1545	25.119	15.848	4	0	0	e522:6	1612	24.800	15.759	4	0	0	e388:1
1546	24.849	15.792	4	0	0	e522:7	1613	25.047	15.649	4	0	0	e388:2
1547	33.872	16.226	4	0	0	e350:0	1614	24.831	15.372	4	0	0	e388:3
1548	34.020	16.450	4	0	0	e350:1	1615	24.653	15.638	4	0	0	e388:4
1549	34.269	16.281	4	0	0	e350:2	1616	24.979	15.743	4	0	0	e388:5
1550	34.120	16.098	4	0	0	e350:3	1617	25.047	15.549	4	0	0	e388:6
1551	33.872	16.326	4	0	0	e350:4	1618	24.731	15.372	4	0	0	e388:7
1552	34.120	16.450	4	0	0	e350:5	1619	30.073	15.353	4	0	0	e400:0
1553	34.269	16.181	4	0	0	e350:6	1620	30.222	15.603	4	0	0	e400:1
1554	34.020	16.098	4	0	0	e350:7	1621	30.472	15.472	4	0	0	e400:2
1555	27.957	16.729	4	0	0	e336:0	1622	30.342	15.204	4	0	0	e400:3
1556	28.106	16.963	4	0	0	e336:1	1623	30.073	15.453	4	0	0	e400:4
1557	28.321	16.699	4	0	0	e336:2	1624	30.322	15.603	4	0	0	e400:5
1558	28.140	16.574	4	0	0	e336:3	1625	30.472	15.334	4	0	0	e400:6
1559	27.957	16.829	4	0	0	e336:4	1626	30.199	15.204	4	0	0	e400:7
1560	28.206	16.963	4	0	0	e336:5	1627	11.526	11.630	4	0	0	e523:0
1561	28.321	16.599	4	0	0	e336:6	1628	11.566	11.847	4	0	0	e523:1
1562	28.040	16.574	4	0	0	e336:7	1629	11.733	11.780	4	0	0	e523:2
1563	24.480	17.219	4	0	0	e330:0	1630	11.890	11.516	4	0	0	e523:3
1564	24.630	17.469	4	0	0	e330:1	1631	11.526	11.730	4	0	0	e523:4
1565	24.880	17.319	4	0	0	e330:2	1632	11.666	11.847	4	0	0	e523:5
1566	24.825	17.174	4	0	0	e330:3	1633	11.733	11.585	4	0	0	e523:6
1567	24.480	17.319	4	0	0	e330:4	1634	11.781	11.510	4	0	0	e523:7
1568	24.730	17.469	4	0	0	e330:5	1635	12.882	10.318	4	0	0	e536:0
1569	24.880	17.219	4	0	0	e330:6	1636	12.903	10.649	4	0	0	e536:1
1570	24.725	17.174	4	0	0	e330:7	1637	13.221	10.631	4	0	0	e536:2
1571	19.914	17.663	4	0	0	e320:0	1638	13.217	10.312	4	0	0	e536:3
1572	20.064	17.914	4	0	0	e320:1	1639	12.855	10.485	4	0	0	e536:4
1573	20.312	17.734	4	0	0	e320:2	1640	13.003	10.649	4	0	0	e536:5
1574	20.177	17.514	4	0	0	e320:3	1641	13.221	10.531	4	0	0	e536:6
1575	19.914	17.763	4	0	0	e320:4	1642	13.117	10.312	4	0	0	e536:7

1643	18.109	12.063	4	0	0	e425:0	1710	23.936	12.583	4	0	0	e500:3
1644	18.080	12.326	4	0	0	e425:1	1711	23.686	12.834	4	0	0	e500:4
1645	18.394	12.407	4	0	0	e425:2	1712	24.073	12.894	4	0	0	e500:5
1646	18.447	12.139	4	0	0	e425:3	1713	24.086	12.733	4	0	0	e500:6
1647	18.109	12.258	4	0	0	e425:4	1714	23.836	12.583	4	0	0	e500:7
1648	18.275	12.326	4	0	0	e425:5	1715	26.085	12.704	4	0	0	e504:0
1649	18.394	12.307	4	0	0	e425:6	1716	26.257	12.856	4	0	0	e504:1
1650	18.347	12.139	4	0	0	e425:7	1717	26.484	12.827	4	0	0	e504:2
1651	27.127	14.987	4	0	0	e440:0	1718	26.334	12.577	4	0	0	e504:3
1652	27.330	15.161	4	0	0	e440:1	1719	26.086	12.854	4	0	0	e504:4
1653	27.511	14.978	4	0	0	e440:2	1720	26.452	12.856	4	0	0	e504:5
1654	27.281	14.790	4	0	0	e440:3	1721	26.484	12.727	4	0	0	e504:6
1655	27.127	15.087	4	0	0	e440:4	1722	26.234	12.577	4	0	0	e504:7
1656	27.469	15.142	4	0	0	e440:5	1723	11.006	13.501	4	0	0	e358:0
1657	27.497	14.837	4	0	0	e440:6	1724	10.997	13.839	4	0	0	e358:1
1658	27.181	14.790	4	0	C	e440:7	1725	11.327	13.839	4	0	0	e358:2
1659	31.370	14.679	4	0	0	e448:0	1726	11.328	13.513	4	0	0	e358:3
1660	31.540	14.960	4	0	0	e448:1	1727	11.006	13.601	4	0	0	e358:4
1661	31.767	14.823	4	0	0	e448:2	1728	11.097	13.839	4	0	0	e358:5
1662	31.618	14.561	4	0	0	e448:3	1729	11.327	13.739	4	0	0	e358:6
1663	31.368	14.828	4	0	0	e448:4	1730	11.228	13.513	4	0	0	e358:7
1664	31.640	14.960	4	0	0	e448:5	1731	23.410	17.219	4	0	0	e328:0
1665	31.767	14.723	4	0	0	e448:6	1732	23.560	17.469	4	0	0	e328:1
1666	31.518	14.561	4	0	0	e448:7	1733	23.810	17.319	4	0	0	e328:2
1667	32.708	14.770	4	0	0	e449:0	1734	23.569	17.091	4	0	0	e328:3
1668	32.858	15.040	4	0	0	e449:1	1735	23.410	17.319	4	0	0	e328:4
1669	33.107	14.890	4	0	0	e449:2	1736	23.660	17.469	4	0	0	e328:5
1670	32.956	14.640	4	0	0	e449:3	1737	23.810	17.219	4	0	0	e328:6
1671	32.707	14.887	4	0	0	e449:4	1738	23.469	17.091	4	0	0	e328:7
1672	32.958	15.040	4	0	0	e449:5	1739	31.444	16.335	4	0	0	e343:0
1673	33.107	14.790	4	0	0	e449:6	1740	31.594	16.585	4	0	0	e343:1
1674	32.856	14.640	4	0	0	e449:7	1741	31.783	16.430	4	0	0	e343:2
1675	33.825	14.739	4	0	0	e451:0	1742	31.634	16.194	4	0	0	e343:3
1676	33.926	14.773	4	0	0	e451:1	1743	31.444	16.435	4	0	0	e343:4
1677	34.200	14.723	4	0	0	e451:2	1744	31.694	16.585	4	0	0	e343:5
1678	34.074	14.569	4	0	0	e451:3	1745	31.783	16.330	4	0	0	e343:6
1679	33.825	14.839	4	0	0	e451:4	1746	31.534	16.194	4	0	0	e343:7
1680	34.121	14.773	4	0	0	e451:5	1747	34.490	16.136	4	0	0	e351:0
1681	34.200	14.623	4	0	0	e451:6	1748	34.640	16.391	4	0	0	e351:1
1682	33.974	14.569	4	0	0	e451:7	1749	34.869	16.152	4	0	0	e351:2
1683	24.206	14.183	4	0	0	e475:0	1750	34.660	16.007	4	0	0	e351:3
1684	24.415	14.403	4	0	0	e475:1	1751	34.494	16.283	4	0	0	e351:4
1685	24.601	14.222	4	0	0	e475:2	1752	34.740	16.391	4	0	0	e351:5
1686	24.407	14.018	4	0	0	e475:3	1753	34.869	16.052	4	0	0	e351:6
1687	24.212	14.314	4	0	0	e475:4	1754	34.560	16.007	4	0	0	e351:7
1688	24.515	14.403	4	0	0	e475:5	1755	23.142	16.741	4	0	0	e519:0
1689	24.601	14.122	4	0	0	e475:6	1756	23.346	16.897	4	0	0	e519:1
1690	24.248	14.043	4	0	0	e475:7	1757	23.507	16.677	4	0	0	e519:2
1691	30.062	14.025	4	0	0	e485:0	1758	23.317	16.519	4	0	0	e519:3
1692	30.210	14.273	4	0	0	e485:1	1759	23.142	16.841	4	0	0	e519:4
1693	30.461	14.139	4	0	0	e485:2	1760	23.483	16.878	4	0	0	e519:5
1694	30.359	13.977	4	0	0	e485:3	1761	23.507	16.577	4	0	0	e519:6
1695	30.062	14.125	4	0	0	e485:4	1762	23.217	16.519	4	0	0	e519:7
1696	30.310	14.273	4	0	0	e485:5	1763	21.322	16.241	4	0	0	e381:0
1697	30.461	14.007	4	0	0	e485:6	1764	21.544	16.426	4	0	0	e381:1
1698	30.164	13.977	4	0	0	e485:7	1765	21.711	16.249	4	0	0	e381:2
1699	31.371	14.024	4	0	0	e487:0	1766	21.530	16.045	4	0	0	e381:3
1700	31.499	14.274	4	0	0	e487:1	1767	21.322	16.341	4	0	0	e381:4
1701	31.771	14.125	4	0	0	e487:2	1768	21.644	16.426	4	0	0	e381:5
1702	31.673	13.956	4	0	0	e487:3	1769	21.711	16.149	4	0	0	e381:6
1703	31.371	14.124	4	0	0	e487:4	1770	21.430	16.045	4	0	0	e381:7
1704	31.640	14.274	4	0	0	e487:5	1771	22.886	15.896	4	0	0	e385:0
1705	31.771	14.025	4	0	0	e487:6	1772	23.074	16.099	4	0	0	e385:1
1706	31.478	13.956	4	0	0	e487:7	1773	23.276	15.912	4	0	0	e385:2
1707	23.686	12.734	4	0	0	e500:0	1774	23.070	15.713	4	0	0	e385:3
1708	23.859	12.894	4	0	0	e500:1	1775	22.886	15.996	4	0	0	e385:4
1709	24.086	12.833	4	0	0	e500:2	1776	23.203	16.091	4	0	0	e385:5

1777	23.276	15.812	4	0	0	e385:6	1844	26.928	14.277	4	0	0	e479:1
1778	22.970	15.713	4	0	0	e385:7	1845	27.104	14.141	4	0	0	e479:2
1779	31.426	15.347	4	0	0	e402:0	1846	26.851	13.920	4	0	0	e479:3
1780	31.554	15.593	4	0	0	e402:1	1847	26.704	14.155	4	0	0	e479:4
1781	31.782	15.444	4	0	0	e402:2	1848	27.049	14.267	4	0	0	e479:5
1782	31.660	15.195	4	0	0	e402:3	1849	27.104	14.041	4	0	0	e479:6
1783	31.426	15.447	4	0	0	e402:4	1850	26.751	13.920	4	0	0	e479:7
1784	31.654	15.593	4	0	0	e402:5	1851	28.750	14.033	4	0	0	e483:0
1785	31.782	15.344	4	0	0	e402:6	1852	28.877	14.277	4	0	0	e483:1
1786	31.560	15.195	4	0	0	e402:7	1853	29.150	14.128	4	0	0	e483:2
1787	32.709	15.342	4	0	0	e405:0	1854	29.046	13.902	4	0	0	e483:3
1788	32.857	15.590	4	0	0	e405:1	1855	28.750	14.133	4	0	0	e483:4
1789	33.109	15.440	4	0	0	e405:2	1856	29.020	14.277	4	0	0	e483:5
1790	32.987	15.192	4	0	0	e405:3	1857	29.150	14.028	4	0	0	e483:6
1791	32.709	15.442	4	0	0	e405:4	1858	28.851	13.902	4	0	0	e483:7
1792	32.957	15.590	4	0	0	e405:5	1859	32.704	14.024	4	0	0	e489:0
1793	33.109	15.340	4	0	0	e405:6	1860	32.855	14.274	4	0	0	e489:1
1794	32.830	15.192	4	0	0	e405:7	1861	33.104	14.124	4	0	0	e489:2
1795	33.810	15.340	4	0	0	e408:0	1862	33.001	13.929	4	0	0	e489:3
1796	33.998	15.586	4	0	0	e408:1	1863	32.704	14.124	4	0	0	e489:4
1797	34.051	15.488	4	0	0	e408:2	1864	32.955	14.274	4	0	0	e489:5
1798	34.094	15.193	4	0	0	e408:3	1865	33.104	14.024	4	0	0	e489:6
1799	33.810	15.440	4	0	0	e408:4	1866	32.806	13.929	4	0	0	e489:7
1800	34.098	15.586	4	0	0	e408:5	1867	20.709	12.751	4	0	0	e496:0
1801	34.051	15.293	4	0	0	e408:6	1868	20.812	12.852	4	0	0	e496:1
1802	33.936	15.191	4	0	0	e408:7	1869	21.081	12.861	4	0	0	e496:2
1803	15.220	10.166	4	0	0	e419:0	1870	20.857	12.615	4	0	0	e496:3
1804	15.231	10.496	4	0	0	e419:1	1871	20.804	12.851	4	0	0	e496:4
1805	15.540	10.504	4	0	0	e419:2	1872	21.007	12.852	4	0	0	e496:5
1806	15.571	10.178	4	0	0	e419:3	1873	21.081	12.741	4	0	0	e496:6
1807	15.220	10.266	4	0	0	e419:4	1874	20.757	12.615	4	0	0	e496:7
1808	15.331	10.496	4	0	0	e419:5	1875	28.741	12.721	4	0	0	e507:0
1809	15.565	10.367	4	0	0	e419:6	1876	28.862	12.968	4	0	0	e507:1
1810	15.390	10.163	4	0	0	e419:7	1877	29.141	12.820	4	0	0	e507:2
1811	19.640	12.924	4	0	0	e427:0	1878	28.991	12.570	4	0	0	e507:3
1812	19.781	13.219	4	0	0	e427:1	1879	28.741	12.821	4	0	0	e507:4
1813	19.955	13.229	4	0	0	e427:2	1880	29.023	12.967	4	0	0	e507:5
1814	20.060	12.923	4	0	0	e427:3	1881	29.141	12.720	4	0	0	e507:6
1815	19.640	13.024	4	0	0	e427:4	1882	28.891	12.570	4	0	0	e507:7
1816	19.881	13.219	4	0	0	e427:5	1883	30.056	12.718	4	0	0	e510:0
1817	19.955	13.129	4	0	0	e427:6	1884	30.179	12.964	4	0	0	e510:1
1818	19.946	12.901	4	0	0	e427:7	1885	30.456	12.816	4	0	0	e510:2
1819	22.535	14.725	4	0	0	e432:0	1886	30.306	12.566	4	0	0	e510:3
1820	22.733	15.003	4	0	0	e432:1	1887	30.056	12.818	4	0	0	e510:4
1821	22.929	14.895	4	0	0	e432:2	1888	30.335	12.964	4	0	0	e510:5
1822	22.738	14.615	4	0	0	e432:3	1889	30.456	12.716	4	0	0	e510:6
1823	22.535	14.825	4	0	0	e432:4	1890	30.206	12.566	4	0	0	e510:7
1824	22.852	14.997	4	0	0	e432:5	1891	10.586	14.096	4	0	0	e302:0
1825	22.929	14.795	4	0	0	e432:6	1892	10.692	14.471	4	0	0	e302:1
1826	22.614	14.622	4	0	0	e432:7	1893	10.898	14.446	4	0	0	e302:2
1827	24.505	15.073	4	0	0	e435:0	1894	10.919	14.116	4	0	0	e302:3
1828	24.679	15.358	4	0	0	e435:1	1895	10.586	14.196	4	0	0	e302:4
1829	24.899	15.244	4	0	0	e435:2	1896	10.792	14.471	4	0	0	e302:5
1830	24.704	14.966	4	0	0	e435:3	1897	10.898	14.346	4	0	0	e302:6
1831	24.502	15.212	4	0	0	e435:4	1898	10.794	14.099	4	0	0	e302:7
1832	24.856	15.331	4	0	0	e435:5	1899	13.107	14.962	4	0	0	e364:0
1833	24.899	15.144	4	0	0	e435:6	1900	13.080	15.266	4	0	0	e364:1
1834	24.581	14.973	4	0	0	e435:7	1901	13.437	15.288	4	0	0	e364:2
1835	28.746	14.697	4	0	0	e443:0	1902	13.107	15.062	4	0	0	e364:4
1836	28.880	14.938	4	0	0	e443:1	1903	13.195	15.280	4	0	0	e364:5
1837	29.146	14.789	4	0	0	e443:2	1904	13.437	15.188	4	0	0	e364:6
1838	28.997	14.539	4	0	0	e443:3	1905	18.521	11.790	4	0	0	e466:0
1839	28.746	14.797	4	0	0	e443:4	1906	18.454	12.055	4	0	0	e466:1
1840	29.023	14.937	4	0	0	e443:5	1907	18.842	12.128	4	0	0	e466:2
1841	29.146	14.689	4	0	0	e443:6	1908	18.730	11.761	4	0	0	e466:3
1842	28.897	14.539	4	0	0	e443:7	1909	18.506	11.913	4	0	0	e466:4
1843	26.705	14.027	4	0	0	e479:0	1910	18.571	12.082	4	0	0	e466:5

1911	18.842	12.028	4	0	0	e466:6
1912	18.630	11.761	4	0	0	e466:7
1913	20.170	13.832	4	0	0	e428:0
1914	20.488	13.971	4	0	0	e428:1
1915	20.520	13.783	4	0	0	e428:2
1916	20.319	13.691	4	0	0	e428:3
1917	20.181	13.982	4	0	0	e428:4
1918	20.588	13.971	4	0	0	e428:5
1919	20.219	13.691	4	0	0	e428:7

Appendix Q: The Link File SCFINAL2.DAT

This is the complete link file that was used in the final simulation runs. The total number of links in the complete expanded network was 3,286. Figure A.7 provides the format of this file.

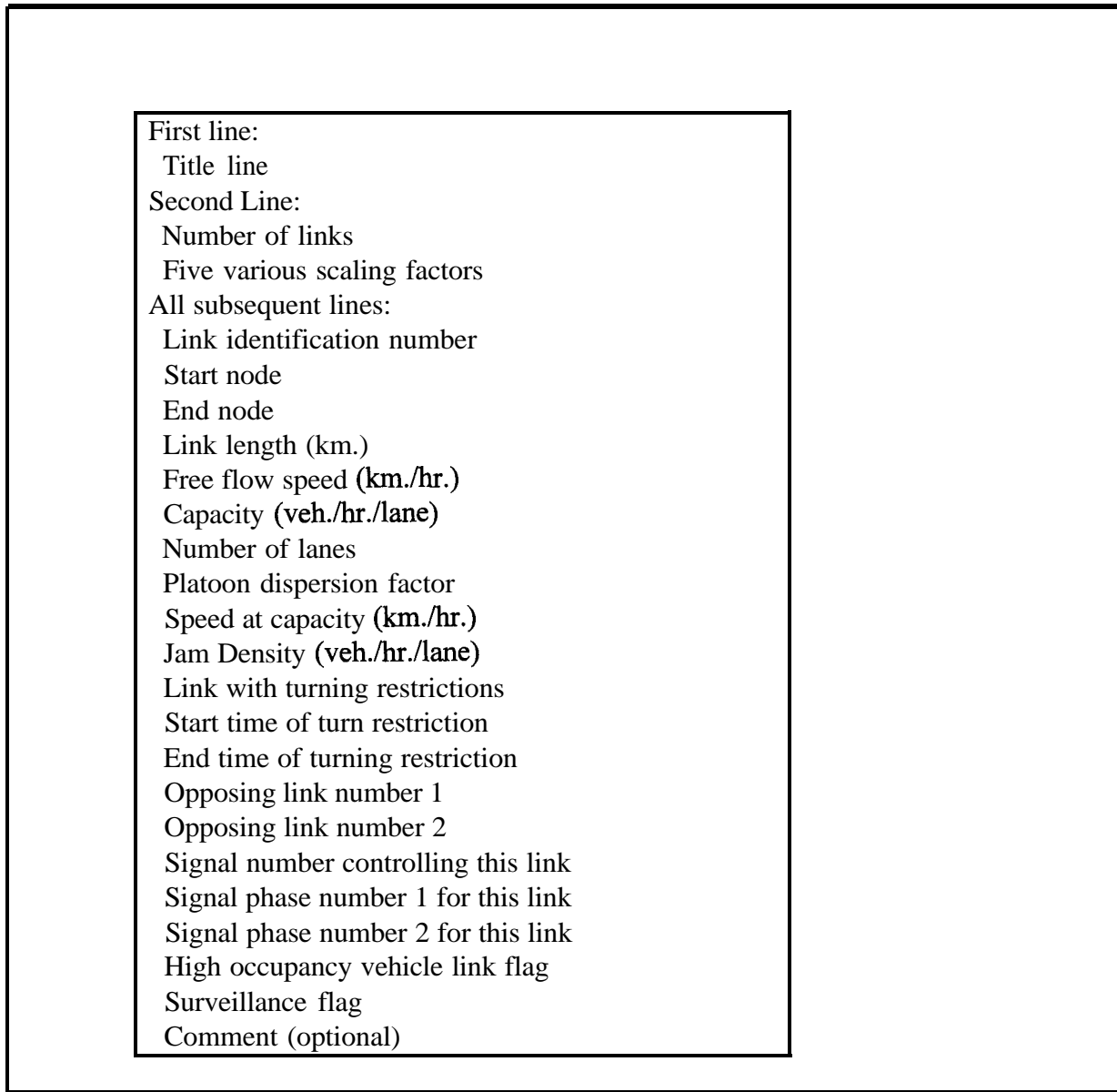


Figure A.7: Format of the INTEGRATION Link File

SCF14_2.DAT

3286	1.0	1.0	1.0	1.0	1.0																		
1	1	1401	0.400	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	001	c			
2	1407	301	0.215	65	2000	3	0.5	39.0	131	133	0	0	50400	0	0	0	273	1	0	0	1	c	
3	301	1891	0.076	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	0	0	001	c			
4	1897	158	0.186	65	2000	3	0.5	39.0	131	131	0	0	50400	0	0	0	0	0	001	c			
5	158	303	0.186	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	275	1	0	0	1	c		
6	303	1309	0.221	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	0	0	001	c			
7	1315	305	0.216	65	2000	3	0.5	39.0	131	128	0	0	50400	0	0	0	0	0	001	c			
8	305	1385	0.075	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	0	0	001	c			
9	1391	307	0.221	65	2000	3	0.5	39.0	131	126	0	0	50400	0	0	0	0	0	001	c			
10	307	1393	0.090	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	0	0	001	c			
11	1399	160	0.222	65	2000	3	0.5	39.0	131	124	0	0	50400	0	0	0	0	0	001	c			
12	160	1083	0.122	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	0	0	001	c			
13	1087	310	0.139	65	2000	3	0.5	39.0	131	122	0	0	50400	0	0	0	0	0	001	c			
14	310	311	0.379	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	394	3	101		c			
15	311	312	0.249	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	0	0	001	c			
16	312	313	0.471	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	396	1	0	0	1	c		
17	313	314	0.302	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	397	1	0	0	1	c		
18	314	315	0.305	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	398	1	0	0	1	c		
19	315	1191	0.489	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c				
20	1197	1199	0.005	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c				
21	1205	317	0.758	65	2000	2	0.5	39.0	131	114	0	0	50400	0	0	400	1	0	0	1	c		
22	317	1131	0.481	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c				
23	1137	163	0.147	65	2000	2	0.5	39.0	131	112	0	0	50400	0	0	0	0	001	c				
24	163	319	0.147	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	2	1001		c				
25	319	1571	0.278	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c				
26	1577	321	0.446	65	2000	2	0.5	39.0	131	109	0	0	50400	0	0	4	1001		c				
27	321	856	0.123	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c				
28	860	166	0.246	65	2000	2	0.5	39.0	131	107	0	0	50400	0	0	0	0	001	c				
29	166	721	0.146	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c				
30	727	1207	0.005	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c				
31	1213	325	0.187	65	2000	3	0.5	39.0	131	104	0	0	50400	0	0	9	1001		c				
32	325	326	0.086	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	10	1001		c				
33	326	327	0.258	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	11	1001		c				
34	327	1731	0.224	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c				
35	1737	170	0.207	65	2000	2	0.5	39.0	131	100	0	0	50400	0	0	0	0	001	c				
36	170	329	0.207	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	13	1	0	0	1	c		
37	329	1563	0.195	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c				
38	1569	331	0.312	65	2000	2	0.5	39.0	131	97	0	0	50400	0	0	15	1	0	0	1	c		
39	331	332	0.232	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	16	1	0	0	1	c		
40	332	173	0.285	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c				
41	173	1183	0.005	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c				
42	1189	661	0.050	65	2000	2	0.5	39.0	131	93	0	0	50400	0	0	18	1	0	0	1	c		
43	661	334	0.261	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c				
44	334	174	0.150	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c				
45	174	335	0.150	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	19	1	0	0	1	c		
46	335	1555	0.365	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c				
47	1561	337	0.233	65	2000	2	0.5	39.0	131	88	0	0	50400	0	0	211	001		c				
48	337	338	0.218	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	22	1001		c				
49	338	848	0.233	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c				
50	854	340	0.348	65	2000	2	0.5	39.0	131	85	0	0	50400	0	0	24	1001		c				
51	340	1579	0.086	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c				
52	1585	342	0.469	65	2000	2	0.5	39.0	131	83	0	0	50400	0	0	26	1001		c				
53	342	1739	0.278	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c				
54	1745	344	0.097	65	2000	2	0.5	39.0	131	81	0	0	50400	0	0	0	0	001	c				
55	344	345	0.098	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	28	1001		c				
56	345	346	0.285	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	29	1	0	0	1	c		
57	346	713	0.199	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c				
58	719	348	0.220	65	2000	2	0.5	39.0	131	77	0	0	50400	0	0	31	1	001	c				
59	348	349	0.139	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	32	1001		c				
60	349	1547	0.221	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c				
61	1553	1747	0.314	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c				
62	1753	352	0.355	65	2000	2	0.5	39.0	131	73	0	0	50400	0	0	35	1	0	0	1	c		
63	352	1293	0.140	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c				
64	1299	354	0.432	65	2000	2	0.5	39.0	131	71	0	0	50400	0	0	403	1	0	0	1	c		

65	354	355	0.329	65	2000	2	0.5	39.0	131	0	0	0	0	0	404	1	001	c
66	355	925	0.204	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
67	930	45	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	001	c
68	45	926	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	001	c
69	928	355	0.304	65	2000	2	0.5	39.0	131	66	0	50400	0	0	404	1	0 0 1	c
70	355	354	0.329	65	2000	2	0.5	39.0	131	0	0	0	0	0	403	3	101	c
71	354	1295	0.432	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
72	1297	352	0.240	65	2000	2	0.5	39.0	131	63	0	50400	0	0	35	1001		c
73	352	1749	0.355	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
74	1751	1549	0.414	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
75	1551	349	0.321	65	2000	2	0.5	39.0	131	60	0	50400	0	0	32	1001		c
76	349	348	0.139	65	2000	2	0.5	39.0	131	0	0	0	0	0	311	001		c
77	348	715	0.220	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
78	717	346	0.299	65	2000	2	0.5	39.0	131	57	0	50400	0	0	29	1001		c
79	346	345	0.285	65	2000	2	0.5	39.0	131	0	0	0	0	0	28	1 0 0 1		c
80	345	344	0.098	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
81	344	1741	0.097	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
82	1743	342	0.378	65	2000	2	0.5	39.0	131	53	0	50400	0	0	26	1001		c
83	342	1581	0.469	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
84	1583	340	0.186	65	2000	2	0.5	39.0	131	51	0	50400	0	0	24	1 0 0 1		c
85	340	850	0.348	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
86	852	338	0.333	65	2000	2	0.5	39.0	131	49	0	50400	0	0	22	1 0 0 1		c
87	338	337	0.218	65	2000	2	0.5	39.0	131	0	0	0	0	0	211	001		c
88	337	1557	0.233	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
89	1559	335	0.465	65	2000	2	0.5	39.0	131	46	0	50400	0	0	19	1 0 0 1		c
90	335	174	0.150	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
91	174	334	0.150	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
92	334	661	0.050	65	2000	2	0.5	39.0	131	0	0	0	0	0	18	1 0 0 1		c
93	661	1185	0.211	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
94	1187	173	0.285	65	2000	2	0.5	39.0	131	41	0	50400	0	0	0	0	001	c
95	173	332	0.285	65	2000	2	0.5	39.0	131	0	0	0	0	0	16	1 0 0 1		c
96	332	331	0.232	65	2000	2	0.5	39.0	131	0	0	0	0	0	15	1 0 0 1		c
97	331	1565	0.312	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
98	1567	329	0.295	65	2000	2	0.5	39.0	131	37	0	50400	0	0	13	1001		c
99	329	170	0.207	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
100	170	1733	0.207	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
101	1735	327	0.324	65	2000	2	0.5	39.0	131	34	0	50400	0	0	11	1 0 0 1		c
102	327	326	0.258	65	2000	2	0.5	39.0	131	0	0	0	0	0	10	1 0 0 1		c
103	326	325	0.086	65	2000	2	0.5	39.0	131	0	0	0	0	0	9	1001		c
104	325	1209	0.187	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
105	1211	723	0.097	65	2000	4	0.5	39.0	131	0	0	0	0	0	0	0	001	c
106	725	166	0.246	65	2000	2	0.5	39.0	131	29	0	50400	0	0	0	0	001	c
107	166	857	0.246	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
108	859	321	0.223	65	2000	2	0.5	39.0	131	27	0	50400	0	0	4	1 001		c
109	321	1573	0.446	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
110	1575	319	0.378	65	2000	2	3.5	39.0	131	25	0	50400	0	0	2	1001		c
111	319	163	0.147	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
112	163	1133	0.147	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
113	1135	317	0.581	65	2000	2	0.5	39.0	131	22	0	50400	0	0	400	4 101		c
114	317	1201	0.100	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
115	1203	1193	0.758	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
116	1195	315	0.589	65	2000	2	0.5	39.0	131	19	0	50400	0	0	398	1 0 0 1		c
117	315	314	0.305	65	2000	2	0.5	39.0	131	0	0	0	0	0	397	1 0 0 1		c
118	314	313	0.302	65	2000	3	0.5	39.0	131	0	0	0	0	0	3 9 6	1 0 0 1		c
119	313	312	0.471	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	001	c
120	312	311	0.249	65	2000	3	0.5	39.0	131	0	0	0	0	0	394	3 101		c
121	311	310	0.379	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	001	c
122	310	1084	0.139	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	001	c
123	1086	160	0.222	65	2000	3	0.5	39.0	131	12	0	50400	0	0	0	0	001	c
124	160	1395	0.222	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	001	c
125	1397	307	0.190	65	2000	3	0.5	39.0	131	10	0	50400	0	0	0	0	001	c
126	307	1387	0.221	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	001	c
127	1389	305	0.175	65	2000	3	0.5	39.0	131	8	0	50400	0	0	0	0	001	c
128	305	1311	0.216	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	001	c
129	1313	303	0.321	65	2000	3	0.5	39.0	131	6	0	50400	0	0	275	1 0 0 1		c
130	303	158	0.186	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	001	c
131	158	1893	0.186	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	001	c

132	1895	301	0.176	65	2000	3	0.5	39.0	131	3	0	50400	0	0	273	3	101	c				
133	301	1403	0.215	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	001	c				
134	1405	1	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	001	c				
135	2	1377	0.400	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	001	c				
136	634	1723	0.095	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c				
137	1729	157	0.198	65	2000	2	0.5	39.0	131	272	0	50400	0	0	0	0	001	c				
138	157	359	0.198	65	2000	2	0.5	39.0	131	0	0	0	0	0	288	1	0	0	1	c		
139	359	1075	0.197	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c				
140	1081	361	0.201	65	2000	2	0.5	39.0	131	269	0	50400	0	0	290	1	0	0	1	c		
141	361	778	0.093	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c				
142	784	363	0.214	65	2000	2	0.5	39.0	131	267	0	50400	0	0	292	1	0	0	1	c		
143	363	1899	0.107	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c				
144	1904	365	0.227	65	2000	2	0.5	39.0	131	265	0	50400	0	0	294	1	0	0	1	c		
145	365	159	0.107	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c				
146	159	750	0.007	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c				
147	755	367	0.206	65	2000	2	0.5	39.0	131	262	0	50400	0	0	296	1	0	0	1	c		
148	367	1095	0.415	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c				
149	1099	1089	0.177	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c				
150	635	636	0.037	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	001	c				
151	636	370	0.549	65	2000	2	0.5	39.0	131	0	0	0	0	0	611	001	c					
152	370	840	0.189	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c				
153	846	1117	0.252	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c				
154	1123	373	0.217	65	2000	2	0.5	39.0	131	255	0	50400	0	0	64	1001	c					
155	373	162	0.141	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c				
156	162	1125	0.041	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c				
157	1130	375	0.034	65	2000	2	0.5	39.0	131	252	0	50400	0	0	66	1001	c					
158	375	1595	0.153	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c				
159	1601	377	0.209	65	2000	2	0.5	39.0	131	250	0	50400	0	0	68	1001	c					
160	377	378	0.190	65	2000	2	0.5	39.0	131	0	0	0	0	0	69	1001	c					
161	378	164	0.156	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c				
162	164	1603	0.056	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c				
163	1609	380	0.406	60	2000	2	0.5	36.0	131	246	0	50400	0	0	7	1	1	0	0	1	c	
164	380	167	0.249	60	2000	2	0.5	36.0	131	0	0	0	0	0	0	0	001	c				
165	167	1763	0.149	60	2000	2	0.5	36.0	131	0	0	0	0	0	0	0	001	c				
166	1769	382	0.308	57	2000	2	0.5	34.2	131	243	0	50400	0	0	73	1001	c					
167	382	169	0.096	57	2000	2	0.5	34.2	131	0	0	0	0	0	0	0	001	c				
168	169	383	0.096	57	2000	2	0.5	34.2	131	0	0	0	0	0	0	0	001	c				
169	383	384	0.154	57	2000	2	0.5	34.2	131	0	0	0	0	0	74	1001	c					
170	384	1771	0.209	57	2000	2	0.5	34.2	131	0	0	0	0	0	0	0	001	c				
171	1777	386	0.391	57	2000	2	0.5	34.2	131	238	0	50400	0	0	76	1	0	0	1	c		
172	386	1153	0.214	57	2000	2	0.5	34.2	131	0	0	0	0	0	0	0	001	c				
173	1159	1611	0.316	57	2000	2	0.5	34.2	131	0	0	0	0	0	0	0	001	c				
174	1617	389	0.126	65	2000	2	0.5	39.0	131	235	0	50400	0	0	0	0	001	c				
175	389	827	0.285	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c				
176	833	1215	0.107	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c				
177	1220	392	0.111	65	2000	2	0.5	39.0	131	232	0	50400	0	0	811	001	c					
178	392	1221	0.108	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c				
179	1227	1587	0.534	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c				
181	1593	396	0.280	65	2000	2	0.5	39.0	131	228	0	50400	0	0	84	1001	c					
182	396	185	0.111	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c				
183	185	397	0.111	65	2000	2	0.5	39.0	131	0	0	0	0	0	85	1001	c					
184	397	1531	0.229	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c				
185	1537	399	0.288	65	2000	2	0.5	39.0	131	224	0	50400	0	0	87	1001	c					
186	399	1619	0.407	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c				
187	1625	1243	0.362	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c				
188	1249	180	0.188	65	2000	2	0.5	39.0	131	221	0	50400	0	0	0	0	001	c				
189	180	1779	0.088	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c				
190	1785	403	0.111	65	2000	2	0.5	39.0	131	219	0	50400	0	0	911	001	c					
191	403	404	0.369	65	2000	2	0.5	39.0	131	0	0	0	0	0	92	1001	c					
192	404	1787	0.196	65	2000	2	0.5	39.0	131	3126	3600	10800	0	0	0	0	001	c				
193	1793	406	0.219	65	2000	2	0.5	39.0	131	216	0	50400	0	0	94	1001	c					
194	406	407	0.259	65	2000	2	0.5	39.0	131	0	0	0	0	0	95	1	0	0	1	c		
195	407	1795	0.112	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c				
196	1801	1523	0.038	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c				
197	1529	410	0.172	65	2000	2	0.5	39.0	131	212	0	50400	0	0	3	1	4	1	0	0	1	c
198	410	1515	0.331	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c				
199	1521	412	0.195	65	2000	2	0.5	39.0	131	210	0	50400	0	0	100	1	0	0	1	c		

268	782	361	0.193	65	2000	2	0.5	39.0	131	141	0	50400	0	0	290	1	3	0	1	c	
269	361	1077	0.201	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
270	1079	359	0.297	65	2000	2	0.5	39.0	131	139	0	50400	0	0	288	1	0	0	1	c	
271	359	157	0.198	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
272	157	1725	0.198	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
273	634	1379	0.195	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
274	1381	2	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	1	c
275	5	1483	0.400	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	1	c
276	1489	798	0.179	75	2000	3	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	1	c
277	804	418	0.695	75	2000	3	0.5	45.0	131	368	0	50400	0	0	147	1	0	0	1	c	
278	418	1803	0.339	75	2000	3	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	1	c
279	1809	957	0.238	75	2000	3	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	1	c
280	963	124	0.200	75	2000	3	0.5	45.0	131	365	0	50400	0	0	0	0	0	0	0	1	c
281	124	1465	0.100	75	2000	3	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	1	c
282	1469	1471	0.071	75	2000	3	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	1	c
283	1475	1477	0.357	75	2000	3	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	1	c
284	1481	125	0.266	75	2000	3	0.5	45.0	131	361	0	50400	0	0	0	0	0	0	0	1	c
285	125	705	0.166	75	2000	3	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	1	c
286	711	1643	0.140	75	2000	3	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	1	c
287	1649	128	0.328	65	2000	3	0.5	39.0	131	358	0	50400	0	0	0	0	0	0	0	1	c
288	128	426	0.328	65	2000	3	0.5	39.0	131	0	0	0	0	0	155	1	0	0	1	c	
289	426	1811	0.370	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
290	1817	633	0.163	65	2000	3	0.5	39.0	131	355	0	50400	0	0	0	0	0	0	0	1	c
291	633	1916	0.193	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
292	1918	151	0.235	65	2000	3	0.5	39.0	131	353	0	50400	0	0	0	0	0	0	0	1	c
293	151	1457	0.135	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
294	1463	430	0.558	65	2000	3	0.5	39.0	131	351	0	50400	0	0	159	1	0	0	1	c	
295	430	431	0.407	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
296	431	1819	0.088	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
297	1825	150	0.328	75	2000	3	0.5	45.0	131	348	0	50400	0	0	0	0	0	0	0	1	c
298	150	1139	0.028	75	2000	3	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	1	c
299	1144	1161	0.221	75	2000	3	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	1	c
300	1167	1827	0.359	75	2000	3	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	1	c
301	1833	436	0.461	75	2000	4	0.5	45.0	131	344	0	50400	0	0	164	1	0	0	1	c	
302	436	149	0.158	75	2000	4	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	1	c
303	149	764	0.058	75	2000	4	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	1	c
304	770	439	0.306	65	2000	3	0.5	39.0	131	341	0	50400	0	0	166	1	0	0	1	c	
305	439	1651	0.424	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
306	1657	1229	0.361	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
307	1233	442	0.230	65	2000	3	0.5	39.0	131	338	0	50400	0	0	169	1	0	0	1	c	
308	442	1835	0.213	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
309	1841	444	0.295	65	2000	2	0.5	39.0	131	336	0	50400	0	0	171	1	0	0	1	c	
310	444	445	0.127	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
311	445	148	0.195	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
312	148	1449	0.095	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
313	1455	1251	0.336	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
314	1257	1659	0.246	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
315	1665	1667	0.723	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
316	1673	450	0.223	65	2000	2	0.5	39.0	131	329	0	50400	0	0	178	1	0	0	1	c	
317	450	1675	0.379	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
318	1681	452	0.329	65	2000	2	0.5	39.0	131	327	0	50400	0	0	181	1	0	0	1	c	
319	452	1441	0.094	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
320	1447	454	0.420	65	2000	2	0.5	39.0	131	325	0	50400	0	0	0	0	0	0	0	1	c
321	454	909	0.434	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
322	915	47	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	1	c
323	47	911	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	1	c
324	913	454	0.534	65	2000	2	0.5	39.0	131	321	0	50400	0	0	0	0	0	0	0	1	c
325	454	1443	0.420	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
326	1445	452	0.194	65	2000	2	0.5	39.0	131	319	0	50400	0	0	181	1	0	0	1	c	
327	452	1677	0.329	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
328	1679	450	0.479	65	2000	2	0.5	39.0	131	317	0	50400	0	0	178	1	0	0	1	c	
329	450	1669	0.223	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
330	1671	1661	0.823	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
331	1663	1253	0.346	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
332	1255	1451	0.436	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
333	1453	148	0.195	65	2000	2	0.5	39.0	131	312	0	50400	0	0	0	0	0	0	0	1	c
334	148	445	0.195	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c

335	445	444	0.127	65	2000	2	0.5	39.0	131	0	0	0	0	0	171	1	0	0	1	c	
336	444	1837	0.295	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
337	1839	442	0.313	65	2000	3	0.5	39.0	131	308	0	50400	0	0	169	1	0	0	1	c	
338	442	1230	0.230	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
339	1232	1653	0.461	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
340	1655	439	0.524	65	2000	3	0.5	39.0	131	305	0	50400	0	0	166	1	0	0	1	c	
341	439	766	0.306	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
342	768	149	0.158	75	2000	3	0.5	45.0	131	303	0	50400	0	0	0	0	0	0	0	1	c
343	149	436	0.158	75	2000	3	0.5	45.0	131	0	0	0	0	0	164	1	0	0	1	c	
344	436	1829	0.461	75	2000	3	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	1	c
345	1831	1163	0.459	75	2000	3	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	1	c
346	1165	1141	0.221	75	2000	3	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	1	c
347	1142	150	0.128	75	2000	3	0.5	45.0	131	298	0	50400	0	0	0	0	0	0	0	1	c
348	150	1821	0.328	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
349	1823	431	0.188	65	2000	3	0.5	39.0	131	296	0	50400	0	0	0	0	0	0	0	1	c
350	431	430	0.407	65	2000	3	0.5	39.0	131	0	0	0	0	0	159	1	0	0	1	c	
351	430	1459	0.558	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
352	1461	151	0.235	65	2000	3	0.5	39.0	131	293	0	50400	0	0	0	0	0	0	0	1	c
353	151	1914	0.135	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
354	1919	633	0.293	65	2000	3	0.5	39.0	131	1355	0	50400	0	0	0	0	0	0	0	1	c
355	633	1813	0.163	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
356	1815	426	0.470	65	2000	3	0.5	39.0	131	289	0	50400	0	0	155	1	0	0	1	c	
357	426	128	0.328	75	2000	3	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	1	c
358	128	1645	0.328	75	2000	3	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	1	c
359	1647	707	0.240	75	2000	3	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	1	c
360	709	125	0.266	75	2000	3	0.5	45.0	131	285	0	50400	0	0	0	0	0	0	0	1	c
361	125	1478	0.266	75	2000	3	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	1	c
362	1480	1472	0.457	75	2000	3	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	1	c
363	1474	1466	0.171	75	2000	3	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	1	c
364	1468	124	0.200	75	2000	3	0.5	45.0	131	281	0	50400	0	0	0	0	0	0	0	1	c
365	124	959	0.200	75	2000	3	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	1	c
366	961	1805	0.338	75	2000	3	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	1	c
367	1807	418	0.439	75	2000	3	0.5	45.0	131	278	0	50400	0	0	147	1	0	0	1	c	
368	418	800	0.695	75	2000	3	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	1	c
369	802	1485	0.279	75	2000	3	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	1	c
370	1487	5	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	1	c
371	6	943	0.400	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	1	c
372	948	457	0.808	65	2000	2	0.5	39.0	131	467	0	50400	0	0	419	1	0	0	1	c	
373	457	949	0.382	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
374	955	965	0.226	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
375	970	460	0.234	65	2000	2	0.5	39.0	131	0	0	0	0	0	422	1	0	0	1	c	
376	460	977	0.070	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
377	982	971	0.079	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
378	976	983	0.192	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
379	988	464	0.319	65	2000	2	0.5	39.0	131	460	0	50400	0	0	426	1	0	0	1	c	
380	464	989	0.143	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
381	995	1905	0.320	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
382	1911	467	0.295	70	2000	2	0.5	42.0	131	457	0	50400	0	0	428	1	0	0	1	c	
383	467	468	0.246	70	2000	2	0.5	42.0	131	0	0	0	0	0	429	1	0	0	1	c	
384	468	997	0.583	70	2000	2	0.5	42.0	131	0	0	0	0	0	0	0	0	0	0	1	c
385	1003	1005	0.016	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
386	1009	1433	0.251	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
387	1439	625	0.037	70	2000	3	0.5	42.0	131	452	0	50400	0	0	0	0	0	0	0	1	c
388	625	1360	0.152	55	2000	3	0.5	33.0	131	0	0	0	0	0	0	0	0	0	0	1	c
389	1362	472	0.294	65	2000	2	0.5	39.0	131	450	0	50400	0	0	187	1	0	0	1	c	
390	472	1425	0.337	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
391	1431	129	0.171	65	2000	2	0.5	39.0	131	448	0	50400	0	0	0	0	0	0	0	1	c
392	129	1169	0.471	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
393	1175	1683	0.417	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
394	1689	476	0.375	65	2000	2	0.5	39.0	131	445	0	50400	0	0	191	1	0	0	1	c	
395	476	132	0.161	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
396	132	477	0.161	65	2000	2	0.5	39.0	131	0	0	0	0	0	192	1	0	0	1	c	
397	477	1417	0.149	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
398	1423	1843	0.462	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
399	1849	480	0.466	65	2000	2	0.5	39.0	131	440	0	50400	0	0	432	1	0	0	1	c	
400	480	1235	0.117	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	1	c
401	1241	135	0.106	65	2000	2	0.5	39.0	131	438	0	50400	0	0	0	0	0	0	0	1	c

402	135	482	0.106	65	2000	2	0.5	39.0	131	0	0	0	0	0	197	1	001	c
403	482	1851	0.232	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
404	1857	484	0.397	65	2000	2	0.5	39.0	131	435	0	50400	0	0	199	1	001	c
405	484	138	0.203	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
406	138	1691	0.103	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
407	1697	1259	0.325	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
408	1264	139	0.173	65	2000	2	0.5	39.0	131	431	0	50400	0	0	0	0	001	c
409	139	1699	0.073	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
410	1705	488	0.400	65	2000	2	0.5	39.0	131	429	0	50400	0	0	204	1	001	c
411	488	1859	0.332	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
412	1865	490	0.361	65	2000	2	0.5	39.0	131	427	0	50400	0	0	206	1	001	c
413	490	1277	0.240	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
414	1283	492	0.156	65	2000	2	0.5	39.0	131	425	0	50400	0	0	0	0	001	c
415	492	1285	0.056	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
416	1291	144	0.180	65	2000	3	0.5	39.0	131	423	0	50400	0	0	0	0	001	c
417	144	494	0.180	65	2000	3	0.5	39.0	131	0	0	0	0	0	436	1	001	c
418	494	1301	0.507	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
419	1307	48	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	001	c
420	48	1303	0.500	-1	2000	5	0.5	a.0	131	0	0	0	0	0	0	0	001	c
421	1305	494	0.607	65	2000	2	0.5	39.0	131	418	0	50400	0	0	436	1	001	c
422	494	144	0.180	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
423	144	1287	0.180	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
424	1289	492	0.156	65	2000	2	0.5	39.0	131	415	0	50400	0	0	0	0	001	c
425	492	1279	0.156	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
426	1281	490	0.340	65	2000	2	0.5	39.0	131	413	0	50400	0	0	206	1	001	c
427	490	1861	0.361	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
428	1863	488	0.432	65	2000	2	0.5	39.0	131	411	0	50400	0	0	204	1	001	c
429	488	1701	0.400	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
430	1703	139	0.173	65	2000	2	0.5	39.0	131	409	0	50400	0	0	0	0	001	c
431	139	1261	0.073	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
432	1262	1693	0.425	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
433	1695	138	0.203	65	2000	2	0.5	39.0	131	406	0	50400	0	0	0	0	001	c
434	138	484	0.203	65	2000	2	0.5	39.0	131	0	0	0	0	0	199	1	001	c
435	484	1853	0.397	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
436	1855	482	0.332	65	2000	2	0.5	39.0	131	403	0	50400	0	0	197	1	001	c
437	482	135	0.106	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
438	135	1237	0.106	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
439	1239	480	0.217	65	2000	2	0.5	39.0	131	400	0	50400	0	0	432	1	001	c
440	480	1845	0.466	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
441	1847	1419	0.562	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
442	1421	477	0.249	65	2000	2	0.5	39.0	131	397	0	50400	0	0	192	1	001	c
443	477	132	0.161	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
444	132	476	0.161	65	2000	2	0.5	39.0	131	0	0	0	0	0	191	1	001	c
445	476	1685	0.375	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
446	1687	1171	0.517	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
447	1173	129	0.571	65	2000	2	0.5	39.0	131	392	0	50400	0	0	0	0	001	c
448	129	1427	0.171	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
449	1429	472	0.437	70	2000	2	0.5	42.0	131	390	0	50400	0	0	187	1	001	c
450	472	1358	0.194	70	2000	2	0.5	42.0	131	0	0	0	0	0	0	0	001	c
451	1363	625	0.252	55	2000	2	0.5	33.0	131	1466	0	50400	0	0	0	0	001	c
452	625	1435	0.037	70	2000	2	0.5	42.0	131	0	0	0	0	0	0	0	001	c
453	1437	1006	0.351	70	2000	2	0.5	42.0	131	0	0	0	0	0	0	0	001	c
454	1008	999	0.116	70	2000	2	0.5	42.0	131	0	0	0	0	0	0	0	001	c
455	1001	468	0.683	70	2000	2	0.5	42.0	131	384	0	50400	0	0	429	1	001	c
456	468	467	0.246	70	2000	2	0.5	42.0	131	0	0	0	0	0	428	1	001	c
457	467	1907	0.295	70	2000	2	0.5	42.0	131	0	0	0	0	0	0	0	001	c
458	1909	991	0.420	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
459	993	464	0.243	65	2000	2	0.5	39.0	131	380	0	50400	0	0	426	1	001	c
460	464	985	0.219	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
461	986	973	0.192	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
462	974	979	0.079	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
463	980	460	0.170	65	2000	2	0.5	39.0	131	0	0	0	0	0	422	1	001	c
464	460	967	0.134	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
465	968	951	0.326	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
466	953	457	0.482	65	2000	2	0.5	39.0	131	373	0	50400	0	0	419	1	001	c
467	457	945	0.708	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
468	946	6	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	001	c

469	1010	1867	0.144	55	2000	2	0.5	33.0	131	0	a	0	0	0	0	0	001	c
470	1873	1409	0.667	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
471	1415	1011	0.473	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
472	1017	499	0.188	55	2000	2	0.5	33.0	131	523	a	50400	0	0	210	1001	c	
473	499	1707	0.236	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
474	1713	501	0.453	55	2000	2	0.5	33.0	131	521	0	50400	0	0	214	1001	c	
475	501	502	0.267	55	2000	2	0.5	33.0	131	0	0	0	0	0	281	1001	c	
476	502	1019	0.111	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
477	1025	1715	0.390	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
478	1721	505	0.618	55	2000	1	0.5	33.0	131	517	0	50400	0	0	219	1001	c	
479	505	1027	0.333	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
480	1032	136	0.277	55	2000	2	0.5	33.0	131	515	0	50400	0	0	0	0	001	c
481	136	1875	0.177	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
482	1881	508	0.405	55	2000	2	0.5	33.0	131	513	0	50400	0	0	224	1001	c	
483	508	137	0.093	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
484	137	509	0.093	55	2000	2	0.5	33.0	131	0	0	0	0	0	225	1001	c	
485	509	1883	0.111	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
486	1889	511	0.312	55	2000	2	0.5	33.0	131	509	0	50400	0	0	229	1001	c	
487	511	140	0.138	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
488	140	512	0.138	55	2000	2	0.5	33.0	131	0	0	0	0	0	230	1001	c	
489	512	1033	0.115	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
490	1039	141	0.187	55	2000	2	0.5	33.0	131	505	0	50400	0	0	0	0	001	c
491	141	514	0.187	55	2000	2	0.5	33.0	131	0	0	0	0	0	439	1001	c	
492	514	1041	0.336	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
493	1047	142	0.359	55	2000	2	0.5	33.0	131	502	0	50400	0	0	0	0	001	c
494	142	729	0.259	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
495	735	143	0.424	55	2000	2	0.5	33.0	131	500	0	50400	0	0	0	0	001	c
496	143	1049	0.324	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
497	1055	49	0.500	-1	2000	2	0.5	8.0	131	0	0	0	0	0	0	0	001	c
498	49	1051	0.500	-1	2000	2	0.5	8.0	131	0	0	0	0	0	0	0	001	c
499	1053	143	0.424	55	2000	2	0.5	33.0	131	496	0	50400	0	0	0	0	001	c
500	143	731	0.424	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
501	733	142	0.359	55	2000	2	0.5	33.0	131	494	0	50400	0	0	0	0	001	c
502	142	1043	0.359	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
503	1045	514	0.436	55	2000	2	0.5	33.0	131	492	0	50400	0	0	439	1001	c	
504	514	141	0.187	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
505	141	1035	0.187	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
506	1037	512	0.215	55	2000	2	0.5	33.0	131	489	0	50400	0	0	230	1001	c	
507	512	140	0.138	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
508	140	511	0.138	55	2000	2	0.5	33.0	131	0	0	0	0	0	229	1001	c	
509	511	1885	0.312	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
510	1887	509	0.211	55	2000	2	0.5	33.0	131	485	0	50400	0	0	225	1001	c	
511	509	137	0.093	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
512	137	508	0.093	55	2000	2	0.5	33.0	131	0	0	0	0	0	224	1001	c	
513	508	1877	0.405	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
514	1879	136	0.277	55	2000	2	0.5	33.0	131	481	0	50400	0	0	0	0	001	c
515	136	1029	0.177	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
516	1030	505	0.433	55	2000	2	0.5	33.0	131	479	0	50400	0	0	219	1001	c	
517	505	1717	0.618	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
518	1719	1021	0.490	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
519	1023	502	0.211	55	2000	2	0.5	33.0	131	476	0	50400	0	0	281	1001	c	
520	502	501	0.267	55	2000	2	0.5	33.0	131	0	0	0	0	0	214	1001	c	
521	501	1709	0.453	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
522	1711	499	0.336	55	2000	2	0.5	33.0	131	473	0	50400	0	0	210	1001	c	
523	499	1013	0.188	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
524	1015	1411	0.573	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
525	1413	1869	0.767	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
526	1a71	1007	0.244	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
527	649	518	0.532	65	2000	3	0.5	39.0	131	0	0	0	0	0	411	001	c	
528	518	1755	0.246	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	001	c
529	1761	171	0.190	65	2000	3	0.5	39.0	131	540	0	50400	0	0	0	0	001	c
530	171	520	0.190	65	2000	3	0.5	39.0	131	0	0	0	0	0	43	1001	c	
531	520	1145	0.216	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	001	c
532	1151	1539	0.246	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	001	c
533	1545	828	0.387	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	001	c
534	834	765	0.163	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	001	c
535	769	830	0.163	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c

536	832	1541	0.487	65	2000	3	0.5	39.0	131	2760	0	50400	0	0	0	0	001	c				
537	1543	1147	0.346	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	001	c				
538	1149	520	0.316	65	2000	3	0.5	39.0	131	531	0	50400	0	0	4	3	1	0	0	1	c	
539	520	171	0.190	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
540	171	1757	0.190	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
541	1759	518	0.346	65	2000	3	0.5	39.0	131	528	0	50400	0	0	4	1	1	0	0	1	c	
542	518	649	0.532	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
543	3	1627	0.400	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	001	c			
544	1633	835	0.005	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
545	838	737	0.032	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
546	741	1507	0.005	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
547	1513	527	0.318	65	2000	2	0.5	39.0	131	578	0	50400	0	0	10	7	1	0	0	1	c	
548	527	183	0.237	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
549	183	790	0.137	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
550	796	742	0.293	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
551	762	1323	0.005	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
552	1327	1101	0.433	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
553	1107	938	0.691	65	2000	1	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
554	942	905	0.005	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
555	907	126	0.183	65	2000	1	0.5	39.0	131	570	0	50400	0	0	0	0	0	001	c			
556	126	532	0.183	65	2000	1	0.5	39.0	131	0	0	0	0	0	1	3	7	1	0	0	1	c
557	532	127	0.179	65	2000	1	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
558	533	811	0.370	65	2000	1	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
559	817	806	0.072	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
560	810	1644	0.057	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
561	1650	1906	0.105	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	0	001	c			
562	1912	32	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	001	c			
563	32	1908	0.400	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	001	c			
564	1910	1646	0.105	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	0	001	c			
565	1648	807	0.057	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
566	809	813	0.172	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
567	815	533	0.470	65	2000	1	0.5	39.0	131	558	0	50400	0	0	1	3	8	1	0	0	1	c
568	127	532	0.179	65	2000	1	0.5	39.0	131	0	0	0	0	0	1	3	7	1	0	0	1	c
569	532	126	0.183	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
570	126	903	0.083	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
571	908	939	0.005	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
572	941	1103	0.691	65	2000	1	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
573	1105	1325	0.433	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
574	1326	758	0.053	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
575	746	792	0.393	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
576	794	183	0.237	65	2000	2	0.5	39.0	131	549	0	50400	0	0	0	0	0	001	c			
577	183	527	0.237	65	2000	2	0.5	39.0	131	0	0	0	0	0	10	7	1	0	0	1	c	
578	527	1509	0.318	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
579	1511	738	0.005	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
580	740	836	0.132	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
581	837	1629	0.085	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
582	1631	3	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	001	c			
583	4	1499	0.400	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	001	c			
584	1505	1635	0.182	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
585	1641	537	0.597	65	2000	2	0.5	39.0	131	598	0	50400	0	0	1	3	2	1	0	0	1	c
586	537	120	0.273	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
587	120	697	0.173	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
588	703	122	0.170	65	2000	2	0.5	39.0	131	595	0	50400	0	0	0	0	0	001	c			
589	122	1109	0.070	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
590	1115	123	0.284	65	2000	2	0.5	39.0	131	593	0	50400	0	0	0	0	0	001	c			
591	123	937	0.184	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
592	940	123	0.284	65	2000	2	0.5	39.0	131	591	0	50400	0	0	0	0	0	001	c			
593	123	1111	0.284	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
594	1113	122	0.170	65	2000	2	0.5	39.0	131	589	0	50400	0	0	0	0	0	001	c			
595	122	699	0.170	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
596	701	120	0.273	65	2000	2	0.5	39.0	131	587	0	50400	0	0	0	0	0	001	c			
597	120	537	0.273	65	2000	2	0.5	39.0	131	0	0	0	0	0	1	3	2	1	0	0	1	c
598	537	1637	0.597	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
599	1639	1501	0.282	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c			
600	1503	4	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	001	c			
601	307	363	0.563	55	1800	1	0.5	33.0	131	0	0	0	0	0	2	9	2	0	0	1	c	
602	363	540	0.514	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	0	001	c			

603	540	154	0.530	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	001	c	
604	154	541	0.530	55	1800	1	0.5	33.0	131	0	0	0	0	0	117	2	001	c
605	541	632	0.597	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	001	c
606	632	902	0.201	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	001	c
607	906	632	0.201	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	001	c
608	632	541	0.597	55	1800	1	0.5	33.0	131	0	0	0	0	0	117	2	001	c
609	541	154	0.530	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	001	c
610	154	540	0.530	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	001	c
611	540	363	0.514	55	1800	1	0.5	33.0	131	0	0	0	0	0	292	2	001	c
612	363	307	0.563	55	1800	1	0.5	33.0	131	0	0	0	0	0	10001	0	001	c
613	543	819	1.979	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	001	c
614	825	545	0.143	55	1800	1	0.5	33.0	131	619	0	50400	0	0	128	1	001	c
615	545	1491	0.766	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	001	c
616	1497	1913	0.275	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
617	1917	1493	0.275	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
618	1495	545	0.866	55	1800	1	0.5	33.0	131	615	0	50400	0	0	128	1	0 0 1	c
619	545	821	0.143	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	001	c
620	823	543	2.079	55	1800	1	0.5	33.0	131	613	0	50400	0	0	0	0	001	c
621	547	548	0.887	45	1800	1	0.5	27.0	131	0	0	0	0	0	115	2	001	c
622	548	549	0.740	45	1800	1	0.5	27.0	131	0	0	0	0	0	116	2	001	c
623	549	431	0.728	45	1800	1	0.5	27.0	131	0	0	0	0	0	10001	0	001	c
624	431	549	0.728	45	1800	1	0.5	27.0	131	0	0	0	0	0	116	2	001	c
625	549	548	0.740	45	1800	1	0.5	27.0	131	0	0	0	0	0	115	2	001	c
626	548	547	0.887	45	1800	1	0.5	27.0	131	0	0	0	0	0	114	2	001	c
627	7	1402	0.400	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	001	c
628	1408	1328	0.101	55	2000	2	Q.5	33.0	131	0	0	0	0	0	0	0	001	c
629	1332	1378	0.111	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
630	1384	552	0.830	65	2000	2	0.5	39.0	131	639	0	50400	0	0	102	1	0 0 1	c
631	552	1628	0.271	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
632	1634	1500	0.949	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
633	1506	1484	0.809	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
634	1490	29	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	001	c
635	29	1486	0.400	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	001	c
636	1488	1502	0.809	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
637	1504	1630	0.949	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
638	1632	552	0.371	65	2000	2	0.5	39.0	131	631	0	50400	0	0	102	1	0 0 1	c
639	552	1380	0.730	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
640	1382	1330	0.111	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
641	1331	1404	0.101	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
642	1406	7	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	001	c
643	301	640	0.191	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	001	c
644	640	634	0.245	55	1800	1	0.5	33.0	131	0	0	0	0	0	10001	0	001	c
645	634	640	0.245	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	001	c
646	640	301	0.191	55	1800	1	0.5	33.0	131	0	0	0	0	0	273	2	001	c
647	8	1892	0.400	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	001	c
648	1898	1724	0.349	70	2000	2	0.5	42.0	131	0	0	0	0	0	0	0	001	c
649	1730	156	0.604	65	2000	2	0.5	39.0	131	664	0	50400	0	0	0	0	001	c
650	156	1508	0.504	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
651	1514	553	0.467	65	2000	2	0.5	39.0	131	662	0	50400	0	0	122	1	0 0 1	c
652	553	184	0.300	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
653	184	1636	0.200	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
654	1642	554	0.344	65	2000	2	0.5	39.0	131	659	0	50400	0	0	144	2	001	c
655	554	119	0.257	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
656	119	799	0.157	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
657	803	119	0.257	65	2000	2	0.5	39.0	131	656	0	50400	0	0	0	0	001	c
658	119	554	0.257	65	2000	2	0.5	39.0	131	0	0	0	0	0	144	2	001	c
659	554	1638	0.244	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
660	1640	184	0.300	65	2000	2	0.5	39.0	131	653	0	50400	0	0	0	0	001	c
661	184	553	0.300	65	2000	2	0.5	39.0	131	0	0	0	0	0	122	1	0 0 1	c
662	553	1510	0.367	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
663	1512	156	0.604	65	2000	2	0.5	39.0	131	650	0	50400	0	0	0	0	001	c
664	156	1726	0.504	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
665	1728	1894	0.349	70	2000	2	0.5	42.0	131	0	0	0	0	0	0	0	001	c
666	1896	8	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	001	c
667	303	359	0.476	55	1800	1	0.5	33.0	131	0	0	0	0	0	288	2	001	c
668	359	303	0.476	55	1800	1	0.5	33.0	131	0	0	0	0	0	275	2	001	c
669	9	1310	0.400	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	001	c

670	1316	1076	0.415	55	1800	2	0.5	33.0	131	0	0	0	0	0	0	0	0	0	001	c	
671	1082	155	0.441	55	1800	2	0.5	33.0	131	676	0	50400	0	0	0	0	0	0	001	c	
672	155	555	0.441	55	1800	2	0.5	33.0	131	0	0	0	0	0	0	306	1	0	0	1	c
673	555	791	0.260	55	1800	2	0.5	33.0	131	0	0	0	0	0	0	0	0	0	001	c	
674	795	555	0.360	55	1800	2	0.5	33.0	131	673	0	50400	0	0	0	306	1	0	0	1	c
675	555	155	0.441	55	1800	2	0.5	33.0	131	0	0	0	0	0	0	0	0	0	001	c	
676	155	1078	0.341	55	1800	2	0.5	33.0	131	0	0	0	0	0	0	0	0	0	001	c	
677	1080	1312	0.415	55	1800	2	0.5	33.0	131	0	0	0	0	0	0	0	0	0	001	c	
678	1314	9	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	001	c	
679	10	1386	0.400	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	001	c	
680	1392	779	0.445	65	2000	1	0.5	39.0	131	0	0	0	0	0	0	0	0	0	001	c	
681	785	559	0.440	65	2000	2	0.5	39.0	131	698	0	50400	0	0	0	300	1	0	0	1	c
682	559	556	0.374	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	301	1	0	0	1	c
683	556	743	0.298	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	001	c	
684	653	757	0.066	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	001	c	
685	763	698	0.693	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	001	c	
686	704	121	0.399	65	2000	1	0.5	39.0	131	693	0	50400	0	0	0	0	0	0	001	c!	
687	121	1804	0.299	65	2000	1	0.5	39.0	131	0	0	0	0	0	0	0	0	0	001	c	
688	1810	950	0.005	55	2000	1	0.5	33.0	131	0	0	0	0	0	0	0	0	0	001	c	
689	956	30	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	001	c	
690	30	952	0.400	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	001	c	
691	954	1806	0.005	55	2000	1	0.5	33.0	131	0	0	0	0	0	0	0	0	0	001	c	
692	1808	121	0.399	65	2000	1	0.5	39.0	131	687	0	50400	0	0	0	0	0	0	001	c	
693	121	700	0.299	65	2000	1	0.5	39.0	131	0	0	0	0	0	0	0	0	0	001	c	
694	702	759	0.693	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	001	c	
695	761	787	0.005	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	0	0	001	c	
696	747	556	0.398	65	2000	2	0.5	39.0	131	683	0	50400	0	0	0	301	1	0	0	1	c
697	556	559	0.374	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	300	1	0	0	1	c
698	559	781	0.340	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	001	c	
699	783	1388	0.445	65	2000	1	0.5	39.0	131	0	0	0	0	0	0	0	0	0	001	c	
700	1390	10	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	001	c	
701	11	1394	0.400	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	001	c	
702	1400	1900	0.520	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	0	0	001	c	
703	1903	1396	0.420	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	0	0	001	c	
704	1398	11	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	001	c	
705	1088	751	0.557	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	0	0	001	c	
706	754	1085	0.557	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	0	0	001	c	
707	12	312	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	001	c	
708	312	654	0.662	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	483	2	0	0	1	c
709	654	1090	0.100	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	0	0	001	c	
1384	1093	654	0.100	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	483	1	0	0	1	c
710	654	312	0.662	65	2000	3	0.5	39.0	131	709	0	50400	0	0	0	0	0	0	001	c	
711	312	12	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	001	c	
712	313	635	0.865	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	480	2	0	0	1	c
713	635	313	0.865	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	396	2	3	0	1	c
714	315	370	0.717	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	612	0	0	0	1	c
715	370	315	0.717	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	398	2	0	0	1	c
716	13	651	0.100	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	001	c	
717	651	1192	0.400	55	2000	2	0.5	33.0	131	0	0	0	0	0	0	0	0	0	001	c	
718	1198	841	0.424	55	1800	2	0.5	33.0	131	0	0	0	0	0	0	0	0	0	001	c	
719	847	562	0.294	55	1800	2	0.5	33.0	131	722	0	50400	0	0	0	110	1	0	0	1	c
720	562	543	0.334	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	0	0	001	c	
721	543	562	0.334	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	110	1	0	0	1	c
722	562	843	0.194	55	1800	2	0.5	33.0	131	0	0	0	0	0	0	0	0	0	001	c	
723	a45	1194	0.424	55	1800	2	0.5	33.0	131	0	0	0	0	0	0	0	0	0	001	c	
724	1196	651	0.100	55	1800	2	0.5	33.0	131	0	0	0	0	0	0	0	0	0	001	c	
725	651	13	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	001	c	
726	317	1126	0.458	40	1800	1	0.5	24.0	131	0	0	0	0	0	0	0	0	0	001	c	
727	1129	317	0.458	40	1800	1	0.5	24.0	131	726	0	50400	0	0	0	400	2	3	0	1	c
728	14	1132	0.400	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	001	c	
729	1138	1596	0.412	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	001	c	
730	1602	563	0.290	65	2000	2	0.5	39.0	131	760	0	50400	0	0	0	112	1	0	0	1	c
731	563	565	0.180	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	284	3	0	0	1	c
732	565	165	0.107	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	001	c	
733	165	547	0.107	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	114	1	0	0	1	c
734	547	566	0.252	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	119	1	0	0	1	c
735	566	153	0.315	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	001	c	

803	549	576	0.282	65	2000	1	0.5	39.0	131	0	0	0	0	0	121	1	0	0	1	c
804	576	1458	0.269	65	2000	1	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	0
805	1464	1365	0.342	75	2000	2	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	0
806	1371	1434	0.098	75	2000	2	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	0
807	1440	1868	0.117	75	2000	2	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	0
808	1874	33	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	0
809	33	1870	0.400	-1	2000	5	0.5	a.0	131	0	0	0	0	0	0	0	0	0	0	0
810	1872	1436	0.117	75	2000	2	0.5	45.0	131	2613	0	50400	0	0	0	0	0	0	0	0
811	1438	1367	0.098	60	2000	2	0.5	36.0	131	0	0	0	0	0	0	0	0	0	0	0
812	1369	1460	0.342	75	2200	2	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	0
813	1462	576	0.369	65	2000	1	0.5	39.0	131	804	0	50400	0	0	121	1	0	0	1	c
814	576	549	0.282	65	2000	1	0.5	39.0	131	0	0	0	0	0	116	1	0	0	1	c
815	549	168	0.337	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	0
816	168	1766	0.237	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	0
817	1768	575	0.487	65	2000	1	0.5	39.0	131	800	0	50400	0	0	40	1	0	0	1	c
818	644	724	0.005	65	2000	1	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	0
819	726	648	0.050	65	2000	1	0.5	39.0	131	798	0	50400	0	0	0	0	0	0	0	0
820	648	16	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	0
821	325	649	0.498	55	1800	1	0.5	33.0	131	0	0	0	0	0	10001	0	0	0	0	0
822	649	382	0.122	55	1800	1	0.5	33.0	131	0	0	0	0	0	73	2	0	0	1	c
823	382	649	0.122	55	1800	1	0.5	33.0	131	0	0	0	0	0	10001	0	0	0	1	c
824	649	325	0.498	55	1800	1	0.5	33.0	131	0	0	0	0	0	9	2	0	0	1	c
825	327	518	0.275	55	1800	1	0.5	33.0	131	0	0	0	0	0	41	2	0	0	1	c
826	518	384	0.642	55	1800	1	0.5	33.0	131	0	0	0	0	0	74	2	0	0	1	c
827	384	518	0.642	55	1800	1	0.5	33.0	131	0	0	0	0	0	41	2	0	0	1	c
828	518	327	0.275	55	1800	1	0.5	33.0	131	0	0	0	0	0	11	2	0	0	1	c
829	17	1732	0.400	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	0
830	1738	1756	0.276	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	0	0	0	0
831	1762	1772	0.525	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	0	0	0	0
832	1778	1820	0.435	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	0	0	0	0
833	1826	1426	0.485	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	0	0	0	0
834	1416	34	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	0
835	34	1412	0.400	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	0
836	1430	1822	0.485	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	0	0	0	0
837	1824	1774	0.435	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	0	0	0	0
838	1776	1758	0.525	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	0	0	0	0
839	1760	1734	0.276	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	0	0	0	0
840	1736	17	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	0
841	329	520	0.533	55	1800	1	5.5	33.0	131	0	0	0	0	0	43	2	0	0	1	c
842	520	386	0.520	55	1800	1	0.5	33.0	131	0	0	0	0	0	76	2	0	0	1	c
843	386	1140	0.391	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	0	0	0	0
844	1143	386	0.391	55	1800	1	0.5	33.0	131	843	0	50400	0	0	76	2	0	0	1	c
845	386	520	0.520	55	1800	1	0.5	33.0	131	0	0	0	0	0	43	2	0	0	1	c
846	520	329	0.533	55	1800	1	0.5	33.0	131	0	0	0	0	0	13	2	0	0	1	c
847	585	1146	0.504	55	2000	1	0.5	33.0	131	0	0	0	0	0	0	0	0	0	0	0
848	1152	172	0.209	55	2000	1	0.5	33.0	131	859	0	50400	0	0	0	0	0	0	0	0
849	172	1154	0.109	55	2000	1	0.5	33.0	131	0	0	0	0	0	0	0	0	0	0	0
850	1160	1162	0.212	55	2000	1	0.5	33.0	131	0	0	0	0	0	0	0	0	0	0	0
851	1168	1170	0.406	55	2000	1	0.5	33.0	131	0	0	0	0	0	0	0	0	0	0	0
852	1176	578	0.198	55	2000	1	0.5	33.0	131	855	0	50400	0	0	0	0	0	0	0	0
853	1018	35	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	0
854	35	1014	0.400	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	0
855	578	1172	0.098	55	2000	1	0.5	33.0	131	0	0	0	0	0	0	0	0	0	0	0
856	1174	1164	0.406	55	2000	1	0.5	33.0	131	0	0	0	0	0	0	0	0	0	0	0
857	1166	1156	0.212	55	2000	1	0.5	33.0	131	0	0	0	0	0	0	0	0	0	0	0
858	1158	172	0.209	55	2000	1	0.5	33.0	131	849	0	50400	0	0	0	0	0	0	0	0
859	172	1148	0.109	55	2000	1	0.5	33.0	131	0	0	0	0	0	0	0	0	0	0	0
860	1150	585	0.604	55	2000	1	0.5	33.0	131	a47	0	50400	0	0	10001	0	0	0	0	0
861	18	1564	0.400	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	0
862	1570	585	0.050	75	2000	3	0.5	45.0	131	889	0	50400	0	0	0	0	0	0	0	0
863	585	579	0.323	75	2000	2	0.5	45.0	131	0	0	0	0	0	46	1	0	0	1	c
864	579	1540	0.358	75	2000	2	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	0
865	1546	1612	0.185	75	2000	2	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	0
866	1618	1828	0.104	75	2000	2	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	0
867	1834	1684	0.555	75	2000	2	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	0
868	1690	1178	0.133	75	2000	2	0.5	45.0	131	0	0	0	0	0	0	0	0	0	0	0
869	1182	652	0.145	75	2000	2	0.5	45.0	131	882	0	50400	0	0	211	1	0	0	1	c

937	1350	1340	0.021	50	2000	3	0.5	30.0	131	0	0	0	0	0	0	0	001	c			
938	1344	1876	0.324	50	2000	1	0.5	30.0	131	0	0	0	0	0	0	0	001	c			
939	1882	39	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	001	c			
940	39	1878	0.400	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	001	c			
941	1880	1341	0.324	50	2000	1	0.5	30.0	131	0	0	0	0	0	0	0	001	c			
942	1342	1347	0.021	50	2000	3	0.5	30.0	131	0	0	0	0	0	0	0	001	c			
943	1349	1854	0.147	50	2000	1	0.5	30.0	131	0	0	0	0	0	0	0	001	c			
944	1856	1838	0.312	50	2000	1	0.5	30.0	131	0	0	0	0	0	0	0	001	c			
945	1840	1534	0.312	55	2000	1	0.5	33.0	131	0	0	0	0	0	0	0	001	c			
946	645	851	0.221	55	2000	1	0.5	33.0	131	0	0	0	0	0	0	0	001	c			
947	853	21	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	001	c			
948	22	1580	0.400	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	001	c			
949	1586	591	0.210	65	2000	2	0.5	39.0	131	968	50400	0	0	49	1	0	0	1	c		
950	591	176	0.200	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	c		
951	176	1620	0.100	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	c		
952	1626	1450	0.312	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	c		
953	1456	592	0.211	65	2000	2	0.5	39.0	131	964	50400	0	0	20	0	1	0	0	1	c	
954	592	1692	0.100	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	c	
955	1698	1317	0.135	70	2000	2	0.5	42.0	131	0	0	0	0	0	0	0	0	0	0	c	
956	1322	1334	0.062	70	2000	4	0.5	42.0	131	0	0	0	0	0	0	0	0	0	0	c	
957	1338	1884	0.307	70	2000	2	0.5	42.0	131	0	0	0	0	0	0	0	0	0	0	c	
958	1890	40	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c	
959	40	1886	0.400	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c	
960	1888	1335	0.307	70	2000	2	0.5	42.0	131	0	0	0	0	0	0	0	0	0	0	c	
961	1336	1319	0.062	70	2000	4	0.5	42.0	131	0	0	0	0	0	0	0	0	0	0	c	
962	1321	1694	0.135	70	2000	2	0.5	42.0	131	0	0	0	0	0	0	0	0	0	0	c	
963	1696	592	0.200	65	2000	2	0.5	39.0	131	954	50400	0	0	20	0	1	0	0	1	c	
964	592	1452	0.111	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	c	
965	1454	1622	0.312	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	c	
966	1624	176	0.200	65	2000	2	0.5	39.0	131	951	50400	0	0	0	0	0	0	0	0	c	
967	176	591	0.200	65	2000	2	0.5	39.0	131	0	0	0	0	49	1	0	0	1	0	c	
968	591	1582	0.110	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	c	
969	1584	22	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c	
970	615	1244	0.300	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	0	0	0	c	
971	1250	1252	0.311	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	0	0	0	c	
972	1258	1260	0.412	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	0	0	0	c	
973	1263	1254	0.312	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	0	0	0	c	
974	1256	1246	0.311	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	0	0	0	c	
975	615	342	0.215	55	1800	1	0.5	33.0	131	0	0	0	0	26	2	0	0	1	0	c	
976	23	1740	0.400	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c	
977	1746	595	0.210	65	2000	1	0.5	39.0	131	994	50400	0	0	5	1	1	0	0	1	c	
978	595	1780	0.311	65	2000	1	0.5	39.0	131	3117	36000	46800	0	0	0	0	0	0	0	1	c
979	1786	596	0.205	65	2000	2	0.5	39.0	131	992	50400	0	0	4	5	4	1	0	0	1	c
980	596	1660	0.094	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	c	
981	1666	1700	0.322	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	c	
982	1706	896	0.142	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	c	
983	901	891	0.061	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	c	
984	895	1034	0.307	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	c	
985	1040	41	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c	
986	41	1036	0.400	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c	
987	1038	892	0.307	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	c	
988	893	898	0.061	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	c	
989	900	1702	0.142	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	c	
990	1704	1662	0.322	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	c	
991	1664	596	0.194	65	2000	2	0.5	39.0	131	980	50400	0	0	4	5	4	1	0	0	1	c
992	596	1782	0.105	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	c	
993	1784	595	0.411	65	2000	1	0.5	39.0	131	978	50400	0	0	5	1	1	0	0	1	c	
994	595	1742	0.110	65	2000	1	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	c	
995	1744	23	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c	
996	24	714	0.400	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c	
997	720	599	0.169	65	2000	2	0.5	39.0	131	1018	50400	0	0	5	2	1	0	0	1	c	
998	599	177	0.200	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	c	
999	177	1788	0.100	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	c	
1000	1794	1668	0.246	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	c	
1001	1674	147	0.234	65	2000	2	0.5	39.0	131	1014	50400	0	0	0	0	0	0	0	0	c	
1002	147	1860	0.134	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	c	
1003	1866	884	0.210	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	0	0	c	

1004	889	879	0.038	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1005	883	602	0.174	65	2000	3	0.5	39.0	131	1010	0	50400	0	0	459	1	0	0	1	c
1006	602	1042	0.099	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1007	1048	42	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	001	c	
1008	42	1044	0.400	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	001	c	
1009	1046	602	0.199	65	2000	3	0.5	39.0	131	1006	0	50400	0	0	459	1	0	0	0	c
1010	602	880	0.074	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1011	881	886	0.038	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1012	888	1862	0.210	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1013	1864	147	0.234	65	2000	2	0.5	39.0	131	1002	0	50400	0	0	0	0	0	001	c	
1014	147	1670	0.134	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1015	1672	1790	0.246	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1016	1792	177	0.200	65	2000	2	0.5	39.0	131	999	0	50400	0	0	0	0	0	001	c	
1017	177	599	0.200	65	2000	2	0.5	39.0	131	0	0	0	0	0	52	1001	0	0	c	
1018	599	716	0.069	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1019	718	24	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	001	c	
1020	348	406	0.568	55	1800	1	0.5	33.0	131	0	0	0	0	0	94	2	0	0	0	c
1021	406	450	0.333	65	2000	2	0.5	39.0	131	0	0	0	0	0	178	2	0	0	0	c
1022	450	406	0.333	65	2000	2	0.5	39.0	131	0	0	0	0	0	94	2	0	0	0	c
1023	406	348	0.568	55	1800	1	0.5	33.0	131	0	0	0	0	0	312	0	0	0	0	c
1024	25	1548	0.400	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	001	c	
1025	1554	617	0.126	65	2000	2	0.5	39.0	131	1052	0	50400	0	0	0	0	0	001	c	
1026	617	178	0.126	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1027	178	603	0.126	65	2000	2	0.5	39.0	131	0	0	0	0	0	57	1	0	0	0	c
1028	603	1796	0.100	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1029	1802	1271	0.144	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1030	1276	1676	0.057	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1031	1682	146	0.211	65	2000	2	0.5	39.0	131	1046	0	50400	0	0	0	0	0	001	c	
1032	146	1278	0.111	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1033	1284	1266	0.101	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1034	1270	873	0.112	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1035	877	1057	0.022	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1036	1062	1063	0.005	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1037	1068	730	0.160	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1038	736	43	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	001	c	
1039	43	732	0.400	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	001	c	
1040	734	1065	0.160	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1041	1066	1059	0.005	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1042	1060	874	0.022	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1043	875	1267	0.112	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1044	1269	1280	0.101	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1045	1282	146	0.211	65	2000	2	0.5	39.0	131	1032	0	50400	0	0	0	0	0	001	c	
1046	146	1678	0.111	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1047	1680	1273	0.057	65	2000	3	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1048	1274	1798	0.144	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1049	1800	603	0.200	65	2000	2	0.5	39.0	131	1028	0	50400	0	0	57	1001	0	0	0	c
1050	603	178	0.126	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1051	178	617	0.126	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1052	617	1550	0.026	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1053	1552	25	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	001	c	
1054	26	1748	0.400	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	001	c	
1055	1754	609	0.191	65	2000	2	0.5	39.0	131	1062	0	50400	0	0	54	1001	0	0	0	c
1056	609	610	0.191	65	2000	2	0.5	39.0	131	0	0	0	0	0	58	1001	0	0	0	c
1057	610	1524	0.082	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1058	1530	1272	0.180	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1059	1275	1526	0.180	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1060	1528	610	0.182	65	2000	2	0.5	39.0	131	1057	0	50400	0	0	58	1001	0	0	0	c
1061	610	609	0.191	65	2000	2	0.5	39.0	131	0	0	0	0	0	54	1001	0	0	0	c
1062	609	1750	0.091	65	2000	2	0.5	39.0	131	0	0	0	0	0	0	0	0	001	c	
1063	1752	26	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	001	c	
1064	27	1294	0.400	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	001	c	
1065	1300	611	0.203	55	1800	1	0.5	33.0	131	1084	0	50400	0	0	55	2	0	0	0	c
1066	611	179	0.091	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	0	001	c	
1067	179	612	0.091	55	1800	1	0.5	33.0	131	0	0	0	0	0	59	1001	0	0	0	c
1068	612	1516	0.087	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	0	001	c	
1069	1522	1442	0.305	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	0	001	c	
1070	1448	613	0.123	55	1800	1	0.5	33.0	131	1079	0	50400	0	0	0	0	0	001	c	

1138	656	630	0.057	55	1800	2	0.5	33.0	131	0	0	0	0	0	10001	0	001	c
1139	591	615	0.461	55	1800	1	0.5	33.0	131	0	0	0	0	0	50	2	001	c
1140	615	595	0.378	55	1800	1	0.5	33.0	131	0	0	0	0	0	51	2	001	c
3141	595	599	0.775	55	1800	1	0.5	33.0	131	0	0	0	0	0	52	2	001	c
1142	599	616	0.231	55	1800	1	0.5	33.0	131	0	0	0	0	0	53	1	0 0 1	c
1143	616	617	0.448	55	1800	1	0.5	33.0	131	0	0	0	0	0	10001	0	001	c
1144	617	616	0.448	55	1800	1	0.5	33.0	131	0	0	0	0	0	53	1	0 0 1	c
1145	616	599	0.231	55	1800	1	0.5	33.0	131	0	0	0	0	0	52	2	001	c
1146	599	595	0.775	55	1800	1	0.5	33.0	131	0	0	0	0	0	51	2	001	c
1147	595	615	0.378	55	1800	1	0.5	33.0	131	0	0	0	0	0	50	2	001	c
1148	615	591	0.461	55	1800	1	0.5	33.0	131	0	0	0	0	0	49	2	001	c
1149	578	131	0.251	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	001	c
1150	131	1177	0.251	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	001	c
1151	1180	131	0.251	55	1800	1	0.5	33.0	131	1150	0	50400	0	0	0	0	001	c
1152	131	578	0.251	55	1800	1	0.5	33.0	131	0	0	0	0	0	10001	0	001	c
1153	617	609	0.500	55	1800	1	0.5	33.0	131	0	0	0	0	0	54	2	001	c
1154	609	611	0.593	55	1800	1	0.5	33.0	131	0	0	0	0	0	55	1	0 0 1	c
1155	611	862	0.335	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	001	c
1156	646	867	0.050	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	001	c
1157	871	620	0.338	55	1800	1	0.5	33.0	131	1159	0	50400	0	0	469	2	201	c
1158	112	620	0.500	-1	2000	5	0.5	8.0	131	0	0	0	0	0	469	3	301	c
1159	620	868	0.238	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	001	c
1160	646	863	0.120	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	001	c
1161	864	611	0.435	55	1800	1	0.5	33.0	131	1155	0	50400	0	0	55	1	0 0 1	c
1162	611	609	0.593	55	1800	1	0.5	33.0	131	0	0	0	0	0	54	2	001	c
1163	609	617	0.500	55	1800	1	0.5	33.0	131	0	0	0	0	0	10001	0	001	c
1164	1061	1069	0.059	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	001	c
1165	1072	1058	0.005	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	001	c
1166	578	130	0.410	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	001	c
1167	130	1012	0.310	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	001	c
1168	1016	130	0.410	55	1800	1	0.5	33.0	131	1167	0	50400	0	0	0	0	001	c
1169	130	578	0.410	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	001	c
1170	540	559	0.251	55	1800	1	0.5	33.0	131	0	0	0	0	0	300	2	001	c
1171	559	540	0.251	55	1800	1	0.5	33.0	131	0	0	0	0	0	10001	0	001	c
1172	647	564	0.035	30	1800	1	0.5	18.0	131	0	0	0	0	0	0	0	0 0 1	c
1173	564	563	0.346	30	1800	1	0.5	18.0	131	0	0	0	0	0	112	2	0 0 1	c
1174	563	564	0.346	30	1800	1	0.5	18.0	131	0	0	0	0	0	0	0	0 0 1	c
1175	647	562	0.113	30	1800	1	0.5	18.0	131	0	0	0	0	0	110	2	0 0 1	c
1176	113	196	0.183	-1	2200	5	0.0	8.0	131	0	0	0	0	0	0	0	0 0 1	c
1177	196	197	0.061	100	2200	3	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1178	197	198	0.183	100	1900	4	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1179	198	199	0.442	100	1833	6	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1180	199	200	0.213	100	1800	7	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1181	200	201	0.350	100	1943	7	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1182	201	202	0.091	100	1800	7	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1183	202	203	0.503	100	1943	7	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1184	203	204	0.183	100	1900	7	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1185	204	205	0.030	100	1900	7	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1186	205	206	0.030	100	1900	6	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1187	206	192	0.122	100	1900	6	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1188	192	193	0.137	100	1886	7	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1189	193	207	0.213	100	1900	6	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1190	207	208	0.168	100	1850	6	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1191	208	194	0.046	100	1933	6	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1192	194	195	0.122	100	1914	7	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1193	195	209	0.472	100	1933	6	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1194	209	210	0.152	100	2000	6	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1195	210	211	0.533	100	1980	5	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1196	211	212	0.884	100	1950	6	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1197	212	213	0.076	100	2000	5	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1198	213	214	0.594	100	2000	5	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1199	214	215	0.808	100	2040	5	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1200	215	216	0.190	100	1960	5	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1201	216	217	0.038	100	1960	5	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1202	217	218	0.091	100	1900	6	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1203	218	219	0.259	100	2000	5	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1204	219	220	1.082	100	1960	5	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c

1205	220	221	0.320	100	2000	5	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1206	221	222	0.229	100	2000	4	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1207	222	223	0.427	100	2000	4	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1208	223	224	0.549	100	2025	4	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1209	224	225	0.594	100	1820	5	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1210	225	226	1.021	300	2050	4	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1211	226	227	0.259	100	1880	5	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1212	227	228	0.350	100	2100	4	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1213	228	229	0.945	100	2100	4	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1214	229	230	0.381	100	1900	4	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1215	230	231	0.625	100	1800	5	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1216	231	232	0.503	100	1925	4	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1217	232	233	0.312	100	2000	3	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1218	233	106	0.152	100	2000	3	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1219	114	234	0.500	-1	2000	5	-0.1	8.0	131	0	0	0	0	0	0	0	001	c
1220	234	235	0.594	100	2067	3	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1221	235	236	0.884	100	2000	5	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1222	236	237	0.427	100	2050	4	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1223	237	238	0.503	100	2000	5	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1224	238	239	0.655	100	2000	5	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1225	239	240	0.350	100	1980	5	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1226	240	241	0.640	100	1980	5	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1227	241	242	0.091	100	1980	5	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1228	242	243	0.579	100	2000	5	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1229	243	244	0.305	100	2000	5	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1230	244	245	0.107	100	2000	6	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1231	245	246	0.091	100	2000	5	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1232	246	247	0.396	100	1980	5	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1233	247	248	0.122	100	1980	5	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1234	248	249	0.472	100	1960	5	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1235	249	250	1.158	100	2000	5	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1236	250	251	0.213	100	2100	5	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1237	251	252	0.107	100	2050	6	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1238	252	253	0.061	100	1960	5	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1239	253	254	0.274	100	2000	5	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1240	254	255	0.762	100	2040	5	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1241	255	256	0.442	100	2040	5	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1242	256	257	0.289	100	2040	5	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1243	257	258	0.838	100	1967	6	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1244	258	259	0.503	100	2080	5	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1245	259	260	0.137	100	1933	6	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1246	260	261	0.366	100	1933	6	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1247	261	262	0.350	100	1942	7	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1248	262	263	0.259	100	1942	7	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1249	263	264	0.122	100	1942	7	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1250	264	265	0.381	100	1967	6	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1251	265	266	0.091	100	1967	6	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1252	266	267	0.107	100	1942	7	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1253	267	268	0.549	100	1942	7	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1254	268	269	0.046	100	1942	7	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1255	269	270	0.122	100	1914	7	0.0	60.0	131	0	0	0	0	0	0	0	0 0 1	c
1256	270	271	0.488	100	1933	6	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1257	271	272	0.137	100	1933	6	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1258	272	273	0.381	100	1900	6	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1259	273	274	0.061	100	1910	5	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1260	274	107	0.206	100	1900	6	0.0	60.0	131	0	0	0	0	0	0	0	001	c
1261	200	275	0.102	64	2000	4	0.0	38.4	131	0	0	0	0	0	0	0	001	c
1262	275	624	0.119	64	1500	2	0.0	38.4	131	0	0	0	0	0	473	2	001	c
1263	202	276	0.133	64	2000	4	0.0	38.4	131	0	0	0	0	0	0	0	001	c
1264	276	885	0.005	64	1500	2	0.0	38	4 131	1882	0	50400	0	0	0	0	001	c
1265	206	277	0.073	64	2000	4	0.0	38	4 131	0	0	0	0	0	0	0	001	c
1266	277	897	0.024	64	1500	2	0.0	38	4 131	1896	0	50400	0	0	0	0	0 0 1	c
1267	208	278	0.108	75	2000	4	0.0	45	0 131	0	0	0	0	0	0	0	001	c
1268	278	1318	0.145	75	1500	2	0.0	45	0 131	2453	0	50400	0	0	0	0	001	c
1269	210	279	0.152	75	2000	4	0.0	45	0 131	0	0	0	0	0	0	0	001	c
1270	279	1346	0.005	75	1500	2	0.0	45	0 131	2482	0	50400	0	0	0	0	0 0 1	c
1271	212	280	0.190	75	2000	4	0.0	45	0 131	0	0	0	0	0	0	0	001	c

1272	280	773	0.053	75	1800	3	0.0	45.0	131	1740	0	50400	0	0	0	0	0	0	1	c
1273	215	281	0.147	75	2000	4	0.0	45.0	131	0	0	0	0	0	0	0	0	0	1	c
1274	281	657	0.119	75	1500	2	0.0	45.0	131	0	0	0	0	0	0	0	0	0	1	c
1275	695	582	0.087	64	1500	1	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1276	220	282	0.160	64	2000	4	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1277	282	1359	0.154	64	1500	2	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1278	221	283	0.192	64	2000	4	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1279	283	1915	0.349	64	1800	2	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1280	225	284	0.221	64	2000	4	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1281	284	933	0.035	64	1500	2	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1282	227	285	0.104	64	2000	4	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1283	285	904	0.153	64	1500	2	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1284	229	286	0.108	75	2000	4	0.0	45.0	131	0	0	0	0	0	0	0	0	0	1	c
1285	286	287	0.135	75	1500	2	0.0	45.0	131	0	0	0	0	0	0	0	0	0	1	c
1286	287	744	0.076	64	1500	3	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1287	696	288	0.301	64	2300	1	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1288	288	186	0.235	64	2000	2	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1289	642	187	0.390	64	2000	2	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1290	236	289	0.192	64	2000	4	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1291	289	1324	0.099	64	1800	2	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1292	691	629	0.411	64	1500	1	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1293	692	639	0.370	64	1500	1	0.0	38.4	131	782	0	50400	0	0	0	0	0	0	1	c
1294	693	638	0.072	64	2000	1	0.0	38.4	131	775	0	50400	0	0	0	0	0	0	1	c
1295	246	290	0.270	64	2000	4	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1296	290	1364	0.306	64	1500	2	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1297	250	291	0.159	64	2000	4	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1298	291	584	0.095	64	1500	2	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1299	694	659	0.072	64	2000	1	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1300	255	292	0.132	64	2000	4	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1301	292	1351	0.114	64	1500	2	0.0	38.4	131	2487	0	50400	0	0	0	0	0	0	1	c
1302	258	293	0.132	64	2000	4	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1303	293	1339	0.053	64	1500	2	0.0	38.4	131	2473	0	50400	0	0	0	0	0	0	1	c
1304	260	294	0.253	64	2000	4	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1305	294	1333	0.053	64	1500	2	0.0	38.4	131	2466	0	50400	0	0	0	0	0	0	1	c
1306	263	295	0.139	64	2000	4	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1307	295	890	0.053	64	1500	2	0.0	38.4	131	1892	0	50400	0	0	0	0	0	0	1	c
1308	267	296	0.101	64	2000	4	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1309	296	878	0.084	64	2000	2	0.0	38.4	131	1873	0	50400	0	0	0	0	0	0	1	c
1310	270	297	0.112	64	2000	4	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1311	297	872	0.026	64	1500	2	0.0	38.4	131	1866	0	50400	0	0	0	0	0	0	1	c
1312	272	108	0.500	-1	2000	5	0.0	8.0	131	0	0	0	0	0	0	0	0	0	1	c
1313	641	188	0.198	64	1900	2	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1314	189	298	0.141	64	2000	3	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1315	298	198	0.276	64	2000	3	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1316	115	299	0.247	-1	2000	5	0.0	8.0	131	0	0	0	0	0	0	0	0	0	1	c
1317	299	199	0.184	64	2000	1	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1318	624	662	0.092	64	1500	2	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1319	662	201	0.112	64	1500	1	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1320	887	663	0.101	64	1500	2	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1321	663	203	0.080	64	1500	1	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1322	899	664	0.018	64	1500	2	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1323	664	207	0.139	64	1500	1	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1324	1320	665	0.052	64	1500	2	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1325	665	209	0.160	64	1500	1	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1326	1348	666	0.143	59	1500	2	0.0	35.4	131	0	0	0	0	0	0	0	0	0	1	c
1327	666	211	0.163	59	1500	1	0.0	35.4	131	0	0	0	0	0	0	0	0	0	1	c
1328	775	667	0.060	64	1500	2	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1329	667	214	0.160	64	1500	1	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1330	658	668	0.145	64	1500	1	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1331	668	217	0.072	64	1500	1	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1332	581	669	0.185	75	1500	2	0.0	45.0	131	0	0	0	0	0	0	0	0	0	1	c
1333	669	219	0.166	75	1500	1	0.0	45.0	131	0	0	0	0	0	0	0	0	0	1	c
1334	1368	670	0.235	80	1500	2	0.0	48.0	131	0	0	0	0	0	0	0	0	0	1	c
1335	670	223	0.240	80	1500	1	0.0	48.0	131	0	0	0	0	0	0	0	0	0	1	c
1336	1374	671	0.213	64	1500	2	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1337	671	224	0.132	64	1500	1	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c
1338	630	672	0.120	64	1500	2	0.0	38.4	131	0	0	0	0	0	0	0	0	0	1	c

1414	562	647	0.113	30	1800	2	0.5	18.0	131	0	0	0	0	0	111	1	001	c
1415	564	647	0.035	30	1800	2	0.5	18.0	131	0	0	0	0	0	111	1	0 0 1	c
1416	1206	1118	0.098	55	1800	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
1417	1124	647	0.113	55	1800	2	0.5	33.0	131	1418	0	50400	0	0	111	2	001	c
1418	647	1120	0.013	55	1800	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
1419	1122	1202	0.098	55	1800	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
1420	563	570	0.750	35	1800	2	0.5	21.0	131	0	0	0	0	0	113	2	001	c
1421	570	563	0.750	35	1800	2	0.5	21.0	131	0	0	0	0	0	112	2	001	c
1422	648	1208	0.005	55	1800	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
1423	1212	648	0.060	55	1800	2	0.5	33.0	131	1422	0	50400	0	0	0	0	001	c
1424	1214	649	0.446	55	1800	2	0.5	33.0	131	1425	0	50400	0	0	0	0	001	c
1425	649	1210	0.346	55	1800	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
1426	651	1200	0.005	55	1800	2	0.5	33.0	131	0	0	0	0	0	0	0 0 0 0 1	c	
1427	1204	651	0.100	55	1800	2	0.5	33.0	131	0	0	0	0	0	0	0	001	c
1428	805	944	0.850	65	1800	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
1429	947	801	0.750	65	1800	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
1430	1432	1410	0.640	65	1800	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
1431	1414	1428	0.640	65	1800	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
1432	653	745	0.005	65	1800	2	0.5	39.0	131	0	0	0	0	0	0	0	001	c
1433	749	653	0.100	65	1800	2	0.5	39.0	131	1432	0	50400	0	0	0	0	001	c
1434	655	656	0.100	55	1800	1	0.5	33.0	131	0	0	0	0	0	0	0	001	c
1435	119	50	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1436	50	119	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1437	120	51	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1438	51	120	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1439	121	51	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1440	51	121	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1441	121	52	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1442	52	121	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1443	122	52	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1444	52	122	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1445	123	53	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1446	53	123	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1447	124	53	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1448	53	124	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1449	125	54	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1450	54	125	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1451	126	80	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1452	80	126	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1453	184	83	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1454	83	184	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1455	183	82	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1456	82	183	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1457	120	82	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1458	82	120	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1459	127	79	0.300	-1	2000	5	0.5	a.0	131	0	0	0	0	0	0	0	000	c
1460	79	127	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1461	128	55	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1462	55	128	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1463	182	78	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1464	78	182	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1465	153	77	0.300	-1	2000	5	0.5	a.0	131	0	0	0	0	0	0	0	000	c
1466	77	153	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1467	153	76	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1468	76	153	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1469	152	76	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1470	76	152	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1471	152	75	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1472	75	152	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1473	151	75	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1474	75	151	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1475	150	74	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1476	74	150	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1477	129	74	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1478	74	129	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1479	129	56	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c
1480	56	129	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	000	c

1481	130	56	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1482	56	130	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1483	130	57	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1484	57	130	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1485	131	57	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	a
1486	57	131	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1487	132	58	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1488	58	132	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1489	133	58	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1490	58	133	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1491	133	59	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1492	59	133	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1493	134	59	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1494	59	134	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1495	149	73	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1496	73	149	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1497	132	73	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1498	73	132	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1499	136	61	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1500	61	136	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1501	137	62	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1502	62	137	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1503	138	62	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1504	62	138	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1505	135	60	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1506	60	135	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1507	140	63	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1508	63	140	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1509	139	63	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1510	63	139	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1511	141	64	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1512	64	141	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1513	142	65	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1514	65	142	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1515	143	66	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1516	66	143	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1517	144	66	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1518	66	144	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1519	144	67	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1520	67	144	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1521	145	67	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1522	67	145	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1523	145	105	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1524	105	145	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1525	146	105	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1526	105	146	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1527	146	68	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1528	68	146	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1529	147	68	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1530	68	147	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1531	181	104	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1532	104	181	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1533	179	104	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1534	104	179	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1535	179	103	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	a
1536	103	179	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1537	178	103	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1538	103	178	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1539	178	102	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1540	102	178	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1541	177	102	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1542	102	177	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1543	177	101	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1544	101	177	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1545	147	69	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1546	69	147	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c
1547	180	69	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	0	c

1548	69	180	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1549	180	100	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1550	100	180	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1551	176	100	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1552	100	176	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1553	175	99	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1554	99	175	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1555	176	99	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1556	99	176	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1557	175	98	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1558	98	175	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1559	185	98	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1560	98	185	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	d
1561	185	72	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1562	72	185	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1563	135	72	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1564	72	135	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1565	148	70	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1566	70	148	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1567	148	71	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1568	71	148	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	d
1569	138	71	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1570	71	138	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	d
1571	174	97	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1572	97	174	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1573	149	92	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1574	92	149	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1575	172	92	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1576	92	172	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1577	173	96	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1578	96	173	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1579	170	95	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1580	95	170	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1581	171	95	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1582	95	171	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1583	171	91	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1584	91	171	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1585	150	91	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1586	91	150	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1587	169	90	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1588	90	169	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1589	168	90	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1590	90	168	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1591	168	89	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1592	89	168	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	d
1593	167	89	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1594	89	167	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1595	167	94	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1596	94	167	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1597	166	94	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1598	94	166	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1599	163	93	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1600	93	163	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1601	164	93	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1602	93	164	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1603	164	88	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1604	88	164	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1605	165	88	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1606	88	165	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1607	162	87	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	d
1608	87	162	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1609	161	80	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	d
1610	80	161	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1611	154	81	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	d
1612	81	154	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1613	161	81	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c
1614	81	161	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c

1615	159	81	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c		
1616	81	159	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c		
1617	159	86	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c		
1618	86	159	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c		
1619	160	86	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c		
1620	86	160	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	d!		
1621	155	84	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c		
1622	84	155	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c		
1623	156	84	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	d!		
1624	84	156	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c		
1625	157	85	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c		
1626	85	157	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c		
1627	158	85	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c		
1628	85	158	0.300	-1	2000	5	0.5	8.0	131	0	0	0	0	0	0	0	0	0	c		
1629	127	533	0.179	65	2000	1	0.5	39.0	131	0	0	0	0	0	138	1	0	0	1	c	
1630	533	127	0.179	65	2000	1	0.5	39.0	131	0	0	0	0	0	0	0	0	0	1	c	
1401	240	691	0.050	70	2000	4	-0.1	42.0	131	0	0	0	0	0	0	0	0	0	0	1	c
1402	243	692	0.050	70	2000	4	-0.1	42.0	131	0	0	0	0	0	0	0	0	0	0	1	c
1403	245	693	0.050	70	2000	4	-0.1	42.0	131	0	0	0	0	0	0	0	0	0	0	1	c
1404	252	694	0.050	70	2000	4	-0.1	42.0	131	0	0	0	0	0	0	0	0	0	0	1	c
180	218	695	0.050	70	2000	4	-0.1	42.0	131	0	0	0	0	0	0	0	0	0	0	1	d!
1383	231	696	0.050	70	2000	4	-0.1	42.0	131	0	0	0	0	0	0	0	0	0	0	1	c
1389	273	641	0.050	70	2000	4	-0.1	42.0	131	0	0	0	0	0	0	0	0	0	0	1	c
1390	232	642	0.050	70	2000	4	-0.1	42.0	131	0	0	0	0	0	0	0	0	0	0	1	c
1631	697	704	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	0	1	587R
1632	697	703	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	133	1	0	0	1	587T	
1633	697	702	0.100	30	1601	1	0.5	18.0	131	0	0	0	1638	0	133	1	0	0	1	587L	
1634	698	701	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	685R	
1635	698	704	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	133	2	0	0	1	685T	
1636	698	703	0.100	30	1601	1	0.5	18.0	131	0	0	0	1641	0	133	2	0	0	1	685L	
1637	699	702	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	595R	
1638	699	701	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	133	1	0	0	1	595T	
1639	699	704	0.100	30	1601	1	0.5	18.0	131	0	0	0	1632	0	133	1	0	0	1	595L	
1640	700	703	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	693R	
1641	700	702	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	133	2	0	0	1	693T	
1642	700	701	0.100	30	1601	1	0.5	18.0	131	0	0	0	1635	0	133	2	0	0	1	693L	
1643	705	712	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	285R	
1644	705	711	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	153	1	0	0	1	285T	
1645	705	710	0.100	30	1601	1	0.5	18.0	131	0	0	0	1650	0	153	1	2	0	1	285L	
1646	706	709	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	744R	
1647	706	712	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	153	4	0	0	1	744T	
1648	706	711	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	153	4	0	0	1	744L	
1649	707	710	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	359R	
1650	707	709	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	153	1	0	0	1	359T	
1651	707	712	0.100	30	1601	1	0.5	18.0	131	0	0	0	1644	0	153	1	2	0	1	359L	
1652	708	711	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	748R	
1653	708	710	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	153	3	0	0	1	748T	
1654	708	709	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	153	3	0	0	1	748L	
1655	713	720	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	57R	
1656	713	719	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	3	0	1	0	0	1	57T
1657	713	718	0.100	30	1601	1	0.5	18.0	131	0	0	0	1662	0	3	0	1	0	0	1	57L
1658	714	717	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	996R	
1659	714	720	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	30	2	0	0	1	996T	
1660	714	719	0.100	30	1601	1	0.5	18.0	131	0	0	0	1665	0	30	2	0	0	1	996L	
1661	715	718	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	77R	
1662	715	717	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	3	0	1	0	0	1	77T
1663	715	720	0.100	30	1601	1	0.5	18.0	131	0	0	0	1656	0	3	0	1	0	0	1	77L
1664	716	719	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1018R	
1665	716	718	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	30	2	0	0	1	1018T	
1666	716	717	0.100	30	1601	1	0.5	18.0	131	0	0	0	1659	0	30	2	0	0	1	1018L	
1667	721	728	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	29R	
1668	721	727	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	71	0	0	0	1	29T	
1669	721	726	0.100	30	1601	1	0.5	18.0	131	0	0	0	1674	0	7	1	0	0	1	29L	
1670	722	725	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	798R	
1671	722	728	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	7	2	0	0	1	798T	
1673	723	726	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	105R	
1674	723	725	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	7	1	0	0	1	105T	

1675	723	728	0.100	30	1601	1	0.5	18.0	131	0	0	0	1668	0	71	0	0	1	105L
1676	724	727	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	818R
1677	724	726	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	7	2	0	1	818T
1678	724	725	0.100	30	1601	1	0.5	18.0	131	0	0	0	1671	0	7	2	0	1	818L
1679	729	736	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	494R
1680	729	735	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	441	1	001		494T
1681	729	734	0.100	30	1601	1	0.5	18.0	131	0	0	0	1686	0	441	1	0	1	494L
1682	730	733	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	1037R
1683	730	736	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	441	2	0	1	1037T
1684	730	735	0.100	30	1601	1	0.5	18.0	131	0	0	0	1689	0	441	2	0	1	1037L
1685	731	734	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	500R
1686	731	733	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	441	1	0	1	500T
1687	731	736	0.100	30	1601	1	0.5	18.0	131	0	0	0	1680	0	441	1	0	1	500L
1688	732	735	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	1039R
1689	732	734	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	441	2	0	1	1039T
1690	732	733	0.100	30	1601	1	0.5	18.0	131	0	0	0	1683	0	441	2	0	1	1039L
1691	737	741	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	105	1001			545T
1692	738	740	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	105	1	0	1	579T
1693	739	741	0.100	50	2800	2	0.5	30.0	131	0	0	0	0	0	0	0	0	1	1386R
1694	739	740	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	105	2	0	1	1386L
1695	742	749	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	550R
1696	742	748	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	302	2	0	1	550T
1697	742	747	0.100	30	1601	1	0.5	18.0	131	0	0	0	1702	0	302	2	0	1	550L
1698	743	746	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	683R
1699	743	749	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	302	1	0	1	683T
1700	743	748	0.100	30	1601	2	0.5	18.0	131	0	0	0	1705	0	302	3	101		683L
1701	744	747	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	1286R
1702	744	746	0.100	50	1800	2	0.0	30.0	131	0	0	0	0	0	302	2	0	1	1286T
1703	744	749	0.100	30	1601	1	0.0	18.0	131	0	0	0	1696	0	302	2	0	1	1286L
1704	745	748	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	1432R
1705	745	747	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	302	1	0	1	1432T
1706	745	746	0.100	30	1601	1	0.5	18.0	131	0	0	0	1699	0	302	1	0	1	1432L
1707	750	755	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	295	1	0	1	146T
1708	750	754	0.100	30	1601	1	0.5	18.0	131	0	0	0	1712	0	295	1	0	1	146L
1709	751	753	0.100	50	2800	2	0.5	30.0	131	0	0	0	0	0	0	0	0	1	705R
1710	751	755	0.100	30	1601	2	0.5	18.0	131	0	0	0	0	0	295	2	0	1	705L
1711	752	754	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	262R
1712	752	753	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	295	1	0	1	262T
1713	756	763	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	1395R
1714	756	762	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	304	2	0	1	1395T
1715	756	761	0.100	30	1601	1	0.5	18.0	131	0	0	0	1720	0	304	2	0	1	1395L
1716	757	760	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	684R
1717	757	763	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	304	1	0	1	684T
1718	757	762	0.100	30	1601	1	0.5	18.0	131	0	0	0	1723	0	304	1	0	1	684L
1719	758	761	0.100	50	2800	2	0.5	30.0	131	0	0	0	0	0	0	0	0	1	574R
1720	758	760	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	304	2	0	1	574T
1721	758	763	0.100	30	1601	1	0.5	18.0	131	0	0	0	1714	0	304	2	0	1	574L
1722	759	762	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	694R
1723	759	761	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	304	1	0	1	694I
1724	759	760	0.100	30	1601	1	0.5	18.0	131	0	0	0	1717	0	304	1	0	1	694L
1725	764	771	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	303R
1726	764	770	0.100	50	1800	4	0.5	30.0	131	0	0	0	0	0	165	1	0	1	303T
1727	764	769	0.100	30	1601	1	0.5	18.0	131	0	0	0	1732	0	165	1	0	1	303L
1728	765	768	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	534R
1730	765	770	0.100	30	1601	2	0.5	18.0	131	0	0	0	0	0	165	3	2	1	534L
1731	766	769	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	341R
1732	766	768	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	165	1	0	1	341T
1733	766	771	0.100	30	1601	1	0.5	18.0	131	0	0	0	1726	0	165	1	0	1	341L
1734	767	770	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	1112R
1737	772	775	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	911R
1738	772	777	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	216	1	0	1	911T
1739	773	776	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	1272R
1740	773	775	0.100	50	1800	1	0.0	30.0	131	0	0	0	0	0	216	2	0	1	1272T
1741	773	777	0.100	30	1601	2	0.0	18.0	131	0	0	0	0	0	216	2	0	1	1272L
1742	774	776	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	216	1	0	1	917T
1743	774	775	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	216	3	1	0	917L
1744	778	785	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001		141R

1745	778	784	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	291	1	0	0	1	141T
1746	778	783	0.100	30	1601	1	0.5	18.0	131	0	0	0	1751	0	291	1	0	0	1	141L
1747	779	782	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	680R
1748	779	785	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	291	3	0	0	1	680T
1749	779	784	0.100	30	1601	1	0.5	18.0	131	0	0	0	1754	0	291	3	0	0	1	680L
1750	780	783	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	267R
1751	780	782	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	291	4	1	0	1	267T
1752	780	785	0.100	30	1601	2	0.5	18.0	131	0	0	0	1745	0	291	4	1	0	1	267L
1753	781	784	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	698R
1754	781	783	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	291	3	0	0	1	698T
1755	781	782	0.100	30	1601	2	0.5	18.0	131	0	0	0	1748	0	291	2	3	0	1	698L
1756	786	789	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	305	1	0	0	1	1397L
1757	787	789	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	695R
1758	787	788	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	305	2	0	0	1	695T
1759	790	797	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	549R
1760	790	796	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	307	1	0	0	1	549T
1761	790	795	0.100	30	1601	1	0.5	18.0	131	0	0	0	1766	0	307	1	0	0	1	549L
1762	791	794	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	673R
1763	791	797	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	307	2	0	0	1	673T
1764	791	796	0.100	30	1601	1	0.5	18.0	131	0	0	0	1769	0	307	2	0	0	1	673L
1765	792	795	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	575R
1766	792	794	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	307	1	0	0	1	575T
1767	792	797	0.100	30	1601	1	0.5	18.0	131	0	0	0	1760	0	307	1	0	0	1	575L
1768	793	796	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1396R
1769	793	795	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	307	2	0	0	1	1396T
1770	793	794	0.100	30	1601	1	0.5	18.0	131	0	0	0	1763	0	307	2	0	0	1	1396L
1771	798	805	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	276R
1772	798	804	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	146	1	0	0	1	276T
1773	798	803	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	146	3	0	0	1	276L
1774	799	802	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	656R
1775	799	805	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	146	2	0	0	1	656T
1776	799	804	0.100	30	1601	1	0.5	18.0	131	0	0	0	1781	0	146	2	0	0	1	656L
1777	800	803	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	368R
1778	800	802	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	146	1	0	0	1	368T
1779	800	805	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	146	3	0	0	1	368L
1780	801	804	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1429R
1781	801	803	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	146	2	0	0	1	1429T
1782	801	802	0.100	30	1601	1	0.5	18.0	131	0	0	0	1775	0	146	2	0	0	1	1429L
1783	806	808	0.100	70	2800	2	0.5	42.0	131	0	0	0	0	0	0	0	0	0	1	559R
1784	806	810	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	141	1	0	0	1	559T
1785	807	809	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	141	1	0	0	1	565T
1786	807	808	0.100	30	1601	1	0.5	18.0	131	0	0	0	1784	0	141	2	101		565L	
1787	811	818	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001			558R
1788	811	817	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	140	2	0	0	1	558T
1789	811	816	0.100	30	1601	1	0.5	18.0	131	0	0	0	1794	0	140	2	0	0	1	558L
1790	812	815	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	740R
1791	812	818	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	140	1	0	0	1	740T
1792	812	817	0.100	30	1601	1	0.5	18.0	131	0	0	0	1797	0	140	3	101			740L
1793	813	816	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	566R
1794	813	815	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	140	2	0	0	1	566T
1795	813	818	0.100	30	1601	1	0.5	18.0	131	0	0	0	1788	0	140	2	0	0	1	566L
1796	814	817	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001			749R
1797	814	816	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	140	1	0	0	1	749T
1798	814	815	0.100	30	1601	1	0.5	18.0	131	0	0	0	1791	0	140	1	0	0	1	749L
1799	819	826	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	613R
1800	819	825	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	124	2	0	0	1	613T
1801	819	824	0.100	30	1601	1	0.5	18.0	131	0	0	0	1806	0	124	2	0	0	1	613L
1802	820	823	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001			736R
1803	820	826	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	124	1	0	0	1	736T
1804	820	825	0.100	30	1601	1	0.5	18.0	131	0	0	0	1809	0	124	1	0	0	1	736L
1805	821	824	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	619R
1806	821	823	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	124	2	0	0	1	619T
1807	821	826	0.100	30	1601	1	0.5	18.0	131	0	0	0	1800	0	124	2	0	0	1	619L
1808	822	825	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	753R
1809	822	824	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	124	1	0	0	1	753T
1810	822	823	0.100	30	1601	1	0.5	18.0	131	0	0	0	1803	0	124	1	0	0	1	753L
1811	827	834	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	175R

1812	827	833	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	79	1	4	0	1	175T
1814	828	831	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	533R
1815	828	834	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	79	3	0	0	1	533T
1816	828	833	0.100	30	1601	1	0.5	18.0	131	0	0	0	1821	0	79	2	3	0	1	533L
1817	829	832	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	1	233R
1818	829	831	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	79	1	4	0	1	233T
1820	830	833	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	1	535R
1821	830	832	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	79	3	0	0	1	535T
1823	835	839	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	1	544R
1824	835	838	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	104	1	0	0	1	544T
1825	836	837	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	104	1	0	0	1	580T
1826	836	839	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	104	1	0	0	1	580L
1827	840	847	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	1	152R
1828	840	846	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	62	1	0	0	1	152T
1829	840	845	0.100	30	1601	1	0.5	18.0	131	0	0	0	1834	0	62	1	0	0	1	152L
1831	841	847	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	62	3	0	0	1	718T
1832	841	846	0.100	30	1601	1	0.5	18.0	131	0	0	0	1837	0	62	3	0	0	1	718L
1833	842	845	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	1	256R
1834	842	844	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	62	1	0	0	1	256T
1835	842	847	0.100	30	1601	1	0.5	18.0	131	0	0	0	1828	0	62	1	0	0	1	256L
1836	843	846	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	1	722R
1837	843	845	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	62	3	0	0	1	722T
1838	843	844	0.100	30	1601	1	0.5	18.0	131	0	0	0	1831	0	62	2	3	0	1	722L
1839	848	855	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	1	49R
1840	848	854	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	23	1	0	0	1	49T
1841	848	853	0.100	30	1601	1	0.5	18.0	131	0	0	0	1846	0	23	1	0	0	1	49L
1842	849	852	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	1	931R
1843	849	855	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	23	2	0	0	1	931T
1844	849	854	0.100	30	1601	1	0.5	18.0	131	0	0	0	1849	0	23	2	0	0	1	931L
1845	850	853	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	1	85R
1846	850	852	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	23	1	0	0	1	85T
1847	850	855	0.100	30	1601	1	0.5	18.0	131	0	0	0	1840	0	23	1	0	0	1	85L
1848	851	854	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	1	946R
1849	851	853	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	23	2	0	0	1	946T
1850	851	852	0.100	30	160-1	1	0.5	18.0	131	0	0	0	1843	0	23	2	0	0	1	946L
1851	856	861	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	1	27R
1852	856	860	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	5	1	0	0	1	27T
1853	857	859	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	5	1	0	0	1	107T
1854	857	861	0.100	30	1601	1	0.5	18.0	131	0	0	0	1852	0	5	1	0	0	1	107L
1855	858	860	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	1	796R
1856	858	859	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	5	2	0	0	1	796L
1857	862	866	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	1	1155R
1858	862	865	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	467	1	0	0	1	1155T
1859	863	864	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	467	3	1	0	1	1160T
1860	863	866	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	467	3	1	0	1	1160L
1861	867	871	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	468	1	0	0	1	1156T
1862	867	870	0.100	30	1601	1	0.5	18.0	131	0	0	0	1866	0	468	1	0	0	1	1156L
1863	868	870	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	1	1159R
1864	868	869	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	468	1	0	0	1	1159T
1865	872	877	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	1	1311R
1866	872	876	0.100	50	1800	1	0.0	30.0	131	0	0	0	0	0	463	2	0	0	1	1311T
1867	872	875	0.100	30	1601	1	0.0	18.0	131	0	0	0	0	0	463	2	0	0	1	1311L
1868	873	877	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	463	3	1	0	1	1034T
1869	873	876	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	463	3	1	0	1	1034L
1870	874	876	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	1	1042R
1871	874	875	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	463	1	0	0	1	1042T
1872	878	883	0.100	50	2800	2	0.0	30.0	131	0	0	0	0	0	0	0	0	1	1	1309R
1873	878	882	0.100	50	1800	1	0.0	30.0	131	0	0	0	0	0	458	3	0	0	1	1309T
1874	878	881	0.100	30	1601	2	0.0	18.0	131	0	0	0	0	0	458	3	0	0	1	1309L
1875	879	883	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	458	2	1	0	1	1004T
1876	879	882	0.100	30	1601	2	0.5	18.0	131	0	0	0	0	0	458	2	1	0	1	1004L
1877	880	882	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	1	1010R
1878	880	881	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	458	1	0	0	1	1010T
1879	884	887	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	1	1003R
1880	884	889	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	457	1	0	0	1	1003T
1881	885	888	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1	1	1264R
1882	885	887	0.100	50	1800	1	0.0	30.0	131	0	0	0	0	0	457	3	0	0	1	1264T

1883	885	889	0.100	30	1601	1	0.0	18.0	131	0	0	0	0	0	457	3	0	0	1	1264L
1884	886	888	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	457	2	1	0	1	1011T
1885	886	887	0.100	30	1601	2	0.5	18.0	131	0	0	0	0	0	457	2	1	0	1	1011L
1886	890	895	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1307R
1887	890	894	0.100	50	1800	1	0.0	30.0	131	0	0	0	0	0	456	2	0	0	1	1307T
1888	890	893	0.100	30	1601	1	0.0	18.0	131	0	0	0	0	0	456	2	0	0	1	1307L
1889	891	895	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	456	3	1	0	1	983T
1890	891	894	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	456	3	1	0	1	983L
1891	892	894	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	9871
1892	892	893	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	456	1	0	0	1	987T
1893	896	899	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	982R
1894	896	901	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	455	1	0	0	1	982T
1895	897	900	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1266R
1896	897	899	0.100	50	1800	1	0.0	30.0	131	0	0	0	0	0	455	2	0	0	1	1266T
1897	897	901	0.100	30	1601	1	0.0	18.0	131	0	0	0	0	0	455	2	0	0	1	1266L
1898	898	900	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	455	3	1	0	1	988T
1899	898	899	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	455	3	1	0	1	988L
1900	902	908	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	606R
1901	902	907	0.100	30	1601	1	0.5	18.0	131	0	0	0	1906	0	136	2	2	0	1	606L
1902	903	906	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	570R
1903	903	908	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	136	1	0	0	1	570T
1904	904	907	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1283R
1905	904	906	0.100	50	1800	1	0.0	30.0	131	0	0	0	0	0	136	3	3	0	1	1283T
1906	904	908	0.100	30	1601	1	0.0	18.0	131	0	0	0	1899	0	136	3	3	0	1	1283L
1907	905	907	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	136	1	0	0	1	554T
1908	905	906	0.100	30	1601	1	0.5	18.0	131	0	0	0	1901	0	136	1	0	0	1	554L
1909	909	916	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	321R
1910	909	915	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	417	2	0	0	1	321T
1911	909	914	0.100	30	1601	1	0.5	18.0	131	0	0	0	1916	0	417	2	0	0	1	321L
1912	910	913	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1089R
1913	910	916	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	417	1	0	0	1	1089T
1914	910	915	0.100	30	1601	1	0.5	18.0	131	0	0	0	1919	0	417	1	0	0	1	1089L
1915	911	914	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	323R
1916	911	913	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	417	2	0	0	1	323T
1917	911	916	0.100	30	1601	1	0.5	18.0	131	0	0	0	1910	0	417	2	0	0	1	323L
1918	912	915	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1099R
1919	912	914	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	417	1	0	0	1	1099T
1920	912	913	0.100	30	1601	1	0.5	18.0	131	0	0	0	1913	0	417	1	0	0	1	1099L
1921	917	924	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	203R
1922	917	923	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	412	1	0	0	1	203T
1923	917	922	0.100	30	1601	1	0.5	18.0	131	0	0	0	1928	0	412	3	1	0	1	203L
1924	918	921	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1088R
1925	918	924	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	412	2	0	0	1	1088T
1926	918	923	0.100	30	1601	1	0.5	18.0	131	0	0	0	1931	0	412	2	0	0	1	1088L
1927	919	922	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	205R
1928	919	921	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	412	1	0	0	1	205T
1929	919	924	0.100	30	1601	1	0.5	18.0	131	0	0	0	1922	0	412	1	0	0	1	205L
1930	920	923	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1100R
1931	920	922	0.100	50	1800	4	0.5	30.0	131	0	0	0	0	0	412	2	0	0	1	1100T
1932	920	921	0.100	30	1601	1	0.5	18.0	131	0	0	0	1925	0	412	2	0	0	1	1100L
1933	925	931	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	66R
1934	925	930	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	405	1	0	0	1	66T
1935	925	929	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	405	1	0	0	1	66L
1936	926	929	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	68R
1937	926	928	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	405	1	0	0	1	68T
1938	926	931	0.100	30	1601	1	0.5	18.0	131	0	0	0	1936	0	405	1	0	0	1	68L
1939	927	930	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1103R
1940	927	929	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	405	2	0	0	1	1103T
1941	927	928	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	405	2	0	0	1	1103L
1942	932	936	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	139	1	0	0	1	739T
1943	933	935	0.100	50	2800	2	0.0	30.0	131	0	0	0	0	0	0	0	0	0	1	1281R
1944	933	936	0.100	30	1601	1	0.0	18.0	131	0	0	0	0	0	139	2	0	0	1	1281L
1945	934	935	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	139	1	0	0	1	750T
1946	937	942	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	135	1	0	0	1	591T
1947	937	941	0.100	30	1601	1	0.5	18.0	131	0	0	0	1951	0	135	1	0	0	1	591L
1948	938	940	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	553R
1949	938	942	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	135	2	0	0	1	553L

1950	939	941	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	571R
1951	939	940	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	135	1	0	0	1	571T
1952	943	948	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	418	2	0	0	1	371T
1953	943	947	0.100	30	1601	1	0.5	18.0	131	0	0	0	1957	0	418	2	0	0	1	371L
1954	944	946	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1428R
1955	944	948	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	418	1	0	0	1	1428L
1956	945	947	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	467R
1957	945	946	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	418	2	0	0	1	467T
1958	949	956	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	373R
1959	949	955	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	420	1	0	0	1	373T
1960	949	954	0.100	30	1601	1	0.5	18.0	131	0	0	0	1965	0	420	3	1	0	1	373L
1961	950	953	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	688R
1962	950	956	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	420	2	0	0	1	688T
1963	950	955	0.100	30	1601	1	0.5	18.0	131	0	0	0	1968	0	420	2	0	0	1	688L
1964	951	954	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	465R
1965	951	953	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	420	1	0	0	1	465T
1966	951	956	0.100	30	1601	1	0.5	18.0	131	0	0	0	1959	0	420	1	0	0	1	465L
1967	952	955	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	690R
1968	952	954	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	420	2	0	0	1	690T
1969	952	953	0.100	30	1601	1	0.5	18.0	131	0	0	0	1962	0	420	2	0	0	1	690L
1970	957	964	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	279R
1971	957	963	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	149	1	0	0	1	279T
1972	957	962	0.100	30	1601	1	0.5	18.0	131	0	0	0	1977	0	149	1	0	0	1	279L
1973	958	961	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1123R
1974	958	964	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	149	2	0	0	1	1123T
1975	958	963	0.100	30	1601	1	0.5	18.0	131	0	0	0	1980	0	149	2	0	0	1	1123L
1976	959	962	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	365R
1977	959	961	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	149	1	0	0	1	365T
1978	959	964	0.100	30	1601	1	0.5	18.0	131	0	0	0	1971	0	149	1	0	0	1	365L
1979	960	963	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1125R
1980	960	962	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	149	2	0	0	1	1125T
1981	960	961	0.100	30	1601	1	0.5	18.0	131	0	0	0	1974	0	149	2	0	0	1	1125L
1982	965	970	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	421	1	0	0	1	374T
1983	965	969	0.100	30	1601	1	0.5	18.0	131	0	0	0	1987	0	421	3	1	0	1	374L
1984	966	968	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1124R
1985	966	970	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	421	2	0	0	1	1124L
1986	967	969	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	464R
1987	967	968	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	421	1	0	0	1	464T
1988	971	976	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	424	1	0	0	1	377T
1989	971	975	0.100	30	1601	1	0.5	18.0	131	0	0	0	1993	0	424	1	0	0	1	377L
1990	972	974	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1107R
1991	972	976	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	424	2	0	0	1	1107L
1992	973	975	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	461R
1993	973	974	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	424	1	0	0	1	461T
1994	977	982	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	423	1	0	0	1	376T
1995	977	981	0.100	30	1601	1	0.5	18.0	131	0	0	0	1999	0	423	1	0	0	1	376L
1996	978	980	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1105R
1997	978	982	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	423	2	0	0	1	1105L
1998	979	981	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	462R
1999	979	980	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	423	1	0	0	1	462T
2000	983	988	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	425	2	0	0	1	378T
2001	983	987	0.100	30	1601	1	0.5	18.0	131	0	0	0	2005	0	425	2	0	0	1	378L
2002	984	986	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1109R
2003	984	988	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	425	3	0	0	1	1109L
2004	985	987	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	460R
2005	985	986	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	425	2	0	0	1	460T
2006	989	996	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	380R
2007	989	995	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	427	1	0	0	1	380T
2008	989	994	0.100	30	1601	1	0.5	18.0	131	0	0	0	2013	0	427	1	0	0	1	380L
2009	990	993	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	427	3	0	0	1	745R
2010	990	996	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	427	3	0	0	1	745T
2011	990	995	0.100	30	1601	1	0.5	18.0	131	0	0	0	2016	0	427	4	3	0	1	745L
2012	991	994	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	458R
2013	991	993	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	427	1	0	0	1	458T
2014	991	996	0.100	30	1601	1	0.5	18.0	131	0	0	0	2007	0	427	1	0	0	1	458L
2015	992	995	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	747R
2016	992	994	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	427	3	0	0	1	747T

2017	992	993	0.100	30	1601	1	0.5	18.0	131	0	0	0	2010	0	427	2	3	0	1	747L	
2018	997	1004	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	0	1	384R
2019	997	1003	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	430	3	0	0	0	1	384T
2020	997	1002	0.100	30	1601	1	0.5	18.0	131	0	0	0	2025	0	430	3	0	0	1		384L
2021	998	1001	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1		777R
2022	998	1004	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	430	1	0	0	1		777I
2023	998	1003	0.100	30	1601	1	0.5	18.0	131	0	0	0	2028	0	430	2	1	0	1		777L
2024	999	1002	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1		454R
2025	999	1001	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	430	3	0	0	1		454T
2026	999	1004	0.100	30	1601	1	0.5	18.0	131	0	0	0	2019	0	430	3	0	0	1		454L
2027	1000	1003	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1		779R
2028	1000	1002	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	430	1	0	0	1		779T
2029	1000	1001	0.100	30	1601	1	0.5	18.0	131	0	0	0	2022	0	430	4	1	0	1		779L
2030	1005	1010	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1		385R
2031	1005	1009	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	431	1	0	0	1		385T
2032	1006	1008	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	431	1	0	0	1		453T
2033	1006	1010	0.100	30	1601	1	0.5	18.0	131	0	0	0	2031	0	431	1	0	0	1		453L
2034	1007	1009	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1		526R
2035	1007	1008	0.100	30	1601	2	0.5	18.0	131	0	0	0	0	0	431	3	0	0	1		526L
2036	1011	1018	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1		471R
2037	1011	1017	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	209	1	0	0	1		471T
2038	1011	1016	0.100	30	1601	1	0.5	18.0	131	0	0	0	2043	0	209	1	0	0	1		471L
2039	1012	1015	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1		1167R
2040	1012	1018	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	209	2	0	0	1		1167T
2041	1012	1017	0.100	30	1601	1	0.5	18.0	131	0	0	0	2046	0	209	2	0	0	1		1167L
2042	1013	1016	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1		523R
2043	1013	1015	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	209	1	0	0	1		523T
2044	1013	1018	0.100	30	1601	1	0.5	18.0	131	0	0	0	2037	0	209	1	0	0	1		523L
2045	1014	1017	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1		854R
2046	1014	1016	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	209	2	0	0	1		854T
2047	1014	1015	0.100	30	1601	1	0.5	18.0	131	0	0	0	2040	0	209	2	0	0	1		854L
2048	1019	1026	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1		476R
2049	1019	1025	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	215	1	0	0	1		476T
2050	1019	1024	0.100	30	1601	1	0.5	18.0	131	0	0	0	2055	0	215	1	0	0	1		476L
2051	1020	1023	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1		898R
2052	1020	1026	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	215	2	0	0	1		898T
2053	1020	1025	0.100	30	1601	1	0.5	18.0	131	0	0	0	2058	0	215	2	0	0	1		898L
2054	1021	1024	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1		518R
2055	1021	1023	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	215	1	0	0	1		518T
2056	1021	1026	0.100	30	1601	1	0.5	18.0	131	0	0	0	2049	0	215	1	0	0	1		518L
2057	1022	1025	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1		900R
2058	1022	1024	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	215	2	0	0	1		900T
2059	1022	1023	0.100	30	1601	1	0.5	18.0	131	0	0	0	2052	0	215	2	0	0	1		900L
2060	1027	1032	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	220	1	0	0	1		479T
2061	1027	1031	0.100	30	1601	1	0.5	18.0	131	0	0	0	2065	0	220	1	0	0	1		479L
2062	1028	1030	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1		1114R
2063	1028	1032	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	220	2	0	0	1		1114L
2064	1029	1031	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1		515R
2065	1029	1030	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	220	1	0	0	1		515T
2066	1033	1040	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1		489R
2067	1033	1039	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	438	1	0	0	1		489T
2068	1033	1038	0.100	30	1601	1	0.5	18.0	131	0	0	0	2073	0	438	1	0	0	1		489L
2069	1034	1037	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1		984R
2070	1034	1040	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	438	2	0	0	1		984T
2071	1034	1039	0.100	30	1601	1	0.5	18.0	131	0	0	0	2076	0	438	2	0	0	1		984L
2072	1035	1038	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1		505R
2073	1035	1037	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	438	1	0	0	1		505T
2074	1035	1040	0.100	30	1601	1	0.5	18.0	131	0	0	0	2067	0	438	1	0	0	1		505L
2075	1036	1039	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1		986R
2076	1036	1038	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	438	2	0	0	1		986T
2077	1036	1037	0.100	30	1601	1	0.5	18.0	131	0	0	0	2070	0	438	2	0	0	1		986L
2078	1041	1048	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1		492R
2079	1041	1047	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	440	1	0	0	1		492T
2080	1041	1046	0.100	30	1601	1	0.5	18.0	131	0	0	0	2085	0	440	1	0	0	1		492L
2081	1042	1045	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1		1006R
2082	1042	1048	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	440	2	0	0	1		1006T
2083	1042	1047	0.100	30	1601	1	0.5	18.0	131	0	0	0	2088	0	440	2	0	0	1		1006L

2084	1043	1046	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	502R
2085	1043	1045	0.100	50	1800	2	5.5	30.0	131	0	0	0	0	0	440	1	0	0	1	502T
2086	1043	1048	0.100	30	1601	1	0.5	18.0	131	0	0	0	2079	0	440	1	0	0	1	502L
2087	1044	1047	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1008R
2088	1044	1046	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	440	2	0	0	1	1008T
2089	1044	1045	0.100	30	1601	1	0.5	18.0	131	0	0	0	2082	0	440	2	0	0	1	1008L
2090	1049	1056	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	496R
2091	1049	1055	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	442	2	0	0	1	496T
2093	1050	1053	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1093R
2094	1050	1056	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	442	1	0	0	1	1093T
2095	1050	1055	0.100	30	1601	1	0.5	18.0	131	0	0	0	2100	0	442	1	0	0	1	1093L
2096	1051	1054	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	498R
2097	1051	1053	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	442	2	0	0	1	498T
2098	1051	1056	0.100	30	1601	1	0.5	18.0	131	0	0	0	2091	0	442	2	0	0	1	498L
2099	1052	1055	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1095R
2100	1052	1054	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	442	1	0	0	1	1095T
2101	1052	1053	0.100	30	1601	1	0.5	18.0	131	0	0	0	2094	0	442	1	0	0	1	1095L
2102	1057	1062	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	461	1	0	0	1	1035T
2103	1057	1061	0.100	30	1601	1	0.5	18.0	131	0	0	0	2107	0	461	1	0	0	1	1035L
2104	1058	1060	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1165R
2105	1058	1062	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	461	2	0	0	1	1165L
2106	1059	1061	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1041R
2107	1059	1060	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	461	1	0	0	1	1041T
2108	1063	1068	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	462	1	0	0	1	1036T
2109	1063	1067	0.100	30	1601	1	0.5	18.0	131	0	0	0	2113	0	462	1	0	0	1	1036L
2110	1064	1066	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1074R
2111	1064	1068	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	462	2	0	0	1	1074L
2112	1065	1067	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1040R
2113	1065	1066	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	462	1	0	0	1	1040T
2114	1069	1074	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1164R
2115	1069	1073	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	465	2	0	0	1	1164L
2116	1070	1072	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1073R
2117	1070	1074	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	465	1	0	0	1	1073T
2118	1071	1073	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	465	1	0	0	1	1075T
2119	1071	1072	0.100	30	1601	1	0.5	18.0	131	0	0	0	2117	0	465	1	0	0	1	1075L
2120	1075	1082	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	139R
2121	1075	1081	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	289	1	0	0	1	139T
2122	1075	1080	0.100	30	1601	1	0.5	18.0	131	0	0	0	2127	0	289	1	0	0	1	139L
2123	1076	1079	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	670R
2124	1076	1082	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	289	3	0	0	1	670T
2125	1076	1081	0.100	30	1601	1	0.5	18.0	131	0	0	0	2130	0	289	2	3	0	1	670L
2126	1077	1080	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	269R
2127	1077	1079	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	289	1	0	0	1	269T
2128	1077	1082	0.100	30	1601	1	0.5	18.0	131	0	0	0	2121	0	289	1	0	0	1	269L
2129	1078	1081	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	676R
2130	1078	1080	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	289	3	0	0	1	676T
2131	1078	1079	0.100	30	1601	1	0.5	18.0	131	0	0	0	2124	0	289	4	3	0	1	676L
2132	1083	1088	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	12R
2133	1083	1087	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	392	1	0	0	1	12T
2134	1084	1086	0.100	50	1800	4	0.5	30.0	131	0	0	0	0	0	392	1	0	0	1	122T
2135	1084	1088	0.100	30	1601	1	0.5	18.0	131	0	0	0	2133	0	392	1	0	0	1	122L
2136	1085	1087	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	706R
2137	1085	1086	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	392	2	0	0	1	706L
2138	1089	1094	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	407	3	1	0	1	149T
2139	1089	1093	0.100	30	1601	2	0.5	18.0	131	0	0	0	2143	0	407	3	1	0	1	149L
2140	1090	1092	0.100	50	2800	2	0.5	30.0	131	0	0	0	0	0	0	0	0	0	1	709R
2141	1090	1094	0.100	30	1601	2	0.5	18.0	131	0	0	0	0	0	407	2	0	0	1	709L
2142	1091	1093	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	259R
2143	1091	1092	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	407	1	0	0	1	259T
2144	1095	1100	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	148R
2145	1095	1099	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	406	3	1	0	1	148T
2146	1096	1098	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	406	3	1	0	1	260T
2147	1096	1100	0.100	30	1601	2	0.5	18.0	131	0	0	0	2145	0	406	3	1	0	1	260L
2148	1097	1099	0.100	70	2800	2	0.5	42.0	131	0	0	0	0	0	0	0	0	0	1	1132R
2149	1097	1098	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	406	2	0	0	1	1132L
2150	1101	1108	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	552R
2151	1101	1107	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	126	1	0	0	1	552T

2152	1101	1106	0.100	30	1601	1	0.5	18.0	131	0	0	0	2157	0	126	1	001	552L
2153	1102	1105	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001	1120R
2154	1102	1108	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	126	2	0	1120T
2155	1102	1107	0.100	30	1601	1	0.5	18.0	131	0	0	0	2160	0	126	2	001	1120L
2156	1103	1106	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	572R
2157	1103	1105	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	126	1	0	572T
2158	1103	1108	0.100	30	1601	1	0.5	18.0	131	0	0	0	2151	0	126	1	0	572L
2159	1104	1107	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1128R
2160	1104	1106	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	126	2	001	1128T
2161	1104	1105	0.100	30	1601	1	0.5	18.0	131	0	0	0	2154	0	126	2	001	1128L
2162	1109	1116	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001	589R
2163	1109	1115	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	134	1	0	589T
2164	1109	1114	0.100	30	1601	1	0.5	18.0	131	0	0	0	2169	0	1	3	4	589L
2165	1110	1113	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001	1121R
2166	1110	1116	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	134	2	0	1121T
2167	1110	1115	0.100	30	1601	1	0.5	18.0	131	0	0	0	2172	0	134	2	0	1121L
2168	1111	1114	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001	593R
2169	1111	1113	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	134	1	0	593T
2170	1111	1116	0.100	30	1601	1	0.5	18.0	131	0	0	0	2163	0	134	1	0	593L
2171	1112	1115	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001	1127R
2172	1112	1114	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	134	2	0	1127T
2173	1112	1113	0.100	30	1601	1	0.5	18.0	131	0	0	0	2166	0	134	2	0	1127L
2174	1117	1124	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001	153R
2175	1117	1123	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	6	3	1	153T
2176	1117	1122	0.100	30	1601	1	0.5	18.0	131	0	0	0	2181	0	63	1	0	153L
2177	1118	1121	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001	1416R
2178	1118	1124	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	63	2	001	1416T
2179	1118	1123	0.100	30	1601	1	0.5	18.0	131	0	0	0	2184	0	63	2	001	1416L
2180	1119	1122	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001	255R
2181	1119	1121	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	6	3	1	255T
2182	1119	1124	0.100	30	1601	1	0.5	18.0	131	0	0	0	2175	0	6	3	1	255L
2183	1120	1123	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001	1418R
2184	1120	1122	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	63	2	001	1418T
2185	1120	1121	0.100	30	1601	1	0.5	18.0	131	0	0	0	2178	0	63	2	001	1418L
2186	1125	1130	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	6	5	1	156T
2187	1125	1129	0.100	30	1601	1	0.5	18.0	131	0	0	0	2191	0	6	5	1	156L
2188	1126	1128	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	726R
2189	1126	1130	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	65	2	001	726L
2190	1127	1129	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001	252R
2191	1127	1128	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	6	5	1	252T
2192	1131	1138	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001	22R
2193	1131	1137	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	401	4	1	22T
2194	1131	1136	0.100	30	1601	1	0.5	18.0	131	0	0	0	2199	0	401	4	1	22L
2195	1132	1135	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001	728R
2196	1132	1138	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	401	2	3	728T
2197	1132	1137	0.100	30	1601	1	0.5	18.0	131	0	0	0	2202	0	401	2	3	728L
2198	1133	1136	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001	112R
2199	1133	1135	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	401	4	1	112T
2200	1133	1138	0.100	30	1601	1	0.5	18.0	131	0	0	0	2193	0	401	4	1	112L
2201	1134	1137	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001	761R
2202	1134	1136	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	401	2	3	761T
2203	1134	1135	0.100	30	1601	1	0.5	18.0	131	0	0	0	2196	0	401	2	3	761L
2204	1139	1144	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	161	1	0	298T
2205	1139	1143	0.100	30	1601	1	0.5	18.0	131	0	0	0	2209	0	161	1	001	298L
2206	1140	1142	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001	843R
2207	1140	1144	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	161	2	0	843L
2208	1141	1143	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001	346R
2209	1141	1142	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	161	1	0	346T
2210	1145	1152	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001	531R
2211	1145	1151	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	44	1	0	531T
2212	1145	1150	0.100	30	1601	1	0.5	18.0	131	0	0	0	2217	0	44	1	0	531L
2213	1146	1149	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001	847R
2214	1146	1152	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	44	2	0	847T
2215	1146	1151	0.100	30	1601	1	0.5	18.0	131	0	0	0	2220	0	44	2	0	847L
2216	1147	1150	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001	537R
2217	1147	1149	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	44	1	0	537T
2218	1147	1152	0.100	30	1601	1	0.5	18.0	131	0	0	0	2211	0	44	1	0	537L

2219	1148	1151	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	859R
2220	1148	1150	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	44	2	0	0	1	859I
2221	1148	1149	0.100	30	1601	1	0.5	18.0	131	0	0	0	2214	0	44	2	0	0	1	859L
2222	1153	1160	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	172R
2223	1153	1159	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	77	1	0	0	1	172T
2224	1153	1158	0.100	30	1601	1	0.5	18.0	131	0	0	0	2229	0	77	1	0	0	1	172L
2225	1154	1157	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	849R
2226	1154	1160	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	77	2	0	0	1	849T
2227	1154	1159	0.100	30	1601	1	0.5	18.0	131	0	0	0	2232	0	77	2	0	0	1	849L
2228	1155	1158	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	236R
2229	1155	1157	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	77	1	0	0	1	236I
2230	1155	1160	0.100	30	1601	1	0.5	18.0	131	0	0	0	2223	0	77	1	0	0	1	236L
2231	1156	1159	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	857R
2232	1156	1158	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	77	2	0	0	1	857T
2233	1156	1157	0.100	30	1601	1	0.5	18.0	131	0	0	0	2226	0	77	2	0	0	1	857L
2234	1161	1168	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	299R
2235	1161	1167	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	162	1	0	0	1	299T
2236	1161	1166	0.100	30	1601	1	0.5	18.0	131	0	0	0	2241	0	162	1	0	0	1	299L
2237	1162	1165	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	850R
2238	1162	1168	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	162	2	0	0	1	850T
2239	1162	1167	0.100	30	1601	1	0.5	18.0	131	0	0	0	2244	0	162	2	0	0	1	850L
2240	1163	1166	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	345R
2241	1163	1165	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	162	1	0	0	1	345T
2242	1163	1168	0.100	30	1601	1	0.5	18.0	131	0	0	0	2235	0	162	1	0	0	1	345L
2243	1164	1167	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	856R
2244	1164	1166	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	162	2	0	0	1	856T
2245	1164	1165	0.100	30	1601	1	0.5	18.0	131	0	0	0	2238	0	162	2	0	0	1	856L
2246	1169	1176	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	392R
2247	1169	1175	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	189	1	0	0	1	392T
2248	1169	1174	0.100	30	1601	1	0.5	18.0	131	0	0	0	2253	0	189	1	0	0	1	392L
2249	1170	1173	0.100	10	1402	1	3.5	6.0	131	0	0	0	0	0	0	0	0	0	1	851R
2250	1170	1176	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	189	2	0	0	1	851T
2251	1170	1175	0.100	30	1601	1	0.5	18.0	131	0	0	0	2256	0	189	2	0	0	1	851L
2252	1171	1174	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	446R
2253	1171	1173	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	189	1	0	0	1	446T
2254	1171	1176	0.100	30	1601	1	0.5	18.0	131	0	0	0	2247	0	189	1	0	0	1	446L
2255	1172	1175	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	855R
2256	1172	1174	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	189	2	0	0	1	855T
2257	1172	1173	0.100	30	1601	1	0.5	18.0	131	0	0	0	2250	0	189	2	0	0	1	855L
2258	1177	1182	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1150R
2259	1177	1181	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	345	2	0	0	1	1150L
2260	1178	1180	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	868R
2261	1178	1182	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	345	1	0	0	1	868T
2262	1179	1181	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	345	1	0	0	1	882T
2263	1179	1180	0.100	30	1601	1	0.5	18.0	131	0	0	0	2261	0	345	1	0	0	1	882L
2264	1183	1190	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	41R
2265	1183	1189	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	17	1	0	0	1	41T
2266	1183	1188	0.100	30	1601	1	0.5	18.0	131	0	0	0	2271	0	17	1	0	0	1	41L
2267	1184	1187	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	891R
2268	1184	1190	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	17	2	0	0	1	891T
2269	1184	1189	0.100	30	1601	1	0.5	18.0	131	0	0	0	2274	0	17	2	0	0	1	891L
2270	1185	1188	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	93R
2271	1185	1187	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	17	1	0	0	1	93T
2272	1185	1190	0.100	30	1601	1	0.5	18.0	131	0	0	0	2265	0	17	1	0	0	1	93L
2273	1186	1189	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	893R
2274	1186	1188	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	17	2	0	0	1	893T
2275	1186	1187	0.100	30	1601	1	0.5	18.0	131	0	0	0	2268	0	17	2	0	0	1	893L
2276	1191	1198	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	19R
2277	1191	1197	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	399	1	0	0	1	19T
2278	1191	1196	0.100	30	1601	1	0.5	18.0	131	0	0	0	2283	0	399	1	0	0	1	19L
2280	1192	1198	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	399	3	0	0	1	717T
2282	1193	1196	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	115R
2283	1193	1195	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	399	1	0	0	1	115T
2284	1193	1198	0.100	30	1601	1	0.5	18.0	131	0	0	0	2277	0	399	1	0	0	1	115L
2285	1194	1197	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	723R
2286	1194	1196	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	399	3	0	0	1	723T
2287	1194	1195	0.100	30	1601	1	0.5	18.0	131	0	0	0	2280	0	399	3	0	0	1	723L

2362	1251	1256	0.100	30	1601	1	0.5	18.0	131	0	0	0	2367	0	173	1	0	0	1	313L
2363	1252	1255	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	971R
2364	1252	1258	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	173	2	0	0	1	971T
2365	1252	1257	0.100	30	1601	1	0.5	18.0	131	0	0	0	2370	0	173	2	0	0	1	971L
2366	1253	1256	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	331R
2367	1253	1255	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	173	1	0	0	1	331T
2368	1253	1258	0.100	30	1601	1	0.5	18.0	131	0	0	0	2361	0	173	1	0	0	1	331L
2369	1254	1257	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	973R
2370	1254	1256	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	173	2	0	0	1	973T
2371	1254	1255	0.100	30	1601	1	0.5	18.0	131	0	0	0	2364	0	173	2	0	0	1	973L
2372	1259	1264	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	202	1	0	0	1	407T
2373	1259	1263	0.100	30	1601	1	0.5	18.0	131	0	0	0	2377	0	202	1	0	0	1	407L
2374	1260	1262	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	972R
2375	1260	1264	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	202	2	0	0	1	972L
2376	1261	1263	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	431R
2377	1261	1262	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	202	1	0	0	1	431T
2378	1265	1270	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1133R
2379	1265	1269	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	460	2	0	0	1	1133L
2380	1266	1268	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1033R
2381	1266	1270	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	460	1	0	0	1	1033T
2382	1267	1269	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	460	3	1	0	1	1043T
2383	1267	1268	0.100	30	1601	1	0.5	18.0	131	0	0	0	2381	0	460	3	1	0	1	1043L
2384	1271	1276	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	98	1	0	0	1	1029T
2386	1272	1274	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1058R
2387	1272	1276	0.100	30	1601	2	0.5	18.0	131	0	0	0	0	0	98	2	0	0	1	1058L
2388	1273	1275	0.100	50	2800	2	0.5	30.0	131	0	0	0	0	0	0	0	0	0	1	1047R
2389	1273	1274	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	98	1	0	0	1	1047T
2390	1277	1284	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	413R
2391	1277	1283	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	433	1	0	0	1	413T
2392	1277	1282	0.100	30	1601	1	0.5	18.0	131	0	0	0	2397	0	433	1	0	0	1	413L
2393	1278	1281	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1032R
2394	1278	1284	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	433	2	0	0	1	1032T
2395	1278	1283	0.100	30	1601	1	0.5	18.0	131	0	0	0	2400	0	433	2	0	0	1	1032L
2396	1279	1282	0.100	35	1400	1	0.5	21.0	131	0	0	0	0	0	0	0	0	0	1	425R
2397	1279	1281	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	433	1	0	0	1	425T
2398	1279	1284	0.100	30	1601	1	0.5	18.0	131	0	0	0	2391	0	433	1	0	0	1	425L
2399	1280	1283	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1044R
2400	1280	1282	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	433	2	0	0	1	1044T
2401	1280	1281	0.100	30	1601	1	0.5	18.0	131	0	0	0	2394	0	433	2	0	0	1	1044L
2402	1285	1292	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	415R
2403	1285	1291	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	482	1	0	0	1	415T
2404	1285	1290	0.100	30	1601	1	0.5	18.0	131	0	0	0	2409	0	482	1	0	0	1	415L
2405	1286	1289	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1072R
2406	1286	1292	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	482	2	0	0	1	1072T
2407	1286	1291	0.100	30	1601	1	0.5	18.0	131	0	0	0	2412	0	482	2	0	0	1	1072L
2408	1287	1290	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	423R
2409	1287	1289	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	482	1	0	0	1	423T
2410	1287	1292	0.100	30	1601	1	0.5	18.0	131	0	0	0	2403	0	482	1	0	0	1	423L
2411	1288	1291	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1076R
2412	1288	1290	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	482	2	0	0	1	1076T
2413	1288	1289	0.100	30	1601	1	0.5	18.0	131	0	0	0	2406	0	482	2	0	0	1	1076L
2414	1293	1300	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	63R
2415	1293	1299	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	36	1	0	0	1	63T
2416	1293	1298	0.100	30	1601	1	0.5	18.0	131	0	0	0	2421	0	36	1	0	0	1	63L
2417	1294	1297	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1064R
2418	1294	1300	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	36	2	0	0	1	1064T
2420	1295	1298	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	71R
2421	1295	1297	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	36	1	0	0	1	71T
2422	1295	1300	0.100	30	1601	1	0.5	18.0	131	0	0	0	2415	0	36	1	0	0	1	71L
2423	1296	1299	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1084R
2424	1296	1298	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	36	2	0	0	1	1084T
2426	1301	1308	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	418R
2427	1301	1307	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	437	1	0	0	1	418T
2428	1301	1306	0.100	30	1601	1	0.5	18.0	131	0	0	0	2433	0	437	1	0	0	1	418L
2429	1302	1305	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1091R
2430	1302	1308	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	437	2	0	0	1	1091T
2431	1302	1307	0.100	30	1601	1	0.5	18.0	131	0	0	0	2436	0	437	2	0	0	1	1091L

2432	1303	1306	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	420R
2433	1303	1305	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	437	3	1	0	1	420T
2434	1303	1308	0.100	30	1601	2	0.5	18.0	131	0	0	0	2427	0	437	3	1	0	1	420L
2435	1304	1307	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1097R
2436	1304	1306	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	437	2	0	0	1	1097T
2437	1304	1305	0.100	30	1601	1	0.5	18.0	131	0	0	0	2430	0	437	2	0	0	1	1097L
2438	1309	1316	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	6R
2439	1309	1315	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	276	1	0	0	1	6T
2440	1309	1314	0.100	30	1601	1	0.5	18.0	131	0	0	0	2445	0	276	1	0	0	1	6L
2441	1310	1313	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	669R
2442	1310	1316	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	276	2	0	0	1	669T
2443	1310	1315	0.100	30	1601	1	0.5	18.0	131	0	0	0	2448	0	276	2	0	0	1	669L
2444	1311	1314	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	128R
2445	1311	1313	0.100	50	1800	4	0.5	30.0	131	0	0	0	0	0	276	1	0	0	1	128T
2446	1311	1316	0.100	30	1601	1	0.5	18.0	131	0	0	0	2439	0	276	1	0	0	1	128L
2447	1312	1315	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	677R
2448	1312	1314	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	276	2	0	0	1	677T
2449	1312	1313	0.100	30	1601	1	0.5	18.0	131	0	0	0	2442	0	276	2	0	0	1	677L
2450	1317	1320	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	955R
2451	1317	1322	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	226	1	0	0	1	955T
2452	1318	1321	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1268R
2453	1318	1320	0.100	50	1800	1	0.0	30.0	131	0	0	0	0	0	226	2	0	0	1	1268T
2454	1318	1322	0.100	30	1601	1	0.0	18.0	131	0	0	0	0	0	226	2	0	0	1	1268L
2455	1319	1321	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	226	3	1	0	1	961T
2456	1319	1320	0.100	30	1601	2	0.5	18.0	131	0	0	0	0	0	226	3	1	0	1	961L
2457	1323	1327	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	303	1	0	0	1	551T
2458	1324	1326	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1291R
2459	1324	1327	0.100	30	1601	1	0.0	18.0	131	0	0	0	0	0	303	2	0	0	1	1291L
2460	1325	1326	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	303	1	0	0	1	573T
2461	1328	1332	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	285	1	0	0	1	628T
2462	1329	1331	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1380R
2463	1329	1332	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	285	2	0	0	1	1380L
2464	1330	1331	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	285	1	0	0	1	640T
2465	1333	1338	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1305R
2466	1333	1337	0.100	50	1800	1	0.0	30.0	131	0	0	0	0	0	227	3	0	0	1	1305T
2467	1333	1336	0.100	30	1601	2	0.0	18.0	131	0	0	0	0	0	227	3	0	0	1	1305L
2468	1334	1338	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	227	2	1	0	1	956T
2469	1334	1337	0.100	30	1601	2	0.5	18.0	131	0	0	0	0	0	227	2	1	0	1	956L
2470	1335	1337	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	960R
2471	1335	1336	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	227	1	0	0	1	960T
2472	1339	1344	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1303R
2473	1339	1343	0.100	50	1800	1	0.0	30.0	131	0	0	0	0	0	222	2	0	0	1	1303T
2474	1339	1342	0.100	30	1601	1	0.0	18.0	131	0	0	0	0	0	222	2	0	0	1	1303L
2475	1340	1344	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	222	1	0	0	1	937T
2476	1340	1343	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	222	3	1	0	1	937L
2477	1341	1343	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	941R
2478	1341	1342	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	222	1	0	0	1	941T
2479	1345	1348	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	936R
2480	1345	1350	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	221	1	0	0	1	936T
2481	1346	1349	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1270R
2482	1346	1348	0.100	50	1800	1	0.0	30.0	131	0	0	0	0	0	221	3	0	0	1	1270T
2483	1346	1350	0.100	30	1601	2	0.0	18.0	131	0	0	0	0	0	221	3	0	0	1	1270L
2484	1347	1349	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	221	1	0	0	1	942T
2485	1347	1348	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	221	2	1	0	1	942L
2486	1351	1356	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1301R
2487	1351	1355	0.100	50	1800	1	0.0	30.0	131	0	0	0	0	0	217	3	0	0	1	1301T
2488	1351	1354	0.100	30	1601	2	0.0	18.0	131	0	0	0	0	0	217	3	0	0	1	1301L
2489	1352	1356	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	217	1	0	0	1	912T
2490	1352	1355	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	217	2	1	0	1	912L
2491	1353	1355	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	916R
2492	1353	1354	0.100	50	1800	4	0.5	30.0	131	0	0	0	0	0	217	1	0	0	1	916T
2493	1357	1363	0.100	40	1400	1	0.5	24.0	131	0	0	0	0	0	0	0	0	0	1	1135R
2494	1357	1362	0.100	30	1601	1	0.5	18.0	131	0	0	0	2499	0	186	2	2	0	1	1135L
2495	1358	1361	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	450R
2496	1358	1363	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	186	1	0	0	1	450T
2497	1359	1362	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1277R
2498	1359	1361	0.100	50	1800	1	0.0	30.0	131	0	0	0	0	0	186	3	3	0	1	1277T

2499	1359	1363	0.100	30	1601	2	0.0	18.0	131	0	0	0	2492	0	186	3	3	0	1	1277L
2500	1360	1362	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	186	1	0	0	1	388T
2501	1360	1361	0.100	30	1601	1	0.5	18.0	131	0	0	0	2494	0	186	1	0	0	1	388L
2502	1364	1371	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1296R
2503	1364	1370	0.100	50	1800	1	0.0	30.0	131	0	0	0	0	0	184	2	0	0	1	1296T
2504	1364	1369	0.100	30	1601	2	0.0	18.0	131	0	0	0	0	0	184	2	0	0	1	1296L
2505	1365	1368	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	805R
2506	1365	1371	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	184	1	0	0	1	805T
2507	1365	1370	0.100	30	1601	1	0.5	18.0	131	0	0	0	2512	0	184	1	0	0	1	805L
2508	1366	1369	0.100	50	2800	2	0.5	30.0	131	0	0	0	0	0	0	0	0	0	1	1136R
2509	1366	1368	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	184	3	0	0	1	1136T
2510	1366	1371	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	184	3	0	0	1	1136L
2511	1367	1370	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	811R
2512	1367	1369	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	184	1	0	0	1	811T
2513	1367	1368	0.100	30	1601	1	0.5	18.0	131	0	0	0	2506	0	184	1	0	0	1	811L
2514	1372	1374	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	773R
2515	1372	1376	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	143	1	0	0	1	773T
2516	1373	1375	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	143	1	0	0	1	783T
2517	1373	1374	0.100	30	1601	1	0.5	18.0	131	0	0	0	2515	0	143	3	1	0	1	783L
2518	1377	1384	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	135R
2519	1377	1383	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	286	1	0	0	1	135T
2520	1377	1382	0.100	30	1601	1	0.5	18.0	131	0	0	0	2525	0	286	1	0	0	1	135L
2521	1378	1381	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	629R
2522	1378	1384	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	286	2	0	0	1	629T
2523	1378	1383	0.100	30	1601	1	0.5	18.0	131	0	0	0	2528	0	286	2	0	0	1	629L
2524	1379	1382	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	273R
2525	1379	1381	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	286	1	0	0	1	273T
2526	1379	1384	0.100	30	1601	1	0.5	18.0	131	0	0	0	2519	0	286	1	0	0	1	273L
2527	1380	1383	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	639R
2528	1380	1382	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	286	2	0	0	1	639T
2529	1380	1381	0.100	30	1601	1	0.5	18.0	131	0	0	0	2522	0	286	2	0	0	1	639L
2530	1385	1392	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	8R
2531	1385	1391	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	277	1	0	0	1	8T
2532	1385	1390	0.100	30	1601	1	0.5	18.0	131	0	0	0	2537	0	277	1	0	0	1	8L
2533	1386	1389	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	679R
2534	1386	1392	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	277	2	0	0	1	679T
2535	1386	1391	0.100	30	1601	1	0.5	18.0	131	0	0	0	2540	0	277	2	0	0	1	679L
2536	1387	1390	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	126R
2537	1387	1389	0.100	50	1800	4	0.5	30.0	131	0	0	0	0	0	277	1	0	0	1	126T
2538	1387	1392	0.100	30	1601	1	0.5	18.0	131	0	0	0	2531	0	277	1	0	0	1	126L
2539	1388	1391	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	699R
2540	1388	1390	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	277	2	0	0	1	699T
2541	1388	1389	0.100	30	1601	1	0.5	18.0	131	0	0	0	2534	0	277	2	0	0	1	699L
2542	1393	1400	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	10R
2543	1393	1399	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	278	1	0	0	1	10T
2544	1393	1398	0.100	30	1601	1	0.5	18.0	131	0	0	0	2549	0	278	1	0	0	1	10L
2545	1394	1397	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	701R
2546	1394	1400	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	278	2	0	0	1	701T
2547	1394	1399	0.100	30	1601	1	0.5	18.0	131	0	0	0	2552	0	278	2	0	0	1	701L
2548	1395	1398	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	124R
2549	1395	1397	0.100	50	1800	4	0.5	30.0	131	0	0	0	0	0	278	1	0	0	1	124T
2550	1395	1400	0.100	30	1601	1	0.5	18.0	131	0	0	0	2543	0	278	1	0	0	1	124L
2551	1396	1399	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	703R
2552	1396	1398	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	278	2	0	0	1	703T
2553	1396	1397	0.100	30	1601	1	0.5	18.0	131	0	0	0	2546	0	278	2	0	0	1	703L
2554	1401	1408	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1R
2555	1401	1407	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	272	1	0	0	1	1T
2556	1401	1406	0.100	30	1601	1	0.5	18.0	131	0	0	0	2561	0	272	1	0	0	1	1L
2557	1402	1405	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	627R
2558	1402	1408	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	272	2	0	0	1	627T
2559	1402	1407	0.100	30	1601	1	0.5	18.0	131	0	0	0	2564	0	272	2	0	0	1	627L
2560	1403	1406	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	133R
2561	1403	1405	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	272	1	0	0	1	133T
2562	1403	1408	0.100	30	1601	1	0.5	18.0	131	0	0	0	2555	0	272	1	0	0	1	133L
2563	1404	1407	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	641R
2564	1404	1406	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	272	2	0	0	1	641T
2565	1404	1405	0.100	30	1601	1	0.5	18.0	131	0	0	0	2558	0	272	2	0	0	1	641L

2566	1409	1416	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	470R
2567	1409	1415	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	208	1	0	0	1	470T
2568	1409	1414	0.100	30	1601	1	0.5	18.0	131	0	0	0	2573	0	208	1	0	0	1	470L
2569	1410	1413	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1430R
2570	1410	1416	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	208	2	0	0	1	1430T
2571	1410	1415	0.100	30	1601	1	0.5	18.0	131	0	0	0	2576	0	208	2	0	0	1	1430L
2572	1411	1414	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	524R
2573	1411	1413	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	208	1	0	0	1	524T
2574	1411	1416	0.100	30	1601	1	0.5	18.0	131	0	0	0	2567	0	208	1	0	0	1	524L
2575	1412	1415	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	835R
2576	1412	1414	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	208	2	0	0	1	835T
2577	1412	1413	0.100	30	1601	1	0.5	18.0	131	0	0	0	2570	0	208	2	0	0	1	835L
2578	1417	1424	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	397R
2579	1417	1423	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	193	1	0	0	1	397T
2580	1417	1422	0.100	30	1601	1	0.5	18.0	131	0	0	0	2585	0	193	1	0	0	1	397L
2581	1418	1421	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	896R
2582	1418	1424	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	193	2	0	0	1	896T
2583	1418	1423	0.100	30	1601	1	0.5	18.0	131	0	0	0	2588	0	193	2	0	0	1	896L
2584	1419	1422	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	441R
2585	1419	1421	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	193	1	0	0	1	441T
2586	1419	1424	0.100	30	1601	1	0.5	18.0	131	0	0	0	2579	0	193	1	0	0	1	441L
2587	1420	1423	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	902R
2588	1420	1422	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	193	2	0	0	1	902T
2589	1420	1421	0.100	30	1601	1	0.5	18.0	131	0	0	0	2582	0	193	2	0	0	1	902L
2590	1425	1432	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	390R
2591	1425	1431	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	188	1	0	0	1	390T
2592	1425	1430	0.100	30	1601	1	0.5	18.0	131	0	0	0	2597	0	188	1	0	0	1	390L
2593	1426	1429	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	833R
2594	1426	1432	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	188	2	0	0	1	833T
2595	1426	1431	0.100	30	1601	1	0.5	18.0	131	0	0	0	2600	0	188	2	0	0	1	833L
2596	1427	1430	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	448R
2597	1427	1429	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	188	1	0	0	1	448T
2598	1427	1432	0.100	30	1601	1	0.5	18.0	131	0	0	0	2591	0	188	1	0	0	1	448L
2599	1428	1431	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1431R
2600	1428	1430	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	188	2	0	0	1	1431T
2601	1428	1429	0.100	30	1601	1	0.5	18.0	131	0	0	0	2594	0	188	2	0	0	1	1431L
2602	1433	1440	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	386R
2603	1433	1439	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	185	1	0	0	1	386T
2604	1433	1438	0.100	30	1601	1	0.5	18.0	131	0	0	0	2609	0	185	1	0	0	1	386L
2605	1434	1437	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	806R
2606	1434	1440	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	185	3	0	0	1	806T
2607	1434	1439	0.100	30	1601	2	0.5	18.0	131	0	0	0	2612	0	185	2	3	0	1	806L
2608	1435	1438	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	452R
2609	1435	1437	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	185	4	1	0	1	452T
2610	1435	1440	0.100	30	1601	2	0.5	18.0	131	0	0	0	2603	0	185	4	1	0	1	452L
2611	1436	1439	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	810R
2612	1436	1438	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	185	3	0	0	1	810T
2613	1436	1437	0.100	30	1601	2	0.5	18.0	131	0	0	0	2606	0	185	3	0	0	1	810L
2614	1441	1448	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	319R
2615	1441	1447	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	182	1	0	0	1	319T
2616	1441	1446	0.100	30	1601	1	0.5	18.0	131	0	0	0	2621	0	182	1	0	0	1	319L
2617	1442	1445	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1069R
2618	1442	1448	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	182	2	0	0	1	1069T
2619	1442	1447	0.100	30	1601	1	0.5	18.0	131	0	0	0	2624	0	182	2	0	0	1	1069L
2620	1443	1446	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	325R
2621	1443	1445	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	182	1	0	0	1	325T
2622	1443	1448	0.100	30	1601	1	0.5	18.0	131	0	0	0	2615	0	182	1	0	0	1	325L
2623	1444	1447	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1079R
2624	1444	1446	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	182	2	0	0	1	1079T
2625	1444	1445	0.100	30	1601	1	0.5	18.0	131	0	0	0	2618	0	182	2	0	0	1	1079L
2626	1449	1456	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	312R
2627	1449	1455	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	172	1	0	0	1	312T
2628	1449	1454	0.100	30	1601	1	0.5	18.0	131	0	0	0	2633	0	172	1	0	0	1	312L
2629	1450	1453	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	952R
2630	1450	1456	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	172	2	0	0	1	952T
2631	1450	1455	0.100	30	1601	1	0.5	18.0	131	0	0	0	2636	0	172	2	0	0	1	952L
2632	1451	1454	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	332R

2633	1451	1453	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	172	1	0	0	1	332T
2634	1451	1456	0.100	30	1601	1	0.5	18.0	131	0	0	0	2627	0	172	1	0	0	1	332L
2635	1452	1455	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	964R
2636	1452	1454	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	172	2	0	0	1	964T
2637	1452	1453	0.100	30	1601	1	0.5	18.0	131	0	0	0	2630	0	172	2	0	0	1	964L
2638	1457	1464	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	293R
2639	1457	1463	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	158	1	0	0	1	293T
2640	1457	1462	0.100	30	1601	1	0.5	18.0	131	0	0	0	2645	0	158	1	0	0	1	293L
2641	1458	1461	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	804R
2642	1458	1464	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	158	2	0	0	1	804T
2643	1458	1463	0.100	30	1601	1	0.5	18.0	131	0	0	0	2648	0	158	2	0	0	1	804L
2644	1459	1462	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	351R
2645	1459	1461	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	158	1	0	0	1	351T
2646	1459	1464	0.100	30	1601	1	0.5	18.0	131	0	0	0	2639	0	158	1	0	0	1	351L
2647	1460	1463	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	812R
2648	1460	1462	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	158	2	0	0	1	812T
2649	1460	1461	0.100	30	1601	1	0.5	18.0	131	0	0	0	2642	0	158	2	0	0	1	812L
2650	1465	1470	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	281R
2651	1465	1469	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	150	1	0	0	1	281T
2652	1466	1468	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	150	1	0	0	1	363T
2653	1466	1470	0.100	30	1601	1	0.5	18.0	131	0	0	0	2651	0	150	1	0	0	1	363L
2654	1467	1469	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1106R
2655	1467	1468	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	150	2	0	0	1	1106L
2656	1471	1476	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	282R
2657	1471	1475	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	151	1	0	0	1	282T
2658	1472	1474	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	151	1	0	0	1	362T
2659	1472	1476	0.100	30	1601	1	0.5	18.0	131	0	0	0	2657	0	151	1	0	0	1	362L
2660	1473	1475	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1108R
2661	1473	1474	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	151	2	0	0	1	1108L
2662	1477	1482	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	283R
2663	1477	1481	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	152	1	0	0	1	283T
2664	1478	1480	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	152	1	0	0	1	361T
2665	1478	1482	0.100	30	1601	1	0.5	18.0	131	0	0	0	2663	0	152	2	1	0	1	361L
2666	1479	1481	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1110R
2667	1479	1480	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	152	3	0	0	1	1110L
2668	1483	1490	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	275R
2669	1483	1489	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	145	1	0	0	1	275T
2670	1483	1488	0.100	30	1601	1	0.5	18.0	131	0	0	0	2675	0	145	4	1	0	1	275L
2671	1484	1487	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	633R
2672	1484	1490	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	145	3	0	0	1	633T
2673	1484	1489	0.100	30	1601	1	0.5	18.0	131	0	0	0	2678	0	145	3	0	0	1	633L
2674	1485	1488	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	369R
2675	1485	1487	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	145	1	0	0	1	369T
2676	1485	1490	0.100	30	1601	1	0.5	18.0	131	0	0	0	2669	0	145	2	1	0	1	369L
2677	1486	1489	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	635R
2678	1486	1488	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	145	3	0	0	1	635T
2679	1486	1487	0.100	30	1601	1	0.5	18.0	131	0	0	0	2672	0	145	3	0	0	1	635L
2680	1491	1498	0.100	50	2800	2	0.5	30.0	131	0	0	0	0	0	0	0	0	0	1	615R
2681	1491	1497	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	142	2	0	0	1	615T
2682	1491	1496	0.100	30	1601	1	0.5	18.0	131	0	0	0	2687	0	142	2	0	0	1	615L
2683	1492	1495	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	772R
2684	1492	1498	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	142	1	0	0	1	772T
2685	1492	1497	0.100	30	1601	1	0.5	18.0	131	0	0	0	2690	0	142	1	0	0	1	772L
2686	1493	1496	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	617R
2687	1493	1495	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	142	2	0	0	1	617T
2688	1493	1498	0.100	30	1601	1	0.5	18.0	131	0	0	0	2681	0	142	2	0	0	1	617L
2689	1494	1497	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	784R
2690	1494	1496	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	142	1	0	0	1	784T
2691	1494	1495	0.100	30	1601	1	0.5	18.0	131	0	0	0	2684	0	142	3	1	0	1	784L
2692	1499	1506	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	583R
2693	1499	1505	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	130	2	0	0	1	583T
2694	1499	1504	0.100	30	1601	1	0.5	18.0	131	0	0	0	2699	0	130	2	0	0	1	583L
2695	1500	1503	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	632R
2696	1500	1506	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	130	1	0	0	1	632T
2697	1500	1505	0.100	30	1601	1	0.5	18.0	131	0	0	0	2702	0	130	1	0	0	1	632L
2698	1501	1504	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	599R
2699	1501	1503	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	130	2	0	0	1	599T

2700	1501	1506	0.100	30	1601	1	0.5	18.0	131	0	0	0	2693	0	130	2	0	0	1	599L
2701	1502	1505	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	636R
2702	1502	1504	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	130	1	0	0	1	636T
2703	1502	1503	0.100	30	1601	1	0.5	18.0	131	0	0	0	2696	0	130	1	0	0	1	636L
2704	1507	1514	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	546R
2705	1507	1513	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	106	1	0	0	1	546T
2706	1507	1512	0.100	30	1601	1	0.5	18.0	131	0	0	0	2711	0	106	1	0	0	1	546L
2707	1508	1511	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	650R
2708	1508	1514	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	106	3	0	0	1	650T
2709	1508	1513	0.100	30	1601	1	0.5	18.0	131	0	0	0	2714	0	106	3	0	0	1	650L
2710	1509	1512	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	578R
2711	1509	1511	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	106	1	0	0	1	578T
2712	1509	1514	0.100	30	1601	1	0.5	18.0	131	0	0	0	2705	0	106	1	0	0	1	578L
2713	1510	1513	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	662R
2714	1510	1512	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	106	3	0	0	1	662T
2715	1510	1511	0.100	30	1601	1	0.5	18.0	131	0	0	0	2708	0	106	2	3	0	1	662L
2716	1515	1522	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	198R
2717	1515	1521	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	99	1	0	0	1	198T
2718	1515	1520	0.100	30	1601	1	0.5	18.0	131	0	0	0	2723	0	99	1	0	0	1	198L
2719	1516	1519	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1068R
2720	1516	1522	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	99	2	0	0	1	1068T
2721	1516	1521	0.100	30	1601	1	0.5	18.0	131	0	0	0	2726	0	99	2	0	0	1	1068L
2722	1517	1520	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	210R
2723	1517	1519	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	99	1	0	0	1	210T
2724	1517	1522	0.100	30	1601	1	0.5	18.0	131	0	0	0	2717	0	99	1	0	0	1	210L
2725	1518	1521	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1080R
2726	1518	1520	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	99	2	0	0	1	1080T
2727	1518	1519	0.100	30	1601	1	0.5	18.0	131	0	0	0	2720	0	99	2	0	0	1	1080L
2728	1523	1530	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	196R
2729	1523	1529	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	97	1	0	0	1	196T
2731	1524	1527	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1057R
2732	1524	1530	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	97	2	0	0	1	1057T
2734	1525	1528	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	212R
2735	1525	1527	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	97	1	0	0	1	212T
2737	1526	1529	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1059R
2738	1526	1528	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	97	2	0	0	1	1059T
2740	1531	1538	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	184R
2741	1531	1537	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	86	1	0	0	1	184T
2742	1531	1536	0.100	30	1601	1	0.5	18.0	131	0	0	0	2747	0	86	1	0	0	1	184L
2743	1532	1535	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	933R
2744	1532	1538	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	86	2	0	0	1	933T
2745	1532	1537	0.100	30	1601	1	0.5	18.0	131	0	0	0	2750	0	86	2	0	0	1	933L
2746	1533	1536	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	224R
2747	1533	1535	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	86	1	0	0	1	224T
2748	1533	1538	0.100	30	1601	1	0.5	18.0	131	0	0	0	2741	0	86	1	0	0	1	224L
2749	1534	1537	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	945R
2750	1534	1536	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	86	2	0	0	1	945T
2751	1534	1535	0.100	30	1601	1	0.5	18.0	131	0	0	0	2744	0	86	2	0	0	1	945L
2752	1539	1546	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	532R
2753	1539	1545	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	45	2	0	0	1	532T
2754	1539	1544	0.100	30	1601	1	0.5	18.0	131	0	0	0	2759	0	45	2	0	0	1	532L
2755	1540	1543	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	864R
2756	1540	1546	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	45	1	0	0	1	864T
2757	1540	1545	0.100	30	1601	1	0.5	18.0	131	0	0	0	2762	0	45	1	0	0	1	864L
2758	1541	1544	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	536R
2759	1541	1543	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	45	2	0	0	1	536T
2760	1541	1546	0.100	30	1601	1	0.5	18.0	131	0	0	0	2753	0	45	2	0	0	1	536L
2761	1542	1545	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	886R
2762	1542	1544	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	45	1	0	0	1	886T
2763	1542	1543	0.100	30	1601	1	0.5	18.0	131	0	0	0	2756	0	45	1	0	0	1	886L
2764	1547	1554	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	60R
2765	1547	1553	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	33	1	0	0	1	60T
2766	1547	1552	0.100	30	1601	1	0.5	18.0	131	0	0	0	2771	0	33	1	0	0	1	60L
2767	1548	1551	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1024R
2768	1548	1554	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	33	2	0	0	1	1024T
2769	1548	1553	0.100	30	1601	1	0.5	18.0	131	0	0	0	2774	0	33	2	0	0	1	1024L
2770	1549	1552	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	74R

2771	1549	1551	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	33	1	0	0	1	74T
2772	1549	1554	0.100	30	1601	1	0.5	18.0	131	0	0	0	2765	0	33	1	0	0	1	74L
2773	1550	1553	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1052R
2774	1550	1552	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	33	2	0	0	1	1052T
2775	1550	1551	0.100	30	1601	1	0.5	18.0	131	0	0	0	2768	0	33	2	0	0	1	1052L
2776	1555	1562	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	46R
2777	1555	1561	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	20	1	0	0	1	46T
2778	1555	1560	0.100	30	1601	1	0.5	18.0	131	0	0	0	2783	0	20	1	0	0	1	46L
2779	1556	1559	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	905R
2780	1556	1562	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	20	2	0	0	1	905T
2781	1556	1561	0.100	30	1601	1	0.5	18.0	131	0	0	0	2786	0	20	2	0	0	1	905L
2782	1557	1560	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	88R
2783	1557	1559	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	20	1	0	0	1	88T
2784	1557	1562	0.100	30	1601	1	0.5	18.0	131	0	0	0	2777	0	20	1	0	0	1	88L
2785	1558	1561	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	923R
2786	1558	1560	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	20	2	0	0	1	923T
2787	1558	1559	0.100	30	1601	1	0.5	18.0	131	0	0	0	2780	0	20	2	0	0	1	923L
2788	1563	1570	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	37R
2789	1563	1569	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	14	1	0	0	1	37T
2790	1563	1568	0.100	30	1601	1	0.5	18.0	131	0	0	0	2795	0	14	1	0	0	1	37L
2791	1564	1567	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	861R
2792	1564	1570	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	14	2	0	0	1	861T
2793	1564	1569	0.100	30	1601	1	0.5	18.0	131	0	0	0	2798	0	14	2	0	0	1	861L
2794	1565	1568	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	97R
2795	1565	1567	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	14	1	0	0	1	97T
2796	1565	1570	0.100	30	1601	1	0.5	18.0	131	0	0	0	2789	0	14	1	0	0	1	97L
2797	1566	1569	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	889R
2798	1566	1568	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	14	2	0	0	1	889T
2799	1566	1567	0.100	30	1601	1	0.5	18.0	131	0	0	0	2792	0	14	2	0	0	1	889L
2800	1571	1578	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	25R
2801	1571	1577	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	3	1	0	0	1	25T
2802	1571	1576	0.100	30	1601	1	0.5	18.0	131	0	0	0	2807	0	3	1	0	0	1	25L
2803	1572	1575	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	763R
2804	1572	1578	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	3	2	0	0	1	763T
2805	1572	1577	0.100	30	1601	1	0.5	18.0	131	0	0	0	2810	0	3	2	0	0	1	763L
2806	1573	1576	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	109R
2807	1573	1575	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	3	1	0	0	1	109T
2808	1573	1578	0.100	30	1601	1	0.5	18.0	131	0	0	0	2801	0	3	1	0	0	1	109L
2809	1574	1577	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	793R
2810	1574	1576	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	3	2	0	0	1	793T
2811	1574	1575	0.100	30	1601	1	0.5	18.0	131	0	0	0	2804	0	3	2	0	0	1	793L
2812	1579	1586	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	51R
2813	1579	1585	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	25	1	0	0	1	51T
2814	1579	1584	0.100	30	1601	1	0.5	18.0	131	0	0	0	2819	0	25	1	0	0	1	51L
2815	1580	1583	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	948R
2816	1580	1586	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	25	2	0	0	1	948T
2817	1580	1585	0.100	30	1601	1	0.5	18.0	131	0	0	0	2822	0	25	2	0	0	1	948L
2818	1581	1584	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	83R
2819	1581	1583	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	25	1	0	0	1	83T
2820	1581	1586	0.100	30	1601	1	0.5	18.0	131	0	0	0	2813	0	25	1	0	0	1	83L
2821	1582	1585	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	968R
2822	1582	1584	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	25	2	0	0	1	968T
2823	1582	1583	0.100	30	1601	1	0.5	18.0	131	0	0	0	2816	0	25	2	0	0	1	968L
2824	1587	1594	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	179R
2825	1587	1593	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	83	1	0	0	1	179T
2826	1587	1592	0.100	30	1601	1	0.5	18.0	131	0	0	0	2831	0	83	1	0	0	1	179L
2827	1588	1591	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	907R
2828	1588	1594	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	83	3	0	0	1	907T
2829	1588	1593	0.100	30	1601	1	0.5	18.0	131	0	0	0	2834	0	83	3	0	0	1	907L
2830	1589	1592	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	228R
2831	1589	1591	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	83	1	0	0	1	228T
2832	1589	1594	0.100	30	1601	1	0.5	18.0	131	0	0	0	2825	0	83	1	0	0	1	228L
2833	1590	1593	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	921R
2834	1590	1592	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	83	3	0	0	1	921T
2835	1590	1591	0.100	30	1601	1	0.5	18.0	131	0	0	0	2828	0	83	2	3	0	1	921L
2836	1595	1602	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	158R
2837	1595	1601	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	67	1	0	0	1	158T

2838	1595	1600	0.100	30	1601	1	0.5	18.0	131	0	0	0	2843	0	67	1	0	0	1	158L
2839	1596	1599	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	729R
2840	1596	1602	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	67	3	0	0	1	729T
2841	1596	1601	0.100	30	1601	1	0.5	18.0	131	0	0	0	2846	0	67	3	0	0	1	729L
2842	1597	1600	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	250R
2843	1597	1599	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	67	1	0	0	1	250T
2844	1597	1602	0.100	30	1601	1	0.5	18.0	131	0	0	0	2837	0	67	1	0	0	1	250L
2845	1598	1601	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	760R
2846	1598	1600	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	67	3	0	0	1	760T
2847	1598	1599	0.100	30	1601	1	0.5	18.0	131	0	0	0	2840	0	67	2	3	0	1	760L
2848	1603	1610	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	162R
2849	1603	1609	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	70	2	0	0	1	162T
2850	1603	1608	0.100	30	1601	1	0.5	18.0	131	0	0	0	2855	0	70	2	0	0	1	162L
2851	1604	1607	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	765R
2852	1604	1610	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	70	1	0	0	1	765T
2853	1604	1609	0.100	30	1601	1	0.5	18.0	131	0	0	0	2858	0	70	1	0	0	1	765L
2854	1605	1608	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	246R
2855	1605	1607	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	70	2	0	0	1	246T
2856	1605	1610	0.100	30	1601	1	0.5	18.0	131	0	0	0	2849	0	70	2	0	0	1	246L
2857	1606	1609	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	791R
2858	1606	1608	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	70	1	0	0	1	791T
2859	1606	1607	0.100	30	1601	1	0.5	18.0	131	0	0	0	2852	0	70	1	0	0	1	791L
2860	1611	1618	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	173R
2861	1611	1617	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	78	2	0	0	1	173T
2862	1611	1616	0.100	30	1601	1	0.5	18.0	131	0	0	0	2867	0	78	2	0	0	1	173L
2863	1612	1615	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	865R
2864	1612	1618	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	78	1	0	0	1	865T
2865	1612	1617	0.100	30	1601	1	0.5	18.0	131	0	0	0	2870	0	78	1	0	0	1	865L
2866	1613	1616	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	235R
2867	1613	1615	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	78	2	0	0	1	235T
2868	1613	1618	0.100	30	1601	1	0.5	18.0	131	0	0	0	2861	0	78	2	0	0	1	235L
2869	1614	1617	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	885R
2870	1614	1616	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	78	1	0	0	1	885T
2871	1614	1615	0.100	30	1601	1	0.5	18.0	131	0	0	0	2864	0	78	1	0	0	1	885L
2872	1619	1626	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	186R
2873	1619	1625	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	88	2	0	0	1	186T
2874	1619	1624	0.100	30	1601	1	0.5	18.0	131	0	0	0	2879	0	88	2	0	0	1	186L
2875	1620	1623	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	951R
2876	1620	1626	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	88	1	0	0	1	951T
2877	1620	1625	0.100	30	1601	1	0.5	18.0	131	0	0	0	2882	0	88	1	0	0	1	951L
2878	1621	1624	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	222R
2879	1621	1623	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	88	2	0	0	1	222T
2880	1621	1626	0.100	30	1601	1	0.5	18.0	131	0	0	0	2873	0	88	2	0	0	1	222L
2881	1622	1625	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	965R
2882	1622	1624	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	88	1	0	0	1	965T
2883	1622	1623	0.100	30	1601	1	0.5	18.0	131	0	0	0	2876	0	88	1	0	0	1	965L
2884	1627	1634	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	543R
2885	1627	1633	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	103	1	0	0	1	543T
2886	1627	1632	0.100	30	1601	1	0.5	18.0	131	0	0	0	2891	0	103	1	0	0	1	543L
2887	1628	1631	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	631R
2888	1628	1634	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	103	3	0	0	1	631T
2889	1628	1633	0.100	30	1601	1	0.5	18.0	131	0	0	0	2894	0	103	2	3	0	1	631L
2890	1629	1632	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	581R
2891	1629	1631	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	103	1	0	0	1	581T
2892	1629	1634	0.100	30	1601	1	0.5	18.0	131	0	0	0	2885	0	103	1	0	0	1	581L
2893	1630	1633	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	637R
2894	1630	1632	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	103	3	0	0	1	637T
2895	1630	1631	0.100	30	1601	1	0.5	18.0	131	0	0	0	2888	0	103	3	0	0	1	637L
2896	1635	1642	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	584R
2897	1635	1641	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	131	2	0	0	1	584T
2898	1635	1640	0.100	30	1601	1	0.5	18.0	131	0	0	0	2903	0	131	2	0	0	1	584L
2899	1636	1639	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	653R
2900	1636	1642	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	131	1	0	0	1	653T
2901	1636	1641	0.100	30	1601	1	0.5	18.0	131	0	0	0	2906	0	131	1	0	0	1	653L
2902	1637	1640	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	598R
2903	1637	1639	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	131	2	0	0	1	598T
2904	1637	1642	0.100	30	1601	1	0.5	18.0	131	0	0	0	2897	0	131	2	0	0	1	598L

2905	1638	1641	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	659R
2906	1638	1640	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	131	1	0	0	1	659T
2907	1638	1639	0.100	30	1601	1	0.5	18.0	131	0	0	0	2900	0	131	1	0	0	1	659L
2908	1643	1650	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	286R
2909	1643	1649	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	154	1	0	0	1	286T
2910	1643	1648	0.100	30	1601	1	0.5	18.0	131	0	0	0	2915	0	154	1	0	0	1	286L
2911	1644	1647	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	560R
2912	1644	1650	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	154	2	0	0	1	560T
2913	1644	1649	0.100	30	1601	1	0.5	18.0	131	0	0	0	2918	0	154	2	0	0	1	560L
2914	1645	1648	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	358R
2915	1645	1647	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	154	1	0	0	1	358T
2916	1645	1650	0.100	30	1601	1	0.5	18.0	131	0	0	0	2909	0	154	1	0	0	1	358L
2917	1646	1649	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	564R
2918	1646	1648	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	154	2	0	0	1	564T
2919	1646	1647	0.100	30	1601	1	0.5	18.0	131	0	0	0	2912	0	154	2	0	0	1	564L
2920	1651	1658	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	305R
2921	1651	1657	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	167	1	0	0	1	305T
2922	1651	1656	0.100	30	1601	1	0.5	18.0	131	0	0	0	2927	0	167	1	0	0	1	305L
2923	1652	1655	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	908R
2924	1652	1658	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	167	3	0	0	1	908T
2925	1652	1657	0.100	30	1601	1	0.5	18.0	131	0	0	0	2930	0	167	3	0	0	1	908L
2926	1653	1656	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	339R
2927	1653	1655	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	167	1	0	0	1	339T
2928	1653	1658	0.100	30	1601	1	0.5	18.0	131	0	0	0	2921	0	167	1	0	0	1	339L
2929	1654	1657	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	920R
2930	1654	1656	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	167	3	0	0	1	920T
2931	1654	1655	0.100	30	1601	1	0.5	18.0	131	0	0	0	2924	0	167	2	3	0	1	920L
2932	1659	1666	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	314R
2933	1659	1665	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	175	1	0	0	1	314T
2934	1659	1664	0.100	30	1601	1	0.5	18.0	131	0	0	0	2939	0	175	1	0	0	1	314L
2935	1660	1663	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	980R
2936	1660	1666	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	175	2	0	0	1	980T
2937	1660	1665	0.100	30	1601	1	0.5	18.0	131	0	0	0	2942	0	175	2	0	0	1	980L
2938	1661	1664	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	330R
2939	1661	1663	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	175	1	0	0	1	330T
2940	1661	1666	0.100	30	1601	1	0.5	18.0	131	0	0	0	2933	0	175	1	0	0	1	330L
2941	1662	1665	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	990R
2942	1662	1664	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	175	2	0	0	1	990T
2943	1662	1663	0.100	30	1601	1	0.5	18.0	131	0	0	0	2936	0	175	2	0	0	1	990L
2944	1667	1674	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	315R
2945	1667	1673	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	177	1	0	0	1	315T
2946	1667	1672	0.100	30	1601	1	0.5	18.0	131	0	0	0	2951	0	177	1	0	0	1	315L
2947	1668	1671	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1000R
2948	1668	1674	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	177	2	0	0	1	1000T
2949	1668	1673	0.100	30	1601	1	0.5	18.0	131	0	0	0	2954	0	177	2	0	0	1	1000L
2950	1669	1672	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	329R
2951	1669	1671	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	177	1	0	0	1	329T
2952	1669	1674	0.100	30	1601	1	0.5	18.0	131	0	0	0	2945	0	177	1	0	0	1	329L
2953	1670	1673	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1014R
2954	1670	1672	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	177	2	0	0	1	1014T
2955	1670	1671	0.100	30	1601	1	0.5	18.0	131	0	0	0	2948	0	177	2	0	0	1	1014L
2956	1675	1682	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	317R
2957	1675	1681	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	180	1	0	0	1	317T
2958	1675	1680	0.100	30	1601	1	0.5	18.0	131	0	0	0	2963	0	180	1	0	0	1	317L
2959	1676	1679	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1030R
2960	1676	1682	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	180	2	0	0	1	1030T
2961	1676	1681	0.100	30	1601	1	0.5	18.0	131	0	0	0	2966	0	180	2	0	0	1	1030L
2962	1677	1680	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	327R
2963	1677	1679	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	180	1	0	0	1	327T
2964	1677	1682	0.100	30	1601	1	0.5	18.0	131	0	0	0	2957	0	180	1	0	0	1	327L
2965	1678	1681	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1046R
2966	1678	1680	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	180	2	0	0	1	1046T
2967	1678	1679	0.100	30	1601	1	0.5	18.0	131	0	0	0	2960	0	180	2	0	0	1	1046L
2968	1683	1690	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	393R
2969	1683	1689	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	190	3	0	0	1	393T
2970	1683	1688	0.100	30	1601	1	0.5	18.0	131	0	0	0	2975	0	190	3	0	0	1	393L
2971	1684	1687	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	867R

2972	1684	1690	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	190	1	0	0	1	867T		
2973	1684	1689	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	190	2	0	0	1	867L		
2974	1685	1688	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001			445R		
2975	1685	1687	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	190	3	0	0	1	445T		
2976	1685	1690	0.100	30	1601	1	0.5	18.0	131	0	0	0	2969	0	190	3	0	0	1	445L		
2977	1686	1689	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001			883R		
2978	1686	1688	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	190	1	0	0	1	883T		
2979	1686	1687	0.100	30	1601	1	0.5	18.0	131	0	0	0	0	0	190	2	0	0	1	883L		
2980	1691	1698	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	406R		
2981	1691	1697	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	201	1	0	0	1	406T		
2982	1691	1696	0.100	30	1601	1	0.5	18.0	131	0	0	0	2987	0	201	1	0	0	1	406L		
2983	1692	1695	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	1		954R		
2984	1692	1698	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	201	2	0	0	1	954T		
2985	1692	1697	0.100	30	1601	1	0.5	18.0	131	0	0	0	2990	0	201	2	0	0	1	954L		
2986	1693	1696	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001			432R		
2987	1693	1695	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	201	1	0	0	1	432T		
2988	1693	1698	0.100	30	1601	1	0.5	18.0	131	0	0	0	2981	0	201	1	0	0	1	432L		
2989	1694	1697	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	962R		
2990	1694	1696	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	201	2	001			962T		
2991	1694	1695	0.100	30	1601	1	0.5	18.0	131	0	0	0	2984	0	201	2	001			962L		
2992	1699	1706	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001			409R		
2993	1699	1705	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	203	1	0	0	1	409T		
2994	1699	1704	0.100	30	1601	1	0.5	18.0	131	0	0	0	2999	0	203	1	0	0	1	409L		
2995	1700	1703	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	981R		
2996	1700	1706	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	203	2	0	0	1	981T		
2997	1700	1705	0.100	30	1601	1	0.5	18.0	131	0	0	0	3002	0	203	2	0	0	1	981L		
2998	1701	1704	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	429R		
2999	1701	1703	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	203	1	0	0	1	429T		
3000	1701	1706	0.100	30	1601	1	0.5	18.0	131	0	0	0	2993	0	203	1	0	0	1	429L		
3001	1702	1705	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001			989R		
3002	1702	1704	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	203	2	0	0	1	989T		
3003	1702	1703	0.100	30	1601	1	0.5	18.0	131	0	0	0	2996	0	203	2	0	0	1	989L		
3004	1707	1714	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001			473R		
3005	1707	1713	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	213	2	0	0	1	473T		
3006	1707	1712	0.100	30	1601	1	0.5	18.0	131	0	0	0	3011	0	213	2	0	0	1	473L		
3007	1708	1711	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	874R		
3008	1708	1714	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	213	1	0	0	1	874T		
3009	1708	1713	0.100	30	1601	1	0.5	18.0	131	0	0	0	3014	0	213	1	0	0	1	874L		
3010	1709	1712	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001			521R		
3011	1709	1711	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	213	2	001			521T		
3012	1709	1714	0.100	30	1601	1	0.5	18.0	131	0	0	0	3005	0	213	2	001			521L		
3013	1710	1713	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001			876R		
3014	1710	1712	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	2	1	3	1	0	0	1	876T
3015	1710	1711	0.100	30	1601	1	0.5	18.0	131	0	0	0	3008	0	213	1	0	0	1	876L		
3016	1715	1722	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	477R		
3017	1715	1721	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	218	2	0	0	1	477T		
3018	1715	1720	0.100	30	1601	1	0.5	18.0	131	0	0	0	3023	0	218	2	0	0	1	477L		
3019	1716	1719	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001			913R		
3020	1716	1722	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	218	1	0	0	1	913T		
3021	1716	1721	0.100	30	1601	1	0.5	18.0	131	0	0	0	3026	0	218	1	0	0	1	913L		
3022	1717	1720	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001			517R		
3023	1717	1719	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	218	2	001			517T		
3024	1717	1722	0.100	30	1601	1	0.5	18.0	131	0	0	0	3017	0	218	2	0	0	1	517L		
3025	1718	1721	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001			915R		
3026	1718	1720	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	218	1	0	0	1	915T		
3027	1718	1719	0.100	30	1601	1	0.5	18.0	131	0	0	0	3020	0	218	1	0	0	1	915L		
3028	1723	1730	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	136R		
3029	1723	1729	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	287	1	0	0	1	136T		
3030	1723	1728	0.100	30	1601	1	0.5	18.0	131	0	0	0	3035	0	287	1	001			136L		
3031	1724	1727	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001			648R		
3032	1724	1730	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	287	2	0	0	1	648T		
3033	1724	1729	0.100	30	1601	1	0.5	18.0	131	0	0	0	3038	0	287	2	0	0	1	648L		
3034	1725	1728	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	001			272R		
3035	1725	1727	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	287	1	0	0	1	272T		
3036	1725	1730	0.100	30	1601	1	0.5	18.0	131	0	0	0	3029	0	287	1	0	0	1	272L		
3037	1726	1729	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	664R		
3038	1726	1728	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	287	2	0	0	1	664T		

3039	1726	1727	0.100	30	1601	1	0.5	18.0	131	0	0	0	3032	0	287	2	0	0	1	664L
3040	1731	1738	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	34R
3041	1731	1737	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	12	1	0	0	1	34T
3042	1731	1736	0.100	30	1601	1	0.5	18.0	131	0	0	0	3047	0	12	1	0	0	1	34L
3043	1732	1735	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	829R
3044	1732	1738	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	12	2	0	0	1	829T
3045	1732	1737	0.100	30	1601	1	0.5	18.0	131	0	0	0	3050	0	12	2	0	0	1	829L
3046	1733	1736	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	100R
3047	1733	1735	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	12	1	0	0	1	100T
3048	1733	1738	0.100	30	1601	1	0.5	18.0	131	0	0	0	3041	0	12	1	0	0	1	100L
3049	1734	1737	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	839R
3050	1734	1736	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	12	2	0	0	1	839T
3051	1734	1735	0.100	30	1601	1	0.5	18.0	131	0	0	0	3044	0	12	2	0	0	1	839L
3052	1739	1746	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	53R
3053	1739	1745	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	27	1	0	0	1	53T
3054	1739	1744	0.100	30	1601	1	0.5	18.0	131	0	0	0	3059	0	27	1	0	0	1	53L
3055	1740	1743	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	976R
3056	1740	1746	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	27	2	0	0	1	976T
3057	1740	1745	0.100	30	1601	1	0.5	18.0	131	0	0	0	3062	0	27	2	0	0	1	976L
3058	1741	1744	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	81R
3059	1741	1743	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	27	1	0	0	1	81T
3060	1741	1746	0.100	30	1601	1	0.5	18.0	131	0	0	0	3053	0	27	1	0	0	1	81L
3061	1742	1745	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	994R
3062	1742	1744	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	27	2	0	0	1	994T
3063	1742	1743	0.100	30	1601	1	0.5	18.0	131	0	0	0	3056	0	27	2	0	0	1	994L
3064	1747	1754	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	61R
3065	1747	1753	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	34	1	0	0	1	61T
3066	1747	1752	0.100	30	1601	1	0.5	18.0	131	0	0	0	3071	0	34	1	0	0	1	61L
3067	1748	1751	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1054R
3068	1748	1754	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	34	2	0	0	1	1054T
3069	1748	1753	0.100	30	1601	1	0.5	18.0	131	0	0	0	3074	0	34	2	0	0	1	1054L
3070	1749	1752	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	73R
3071	1749	1751	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	34	1	0	0	1	73T
3072	1749	1754	0.100	30	1601	1	0.5	18.0	131	0	0	0	3065	0	34	1	0	0	1	73L
3073	1750	1753	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1062R
3074	1750	1752	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	34	2	0	0	1	1062T
3075	1750	1751	0.100	30	1601	1	0.5	18.0	131	0	0	0	3068	0	34	2	0	0	1	1062L
3076	1755	1762	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	528R
3077	1755	1761	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	42	1	0	0	1	528T
3078	1755	1760	0.100	30	1601	1	0.5	18.0	131	0	0	0	3083	0	42	1	0	0	1	528L
3079	1756	1759	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	830R
3080	1756	1762	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	42	2	0	0	1	830T
3081	1756	1761	0.100	30	1601	1	0.5	18.0	131	0	0	0	3086	0	42	2	0	0	1	830L
3082	1757	1760	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	540R
3083	1757	1759	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	42	1	0	0	1	540T
3084	1757	1762	0.100	30	1601	1	0.5	18.0	131	0	0	0	3077	0	42	1	0	0	1	540L
3085	1758	1761	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	838R
3086	1758	1760	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	42	2	0	0	1	838T
3087	1758	1759	0.100	30	1601	1	0.5	18.0	131	0	0	0	3080	0	42	2	0	0	1	838L
3088	1763	1770	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	165R
3089	1763	1769	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	72	1	0	0	1	165T
3090	1763	1768	0.100	30	1601	1	0.5	18.0	131	0	0	0	3095	0	72	1	0	0	1	165L
3091	1764	1767	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	800R
3092	1764	1770	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	72	2	0	0	1	800T
3093	1764	1769	0.100	30	1601	1	0.5	18.0	131	0	0	0	3098	0	72	2	0	0	1	800L
3094	1765	1768	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	243R
3095	1765	1767	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	72	1	0	0	1	243T
3096	1765	1770	0.100	30	1601	1	0.5	18.0	131	0	0	0	3089	0	72	1	0	0	1	243L
3097	1766	1769	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	816R
3098	1766	1768	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	72	2	0	0	1	816T
3099	1766	1767	0.100	30	1601	1	0.5	18.0	131	0	0	0	3092	0	72	2	0	0	1	816L
3100	1771	1778	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	170R
3101	1771	1777	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	75	1	0	0	1	170T
3102	1771	1776	0.100	30	1601	1	0.5	18.0	131	0	0	0	3107	0	75	1	0	0	1	170L
3103	1772	1775	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	831R
3104	1772	1778	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	75	2	0	0	1	831T
3105	1772	1777	0.100	30	1601	1	0.5	18.0	131	0	0	0	3110	0	75	2	0	0	1	831L

3106	1773	1776	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	238R
3107	1773	1775	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	75	1	0	0	1	238T
3108	1773	1778	0.100	30	1601	1	0.5	18.0	131	0	0	0	3101	0	75	1	0	0	1	238L
3109	1774	1777	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	837R
3110	1774	1776	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	75	2	0	0	1	837T
3111	1774	1775	0.100	30	1601	1	0.5	18.0	131	0	0	0	3104	0	75	2	0	0	1	837L
3112	1779	1786	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	189R
3113	1779	1785	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	90	1	0	0	1	189T
3114	1779	1784	0.100	30	1601	1	0.5	18.0	131	0	0	0	3119	0	90	1	0	0	1	189L
3115	1780	1783	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	978R
3116	1780	1786	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	90	2	0	0	1	978T
3117	1780	1785	0.100	30	1601	1	0.5	18.0	131	0	0	0	3122	0	90	2	0	0	1	978L
3118	1781	1784	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	219R
3119	1781	1783	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	90	1	0	0	1	219T
3120	1781	1786	0.100	30	1601	1	0.5	18.0	131	0	0	0	3113	0	90	1	0	0	1	219L
3121	1782	1785	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	992R
3122	1782	1784	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	90	2	0	0	1	992T
3123	1782	1783	0.100	30	1601	1	0.5	18.0	131	0	0	0	3116	0	90	2	0	0	1	992L
3124	1787	1794	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	192R
3125	1787	1793	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	93	2	0	0	1	192T
3126	1787	1792	0.100	30	1601	1	0.5	18.0	131	0	0	0	3131	0	93	2	0	0	1	192L
3127	1788	1791	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	999R
3128	1788	1794	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	93	1	0	0	1	999T
3129	1788	1793	0.100	30	1601	1	0.5	18.0	131	0	0	0	3134	0	93	1	0	0	1	999L
3130	1789	1792	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	216R
3131	1789	1791	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	93	2	0	0	1	216T
3132	1789	1794	0.100	30	1601	1	0.5	18.0	131	0	0	0	3125	0	93	2	0	0	1	216L
3133	1790	1793	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1015R
3134	1790	1792	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	93	1	0	0	1	1015T
3135	1790	1791	0.100	30	1601	1	0.5	18.0	131	0	0	0	3128	0	93	1	0	0	1	1015L
3136	1795	1802	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	195R
3137	1795	1801	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	96	1	0	0	1	195T
3138	1795	1800	0.100	30	1601	1	0.5	18.0	131	0	0	0	3143	0	96	1	0	0	1	195L
3139	1796	1799	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1028R
3140	1796	1802	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	96	2	0	0	1	1028T
3141	1796	1801	0.100	30	1601	1	0.5	18.0	131	0	0	0	3146	0	96	2	0	0	1	1028L
3142	1797	1800	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	213R
3143	1797	1799	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	96	1	0	0	1	213T
3144	1797	1802	0.100	30	1601	1	0.5	18.0	131	0	0	0	3137	0	96	1	0	0	1	213L
3145	1798	1801	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1048R
3146	1798	1800	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	96	2	0	0	1	1048T
3147	1798	1799	0.100	30	1601	1	0.5	18.0	131	0	0	0	3140	0	96	2	0	0	1	1048L
3148	1803	1810	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	278R
3149	1803	1809	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	148	1	0	0	1	278T
3150	1803	1808	0.100	30	1601	1	0.5	18.0	131	0	0	0	3155	0	148	1	0	0	1	278L
3151	1804	1807	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	687R
3152	1804	1810	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	148	2	0	0	1	687T
3153	1804	1809	0.100	30	1601	1	0.5	18.0	131	0	0	0	3158	0	148	2	0	0	1	687L
3154	1805	1808	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	366R
3155	1805	1807	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	148	1	0	0	1	366T
3156	1805	1810	0.100	30	1601	1	0.5	18.0	131	0	0	0	3149	0	148	1	0	0	1	366L
3157	1806	1809	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	691R
3158	1806	1808	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	148	2	0	0	1	691T
3159	1806	1807	0.100	30	1601	1	0.5	18.0	131	0	0	0	3152	0	148	2	0	0	1	691L
3160	1811	1818	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	289R
3161	1811	1817	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	156	1	0	0	1	289T
3162	1811	1816	0.100	30	1601	1	0.5	18.0	131	0	0	0	3167	0	156	1	2	0	1	289L
3163	1812	1815	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	776R
3164	1812	1818	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	156	3	0	0	1	776T
3165	1812	1817	0.100	30	1601	1	0.5	18.0	131	0	0	0	3170	0	156	3	0	0	1	776L
3166	1813	1816	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	355R
3167	1813	1815	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	156	1	0	0	1	355T
3168	1813	1818	0.100	30	1601	1	0.5	18.0	131	0	0	0	3161	0	156	1	2	0	1	355L
3169	1814	1817	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	780R
3170	1814	1816	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	156	3	0	0	1	780T
3171	1814	1815	0.100	30	1601	1	0.5	18.0	131	0	0	0	3164	0	156	3	0	0	1	780L
3172	1819	1826	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	296R

3173	1819	1825	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	160	1	0	0	1	296T
3174	1819	1824	0.100	30	1601	1	0.5	18.0	131	0	0	0	3179	0	160	1	0	0	1	296L
3175	1820	1823	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	832R
3176	1820	1826	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	160	2	0	0	1	832T
3177	1820	1825	0.100	30	1601	1	0.5	18.0	131	0	0	0	3182	0	160	2	0	0	1	832L
3178	1821	1824	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	348R
3179	1821	1823	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	160	1	0	0	1	348T
3180	1821	1826	0.100	30	1601	1	0.5	18.0	131	0	0	0	3173	0	160	1	0	0	1	348L
3181	1822	1825	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	836R
3182	1822	1824	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	160	2	0	0	1	836T
3183	1822	1823	0.100	30	1601	1	0.5	18.0	131	0	0	0	3176	0	160	2	0	0	1	836L
3184	1827	1834	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	300R
3185	1827	1833	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	163	2	0	0	1	300T
3186	1827	1832	0.100	30	1601	1	0.5	18.0	131	0	0	0	3191	0	163	2	0	0	1	300L
3187	1828	1831	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	866R
3188	1828	1834	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	163	1	0	0	1	866T
3189	1828	1833	0.100	30	1601	1	0.5	18.0	131	0	0	0	3194	0	163	1	0	0	1	866L
3190	1829	1832	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	344R
3191	1829	1831	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	163	2	0	0	1	344T
3192	1829	1834	0.100	30	1601	1	0.5	18.0	131	0	0	0	3185	0	163	2	0	0	1	344L
3193	1830	1833	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	884R
3194	1830	1832	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	163	1	0	0	1	884T
3195	1830	1831	0.100	30	1601	1	0.5	18.0	131	0	0	0	3188	0	163	1	0	0	1	884L
3196	1835	1842	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	308R
3197	1835	1841	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	170	1	0	0	1	308T
3198	1835	1840	0.100	30	1601	1	0.5	18.0	131	0	0	0	3203	0	170	1	0	0	1	308L
3199	1836	1839	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	934R
3200	1836	1842	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	170	2	0	0	1	934T
3202	1837	1840	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	336R
3203	1837	1839	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	170	1	0	0	1	336T
3204	1837	1842	0.100	30	1601	1	0.5	18.0	131	0	0	0	3197	0	170	1	0	0	1	336L
3205	1838	1841	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	944R
3206	1838	1840	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	170	2	0	0	1	944T
3208	1843	1850	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	398R
3209	1843	1849	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	194	2	0	0	1	398T
3210	1843	1848	0.100	30	1601	1	0.5	18.0	131	0	0	0	3215	0	194	2	0	0	1	398L
3211	1844	1847	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	909R
3212	1844	1850	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	194	1	0	0	1	909T
3213	1844	1849	0.100	30	1601	1	0.5	18.0	131	0	0	0	3218	0	194	1	0	0	1	909L
3214	1845	1848	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	440R
3215	1845	1847	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	194	2	0	0	1	440T
3216	1845	1850	0.100	30	1601	1	0.5	18.0	131	0	0	0	3209	0	194	2	0	0	1	440L
3217	1846	1849	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	919R
3218	1846	1848	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	194	1	0	0	1	919T
3219	1846	1847	0.100	30	1601	1	0.5	18.0	131	0	0	0	3212	0	194	3	1	0	1	919L
3220	1851	1858	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	403R
3221	1851	1857	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	198	1	0	0	1	403T
3222	1851	1856	0.100	30	1601	1	0.5	18.0	131	0	0	0	3227	0	198	1	0	0	1	403L
3223	1852	1855	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	935R
3224	1852	1858	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	198	2	0	0	1	935T
3226	1853	1856	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	435R
3227	1853	1855	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	198	1	0	0	1	435T
3228	1853	1858	0.100	30	1601	1	0.5	18.0	131	0	0	0	3221	0	198	1	0	0	1	435L
3229	1854	1857	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	943R
3230	1854	1856	0.100	50	1800	1	0.5	30.0	131	0	0	0	0	0	198	2	0	0	1	943T
3232	1859	1866	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	411R
3233	1859	1865	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	205	2	0	0	1	411T
3234	1859	1864	0.100	30	1601	1	0.5	18.0	131	0	0	0	3239	0	205	2	0	0	1	411L
3235	1860	1863	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1002R
3236	1860	1866	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	205	1	0	0	1	1002T
3237	1860	1865	0.100	30	1601	1	0.5	18.0	131	0	0	0	3242	0	205	1	0	0	1	1002L
3238	1861	1864	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	427R
3239	1861	1863	0.100	50	1800	3	0.5	30.0	131	0	0	0	0	0	205	2	0	0	1	427T
3240	1861	1866	0.100	30	1601	1	0.5	18.0	131	0	0	0	3233	0	205	2	0	0	1	427L
3241	1862	1865	0.100	10	1402	1	0.5	6.0	131	0	0	0	0	0	0	0	0	0	1	1012R
3242	1862	1864	0.100	50	1800	2	0.5	30.0	131	0	0	0	0	0	205	1	0	0	1	1012T
3243	1862	1863	0.100	30	1601	1	0.5	18.0	131	0	0	0	3236	0	205	1	0	0	1	1012L

3244	1867	1874	0.100	10	1402	1	0	5	6.0	131	0	0	0	0	0	0	0	001	469R		
3245	1867	1873	0.100	50	1800	1	0	5	30.0	131	0	0	0	0	0	207	2	0	0	1	469T
3246	1867	1872	0.100	30	1601	1	0	5	18.0	131	0	0	0	3251	0	207	2	0	0	1	469L
3247	1868	1871	0.100	10	1402	1	0	5	6.0	131	0	0	0	0	0	0	0	0	0	1	807R
3248	1868	1874	0.100	50	1800	2	0	5	30.0	131	0	0	0	0	0	207	1	0	0	1	807T
3249	1868	1873	0.100	30	1601	1	0	5	18.0	131	0	0	0	3254	0	207	1	0	0	1	807L
3250	1869	1872	0.100	10	1402	1	0	5	6.0	131	0	0	0	0	0	0	0	0	0	1	525R
3251	1869	1871	0.100	50	1800	2	0	5	30.0	131	0	0	0	0	0	207	2	0	0	1	525T
3252	1869	1874	0.100	30	1601	1	0	5	18.0	131	0	0	0	3245	0	207	2	0	0	1	525L
3253	1870	1873	0.100	10	1402	1	0	5	6.0	131	0	0	0	0	0	0	0	0	0	1	809R
3254	1870	1872	0.100	50	1800	2	0	5	30.0	131	0	0	0	0	0	207	1	0	0	1	809T
3255	1870	1871	0.100	30	1601	1	0	5	18.0	131	0	0	0	3248	0	207	1	0	0	1	809L
3256	1875	1882	0.100	10	1402	1	0	5	6.0	131	0	0	0	0	0	0	0	0	0	1	481R
3257	1875	1881	0.100	50	1800	2	0	5	30.0	131	0	0	0	0	0	223	1	0	0	1	481T
3258	1875	1880	0.100	30	1601	1	0	5	18.0	131	0	0	0	3263	0	223	1	0	0	1	481L
3259	1876	1879	0.100	10	1402	1	0	5	6.0	131	0	0	0	0	0	0	0	0	0	1	938R
3260	1876	1882	0.100	50	1800	1	0	5	30.0	131	0	0	0	0	0	223	2	0	0	1	938T
3262	1877	1880	0.100	10	1402	1	0	5	6.0	131	0	0	0	0	0	0	0	0	0	1	513R
3263	1877	1879	0.100	50	1800	2	0	5	30.0	131	0	0	0	0	0	223	1	0	0	1	513T
3264	1877	1882	0.100	30	1601	1	0	5	18.0	131	0	0	0	3257	0	223	1	0	0	1	513L
3265	1878	1881	0.100	10	1402	1	0	5	6.0	131	0	0	0	0	0	0	0	0	0	1	940R
3266	1878	1880	0.100	50	1800	1	0	5	30.0	131	0	0	0	0	0	223	2	0	0	1	940T
3268	1883	1890	0.100	10	1402	1	0	5	6.0	131	0	0	0	0	0	0	0	0	0	1	485R
3269	1883	1889	0.100	50	1800	2	0	5	30.0	131	0	0	0	0	0	228	2	0	0	1	485T
3270	1883	1888	0.100	30	1601	1	0	5	18.0	131	0	0	0	3275	0	228	2	0	0	1	485L
3271	1884	1887	0.100	10	1402	1	0	5	6.0	131	0	0	0	0	0	0	0	0	0	1	957R
3272	1884	1890	0.100	50	1800	2	0	5	30.0	131	0	0	0	0	0	228	1	0	0	1	957T
3273	1884	1889	0.100	30	1601	1	0	5	18.0	131	0	0	0	0	0	228	3	0	0	1	957L
3274	1885	1888	0.100	10	1402	1	0	5	6.0	131	0	0	0	0	0	0	0	0	0	1	509R
3275	1885	1887	0.100	50	1800	2	0	5	30.0	131	0	0	0	0	0	228	2	0	0	1	509T
3276	1885	1890	0.100	30	1601	1	0	5	18.0	131	0	0	0	3269	0	228	2	0	0	1	509L
3277	1886	1889	0.100	10	1402	1	0	5	6.0	131	0	0	0	0	0	0	0	0	0	1	959R
3278	1886	1888	0.100	50	1800	2	0	5	30.0	131	0	0	0	0	0	228	1	0	0	1	959T
3279	1886	1887	0.100	30	1601	1	0	5	18.0	131	0	0	0	0	0	228	3	0	0	1	959L
3280	1891	1898	0.100	10	1402	1	0	5	6.0	131	0	0	0	0	0	0	0	0	0	1	3R
3281	1891	1897	0.100	50	1800	3	0	5	30.0	131	0	0	0	0	0	274	1	0	0	1	3T
3282	1891	1896	0.100	30	1601	1	0	5	18.0	131	0	0	0	3287	0	274	1	0	0	1	3L
3283	1892	1895	0.100	10	1402	1	0	5	6.0	131	0	0	0	0	0	0	0	0	0	1	647R
3284	1892	1898	0.100	50	1800	2	0	5	30.0	131	0	0	0	0	0	274	2	0	0	1	647T
3285	1892	1897	0.100	30	1601	1	0	5	18.0	131	0	0	0	3290	0	274	2	0	0	1	647L
3286	1893	1896	0.100	10	1402	1	0	5	6.0	131	0	0	0	0	0	0	0	0	0	1	131R
3287	1893	1895	0.100	50	1800	3	0	5	30.0	131	0	0	0	0	0	274	1	0	0	1	131T
3288	1893	1898	0.100	30	1601	1	0	5	18.0	131	0	0	0	3281	0	274	1	0	0	1	131L
3289	1894	1897	0.100	10	1402	1	0	5	6.0	131	0	0	0	0	0	0	0	0	0	1	665R
3290	1894	1896	0.100	50	1800	2	0	5	30.0	131	0	0	0	0	0	274	2	0	0	1	665T
3291	1894	1895	0.100	30	1601	1	0	5	18.0	131	0	0	0	3284	0	274	2	0	0	1	665L
3292	1899	1904	0.100	50	1800	3	0	5	30.0	131	0	0	0	0	0	293	1	0	0	1	143T
3293	1899	1903	0.100	30	1601	1	0	5	18.0	131	0	0	0	3297	0	293	1	0	0	1	143L
3294	1900	1902	0.100	10	1402	1	0	5	6.0	131	0	0	0	0	0	0	0	0	0	1	702R
3295	1900	1904	0.100	30	1601	1	0	5	18.0	131	0	0	0	0	0	293	2	0	0	1	702L
3296	1901	1903	0.100	10	1402	1	0	5	6.0	131	0	0	0	0	0	0	0	0	0	1	265R
3297	1901	1902	0.100	50	1800	3	0	5	30.0	131	0	0	0	0	0	293	1	0	0	1	265T
3298	1905	1912	0.100	10	1402	1	0	5	6.0	131	0	0	0	0	0	0	0	0	0	1	381R
3299	1905	1911	0.100	50	1800	2	0	5	30.0	131	0	0	0	0	0	354	1	0	0	1	381T
3300	1905	1910	0.100	30	1601	1	0	5	18.0	131	0	0	0	3305	0	354	1	0	0	1	381L
3301	1906	1909	0.100	10	1402	1	0	5	6.0	131	0	0	0	0	0	0	0	0	0	1	561R
3302	1906	1912	0.100	50	1800	2	0	5	30.0	131	0	0	0	0	0	354	2	0	0	1	561T
3303	1906	1911	0.100	30	1601	1	0	5	18.0	131	0	0	0	3308	0	354	2	0	0	1	561L
3304	1907	1910	0.100	10	1402	1	0	5	6.0	131	0	0	0	0	0	0	0	0	0	1	457R
3305	1907	1909	0.100	50	1800	2	0	5	30.0	131	0	0	0	0	0	354	1	0	0	1	457T
3306	1907	1912	0.100	30	1601	1	0	5	18.0	131	0	0	0	3299	0	354	1	0	0	1	457L
3307	1908	1911	0.100	10	1402	1	0	5	6.0	131	0	0	0	0	0	0	0	0	0	1	563R
3308	1908	1910	0.100	50	1800	2	0	5	30.0	131	0	0	0	0	0	354	2	0	0	1	563T
3309	1908	1909	0.100	30	1601	1	0	5	18.0	131	0	0	0	3302	0	354	2	0	0	1	563L
3310	1913	1919	0.100	10	1402	1	0	5	6.0	131	0	0	0	0	0	0	0	0	0	1	616R
3311	1913	1918	0.100	30	1601	2	0	5	18.0	131	0	0	0	0	0	157	2	0	0	1	616L
3312	1914	1917	0.100	10	1402	1	0	5	6.0	131	0	0	0	0	0	0	0	0	0	1	353R

3313	1914	1919	0	100	50	1800	3	0.5	30	0	131	0	0	0	0	0	157	1	0	0	1	353T
3314	1915	1918	0	100	10	1402	1	0.5	6	0	131	0	0	0	0	0	0	0	0	0	1	1279R
3315	1915	1917	0	100	50	1800	2	0.0	30	0	131	0	0	0	0	0	157	3	0	0	1	1279T
3316	1915	1919	0	100	30	1601	1	0.0	18	0	131	0	0	0	3309	0	157	3	0	0	1	1279L
3317	1916	1918	0	100	50	1800	4	0.5	30	0	131	0	0	0	0	0	157	1	0	0	1	291T
3318	1916	1917	0	100	30	1601	1	0.5	18	0	131	0	0	0	3311	0	157	1	0	0	1	291L

Appendix R: The TOSEED Program

This program is used to generate a seed origin/destination matrix that is used with the **QUEENSOD** program. The program reads a node file and a file that contains weighting factors for each of the zones in the network. The program then assigns a seed value to each feasible origin/destination pair in the network. The program may be run for multiple time slices.

The C++ source code for the program is given below.

```
// file = TOSEED.CPP

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

#define MAXNUMOFZONES 300
#define MAXNUMOFNODES 4000

void getslicefiles();
void openzonefile(int i);
void opennodefile();
int getdefaultflow();
int getLargeFlowFactor();
float getseedfactor();
void writeseedfile(int i);
int destinations[MAXNUMOFZONES];
int origins[MAXNUMOFZONES];
struct azone{
    int nodenum;
    int oflow;
    int dflow;
} thezones[MAXNUMOFZONES];

int demand, defaultflow, numofflows, numofslices, slicetime;
int numoforigins = 0, numofdests = 0, numofones = 0, largeFlowFactor;
float seedfactor;
char ch, tsfile[40][12], odfile[12];
FILE *nodein, *flowin;
long int odpair = 1;
main()
{
    int i;
    defaultflow = getdefaultflow();
    largeFlowFactor = getLargeFlowFactor();
    seedfactor = getseedfactor();
    opennodefile();
    getslicefiles();
    for (i = 0; i < numofslices; i ++){
        openzonefile(i);
        writeseedfile(i);
        printf("\nProcessing file %s.\n", tsfile[1]);
    }
    printf ("\n\nI counted %d origins and ", numoforigins);
    printf ("%d destinations.", numofdests);
    printf ("\n\nI created a seed demand file using ");
    printf ("%d timeslice(s).", numofslices);
    printf ("\n\nThis file is called %s.", odfile);

    return (0);
}
```

```

void getslicefiles(void)

    char tsfilename[12];
    FILE *tsnamein;
    int i;

    printf("\n\nEnter the name of the file with the time slice files: ");
    scanf("%s", tsfilename);
    printf("\n\nEnter the length of each time slice in seconds: ");
    scanf("%d", &slicetime);
    if ((tsnamein = fopen(tsfilename, "rt")) == NULL) {
        printf ("Time slice file list file not opened!");
        exit(1);

        fscanf(tsnamein, "%d", &numofslices);
        if (numofslices > 40) {
            printf("\n\nFATAL ERROR - The maximum number of time slices is 40");
            exit(1);
        }
        ch = 'a'; while (ch != 10) {ch = fgetc(tsnamein);} ch = 'a';
        for(i = 0; i < numofslices; i ++){
            fscanf(tsnamein, "%s", tsfile[i]);
            ch = 'a'; while (ch != 10) {ch = fgetc(tsnamein);} ch = 'a';

            fclose(tsnamein);
        }
}

void openzonefile(int i)
{
    int k, j, node, origin, flow, fileOflow, fileDflow;
    float xco, yco;
    if ((flowin = fopen(tsfile[i], "rt")) == NULL) {
        printf ("\n\nFlow data input file not opened!");
        exit(1);

        fscanf(flowin, "%d", &numofflows);
        if (numofflows > MAXNUMOFZONES) {
            printf("You have more flows than are allowed");
            exit(0);

            ch = 'a'; while (ch != 10) {ch = fgetc(flowin);} ch = 'a';
            for (k = 0; k < numofflows; k ++){
                fscanf(flowin, "%d %d %d", &node, &fileOflow, &fileDflow);
                if (fileOflow == 0) fileOflow = defaultflow;
                if (fileDflow == 0) fileDflow = defaultflow;
                thezones[k].nodenum = node;
                thezones[k].oflow = fileOflow;
                thezones[k].dflow = fileDflow;
                while (ch != 10) {ch = fgetc(flowin);} ch = 'a';
            }
            fclose(flowin);
        }
}

void opennodefile()
{
    int k, m = 0, n = 0, nodenum, nodetype, numofnodes;
    float xco, yco;
    char nodefile[12];
    clrscr();
    printf("\n\nEnter the name of the INTEGRATION file with the node data. ");
    scanf("%s", nodefile);
    if ((nodein = fopen(nodefile, "rt")) == NULL) {
        printf ("\n\nNode data input file not opened!");
        exit(1);
    }
}

```

```

}
ch = 'a'; while (ch != 10) {ch = fgetc(nodein);} ch = 'a';
fscanf(nodein, "%d", &numofnodes);
if (numofnodes > MAXNUMOFNODES) {
    printf("You may only have %d nodes", MAXNUMOFNODES);
    exit(0);
}
ch = 'a'; while (ch != 10) {ch = fgetc(nodein);} ch = 'a';
for (k = 0; k < numofnodes; k ++){
    fscanf(nodein, "%d %f %f %d", &nodenum, &xco, &yco, &nodetype);
    ch = 'a'; while (ch != 10) (ch = fgetc(nodein)); ch = 'a';
    if (nodetype == 1 || nodetype == 3) {
        origins[m] = nodenum;
        m ++; numoforigins ++;
    }
    if (nodetype == 1 || nodetype == 2) {
        destinations[n] = nodenum;
        n ++; numofdests ++;
    }
    if (nodetype == 1) numofones ++;
}
fclose(nodein);
}

int getdefaultflow()
{
    int defaultflow;
    printf("\n\nEnter the default flow for those zones with no link flows: ");
    scanf("%d", &defaultflow);
    return defaultflow;
}

int getLargeFlowFactor()
{
    int largeFlowFactor;
    printf("\n\nEnter the large flow factor (200 was first used): ");
    scanf("%d", &largeFlowFactor);
    return largeFlowFactor;
}

float getseedfactor()
{
    float seedfactor;
    printf("\n\nEnter the seed factor (2 was first value used): ");
    scanf("%f", &seedfactor);
    return seedfactor;
}

void writeseedfile(int i)

    int m, n, p, q, numofodpairs, oflow;
    long int flow1, flow2, lfflow1, lfflow2;
    float finalseed = 0;
    FILE *odout;
    if (i == 0) {
        numofodpairs = ((numoforigins) * (numofdests)) - numofones;
        numofodpairs *= numofslices;
        printf("\n\nWhat is the name of the seed file to be created? ");
        scanf("%s", odfile);
        if ((odout = fopen(odfile, "wt")) == NULL) {
            printf("\n\nO/D data input cannot be created!");
            exit(0);
        }
        fprintf(odout, "Seed file, SF = %f, DF = %d, lf = %d.\n", \
            seedfactor, defaultflow, largeFlowFactor);
        fprintf(odout, "%d 0 0 1.0\n", numofodpairs);

```

```

else odout = fopen(odfile, "at");

for (m = 0; m < numoforigins; m ++){
    flow1 = defaultflow;
    for (p = 0; p < numofflows; p ++){
        if (thezones[p].nodenum == origins[m]){
            flow1 = thezones[p].oflow;
            break;
        }
    }
    oflow = flow1;
    for (n = 0; n < numofdests; n ++){
        if (origins[m] != destinations[n]){
            flow2 = defaultflow;
            flow1 = oflow;
            for (q = 0; q < numofflows; q ++){
                if (thezones[q].nodenum == destinations[n]){
                    flow2 = thezones[q].dflow;
                    break;
                }
            }
            lfflow1 = flow1/500;
            lfflow2 = flow2/500;
            flow1 t= (lfflow1 * largeFlowFactor);
            flow2 t= (lfflow2 * largeFlowFactor);
            finalseed = ((flow1 * flow2) / (100 * seedfactor*seedfactor));
            fprintf(odout, "%51d %5d %5d %5.0f 1.0 ",
                odpair, origins[m], destinations[n], finalseed);
            fprintf(odout, "%5d %5d ",
                (i * slicetime), ((i+1) * slicetime));
            fprintf(odout, "1.0 0.0 0.0 0.0 0.0 0.0 1.0\n");
            odpair ++;
        }
    }
}
fclose (odout);

```

Appendix S: Final Freeway and Arterial Link Flows

This appendix contains the final link flow values that were used in the demand estimation process. There are data for each half hour time slice from 6:00 a.m. to 8:00 p.m.

Link #	Street	From	To	Direction	5:00	6:30
1	Olmypic	WEST END	Sawtelle	E	210	276
3	Olmypic	Conter	Sepulveda	E	180	272
8	Olmypic	Malcolm	Overland	E	382	400
25	Olmypic	Bedford	La Cienega	E	232	300
27	Olmypic	La Jolla	Crescent Hts.	E	232	284
29	Olmypic	MB48	Fairfax	E	212	272
34	Olmypic	Curson	Hauser	E	262	372
37	Olmypic	Cochran	La Brea	E	128	240
39	Olmypic	Mansfield	Highland	E	236	348
46	Olmypic	Lucerne	Crenshaw	E	336	576
49	Olmypic	4th	Arlington	E	240	384
53	Olmypic	Harvard	Normandie	E	496	748
57	Olmypic	Catalina	Vermont	E	500	656
60	Olmypic	Elden	Hoover	E	484	796
61	Olmypic	Hoover	Alvarado	E	416	696
73	Olmypic	Burlington	Alvarado	W	454	492
74	Olmypic	Alvarado	Hoover	W	290	436
81	Olmypic	Mariposa	Normandie	W	472	696
83	Olmypic	Harvard	Western	W	576	856
88	Olmypic	Norton	Crenshaw	W	324	576
95	Olmypic	MB55	Hingland	W	110	224
100	Olmypic	MB52	Hauser	W	304	596
107	Olmypic	MB48	Crescent Hts.	W	246	528
126	Olmypic	Manning	Overland	W	336	500
131	Olmypic	MB40	Sepulveda	W	196	316
131	Olmypic	Conter	Sawtelle	W	270	444
135	Pico	WEST END	Sawtelle	E	170	392
136	Pico	Cotner	Sepulveda	E	246	616
141	Pico	Malcolm	Overland	E	206	360
141	Pico	Manning	Prosser	E	356	548
144	Pico	Prosser	Patricia	E	418	624
146	Pico	MB41	Beverly Glen	E	216	336
152	Pico	Roxbury	Beverwil	E	122	224
158	Pico	Livonia	Robertson	E	188	252
162	Pico	MB46	La Cienega	E	642	856
170	Pico	Curson	Hauser	E	140	216
175	Pico	Mansfield	San Vicente	E	162	212
184	Pico	4th	Arlington	E	280	452
186	Pico	Wilton	Western	E	196	304
182	Pico	MB62	Normandie	E	196	340
195	Pico	Magnolia	Hoover	E	242	304
196	Pico	Hoover	Alvarado	E	210	376
212	Pico	Bonnie Brae	Alvarado	W	222	300
213	Pico	Alvarado	ioover	W	210	336
219	Pico	Mariposa	Jormandie	W	196	384
222	Pico	Harvard	N e s t e r n	W	184	332
224	Pico	Wilton	Arlington	W	272	476
228	Pico	Norton	Crenshaw	W	248	328
233	Pico	Rimpau	San Vicente	W	256	356
238	Pico	Cochran	Hauser	W	214	404
243	Pico	Gennesse	Fairfax	W	412	844
246	Pico	Crescent Hts.	La Cienega	W	432	660
250	Pico	Shenandoah	Robertson	W	234	428
256	Pico	3everly Dr.	3everwil	W	294	512
262	Pico	Kerwood	3everly Glen	W	206	316
264	Pico	WB41	Patricia	W	248	396
265	Pico	Patricia	Prosser	W	230	332
267	Pico	Manning	Overland	W	200	272

ink #	Street	From	To	Direction	6:00	6:30
272	Pico	MB39	Sepulveda	W	188	276
273	Pico	Cotner	Sawtelle	W	192	268
275	Venice	WEST END	Sawtelle	E	528	832
276	Venice	Sawtelle	Sepulveda	E	358	608
278	Venice	Midvale	Overland	E	440	668
281	Venice	MB6	Clarington	E	302	476
285	Venice	MB7	Robertson	E	462	912
286	Venice	Robertson	National	E	450	840
289	Venice	Cattaraugus	La Cienega	E	372	476
291	Venice	I - 10 South	Cadillac	E	148	324
293	Venice	MB33	Fairfax	E	218	428
296	Venice	Airdrome	Hauser	E	240	452
300	Venice	Redondo	La Brea	E	182	400
303	Venice	MB31	Vineyard	E	132	204
305	Venice	Lafayette	Crenshaw	E	268	420
308	Venice	4th	Arlington	E	184	312
312	Venice	MB30	Western	E	194	308
314	Venice	Harvard	Normandie	E	410	688
315	Venice	Normandie	Vermont	E	204	392
317	Venice	Westmoreland	Hoover	E	1478	1996
319	Venice	Burlington	Union	E	190	392
325	Venice	Oak	Union	W	120	176
327	Venice	Burlington	Hoover	W	154	228
329	Venice	Westmoreland	Vermont	W	176	272
330	Venice	Vermont	Normandie	W	440	616
332	Venice	Harvard	Western	W	226	416
339	Venice	7th/6th	Crenshaw	W	428	676
341	Venice	Lafayette	S. V. (Vineyard)	W	246	480
344	Venice	Rimpau	La Brea	W	186	380
348	Venice	MB32	Hauser	W	330	536
351	Venice	Thurman	Fairfax	W	386	648
353	Venice	MB33	Cadillac	W	314	356
355	Venice	I - 10 South	La Cienega	W	384	732
358	Venice	MB10	National	W	390	532
359	Venice	National	Robertson	W	486	680
363	Venice	Hughes	Clarington	W	350	596
366	Venice	Motor	Overland	W	376	576
368	Venice	Midvale	Sepulveda	W	366	608
369	Venice	Sepulveda	Sawtelle	W	378	596
386	Washington	Adams	Fairfax	E	136	192
388	Washington	I-10 South	I - 10 North	E	90	172
390	Washington	Curson	Hauser	E	188	228
393	Washington	Redondo	La Brea	E	628	1108
397	Washington	S. V. (Vineyard)	W e s t	E	176	284
398	Washington	Nest	Crenshaw	E	392	596
403	Washington	4th	Arlington	E	206	308
406	Washington	MB20	Western	E	588	912
409	Washington	MB21	Normandie	E	446	620
411	Washington	3udlong	Vermont	E	116	244
429	Washington	3udlong	Normandie	W	382	588
432	Washington	Harvard	Western	W	430	560
435	Washington	Gramercy	Arlington	W	220	372
440	Washington	Edgehill	Crenshaw	W	286	464
441	Washington	Crenshaw	West	W	196	344
445	Washington	Rimpau	La Brea	W	642	944
450	Washington	Curson	I - 10 North	W	376	588
452	Washington	- 10 south	Fairfax	W	1036	1416
469	Adams	Nashington	Fairfax	E	50	44

ink #	Street	From	To	Direction	∫:00	∫:30
470	Adams	Fairfax	Hauser	E	106	172
473	Adams	Sycamore	La Brea	E	404	564
475	Adams	Harcourt	Hillcrest	E	132	200
477	Adams	Buckingham	Crenshaw	E	358	632
481	Adams	MB18	Arlington	E	230	392
485	Adams	St. Andrews	Western	E	372	524
509	Adams	La Salle	Western	W	180	256
513	Adams	Gramercy	Arlington	W	144	232
517	Adams	Edgehill	Crenshaw	W	232	356
519	Adams	Buckingham	Hillcrest	W	202	396
521	Adams	Harcourt	La Brea	W	244	408
524	Adams	Redondo	Hauser	W	176	308
525	Adams	Hauser	Fairfax	W	204	304
528	San Vicente	Curson	Hauser	E	100	148
531	San Vicente	Cochran	Redondo	E	280	384
533	San Vicente	La Brea	Pico	E	204	336
534	San Vicente	Pico	Venice (Viney.)	E	122	192
535	San Vicente	Venice (Viney.)	Pico	W	238	428
537	San Vicente	La Brea	Hauser	W	174	332
540	San Vicente	MB53	Redondo	W	232	388
543	National	WEST END	Sawtelle	E	226	324
544	National	Sawtelle	405 SB	E	280	444
546	National	405 NB	Sepulveda	E	354	536
549	National	MB65	Westwood	E	270	440
550	National	Westwood	Over. N	E	168	252
551	National	Over. S	I- 10 South (Over.)	E	28	64
553	National	Motor	Palms	E	64	96
558	National	Bagley	Robertson	E	184	316
559	National	Robertson	I- 10 South (Rob.)	E	214	384
560	National	I- 10 South (Rob.)	Venice	E	236	404
564	National	Washington	Venice	W	100	164
565	National	Venice	I- 10 South (Rob.)	W	176	304
566	National	I- 10 South (Rob.)	Robertson	W	78	124
570	National	MB8	Manning	W	150	204
571	National	Manning	Palms	W	232	328
573	National	Motor	I- 10 South (Over.)	W	174	292
575	National	Over. N	Westwood	W	254	360
578	National	Military	Sepulveda	W	264	364
580	National	405 NB	405 SB	W	436	640
581	National	405 SB	Sawtelle	W	358	448
583	Palms	WEST END	Sawtelle	E	104	152
584	Palms	Sawtelle	Sepulveda	E	102	152
587	Palms	MB2	Overland	E	176	276
595	Palms	MB4	Overland	W	220	312
598	Palms	Kelton	Sepulveda	W	184	264
599	Palms	Sepulveda	Sawtelle	W	152	168
606	Manning	On Ramp EB	National	E	14	20
613	Hillsboro	Beverwil	Robertson	E	54	104
615	Hillsboro	Shenandoah	La Cienega	E	238	344
616	Cadillac	La Cienega	Venice	E	44	80
617	Cadillac	Venice	La Cienega	W	484	632
619	Hillsboro	Shenandoah	Robertson	W	128	200
627	Sawtelle	NORTH END	Olympic	S	76	108
629	Sawtelle	Tennessee	Pico	S	218	272
631	Sawtelle	Ivy	National	S	160	176
632	Sawtelle	National	Palms	S	74	76
633	Sawtelle	Palms	Venice	S	118	156
635	Sawtelle	SOUTH END	Venice	N	190	204

Link #	Street	From	To	Direction	:00	6:30
636	Sawtelle	Venice	Palms	N	122	176
637	Sawtelle	Palms	National	N	182	236
639	Sawtelle	Ivy	Pico	N	348	448
641	Sawtelle	Tennessee	Olympic	N	472	556
648	Sepulveda	Olympic	Pico	S	94	236
650	Sepulveda	MB38	National	S	100	220
653	Sepulveda	MB66	Palms	S	88	136
656	Sepulveda	MB1	Venice	S	118	228
659	Sepulveda	Charnock	Palms	N	246	380
662	Sepulveda	Queensland	National	N	236	484
664	Sepulveda	MB38	Pico	N	662	908
665	Sepulveda	Pico	Olympic	N	186	408
673	Westwood	Coventry	National	S	74	96
679	Overland	NORTH END	Olympic	S	52	88
680	Overland	Olympic	Pico	S	124	192
683	Overland	Coventry	Ramps WB	S	334	580
685	Overland	National	Palms	S	164	236
687	Overland	MB3	Venice	S	156	240
691	Overland	Washington	Venice	N	148	244
693	Overland	MB3	Palms	N	226	376
695	Overland	National	EB Ramps	N	650	1012
698	Overland	Ashby	Pico	N	230	472
699	Overland	Pico	Olympic	N	306	360
718	Beverwil	Olympic	Pico	S	54	44
722	Bevemil	Cashio	Pico	N	92	152
729	Robertson	Olympic	Pico	S	154	248
736	Robertson	MB35	Cadillac	S	236	432
737	Robertson	Cadillac	Cattaraugus	S	296	552
739	Robertson	MB64	Kincardine	S	322	568
740	Robertson	Kincardine	National	S	640	1108
744	Robertson	U-Turn Node	Venice	S	306	392
748	Robertson	Washington	Venice	N	84	124
749	Robertson	I - 10 South	National	N	176	204
750	Robertson	National	Kincardine	N	240	340
752	Robertson	MB64	Cattaraugus	N	492	720
760	Robertson	Cashio	Pico	N	424	592
763	La Cienega	NORTH END	Olympic	S	398	548
765	La Cienega	Whitworth	Pico	S	162	212
768	La Cienega	Airdrome	18th	S	546	772
772	La Cienega	Guthrie	Cadillac	S	590	880
773	La Cienega	Cadillac	I - 10 North	S	930	1312
776	La Cienega	I - 10 South S	Venice	S	632	740
780	La Cienega	Washington	Venice	N	116	108
783	La Cienega	I - 10 South N	I - 10 North	N	450	728
784	La Cienega	I - 10 North	Cadillac	N	604	856
788	La Cienega	Sawyer	18th	N	708	972
791	La Cienega	Cashio	Pico	N	160	380
793	La Cienega	Whitworth	Olympic	N	546	788
796	Crescent Hts.	Whitworth	Olympic	N	42	76
800	Fairfax	Whitworth	Pico	S	86	76
804	Fairfax	18th	Venice	S	432	736
805	Fairfax	Venice	I - 10 North	S	466	720
806	Fairfax	I - 10 North	Washington	S	572	772
807	Fairfax	Washington	Adams	S	740	1036
809	Fairfax	SOUTH END	Adams	N	696	1032
810	Fairfax	Adams	Washington	N	226	448
811	Fairfax	Washington	- 10 North	N	312	492
812	Fairfax	I - 10 North	Venice	N	690	1128

Link #	Street	From	To	Direction	6:00	6:30
816	Fairfax	MB50	Pico	N	386	428
818	Fairfax	Mid-block	Olympic (S.V.)	N	360	464
829	Hauser	NORTH END	Olympic	S	74	168
830	Hauser	Olympic	San Vicente	S	50	96
831	Hauser	San Vicente	Pico	S	78	116
832	Hauser	Pico	Venice	S	106	224
833	Hauser	Venice	Washington	S	106	160
835	Hauser (2)	SOUTH END	Adams	N	96	168
836	Hauser	Washington	Venice	N	110	180
837	Hauser	Venice	Pico	N	84	176
838	Hauser	Pico	San Vicente	N	58	144
839	Hauser	San Vicente	Olympic	N	130	216
847	Redondo	La Brea	San Vicente	S	534	952
859	Redondo	MB54	San Vicente	N	668	952
861	La Brea	NORTH END	Olympic	S	166	256
866	La Brea	Pico	Venice	S	496	888
870	La Brea	23rd	I - 10 North N	S	556	840
874	La Brea	I - 10 South S	Adams	S	982	1280
878	La Brea	I - 10 South S	I - 10 South N	N	550	888
883	La Brea	21st	Washington	N	646	876
884	La Brea	Washington	Venice	N	804	1108
889	La Brea	Redondo	Olympic	N	278	556
896	West	Pico	Washington	S	172	204
902	West	MB15	Washington	N	94	164
905	Crenshaw	NORTH END	Olympic	S	294	464
908	Crenshaw	Pico	Venice	S	460	696
911	Crenshaw	MB16	I - 10 North	S	762	1144
913	Crenshaw	I - 10 South	Adams	S	592	848
915	Crenshaw	SOUTH END	Adams	N	630	94a
916	Crenshaw	Adams	I - 10 South	N	584	800
919	Crenshaw	MB16	Washington	N	798	1184
920	Crenshaw	Washington	Venice	N	632	1068
921	Crenshaw	Venice	Pico	N	390	652
923	Crenshaw	Country Club	Olympic	N	288	504
931	Wilton Pl.	NORTH END	Olympic	S	304	524
933	Arlington	MB57	Pico	S	188	264
934	Arlington	Pico	Venice	S	308	528
935	Arlington	Venice	Washington	S	400	636
936	Arlington	Washington	I - 10 North	S	354	488
938	Arlington	I - 10 South	Adams	S	160	200
940	Arlington	SOUTH END	Adams	N	424	720
941	Arlington	Adams	I - 10 South	N	538	928
944	Arlington	Washington	Venice	N	314	460
945	Arlington	Venice	Pico	N	60	180
946	Arlington	Country Club	Olympic	N	78	268
948	Western	NORTH END	Olympic	S	356	524
951	Western	MB58	Pico	S	608	828
952	Western	Pico	Venice	S	580	904
954	Nestern	18th	Nashington	S	100	152
955	Western	Washington	- 10 North	S	216	264
956	Nestern	I - 10 North	- 10 South	S	470	576
957	Nestern	I - 10 South	Adams	S	154	124
959	Nestern	SOUTH END	Adams	N	684	940
960	Nestern	Adams	- 10 North	N	616	784
961	Nestern	- 10 North	- 10 South	N	778	1196
962	Nestern	- 10 South	Nashington	N	522	824
964	Nestern	18th	Jenice	N	568	652
968	Nestern	11th	Olympic	N	396	628

.ink #	Street	From	To	Direction	6:00	6:30
976	Vormandie	NORTH END	Olympic	S	210	340
978	Vormandie	11th	Pico	S	402	556
980	Normandie	15th	Venice	S	194	384
989	Normandie	I - 10 South	Washington	N	458	796
990	Normandie	Washington	Venice	N	148	304
992	Normandie	15th	Pico	N	262	452
994	Normandie	11th	Olympic	N	138	244
996	Vermont	NORTH END	Olympic	S	222	360
1000	Vermont	Pico	Venice	S	654	1032
1012	Vermont	I - 10 South	Washington	N	588	848
1014	Vermont	MB29	Venice	N	566	624
1018	Vermont	11th	Olympic	N	354	600
1024	Hoover	NORTH END	Olympic	S	378	716
1028	Hoover	12th	Pico	S	372	672
1030	Hoover	Alvarado	Venice	S	556	876
1046	Hoover	MB28	Venice	N	386	624
1048	Hoover	Alvarado	Pico	N	224	380
1052	Hoover	11th	Olympic	N	252	412
1054	Alvarado	NORTH END	Olympic	S	612	864
1057	Alvarado	12th	Pico	S	224	304
1059	Alvarado	Hoover	Pico	N	200	244
1062	Alvarado	11th	Olympic	N	274	388
1112	Vineyard	Washington	Venice (S.V.)	N	220	316
1117	Motor	Pico	Wente Mar	S	298	372
1131	Motor	MB43	Wente Mar	N	332	448
1135	RAMPS (Apple)	Fairfax	Nashington	E	910	1032
1136	RAMPS (Apple j)	Washington	Fairfax	W	406	400
1176	WB ML from start to I-10 overpass				3812	4638
1177	WB ML from I-10 overpass to connector ramp				3812	4638
1178	WB ML from connector ramp to I-10 SB on-ramp				3812	4638
1179	WB ML from I-10 SB on-ramp to I-10 NB on-ramp				7812	7364
1180	WB ML from I-10 NB on-ramp to 20th off-ramp				9066	8446
1181	WB ML between 20th off-ramp and on-ramp				8920	8190
1182	WB ML between 20th on-ramp and Vermont off-ramp				9334	8678
1183	WB ML between Vermont off-ramp and on-ramp				9076	8390
1184	WB ML between Vermont on-ramp and Normandie detector				9508	8842
1185	WB ML between Normandie detector and lane drop near Vermont				9508	8842
1186	WB ML from lane drop to Normandie off-ramp				9508	8842
1187	WB ML between Normandie off-ramp and lane add				9280	8554
1188	WB ML between Normandie lane add and lane drop				9280	8554
1189	WB ML between Normandie lane drop and on-ramp				9280	8554
1190	WB ML between Normandie on-ramp and Western off-ramp				9660	9106
1191	WB ML between Western off-ramp and lane add				9036	8350
1192	WB ML between Western lane add and lane drop				9036	8350
1193	WB ML between Western lane drop and on-ramp				9036	8350
1194	WB ML between Western on-ramp and Arlington off-ramp				9328	8700
1195	NB ML between Arlington off-ramp and on-ramp				9092	8272
1196	NB ML between Arlington on-ramp and Crenshaw off-ramp				9374	8782
1197	NB ML between Crenshaw off-ramp and detector				8810	8054
1198	NB ML between Crenshaw detector and on-ramp				8810	8054
1199	NB ML between Crenshaw on-ramp and NB La Brea off-ramp				9254	8680
1200	NB ML between La Brea NB off-ramp and detector				8838	8234
1201	NB ML between La Brea detector and NB on-ramp				8838	8234
1202	NB ML between La Brea NB on-ramp and SB off-ramp				9010	8614
1203	NB ML between La Brea SB off-ramp and SB on-ramp				8270	8044
1204	NB ML between La Brea SB on-ramp and Washington off-ramp				8550	8424
1205	NB ML between Washington off-ramp and Venice off-ramp				7314	7358
1206	NB ML between Venice off-ramp and Fairfax detector				6394	6468

Link #	Street	From	To	Direction	6:00	6:30
1207	WB ML between Fairfax detector and on-ramp				6394	6468
1208	WB ML between Fairfax on-ramp and La Cienaga on-ramp				6790	7064
1209	WB ML between La Cienaga on-ramp and Robertson off-ramp				7076	7620
1210	WB ML between Robertson off-ramp and on-ramp				6346	6982
1211	WB ML between Robertson on-ramp and National off-ramp				6678	7652
1212	WB ML between National off-ramp and Manning detector				6176	7168
1213	WB ML between Manning detector and Overland off-ramp				6176	7168
1214	WB ML between Overland off-ramp and Overland on-ramp				5528	6484
1215	WB ML between Overland on-ramp and SB I-405 off-ramp				5968	7298
1216	WB ML from I-405 SB off-ramp to I-405 NB off-ramp				4040	5682
1217	WB ML from I-405 NB off-ramp to I-405 overpass				1242	2726
1218	WB ML from I-405 overpass to end				1242	2726
1219	EB ML from start to I-405 overpass				2612	3004
1220	EB ML from I-405 overpass to I-405 on-ramp (NB and SB)				2612	3004
1221	EB ML from I-405 on-ramps to National off-ramp				5434	6158
1222	EB ML between National off-ramp and Overland on-ramp				5034	5608
1223	EB ML between Overland on-ramp and Motor detector				5700	6460
1224	EB ML between Motor detector and Manning on-ramp				5700	6460
1225	EB ML between Manning on-ramp and Robertson off-ramp				5892	6950
1226	EB ML between Robertson off-ramp and National detector				5560	6412
1227	EB ML between National detector and on-ramp				5560	6412
1228	EB ML between National on-ramp and SB La Cienaga off-ramp				5820	6880
1229	EB ML between La Cienaga SB off-ramp and SB on-ramp				5758	6778
1230	EB ML between La Cienaga SB on-ramp and NB off-ramp				6146	7318
1231	EB ML between NB La Cienaga off-ramp and Fairfax off-ramp				5962	7016
1232	EB ML between Fairfax off-ramp and detector				5788	6742
1233	EB ML between Fairfax detector and Venice on-ramp				5788	6742
1234	EB ML between Venice on-ramp and Washington on-ramp				6046	7032
1235	EB ML between Washington on-ramp and SB La Brea off-ramp				6990	7990
1236	EB ML between La Brea SB off-ramp and SB on-ramp				6918	7894
1237	EB ML between La Brea SB on-ramp and NB off-ramp				7360	8424
1238	EB ML between La Brea NB off-ramp and detector				7176	8170
1239	EB ML between La Brea detector and NB on-ramp				7176	8170
1240	EB ML between NB La Brea on-ramp and Crenshaw off-ramp				7766	9108
1241	EB ML between Crenshaw off-ramp and detector				7500	8824
1242	EB ML between Crenshaw detector and on-ramp				7500	8824
1243	EB ML between Crenshaw on-ramp and Arlington off-ramp				8270	9854
1244	EB ML between Arlington off-ramp and on-ramp				8176	9668
1245	EB ML between Arlington on-ramp and Western off-ramp				8858	10498
1246	EB ML between Western off-ramp and ramp from C/D road (lane add)				8650	10302
1247	EB ML between ramp from C/D road (lane add) and Western on-ramp				8650	10302
1248	EB ML between Western on-ramp and Normandie off-ramp				9170	11040
1249	EB ML between Normandie off-ramp and ramp from C/D road (lane drop)				9018	10798
1250	EB ML between ramp from C/D road (lane drop) and Normandie on-ramp				9018	10798
1251	EB ML between Normandie on-ramp and ramp from C/D road near Vermont				9526	11310
1252	EB ML between ramp from C/D road (lane add) and Vermont off-ramp				9526	11310
1253	EB ML between Vermont off-ramp and detector				9340	10974
1254	EB ML between Vermont detector and on-ramp				9340	10974
1255	EB ML between Vermont on-ramp and Hoover off-ramp				9800	11654
1256	EB ML between Hoover off-ramp and on-ramp				9506	11168
1257	EB ML between Hoover on-ramp and SB I-I 10 off-ramp				9922	11598
1258	EB ML between SB I-I 10 off-ramp and NB I-I 10 off-ramp				8592	10228
1259	EB ML from NB I-I 10 off-ramp to I-I 10 overpass				5626	7208
1260	EB ML from I-I 10 overpass to end				5626	7208
1262	Off-ramp (WB) to 20th				152	262
1264	Off-ramp (WB) to Vermont				264	299
1266	Off-ramp (WB) to Normandie				234	295
1268	Off-ramp (WB) to Western				637	777

nk #	Street	From	To	Direction	:00	:30
1270	Off-ramp (WB) to Arlington				241	442
1272	Off-ramp (WB) to Crenshaw				572	749
1274	X-ramp (WB) to NB La Brea				420	460
1275	Off-ramp (WB) to SB La Brea				748	589
1277	Off-ramp (WB) to Washington				1249	1101
1279	Off-ramp (WB) to Venice				928	916
1281	Off-ramp (WB) to Robertson				733	657
1283	Off-ramp (WB) to National				504	500
1286	Off-ramp (WB) to Overland/National				651	705
1288	Connector (WB) to SB I-405				1924	1665
1289	Connector (WB) to NB I-405				2794	3045
1291	Off-ramp (EB) to National				388	540
1292	Off-ramp (EB) to Venice (SPLIT - now from mid-ramp)				323	528
1293	Off-ramp (EB) to SB La Cienaga(SPLIT - now from mid-ramp)				59	100
1294	Off-ramp (EB) to NB La Cienaga (SPLIT - now from mid-ramp)				179	297
1296	Off-ramp (EB) to Fairfax				170	270
1298	Off-ramp (EB) to SB La Brea				69	95
1299	Off-ramp (EB) to NB La Brea (SPLIT - now from mid-ramp)				180	250
1301	Off-ramp (EB) to Crenshaw				258	280
1303	Off-ramp (EB) to Arlington				89	183
1305	Off-ramp (EB) to Western				203	191
1307	Off-ramp (EB) to Normandie				148	238
1309	Off-ramp (EB) to Vermont				180	330
1311	Off-ramp (EB) to Hoover				285	478
1312	Connector (EB) to SB I-10				1295	1346
1313	Connector (EB) to NB I-10				2888	2968
1314	Connector (WB) I-10 SB to ramp meter				2835	2725
1316	Connector (WB) I-10 NB to ramp meter				1255	1083
1318	On-ramp (WB) 20th to ramp meter				414	49c
1320	On-ramp (WB) Vermont to ramp meter				433	451
1322	On-ramp (WB) Normandie to ramp meter				38C	55C
1324	On-ramp (WB) Western to ramp meter				29C	35c
1326	On-ramp (WB) Arlington to ramp meter				28C	51C
1328	On-ramp (WB) Crenshaw to ramp meter				443	626
1330	On-ramp (WB) NB La Brea to ramp meter				17c	38C
1332	On-ramp (WB) SB La Brea to ramp meter				28C	382
1334	On-ramp (WB) Fairfax to ramp meter				397	596
1336	On-ramp (WB) La Cienaga to ramp meter				286	55i
1338	On-ramp (WB) Robertson to ramp meter				33:	67C
1340	On-ramp (WB) Overland/National to ramp meter				44C	815
1342	Connector (EB) from NB I-405 to ramp meter				155:	1474
1344	Connector (EB) from SB I-405 to ramp meter				1265	1679
1346	Connector (EB) from NB,SB I-405 merge to ML				3193	3356
1347	On-ramp (EB) Overland to ramp meter				665	852
1349	On-ramp (EB) Manning to ramp meter				19C	49C
1351	On-ramp (EB) National to ramp meter				26C	47C
1353	On-ramp (EB) SB La Cienaga to ramp meter				39C	54C
1355	On-ramp (EB) Venice/Exposition to ramp meter				256	289
1357	On-ramp (EB) Washington to ramp meter				944	956
1359	On-ramp (EB) SB La Brea to ramp meter				44C	53C
1361	On-ramp (EB) NB La Brea to ramp meter				59C	94C
1363	On-ramp (EB) Crenshaw to ramp meter				77C	103C
1365	On-ramp (EB) Arlington to ramp meter				68:	829
1367	On-ramp (EB) Western to ramp meter				52C	74C
1369	On-ramp (EB) Normandie to ramp meter				50:	51;
1371	On-ramp (EB) Vermont to ramp meter				46C	68C
1373	On-ramp (EB) Hoover to ramp meter				41:	431
1386	405 NB - off ramp to National				506	588

Link #	Street	From	(To	Direction	6:00	6:30
1388	405 SB - on-ramp from	National			220	352
1396	National Pl.	Overland S.	Westwood	W	62	96
1429	Sepulveda	Venice	Washington	N	304	424
1431	Hauser	Adams	Washington	N	78	140

Link #	7:00	7:30	8:00	8:30	9:00	9:30	10:00	10:30	11:00	11:30
1	758	1132	1748	1940	1500	1540	1406	1492	1706	1584
3	644	936	1188	1188	1210	1052	1064	1228	1088	916
8	752	1148	1608	1764	1628	1412	1344	1396	1494	1268
25	720	1392	1400	1432	1208	1048	964	124	304	256
27	676	1248	1554	1636	1246	1112	988	576	690	760
29	594	1112	1304	1400	1032	920	792	480	574	668
34	720	1308	1468	1552	1206	1076	910	548	744	784
37	500	552	764	884	668	620	520	532	614	612
39	928	1128	1224	1292	1018	1120	822	716	730	508
46	1030	1472	1436	1324	1048	1088	846	860	870	760
49	838	1176	1404	1440	1020	872	622	536	652	620
53	1084	1440	1640	1500	1338	1280	1274	1160	1176	1160
57	990	1132	1280	1204	1252	1184	1226	936	1088	1048
60	1240	1636	1784	1860	1438	1424	1208	1104	1184	1164
61	1108	1476	1642	1604	1344	1328	1154	1076	1106	1216
73	742	928	1180	1848	922	1128	1086	932	1168	1004
74	772	856	906	900	882	776	1002	1024	1066	1160
81	1026	1168	1138	1024	940	944	952	984	938	976
83	1096	1308	1096	984	1072	952	1068	1112	1062	976
88	1074	1492	1388	1192	956	852	752	788	722	768
95	312	404	466	432	170	292	308	236	286	0
100	1010	1404	1510	1552	1222	1076	890	900	838	916
107	798	1332	1482	1464	1236	1152	924	880	778	944
126	1074	1792	2078	2228	1978	1652	1504	1484	1620	1788
131	640	1136	1442	1580	1434	1204	1080	1100	1110	1276
133	876	1520	1978	2028	2100	1748	1612	1712	1800	1768
135	768	1200	1478	1536	1328	1300	878	924	1180	1136
136	1060	1280	1458	1420	1266	1308	974	1268	1424	1132
141	472	704	1056	1108	1036	948	772	828	986	888
143	738	924	1512	1376	1524	1304	1126	1240	1264	1304
144	814	1060	1854	1704	1678	1416	1224	1352	1336	1444
146	438	592	932	884	914	716	656	720	688	744
152	458	596	858	764	502	508	318	456	330	412
158	470	804	972	1000	1040	1000	1004	1004	1114	1160
162	1034	1224	1642	1568	1856	1624	1586	1472	1586	1592
170	522	788	930	880	840	916	776	816	874	848
175	462	792	746	748	646	608	550	572	648	652
184	1068	1484	1504	1340	1074	1064	782	432	602	944
186	478	752	758	732	632	552	552	304	312	608
189	504	692	774	772	578	560	596	540	590	648
195	500	696	786	664	522	408	594	536	612	624
196	638	668	610	560	514	492	598	568	666	608
212	554	632	628	824	656	544	384	192	350	408
213	580	684	652	796	650	556	632	452	632	600
219	686	732	610	604	496	592	536	552	620	548
222	664	748	588	552	440	472	468	432	502	516
224	854	1172	956	956	638	660	618	236	456	520
228	922	1124	938	900	622	608	562	648	528	456
233	848	1252	1234	1072	824	868	728	744	806	732
238	734	1076	1108	1024	784	800	596	736	650	680
243	1246	1456	1540	1608	1190	1172	1108	1224	1274	1284
246	1018	1048	1284	1284	1288	1372	1270	1312	1452	1412
250	764	1112	1240	1280	1190	972	960	1272	1110	1144
256	856	1244	1508	1336	1498	1308	1092	1124	1186	1148
262	692	952	1180	1332	1302	1232	1038	1232	1326	1276
264	836	1224	1534	1464	1406	1248	1194	1348	1452	1356
265	790	1124	1340	1432	1364	1132	1108	1260	1408	1284
267	650	968	1056	1140	1098	980	1064	1152	1282	1144

Link #	7:00	7:30	8:00	8:30	9:00	9:30	10:00	10:30	11:00	11:30
272)	660	960	1054	1004	1170	916	1028	1000	1234	1188
273	618	940	988	1028	1110	812	1126	1060	1222	1368
275	1448	1864	2142	2216	1606	1628	1132	1256	1254	1156
276	1038	1504	1864	1908	1422	1324	1022	1108	1111	1032
278	1250	1800	2238	2252	1854	1756	1386	1412	1514	1368
281	798	1172	1460	1496	1170	1036	840	864	888	976
285	1628	2256	2860	3012	2186	1860	1636	1608	1606	1816
286	1486	2144	2500	2732	2062	1680	1394	1460	1408	1536
289	860	1056	1154	1056	932	880	888	776	892	924
291	538	844	982	1100	824	676	470	536	530	532
293	762	1436	1596	1728	1248	968	694	732	824	876
296	876	1456	1716	1684	1268	1024	676	716	772	796
300	640	1196	1258	1256	918	748	482	512	576	672
303	494	944	1022	968	678	516	322	328	374	432
305	1038	1804	1938	1680	1294	1024	768	720	828	864
308	698	1268	1474	1236	948	776	554	512	596	648
312	648	1120	1314	1108	850	720	536	436	508	616
314	1108	1428	1426	1328	1172	952	994	992	990	992
315	672	1068	1264	1016	808	632	454	428	540	536
317	3760	4076	810	552	646	548	1142	408	470	416
319	672	960	988	824	730	484	418	376	458	432
325	330	496	408	380	332	360	404	380	474	476
327	352	568	478	448	340	332	408	352	424	536
329	534	816	614	644	436	404	386	524	496	540
330	922	1176	910	744	650	648	672	676	710	684
332	770	1004	948	788	554	484	412	528	496	444
339	1332	1960	1500	1176	916	876	678	680	644	636
341	812	1260	1102	908	692	588	532	480	462	472
344	720	1292	1048	876	578	576	450	456	398	380
348	1192	1864	1636	1456	1002	932	750	756	766	680
351	1336	1972	1732	1640	1164	1024	830	760	830	756
353	1148	2044	1822	1692	1176	1036	900	840	874	904
355	1246	1864	1982	2060	1616	1384	1074	1140	1136	1192
358	1160	1860	1780	1688	1444	1084	1056	992	994	1164
359	1290	2016	1946	1892	1612	1328	1228	1100	1188	1308
363	994	1740	1690	1636	1332	1016	1124	1040	1166	1292
366	1086	1716	1688	1740	1400	1100	1264	1124	1226	1444
368	1068	1668	1634	1508	1144	1028	1106	1092	1176	1224
369	976	1704	1710	1624	1214	1044	1094	1040	1138	1332
386	234	236	288	288	216	188	238	296	334	280
388	282	508	654	660	454	308	328	284	332	268
390	376	628	852	736	590	420	468	396	416	416
393	1146	1468	1368	1356	1266	1200	1184	1160	1440	1636
397	542	928	830	808	616	500	494	572	530	520
398	1006	1344	1304	1212	982	904	846	876	812	820
403	662	972	1034	940	690	660	580	608	696	636
406	1120	1288	1094	924	694	980	996	932	1074	1092
409	940	1220	1006	772	714	680	688	768	766	808
411	374	548	566	468	334	324	284	248	272	252
429	1234	1876	1538	1480	1156	1072	1044	1104	1176	1252
432	1190	1900	1574	1300	1048	1084	1140	1056	1246	1252
435	736	1372	936	776	624	600	564	648	576	700
440	1008	1352	1374	1140	882	824	720	664	714	784
441	602	824	914	768	546	460	492	456	462	556
445	1480	1736	1812	1620	1388	1204	1270	1228	1308	1320
450	896	1328	1170	892	730	656	598	580	590	664
452	1780	1856	1668	1768	1584	1436	1206	1180	1322	1276
469	128	300	278	268	220	144	170	192	176	200

Link #	7:00	7:30	3:00	3:30	9:00	9:30	10:00	10:30	11:00	11:30
470	246	348	464	424	380	268	280	304	342	284
473	816	940	818	844	714	640	590	632	650	692
475	454	748	684	692	506	400	420	460	444	428
477	780	976	998	944	664	620	582	596	682	496
481	618	980	898	860	538	544	446	392	442	404
485	1004	1472	1358	1180	906	892	952	968	940	1044
509	656	1012	724	520	426	416	434	484	486	484
513	580	1084	830	660	432	368	418	396	422	452
517	698	1200	1032	792	466	340	458	404	342	540
519	748	1248	1158	928	610	476	468	536	462	508
521	760	1320	1126	908	570	500	486	496	436	516
524	608	1000	792	628	400	352	356	352	392	340
525	636	1024	732	564	346	308	282	344	310	292
528	346	548	618	632	504	476	384	412	406	388
531	616	964	924	692	1164	1292	1570	476	2086	2216
533	814	1340	1362	1224	978	904	740	708	714	708
534	496	832	854	692	560	456	460	404	496	400
535	638	968	1006	868	652	596	412	372	362	344
537	498	808	898	816	544	448	316	304	304	304
540	578	932	1030	1036	662	612	434	384	386	400
543	596	808	1000	1056	814	748	626	548	664	704
544	768	1072	1350	1292	1082	1040	982	920	1078	1188
546	884	1132	1428	1432	1186	1052	908	844	1018	1072
549	780	916	1278	1396	1132	1024	822	852	910	1160
550	524	504	684	748	570	580	508	560	494	624
551	126	204	184	284	234	200	204	244	286	300
553	164	288	316	312	230	2 4 8	220	184	232	224
558	492	656	668	616	552	544	438	384	470	504
559	490	676	790	628	666	728	654	604	676	760
560	746	1236	1238	1204	736	5 6 8	494	492	508	444
564	356	632	628	460	400	424	328	248	338	400
565	692	1136	1200	1168	740	584	520	508	488	516
566	274	448	552	516	294	224	198	188	190	208
570	558	1068	1118	828	740	648	654	556	550	624
571	680	1148	1078	916	844	816	814	812	780	840
573	504	908	850	864	632	540	470	548	488	500
575	520	468	668	476	508	492	484	492	578	564
578	588	744	916	856	788	684	694	676	804	808
580	1082	1172	1434	1400	1190	1112	1056	1060	1224	1240
581	792	920	1102	1088	946	856	840	852	950	960
583	394	876	826	924	590	504	424	348	436	388
584	412	792	886	912	598	596	484	408	466	452
587	680	1144	1298	1316	944	856	730	576	738	684
595	686	1252	1082	892	700	644	616	588	606	712
598	572	980	1012	980	680	576	554	512	566	596
599	442	768	888	800	560	508	494	476	490	528
606	54	128	114	140	118	92	78	108	80	108
613	164	204	262	340	266	240	196	208	182	140
615	486	556	648	636	418	448	298	300	344	344
616	108	276	170	124	184	252	202	168	198	252
617	590	628	874	904	840	8 8 4	910	900	836	1044
619	400	584	650	592	474	380	318	208	258	220
627	186	304	398	716	836	684	702	636	652	736
629	384	580	706	624	776	840	934	880	1158	1140
631	314	560	668	632	598	620	696	712	814	796
632	218	360	444	432	436	372	402	388	450	488
633	356	464	530	528	478	364	376	328	376	440
635	462	668	792	836	612	576	518	580	552	560

-ink #	7:00	7:30	3:00	3:30	9:00	3:30	10:00	10:30	11:00	11:30
636	482	904	964	1096	670	4 8 0	410	352	360	408
637	598	1076	1132	1212	826	632	494	504	436	484
639	842	1264	1388	1584	1196	1024	838	780	838	932
641	798	984	1244	1232	1274	1084	1010	892	890	968
648	386	500	512	512	652	560	824	888	958	908
650	342	456	492	300	356	332	366	412	874	876
653	308	448	520	516	572	524	586	716	790	728
656	368	572	668	648	576	508	664	636	728	652
659	828	1236	1610	1652	1230	856	750	864	904	976
662	884	1368	1714	1884	1510	1192	1130	1200	1094	1212
664	1338	1672	1536	1444	1216	924	842	928	1030	1372
665	748	1192	1426	1472	1398	1052	852	916	894	1044
673	238	368	440	488	422	480	382	440	492	596
679	250	348	492	472	372	376	278	296	316	340
680	380	584	618	636	550	544	552	648	618	656
683	912	1220	1408	1512	1252	1044	1020	1252	1426	1492
685	390	692	762	776	742	584	658	672	786	968
687	382	592	650	660	626	652	600	584	624	744
691	424	740	728	824	632	660	478	628	626	680
693	604	968	966	952	800	752	590	684	688	732
695	1656	2104	2320	232a	2048	1764	1518	1576	1576	1732
698	692	744	1154	1424	1492	1184	1078	1076	1138	1312
699	598	820	580	752	532	820	726	748	768	848
718	132	188	222	268	230	276	240	300	280	336
722	416	644	960	984	846	668	456	372	420	416
729	458	708	910	980	898	848	838	756	910	972
736	632	1132	1196	1216	1118	1052	1064	1072	1246	1248
737	860	1344	1300	1388	1250	1244	1206	1248	1364	1456
739	934	1324	1444	1400	1336	1260	1258	1276	1360	1436
740	1348	1920	1820	1792	1864	1788	1828	1896	1922	1904
744	572	812	794	804	986	828	874	852	898	764
748	240	444	500	472	338	312	286	304	314	356
749	426	592	768	880	862	888	806	864	840	808
750	758	1092	1254	1492	1288	1316	1138	1036	1184	1084
752	940	960	1476	1484	1564	1748	1568	1444	1540	1512
760	780	852	1378	1416	1348	1424	1382	1220	1314	1332
763	890	976	1124	1060	1168	1068	1068	788	790	844
765	510	824	1080	1116	986	968	976	1048	1030	1088
768	1224	1272	1450	1424	1602	1720	1586	1404	1504	1476
772	1346	1496	1598	1548	1576	1848	1560	1364	1636	1976
773	2178	2460	2676	2580	2330	2268	2180	2020	2156	2300
776	1514	2332	2326	1996	1858	1572	1436	1352	1398	1432
780	284	472	616	612	490	544	368	436	398	440
783	1316	2012	2360	2052	1660	1736	1332	1304	1270	1356
784	1558	2368	2894	2664	2268	2416	1846	1824	1796	1740
788	1170	1368	1664	1708	1832	1924	1702	1552	1674	1672
791	560	904	926	724	838	852	736	660	668	852
793	872	1040	1208	1084	1340	1144	1290	1048	1142	1224
796	184	364	628	636	614	560	448	456	418	532
800	244	392	532	444	462	440	376	416	438	476
804	1132	1468	1374	1360	1046	976	930	1056	1090	1056
805	944	1260	1134	1116	860	784	732	940	892	912
806	990	1044	978	892	798	724	660	604	668	688
807	1170	1348	1178	1168	970	944	772	804	866	844
809	1342	1432	1382	1504	1342	1204	912	916	1096	1064
810	628	988	1026	964	810	780	598	652	694	628
811	776	964	948	1012	888	796	586	576	590	680
812	1296	1280	826	444	2130	2520	1934	2056	3452	4092

Link #	7:00	7:30	8:00	8:30	9:00	9:30	10:00	10:30	11:00	11:30
816	628	792	980	940	1006	1044	788	848	860	1000
818	634	812	912	956	1052	1004	850	908	902	940
829	330	476	534	496	356	296	340	372	438	500
830	246	452	446	472	360	296	274	312	308	312
831	304	464	502	496	386	344	328	360	312	360
832	384	552	570	520	410	356	372	360	342	428
833	292	404	414	352	294	244	258	292	252	304
835	302	428	352	308	198	148	156	204	220	136
836	330	460	496	496	378	264	264	220	280	296
837	340	448	528	644	494	360	306	308	330	344
838	262	316	40a	496	402	336	272	244	242	276
839	468	612	712	684	656	588	482	480	516	508
847	1214	1752	1588	1212	1532	1416	1436	936	1362	1504
859	1268	1624	1552	1672	1908	1800	1600	1656	1712	1704
861	406	596	622	576	644	228	a	a	a	8
866	1210	1492	1430	1492	1690	1632	1586	1496	1628	1728
870	1096	1388	1352	1324	1160	1008	1122	1024	1236	1204
874	1618	1948	1814	1732	1598	1756	1642	1552	1622	1696
878	1428	1640	1812	1848	1658	1384	1254	1360	1226	1392
883	1268	1688	1630	1656	1542	1348	1328	1436	1366	1264
884	1714	2008	2030	2084	1972	1800	1708	1744	1752	1716
889	566	884	1012	1032	1736	1484	1434	1452	1514	1592
896	430	512	554	404	272	328	334	352	316	344
902	416	664	47a	460	292	300	240	192	226	268
905	832	1248	1530	1492	1140	1040	882	672	810	724
908	1030	1248	966	684	740	668	650	776	604	972
911	1526	1880	1096	588	1564	1524	1552	1616	1566	2152
913	1148	1328	1598	1712	1424	1448	1460	1328	1596	1512
915	1382	1548	1540	1512	1214	1536	1300	1312	1376	1356
916	1166	1132	1194	1192	984	1116	984	996	1014	1076
919	1648	2220	2166	2228	1666	1664	1504	1556	1614	1528
920	1468	1880	1980	1872	1552	1488	1270	1236	1276	1216
921	1042	1368	1354	1276	1016	1004	804	804	840	776
923	974	1384	1082	976	828	604	730	828	780	732
931	910	1424	1782	1716	1378	1276	1160	992	1074	948
933	486	716	818	852	588	576	578	436	610	584
934	828	1124	874	884	646	628	588	632	562	592
935	928	1292	1000	1000	792	696	628	700	618	632
936	580	496	572	656	510	500	462	504	498	508
938	454	604	738	612	488	428	458	524	512	556
940	988	1076	906	972	676	580	542	584	470	436
941	1194	1372	1102	1116	800	720	648	616	548	584
944	806	988	1070	960	1022	964	728	732	684	752
945	316	412	320	316	194	240	166	300	608	628
946	264	332	568	548	964	1320	570	492	430	496
948	1006	1328	1642	1732	1422	1084	1186	1048	1056	1060
951	1136	1332	1044	952	966	888	1042	932	990	968
952	1202	1316	1070	904	944	816	984	864	938	968
954	176	240	222	248	168	192	140	184	186	176
955	546	764	576	464	376	372	374	420	558	508
956	924	1084	986	856	764	836	750	840	848	836
957	312	456	360	340	240	272	274	264	256	300
959	1456	1800	1596	1456	1072	1028	964	1004	1028	1052
960	1158	1380	1166	980	802	796	778	844	824	824
961	1410	1528	1342	1168	1188	1080	1296	1176	1454	1408
962	1182	1408	1514	1560	1284	1188	1016	1072	1014	1276
964	974	1060	1202	1088	1008	1040	1056	876	1112	996
968	996	1384	1036	1012	932	888	1040	992	1070	1064

ink #	7:00	7:30	8:00	8:30	9:00	9:30	10:00	10:30	11:00	11:30
976	534	644	878	896	740	664	580	536	592	544
978	752	864	780	656	680	572	612	580	604	624
980	668	1232	1344	1092	868	704	496	484	548	612
989	1300	1868	1830	1640	1312	1140	994	996	1050	968
990	542	844	712	668	428	384	320	464	472	592
992	602	740	1112	1060	898	760	692	676	718	628
994	460	576	520	588	518	480	536	596	498	584
996	570	736	832	868	682	712	574	480	534	524
1000	1356	1480	1080	1196	978	1124	1144	1080	1242	1224
1012	1320	1616	1540	1572	1404	1520	1288	1312	1384	1420
1014	1096	1204	1308	1308	1262	1244	1250	1028	1150	1116
1018	1048	1044	1012	976	1094	936	1046	1144	1162	1228
1024	1050	1324	1066	928	868	792	826	836	840	948
1028	974	1156	922	924	754	656	752	816	800	844
1030	1454	1620	1138	1344	1068	968	1050	1156	1142	1196
1046	1098	1388	1314	1344	1172	1184	1138	920	1132	1076
1048	716	944	944	912	798	784	752	564	768	772
1052	746	1020	1044	1000	830	776	698	592	696	664
1054	1290	1304	1362	1192	1226	1216	1366	1148	1348	1224
1057	548	780	856	852	648	416	590	564	698	816
1059	470	544	614	584	606	564	502	404	504	432
1062	616	616	74a	844	800	816	700	616	786	760
1112	512	596	694	692	574	540	508	448	550	540
1117	642	900	1158	1136	1170	992	800	700	704	560
1131	728	1072	1242	1228	1222	1152	1016	928	930	776
1135	998	680	638	640	858	828	818	632	858	876
1136	504	648	596	392	478	520	438	508	488	436
1176	5268	5612	6294	6288	6408	6332	5850	5642	5752	6116
1177	5268	5612	6294	6288	6408	6332	5850	5642	5752	6116
1178	5268	5612	6294	6288	6408	6332	5850	5642	5752	6116
1179	7874	8124	8698	8706	8974	8982	8584	8406	8478	8828
1180	8740	8908	9486	9536	9920	10032	9780	9632	9610	9940
1181	8332	8438	9076	9138	9478	9582	9336	9194	9186	9518
1182	8928	9052	9600	9630	9942	10064	9898	9790	9792	10134
1183	8590	8660	9140	9154	9454	9576	9398	9298	9334	9688
1184	9072	9136	9570	9568	9876	10006	9838	9764	9848	10230
1185	9072	9136	9570	9568	9876	10006	9838	9764	9848	10230
1186	9072	9136	9570	9568	9876	10006	9838	9764	9848	10230
1187	8692	8714	9184	9180	9424	9522	9336	9226	9240	9602
1188	8692	8714	9184	9180	9424	9522	9336	9226	9240	9602
1189	8692	8714	9184	9180	9424	9522	9336	9226	9240	9602
1190	9252	9144	9574	9560	9754	9902	9666	9636	9590	10040
1191	8278	8052	8486	8478	8578	8748	8606	8586	8458	8896
1192	8278	8052	8486	8478	8578	8748	8606	8586	8458	8896
1193	8278	8052	8486	8478	8578	8748	8606	8586	8458	8896
1194	8638	8352	8708	8746	8820	9058	8946	8976	8778	9266
1195	7948	7488	7814	7756	7460	7808	8276	8464	8174	8662
1196	8478	8098	8254	8156	7840	8146	8546	8784	8512	9032
1197	7528	6930	6852	6696	6266	6728	7576	7970	7666	8196
1198	7528	6930	6852	6696	6266	6728	7576	7970	7666	8196
1199	8386	7868	7712	7496	6956	7342	8118	8482	8194	8714
1200	8066	7540	7326	7130	6458	6788	7612	7886	7600	8176
1201	8066	7540	7326	7130	6458	6788	7612	7886	7600	8176
1202	8454	7930	7666	7460	6728	6998	7842	8098	7770	8306
1203	8062	7598	7328	7050	6026	6236	7190	7492	7216	7752
1204	8564	8160	7900	7596	6476	6646	7582	7880	7638	8180
1205	7606	7286	7128	6834	5538	5652	6546	6844	6644	7208
1206	6698	6330	6080	5794	4484	4606	5488	5794	5590	6196

ink #	7:00	7:30	8:00	8:30	9:00	9:30	10:00	10:30	11:00	11:30
1207	6698	6330	6080	5794	4484	4606	5488	5794	5590	6196
1208	7552	7262	6908	6552	5146	5176	5942	6218	6056	6692
1209	8438	8306	7928	7584	618E	6162	6762	7002	6922	7598
1210	7874	7642	7338	6938	5332	5252	5912	6148	6014	6692
1211	8894	8934	8906	8272	638C	6186	6790	7022	6970	7670
1212	8400	8400	8290	7642	5734	5538	6140	6372	6298	7016
1213	8382	8356	8302	7642	5734	5538	6140	6372	6298	7016
1214	7600	7514	7404	6772	4812	4648	5338	5586	5454	6170
1215	8864	8952	8908	8490	5954	5672	6328	6556	6444	7182
1216	7484	7700	7700	7258	4292	3802	4258	4402	4326	5070
1217	3998	4162	4388	4264	1686	1348	1720	1882	1804	2604
1218	3998	4162	4388	4264	1686	1348	1720	1882	1804	2604
1219	4576	5308	5926	5734	442a	3600	3624	3932	4328	4496
122c	4576	5308	5926	5734	442C	3600	3624	3932	4328	4496
1221	8154	9022	9236	9326	835C	7596	7596	7862	8106	8226
1222	7426	8150	8200	8200	7186	6494	6706	6914	6802	6764
1223	8520	9330	8970	9298	8156	7426	7656	7820	7584	7550
1224	8520	9330	8970	9298	8156	7426	7656	7820	7584	7550
1225	9088	9852	9438	9788	8686	7898	8116	8218	7982	7980
1226	8288	7792	8380	8708	7812	7082	7204	7284	7062	7072
1227	8288	7792	8380	8708	7812	7082	7204	7284	7062	7072
1228	8728	8182	8760	9098	8352	7710	7892	8084	7792	7742
122s	8570	8002	8592	8920	8162	7518	7696	7874	7548	7492
123C	9228	8562	9194	9538	8742	7918	8294	8466	7968	8172
1231	8792	8004	8490	8768	8040	7246	7600	7782	7330	7534
1232	8452	7538	8062	8162	8312	6838	7034	7298	6768	7044
1233	8452	7538	8062	8162	8312	6838	7034	7298	6768	7044
1234	8256	7888	8448	8560	8704	7214	7364	7636	7186	7484
1235	9236	8854	9340	9454	9672	8204	8344	8632	8218	8546
1236	9096	8712	9234	9340	9514	8032	8162	8442	8030	8352
1237	9534	9192	9634	9790	9934	8522	8674	8960	8600	8892
1238	9162	8700	9430	9410	9480	7962	7812	8398	7996	8344
1239	9162	8700	9430	9410	9480	7962	7812	8398	7996	8344
1240	10200	9542	10162	10200	10200	9728	8490	8848	8656	8914
1241	9896	9226	9828	9908	9744	9250	7994	8350	8184	8442
1242	9602	9226	9828	9908	9744	9250	7994	8350	8184	8442
1243	10704	10288	10718	10738	10796	10282	8834	9150	8976	9322
1244	10400	9942	10400	10400	10400	9900	8560	8888	8674	9016
1245	10480	10996	11332	11170	11000	10392	8972	9270	9096	9442
1246	10310	10828	11168	10948	10604	9934	8576	8862	8670	9028
1247	10310	10828	11168	10948	10604	9934	8576	8862	8670	9028
1248	11090	11540	12002	11426	11094	10494	9076	9412	9270	9686
1249	10774	11120	11356	10778	10734	10212	8766	9098	8970	9394
1250	10774	11120	11356	10778	10734	io212	8766	9098	8970	9394
1251	11298	11624	11800	11512	11174	10638	9146	9482	9412	9852
1252	11298	11624	11834	11512	11174	10638	9146	9482	9412	9852
1253	10932	11234	11412	10978	10454	10104	8680	8990	8908	9374
1254	10932	11234	11412	10978	10454	10104	8680	8990	8908	9374
1255	11832	11954	12054	11476	10904	10644	9250	9550	9488	9924
1256	11152	11150	11160	11030	9876	9678	8526	8894	8854	9304
1257	11600	11600	11600	11450	10242	10030	8890	9282	9302	9782
1258	10250	10290	10484	10314	9060	8778	7590	7876	7824	8286
1259	7358	7384	7674	7230	5600	5304	4692	5066	5052	5502
1260	7358	7384	7674	7230	5600	5304	4692	5066	5052	5502
1262	396	447	392	402	455	469	449	441	429	431
1264	329	372	441	482	502	510	504	494	465	453
1266	370	400	368	392	465	504	508	542	614	640
1268	944	1034	1042	1095	1209	1205	1070	1056	1143	1167

Link #	7:00	7:30	8:00	8:30	9:00	9:30	10:00	10:30	11:00	11:30
1270	669	821	854	1002	1398	1305	675	516	611	617
1272	920	1107	134c	1476	1617	148C	978	817	854	852
1274	310	310	37c	370	510	58C	51c	60C	60C	550
1275	382	315	323	417	723	794	65E	61C	561	563
1277	928	827	735	769	964	103E	1044	1042	100E	990
1279	882	908	1002	1052	1083	1091	106E	105E	106E	1032
1281	548	526	564	653	880	94E	858	86C	92C	924
1283	477	506	58E	636	664	677	65E	65E	677	668
1286	761	807	83E	878	946	93c	809	791	854	862
1288	1348	1203	1115	1247	1707	195c	2089	2171	2141	2155
1289	3404	3428	310E	2910	2677	2561	2561	2536	2546	2513
1291	683	842	1052	1159	1209	1161	898	944	1297	1460
1292	752	926	1091	1101	900	85E	92C	925	91E	906
1293	148	173	173	181	197	203	199	20E	242	250
1294	410	538	72E	788	725	707	699	681	634	636
1296	320	450	44c	620	620	53c	57c	48C	56C	490
1298	132	138	10E	116	163	181	185	189	187	195
1299	350	260	21c	390	470	59C	87C	56C	60C	550
1301	299	325	354	396	471	504	502	496	469	473
1303	299	354	33E	345	376	35E	276	26G	301	305
1305	173	173	17E	226	370	423	40c	404	425	414
1307	315	431	67E	658	339	261	313	311	299	293
1309	380	410	43c	530	680	50c	47c	49c	50c	480
1311	699	842	914	958	974	91C	729	653	633	619
1312	1446	1390	1141	1077	1125	1185	1314	139E	147C	1494
1313	3089	3081	287C	2920	3294	328E	2924	2796	276C	2782
1314	2605	2513	2404	2416	2567	2651	2735	2762	2725	2713
1316	868	783	787	831	948	105C	1195	1227	1133	1111
1318	597	614	524	492	467	484	563	597	605	616
1320	482	475	425	414	423	428	439	465	514	540
1322	560	430	39c	380	330	38C	330	410	35a	440
1324	360	300	22C	270	240	31C	340	390	320	370
1326	530	610	44c	400	380	34a	270	320	34a	370
1328	857	939	861	798	689	614	542	512	528	520
1330	390	390	34C	330	270	210	230	210	170	130
1332	502	561	571	545	451	410	392	390	423	427
1334	854	933	829	758	661	5 7 1	455	423	467	498
1336	886	1044	102C	1032	1042	985	819	786	868	906
1338	1065	1292	1382	1333	1048	933	880	874	955	979
1340	1263	1486	1504	1428	1141	1024	990	970	990	1012
1342	1418	1291	966	1048	1611	1834	1846	1878	1878	1806
1344	2161	2420	2480	2460	2259	2163	2129	2053	1900	1926
1346	3587	3615	3336	3434	3916	4099	4067	4135	4280	4284
1347	1093	1181	1113	1066	970	930	950	906	783	787
1349	570	520	470	490	530	470	460	400	400	430
1351	440	390	380	390	540	630	690	800	730	670
1353	660	560	600	620	580	500	600	590	600	680
1355	317	349	386	400	392	376	331	339	417	439
1357	980	966	890	892	968	992	980	996	1032	1064
1359	440	480	400	450	420	490	510	520	570	540
1361	1040	840	730	790	720	630	680	450	660	570
1363	1100	1060	890	830	860	840	840	800	790	880
1365	1036	1056	894	769	599	492	412	382	422	426
1367	780	710	560	480	490	560	500	550	600	660
1369	524	502	445	427	439	427	380	384	443	459
1371	900	720	640	500	450	540	570	560	580	550
1373	449	451	441	419	366	351	364	388	449	479
1386	884	836	900	840	958	704	550	496	660	632

Link #	7:00	7:30	8:00	8:30	9:00	9:30	10:00	10:30	11:00	11:30
1388	494	516	626	572	516	516	526	552	584	716
1396	212	408	628	688	468	424	296	268	316	320
1429	856	1040	1374	1440	1200	1008	856	916	962	1044
1431	262	308	340	312	210	156	152	168	180	176

ink #	12:00	12:30	13:00	13:30	14:00	14:30	15:00	15:30	16:00	16:30
1	1938	1716	1904	1764	1854	1932	1898	2044	1924	2156
3	1068	1012	1070	920	1094	1060	1132	1196	1244	1300
a	1424	1456	1408	1400	1558	1408	1462	1428	1524	1444
25	1250	1280	1332	1292	1390	1340	1452	1508	1482	1408
27	1194	1180	1288	1272	1316	1416	1410	1604	1538	1484
29	958	968	956	896	1008	1016	1110	1304	1250	1276
34	1010	1000	956	988	1054	980	1112	1392	1506	1452
37	594	516	604	608	734	736	810	988	850	900
39	818	996	626	540	656	528	514	948	1108	1144
46	894	1016	476	820	822	1020	1062	1092	1120	112E
49	416	828	1038	1148	1006	1172	1358	1452	1468	152E
53	1216	1316	1240	1264	1164	1252	1218	1208	1302	1504
57	1122	1116	1034	1112	976	1008	1130	1220	1180	116C
6C	1190	1216	1170	1200	1198	1216	1166	1400	1282	126E
61	1088	572	610	556	570	620	566	768	658	63E
73	1000	868	896	2336	3712	3804	3474	3808	3282	351E
74	1212	1204	1060	944	1112	1056	1176	1228	1266	136C
81	968	872	1218	1208	1074	1112	1166	1232	1322	1424
83	1130	992	1166	1176	1180	1224	1014	1196	1150	1232
86	798	836	740	768	1028	1052	1206	1260	1274	139E
95	310	312	136	0	162	0	0	356	432	432
10C	878	964	956	916	942	924	958	1212	1142	1272
107	944	956	912	916	976	960	930	1044	1130	1204
126	1890	1912	2072	2004	1958	2092	2132	2060	2264	214E
131	1304	1380	1362	1420	1330	1360	1166	1108	1212	1164
133	1960	1924	2044	2108	1986	2176	1934	2024	2044	1904
135	1248	1200	1272	1404	1230	1116	1290	1140	1198	1312
136	1540	1588	1230	1268	1186	1208	1212	1096	1134	1192
141	990	1192	1006	1104	1134	996	1036	924	1060	100C
143	1306	1364	1318	1480	1442	1364	1326	1340	1370	123E
144	1426	1560	1484	1636	1606	1480	1488	1552	1502	138C
146	748	800	790	868	858	772	778	776	742	692
152	492	504	468	400	388	548	678	744	776	56C
15E	1172	1168	1208	1164	1222	1288	1340	1360	1430	1444
162	1488	1668	1408	1588	1544	1544	1550	1748	1360	1472
170	926	800	950	968	964	1000	1086	1084	1210	1300
175	610	640	642	696	690	768	760	764	800	724
184	1150	508	614	548	630	544	800	752	678	476
186	482	572	560	608	628	484	488	616	770	796
189	536	588	562	576	584	636	664	616	758	784
195	678	632	568	612	632	668	716	688	750	560
196	660	508	636	584	592	612	732	696	756	704
212	92	24	166	368	362	376	448	480	568	392
213	680	660	608	708	682	664	792	832	940	896
219	572	624	554	680	564	656	602	688	710	700
222	480	544	414	584	516	568	602	556	636	592
224	440	492	412	520	546	480	552	532	650	616
228	540	532	504	560	594	560	752	648	714	640
233	742	716	732	760	770	956	898	892	996	968
238	624	744	680	652	688	800	704	768	756	824
243	1382	1300	1366	1256	1384	1400	1032	1320	1340	1280
246	1512	1452	1432	1408	1410	1428	1716	1472	1452	1496
250	958	988	958	1032	962	988	956	1012	974	824
256	1228	1248	1246	1236	1202	1404	1340	1264	1248	1152
262	1618	1636	1522	1576	1438	1556	1622	1672	1530	1584
264	1676	1684	1554	1612	1562	1688	1750	1804	1606	1660
265	1688	1672	1564	1560	1500	1616	1680	1684	1480	1612
267	1474	1356	1410	1404	1270	1460	1356	1452	1184	1268

Link #	12:00	12:30	13:00	13:30	14:00	14:30	15:00	15:30	16:00	16:30
272	1342	1236	1462	1320	1402	1372	1338	1348	1256	1348
273	1438	1272	1338	143E	1438	1400	1394	1384	1420	1596
275	1240	1224	1220	1352	1388	1328	1550	1624	1484	1464
276	1150	1064	1054	1080	1230	1144	1336	1376	1376	1436
278	1644	1504	1522	1500	1600	1736	1776	1908	1858	1960
281	1030	904	1004	996	980	1048	1118	1136	1086	1280
285	1858	1756	1926	1824	1932	1944	2250	2260	2150	2356
286	1594	1516	1658	1504	1610	1696	1974	2140	1842	2200
289	1000	864	900	928	996	1048	798	688	1314	1224
291	552	452	546	552	572	604	696	696	682	804
293	826	720	798	756	874	904	1104	1156	1174	1244
296	800	704	736	744	832	860	1104	1204	1312	1332
300	596	596	586	572	584	584	824	936	920	1048
303	420	372	400	332	436	404	572	624	702	744
305	1256	964	854	908	996	1024	1346	1472	1498	1608
308	534	648	596	520	668	664	908	976	1004	1104
312	552	612	504	460	634	624	784	768	888	912
314	944	1072	996	848	1016	1016	1080	1080	1134	1164
315	496	528	426	464	534	476	628	636	738	792
317	460	460	424	344	488	420	326	484	482	460
319	498	504	356	416	438	456	470	452	624	668
325	448	468	384	452	444	500	540	572	750	872
327	500	524	430	540	470	408	578	624	768	860
329	528	436	424	540	522	524	668	672	806	884
330	782	768	818	900	808	912	994	976	1204	1272
332	572	560	564	464	560	628	646	700	794	880
339	694	604	740	580	796	752	900	968	982	1216
341	492	456	562	460	532	568	596	636	742	784
344	432	436	482	448	556	464	530	676	714	800
348	714	812	782	760	768	980	938	1112	1108	1264
351	782	824	816	780	886	980	1002	1192	1150	1124
353	936	992	878	908	928	1088	1084	1372	1332	1424
355	1158	1152	1188	1120	1192	1228	1366	1476	1424	1612
358	1100	1248	1036	976	1104	1144	1266	1252	1430	1480
359	1306	1452	1216	1212	1280	1324	1410	1468	1634	1608
363	1368	1396	1362	1204	1324	1320	1392	1464	1564	1648
366	1516	1548	1610	1464	1404	1460	1610	1688	1758	1848
368	1222	1160	1300	1256	1230	1152	1384	1484	1608	1536
369	1286	1248	1300	1288	1250	1188	1496	1496	1634	1548
386	276	304	264	328	300	388	292	348	240	236
388	368	384	328	424	394	464	568	536	552	564
390	476	432	450	504	544	664	784	672	776	836
393	1558	1732	1614	1740	1586	1696	1724	1632	1522	1504
397	590	536	526	544	602	740	800	812	812	808
398	898	940	900	972	920	964	1068	1156	1134	1068
403	668	648	634	616	730	772	926	836	916	996
406	1182	988	1042	1276	1140	1200	644	1020	1228	1208
409	832	828	912	952	862	992	1102	1104	1368	1372
411	232	180	286	292	290	328	1038	360	370	360
429	1164	1220	1058	1116	1208	1208	1328	1212	1476	1456
432	1182	1176	1112	1200	1350	1312	1354	1340	1564	1608
435	622	696	598	660	682	648	784	596	866	884
440	802	1012	840	872	980	1140	1200	912	1170	1152
441	484	488	444	500	530	568	668	544	694	632
445	1290	1308	1260	1256	1296	1280	1410	1352	1448	1440
450	548	676	520	636	732	660	690	688	698	708
452	1406	1420	1292	1468	1598	1536	1732	1692	1682	1824
469	268	216	190	188	258	188	306	280	310	300

ink #	12:00	12:30	13:00	13:30	14:00	14:30	15:00	15:30	16:00	16:30
470	382	400	332	352	398	488	582	608	638	676
473	736	772	612	668	798	752	904	1076	1002	1108
475	488	560	520	492	516	476	626	684	824	960
477	678	796	774	692	766	768	848	764	1076	1156
481	516	580	494	452	542	588	616	652	818	836
485	1160	1028	1246	1120	1080	1168	1378	1476	1442	1472
509	486	580	558	488	508	476	636	712	702	908
513	462	420	436	496	484	620	598	636	688	784
517	564	600	592	500	566	560	648	668	500	664
519	576	496	586	508	592	584	702	728	714	812
521	530	508	500	552	614	580	628	708	676	678
524	434	416	372	448	470	376	434	488	482	532
525	314	328	316	328	344	332	306	332	346	420
528	482	536	448	524	546	692	782	792	810	840
531	1598	2828	2710	560	2686	2940	812	2904	2432	2456
533	796	840	742	940	852	1056	1252	1312	1326	1540
534	460	580	448	548	544	632	706	924	898	1000
535	386	448	356	348	404	468	380	568	524	492
537	280	300	298	296	280	396	324	376	420	412
540	376	380	396	436	400	484	484	520	424	444
543	764	672	698	736	724	840	840	912	852	928
544	1246	1080	1124	1200	1218	1288	1340	1284	1250	1292
546	1102	1004	938	1048	1006	1116	1108	1040	1112	1124
549	1086	1020	1018	956	1064	1168	1184	1032	1126	1064
550	582	648	606	648	678	752	706	580	696	696
551	398	396	346	300	330	412	478	388	466	588
553	322	376	278	332	278	308	354	348	390	392
558	468	552	514	536	496	604	726	612	614	732
559	668	804	680	732	708	772	1010	912	806	952
560	584	604	558	560	604	676	652	708	752	816
564	320	392	328	396	396	428	626	560	536	636
565	616	540	600	536	596	660	668	632	672	768
566	216	212	276	224	218	244	238	248	296	312
570	674	564	682	688	688	828	900	892	864	940
571	952	792	918	928	1002	1168	1210	1232	1278	1288
573	480	544	502	492	540	692	528	524	534	544
575	570	488	620	576	552	620	690	708	732	808
578	850	892	902	868	780	1008	1034	1012	1076	1052
580	1286	1272	1286	1240	1282	1500	1396	1456	1498	1432
581	1082	1032	1024	936	956	1188	1118	1088	1120	1040
583	496	496	464	396	700	684	676	580	568	636
584	522	484	466	484	672	768	776	660	740	872
587	680	968	834	908	980	920	1164	1028	926	980
595	708	668	654	648	760	972	922	984	894	900
598	704	628	682	640	714	728	934	896	908	912
599	600	668	646	600	634	664	750	784	806	836
606	124	164	138	84	140	176	172	156	162	168
613	162	172	182	184	202	224	302	280	304	372
615	304	344	390	368	434	328	480	480	480	528
616	194	156	140	168	240	232	274	432	324	320
617	828	808	786	768	828	1052	868	824	1050	1116
619	260	188	228	252	306	280	366	428	340	364
627	708	616	650	700	702	272	580	648	652	636
629	1204	1128	1190	1164	1150	1144	1218	1392	1400	1408
631	832	768	812	844	840	908	984	948	1074	1164
632	560	508	496	500	630	688	750	748	872	968
633	500	380	472	452	574	604	610	556	718	712
635	686	612	556	540	614	636	618	592	580	588

ink #	12:00	12:30	13:00	13:30	14:00	14:30	15:00	15:30	16:00	16:30
636	478	392	332	436	554	484	456	360	476	500
637	528	508	454	488	660	476	520	524	494	512
639	892	1016	882	1032	896	896	948	812	812	828
641	958	912	904	948	918	960	910	936	932	876
648	1006	916	910	912	942	956	978	1128	1192	1208
650	722	424	604	456	452	508	442	544	796	976
653	850	848	832	820	858	720	864	1012	1030	1212
656	720	624	680	676	728	692	810	836	856	944
659	872	932	806	916	938	904	980	1072	916	1224
662	1250	1164	1126	1292	1280	1200	1128	1476	1236	1380
664	1128	940	1244	1008	1298	1080	972	1136	1010	968
665	1058	1044	1042	1208	1136	1100	1078	1152	1190	1156
673	588	612	610	604	638	708	696	696	770	796
679	392	444	420	340	436	440	444	384	436	456
680	720	628	704	668	682	724	710	672	748	644
683	1538	1484	1456	1492	1444	1776	1744	1676	1770	1840
685	864	912	816	844	768	1000	1052	960	1038	1076
687	610	784	696	600	674	652	748	776	740	812
691	708	708	722	704	698	676	672	684	714	648
693	782	856	738	716	808	820	818	744	740	812
695	1600	1696	1620	1652	1606	1808	1764	1676	1642	1752
698	1148	1220	1208	1244	1164	1272	1142	1160	1066	1072
699	774	768	846	808	826	640	802	556	768	736
718	316	284	332	308	334	348	390	388	408	432
722	366	472	396	376	362	348	466	472	360	416
729	1102	1024	982	1060	1106	1036	1060	1004	992	1076
736	1310	1224	1284	1292	1310	1448	1434	1384	1312	1360
737	1380	1376	1348	1320	1430	1520	1608	1656	1424	1392
739	1410	1448	1396	1320	1394	1448	1672	1776	1506	1472
740	1976	1932	1882	1892	1902	2008	2128	2100	2024	1844
744	832	748	832	888	852	816	826	936	848	884
748	404	368	376	268	354	404	398	416	432	472
749	866	756	834	856	814	896	892	960	888	968
750	1188	1036	1186	1256	1110	1236	1266	1224	1228	1236
752	1498	1612	1428	1568	1532	1644	1552	1456	1530	1668
760	1264	1328	1232	1456	1290	1396	1240	1344	1284	1360
763	1132	1120	1108	1104	1088	960	1294	1128	1280	1328
765	1090	1048	1082	1024	1080	1120	1266	1228	1372	1408
768	1570	1584	1586	1612	1584	1660	1666	920	1540	1508
772	2028	1816	1856	1976	1928	1788	1396	912	1792	1728
773	2358	2204	2244	2116	2382	2392	1916	1404	2714	2632
776	1430	1448	1298	1316	1388	1560	1718	1928	1750	1896
780	408	488	444	524	466	444	494	548	426	492
783	1336	1332	1294	1388	1334	1236	1424	1584	1468	1540
784	1830	1864	1836	1884	1900	1712	2090	2208	2038	2052
788	1520	1728	1540	1624	1552	1540	1526	1688	1466	1492
791	940	912	892	976	904	984	926	1016	864	904
793	1344	1324	1318	1268	1292	1340	1550	1664	1280	1292
796	486	428	450	548	404	440	480	600	544	488
800	484	488	478	480	436	556	568	540	618	660
804	1178	1072	1180	1244	1208	1288	1358	1368	1218	1244
805	974	888	948	1008	1008	1012	1066	1160	1030	1024
806	700	636	618	844	744	752	818	796	798	860
807	948	844	892	976	1112	1052	1200	1316	1280	1248
809	1164	1236	1050	1192	1276	1372	1432	1492	1394	1564
810	750	760	780	768	800	920	1048	952	1102	1064
811	674	696	646	780	690	696	744	700	704	884
812	4028	3952	4126	4064	3916	4064	2406	3320	2832	3088

ink #	12:00	12:30	13:00	13:30	4:00	14:30	15:00	15:30	16:00	16:30
8161	9461	912	786	780	904	960	850	976	878	992
818	842	816	838	836	868	792	846	804	866	976
829	426	420	444	484	422	472	448	544	598	572
830	334	336	382	312	368	424	326	484	412	472
831	360	344	400	304	382	396	360	472	484	476
832	430	408	432	372	416	452	470	496	536	568
833	342	284	346	304	350	360	394	404	452	504
835	186	212	200	288	192	196	238	292	252	304
836	296	332	248	328	286	356	382	412	368	412
837	346	336	322	336	368	412	404	468	378	412
838	240	232	242	296	268	328	344	360	296	316
839	544	548	556	500	580	604	594	652	632	604
847	1128	1536	1494	1060	1432	1576	1198	1716	1914	1844
859	1738	1768	1670	1700	1622	1632	1760	1944	1770	1488
861	8	12	a	8	10	a	8	8	10	8
866	1658	1804	1766	1820	1724	1896	1934	1888	1864	1656
870	1272	1368	1310	1428	1270	1940	1824	1428	1490	1564
874	1722	1876	1764	1868	1802	1892	2102	2172	2486	2672
878	1370	1404	1408	1260	1388	1564	1540	1700	1560	1548
883	1366	1376	1428	1400	1392	1532	1720	1660	1760	1796
884	1728	1736	1812	1684	1758	1672	1798	2028	1872	1896
889	1602	1584	1678	1624	1514	1324	1248	1336	1220	1244
896	344	384	348	424	392	452	590	492	616	564
902	302	312	294	276	338	380	368	388	386	352
905	740	800	480	1368	830	848	982	988	1060	1160
908	612	608	600	736	856	1036	552	848	904	1208
911	1634	1496	1702	1880	1856	1852	508	1756	1802	1808
913	1630	1740	1564	1636	1628	1764	1842	2024	2022	2128
915	1372	1420	1258	1496	1504	1620	1566	1380	1236	1200
916	1136	1172	1044	1044	1128	1276	1258	1240	1228	1368
919	1618	1484	1350	1320	1608	1628	1724	1696	1730	1876
920	1202	1284	1158	1192	1388	1472	1512	1444	1486	1656
921	840	848	742	800	866	972	1092	1068	1002	1160
923	786	924	616	808	910	1068	1096	1080	1082	1284
931	622	156	436	392	352	392	490	508	580	584
933	658	588	662	684	666	704	810	892	1058	1084
934	616	576	618	684	666	676	816	844	948	1012
935	626	592	652	716	750	592	860	888	994	1092
936	524	488	582	552	612	516	644	560	580	576
938	544	552	564	496	678	636	800	840	908	1032
940	500	492	524	460	586	596	546	656	592	660
941	630	676	600	632	744	748	738	808	730	812
944	734	748	642	572	824	728	918	1044	1194	944
945	574	640	570	552	606	552	618	492	588	504
946	630	872	590	608	894	980	874	1036	536	564
948	1062	1132	1076	1144	1084	1116	1170	1200	1150	1204
951	1030	940	1116	1100	1084	1192	1044	1144	1206	1240
952	1080	904	1166	1128	1126	1160	1140	1200	1218	1248
954	148	140	240	324	158	180	192	184	182	204
955	454	452	584	528	560	556	550	576	692	632
956	848	924	866	836	938	904	918	1028	1004	984
957	304	304	328	380	282	328	348	364	328	360
959	964	1160	1108	1060	1172	1208	1192	1236	1288	1308
960	828	876	892	864	1008	952	962	996	976	1048
961	1584	1380	1700	1688	1634	1520	1668	1772	1580	1644
962	1240	1324	1056	1144	1284	1484	1448	1472	1552	1440
964	1088	1200	1080	1072	1144	1184	1134	1280	1244	1200
968	1190	1260	1150	1108	1204	1088	1348	1264	1304	1428

.ink #	12:00	12:30	13:00	13:30	14:00	14:30	15:00	15:30	16:00	16:30
976	602	640	612	568	584	596	660	632	686	656
978	672	692	726	652	746	836	780	836	850	772
980	54%	660	534	484	606	592	726	724	852	924
989	966	1044	996	976	1052	1124	1294	1196	1296	1280
990	580	472	458	468	484	528	600	664	830	94c
992	622	764	606	564	672	604	724	796	916	792
994	610	592	596	532	626	664	632	564	654	704
996	606	716	668	664	588	620	638	772	612	62C
1000	1266	1312	1310	1192	1364	1180	1242	1416	1216	1256
1012	126	1304	1314	1272	1292	1356	1120	1476	1366	1492
1014	1152	1260	1064	1116	1076	1156	1162	1348	1286	1248
1018	1416	1292	1080	1048	1210	1176	1292	1248	1340	1472
1024	1000	928	972	1012	932	1024	1084	1200	1178	1436
1028	860	904	916	940	836	960	992	1036	1128	1276
1030	1302	1166	1200	1180	1248	1380	1430	1444	1556	1612
1046	1086	1000	992	972	1070	1168	1238	1412	1398	1616
1048	750	644	698	688	778	764	824	1008	974	1068
1052	698	696	638	736	776	740	834	1044	1050	1116
1054	1354	1240	1144	1208	1254	1316	1422	1280	1390	1264
1057	662	596	592	656	628	636	772	644	798	572
1059	47C	432	378	432	454	556	526	532	656	676
1062	850	724	706	760	712	844	796	708	942	912
1112	560	548	554	528	556	688	626	660	640	584
1117	650	616	688	752	688	668	602	656	574	568
1131	886	872	914	920	996	896	938	1004	852	920
1135	942	952	916	1052	1010	1084	1076	1016	1014	944
1136	452	560	496	476	544	616	554	516	486	484
1176	5640	5792	5744	5204	5006	5514	5548	6324	6594	6578
1177	5640	5792	5744	5204	5006	5514	5548	6324	6594	6578
1178	5640	5792	5744	5204	5006	5514	5548	6324	6594	6578
1179	8336	8498	8490	7976	7760	8396	8804	9622	9638	9286
1180	9462	9604	9508	9024	8986	9658	9982	10762	10754	10352
1181	9004	9136	9054	8600	8602	9282	9590	10398	10120	10088
1182	9624	9764	9670	9270	9420	10148	10420	11218	10918	10872
1183	9168	9318	9266	8880	9006	9724	10006	10830	10564	10562
1184	9718	9884	9860	9488	9620	10344	10636	11222	11156	11140
1185	9718	9884	9860	9488	9620	10344	10636	11222	11156	11140
1186	9718	9884	9860	9488	9620	10344	10636	11222	11156	11140
1187	9080	9260	9296	8970	9134	9854	10114	10710	10690	10680
1188	9080	9260	9296	8970	9134	9854	10114	10710	10690	10680
1189	9080	9260	9296	8970	9134	9854	10114	10710	10690	10680
1190	9520	9702	9706	9450	9834	10404	10572	11100	11100	11100
1191	8360	8550	8626	8426	8846	9416	9546	10174	10354	10572
1192	8360	8550	8626	8426	8846	9416	9546	10174	10354	10380
1193	8360	8550	8626	8426	8846	9416	9546	10174	10354	10380
1194	8802	9000	8966	8766	9246	9786	9948	10524	10660	10660
1195	8188	8400	8438	8262	8744	9238	9300	9828	9900	9900
1196	8488	8730	8798	8632	9074	9530	9700	10180	10362	10350
1197	7592	7850	7928	7898	8320	8682	8634	9102	9412	9460
1198	7592	7850	7928	7898	8320	8682	8634	9102	9412	9460
1199	8084	8356	8500	8500	8942	9282	9164	9588	9858	9882
1200	7550	7696	7936	7964	8396	8664	8614	9042	9352	9406
1201	7550	7696	7936	7964	8396	8664	8614	9042	9352	9406
1202	7728	7836	8066	8114	8556	8884	8842	9282	9524	9626
1203	7090	7186	7498	7534	7854	8126	8102	8532	8714	8824
1204	7506	7592	7890	7926	8278	8552	8510	8940	9136	9242
1205	6546	6588	6792	6798	7122	7370	7336	7832	8126	8308
1206	5616	5668	5842	5832	6104	6288	6190	6732	7152	7404

.ink #	12:00	12:30	13:00	13:30	14:00	14:30	15:00	15:30	16:00	16:30
1207	5616	5668	5842	5832	6104	6288	6190	6732	7152	7404
1208	6174	6236	6356	6360	6714	6920	6804	7322	7692	7928
1209	7128	7202	7292	7298	7686	7878	7718	8204	8524	8758
1210	6262	6336	6414	6442	6854	7036	6846	7378	7796	8096
1211	7234	7312	7376	7420	7874	8082	7930	8466	8828	9146
1212	6624	6682	6688	6700	7108	7284	7148	7696	8062	837C
1213	6624	6682	6688	6700	7108	7284	7148	7696	8062	837C
1214	5796	5850	5874	5876	6222	6332	6160	6678	6990	729c
1215	6848	6928	6972	6974	729a	7404	7260	7794	8114	8426
1216	4692	4732	4788	4804	5032	5128	5096	5760	6218	6638
1217	2308	233a	2342	2310	2356	2364	2408	3232	3944	4476
1218	2308	2330	2342	2310	2356	2364	2408	3232	3944	4476
1219	4708	4604	4054	3998	4106	4322	5134	5836	5854	5722
1220	4708	4604	4054	3998	4106	4322	5134	5836	5854	5722
1221	8464	8398	7936	7924	8164	8210	8434	9026	9226	928C
1222	6868	6900	6900	7028	7164	7206	7408	7980	8200	8200
1223	7780	7866	7892	8024	alla	ala4	8478	9068	9234	9244
1224	7780	7866	7892	8024	8118	8184	8478	9068	9234	9244
1225	8160	8306	8274	8442	8636	8724	9008	9526	9676	9706
1226	7292	7434	7368	7536	7744	7798	7938	8442	8740	8782
1227	7292	7434	7368	7536	7744	7798	7938	8442	8740	8782
1228	7952	8104	8008	8258	8464	8540	8628	9050	9360	944C
1229	7714	7874	7810	8054	8214	8278	8374	8798	9102	9202
1230	8326	8454	8390	8704	8854	8886	9004	9298	9682	9752
1231	7648	7748	7664	7990	8162	8218	8332	8636	9072	9138
1232	7144	7200	7112	7432	7566	7630	7634	8002	8438	8514
1233	7144	7200	7112	7432	7566	7630	7634	8002	8438	8382
1234	7568	7616	7506	7834	8014	8092	8074	8450	8900	8888
1235	8680	8742	8606	8932	9130	9206	9154	9530	10000	9784
1236	8474	8526	8380	8698	8882	8952	8908	9258	9682	9432
1237	9034	9146	8960	9256	9550	9644	9446	9818	10162	988C
1238	8510	8636	8428	8688	9030	9088	8914	9296	9682	9430
1239	8510	8636	8428	8688	9030	9088	8914	9296	9682	9430
1240	9070	9226	9090	9320	9738	9814	9730	10122	10144	1020C
1241	8566	8698	8524	8766	9246	9330	9226	9572	9528	9526
1242	8566	8698	8524	8766	9246	9330	9226	9572	9528	9526
1243	9436	9508	9472	9748	10198	10178	10088	10572	10438	10386
1244	9172	9234	9136	9394	9862	9822	9684	10130	9978	9912
1245	9582	9650	9566	9852	10294	10346	10110	10532	10392	10326
1246	9218	9284	9148	9430	9882	9954	9770	10184	9986	9902
1247	9218	9284	9148	9430	9882	9954	9770	10184	9986	9902
1248	9796	9894	9790	10070	10512	10614	10380	10834	10616	10444
1249	9504	9602	9520	9796	10210	10278	9982	10400	10176	9996
1250	9504	9602	9520	9796	10210	10278	9982	10400	10176	9996
1251	9952	10066	10028	10322	10744	10800	10458	10852	10606	10426
1252	9952	10066	10028	10322	10744	10800	10458	10852	10606	10426
1253	9378	9558	9506	9794	10240	10262	10006	10298	10058	9950
1254	9378	9558	9506	9794	10240	10262	10006	10298	10058	9950
1255	10010	10198	10198	10532	10980	10962	10644	10878	10578	10512
1256	9376	9556	9548	9874	10282	10252	9940	10130	9768	9642
1257	9856	10062	10106	10462	10882	10842	10486	10642	10226	10100
1258	8440	8608	8544	8834	9106	9064	8780	8996	8740	8674
1259	5522	5666	5736	6098	6296	6370	6316	6606	6352	6402
1260	5522	5666	5736	6098	6296	6370	6316	6606	6352	6402
1262	463	467	451	427	386	370	384	360	307	268
1264	461	445	400	392	417	417	404	382	354	313
1266	644	622	561	522	488	482	512	508	471	465
1268	1175	1149	1074	1030	994	974	1004	934	769	657

Link #	12:00	12:30	13:00	13:30	14:00	14:30	15:00	15:30	16:00	16:30
1270	621	597	526	506	504	540	635	699	781	783
1272	906	878	763	739	759	835	1044	1085	978	936
1274	540	660	560	540	550	610	540	550	520	500
1275	646	648	565	583	705	746	725	752	833	839
1277	974	1002	1091	1135	1161	1163	1151	1115	1036	978
1279	942	918	942	970	1024	1064	1123	1105	998	948
1281	878	864	872	860	836	829	854	825	745	691
1283	620	628	685	723	770	786	766	768	784	807
1286	838	829	811	831	892	936	966	1016	1093	1123
1288	2185	2193	2167	2185	2269	2243	2121	2028	1926	1852
1289	2414	2398	2434	2509	2691	2721	2633	2523	2314	2239
1291	1615	1506	1032	900	988	1008	1044	1060	1042	1083
1292	878	876	902	910	882	929	1087	1097	959	931
1293	242	232	199	205	248	262	258	256	262	238
1294	687	709	723	717	683	670	683	670	624	618
1296	510	550	550	560	589	591	709	642	650	630
1298	209	217	226	234	246	254	250	274	327	360
1299	530	510	430	570	513	558	542	527	490	460
1301	510	530	563	555	488	482	514	557	630	699
1303	268	276	335	354	333	356	412	449	469	492
1305	368	368	417	425	410	392	347	351	414	437
1307	297	291	268	274	299	335	404	439	451	463
1309	580	510	520	530	500	540	460	560	560	590
1311	639	645	647	663	691	713	717	757	827	896
1312	1434	1460	1555	1635	1755	1781	1735	1665	1516	1468
1313	2954	2954	2798	2745	2776	2699	2504	2418	2436	2340
1314	2695	2705	2745	2772	2754	2882	3254	3298	3045	2952
1316	1127	1105	1018	1050	1227	1261	1177	1139	1115	1066
1318	620	628	616	670	819	868	831	819	794	784
1320	549	567	595	607	614	620	630	622	595	579
1322	440	440	410	480	700	550	460	390	410	420
1324	440	450	340	340	400	370	400	350	330	280
1326	300	330	360	370	330	290	400	350	460	450
1328	492	506	571	601	622	601	530	486	447	425
1330	180	140	130	150	160	220	230	240	170	220
1332	417	408	392	394	423	425	408	408	421	419
1334	559	567	514	528	610	632	614	591	541	524
1336	953	967	935	937	971	959	914	882	833	831
1338	973	975	963	977	1020	1046	1083	1087	1032	1052
1340	1052	1079	1099	1099	1068	1072	1101	1117	1125	1137
1342	1591	1575	1749	1796	1816	1731	1504	1382	1303	1269
1344	2163	2219	2131	2131	2243	2159	1798	1808	2147	2332
1346	4119	4109	4206	4274	4412	4350	4077	3924	3806	3728
1347	912	966	992	1000	956	978	1070	1087	1034	1044
1349	380	440	380	420	520	540	530	460	440	460
1351	660	670	640	720	720	740	690	610	620	660
1353	610	580	580	650	640	610	630	500	580	550
1355	425	417	394	402	449	461	441	449	467	504
1357	1113	1125	1099	1097	1117	1113	1081	1079	1101	1113
1359	560	620	580	560	669	694	539	557	480	450
1361	560	590	660	630	709	725	817	825	760	770
1363	870	810	950	980	954	848	860	1001	910	860
1365	410	414	430	458	532	524	428	402	414	414
1367	580	610	640	640	630	660	610	650	630	540
1369	447	463	508	526	534	520	475	451	431	429
1371	630	640	690	740	740	700	640	580	520	560
1373	479	506	559	587	599	589	545	512	459	459
1386	778	780	534	584	520	664	620	524	654	628

Link #	12:00	12:30	13:00	13:30	14:00	14:30	15:00	15:30	16:00	16:30
1388	638	636	652	684	762	756	786	824	826	908
1396	364	444	332	336	328	348	356	364	348	336
1429	1026	1060	916	964	994	1092	1080	1060	1022	1344
1431	182	180	178	180	146	164	210	244	228	268

Link #	17:00	17:30	18:00	18:30	19:00	19:30
	2098	2420	2	1732	1390	1192
3	1312	1408	1248	1224	a72	692
8	1590	1792	1586	1544	1288	1012
25	1482	1664	1606	1520	1322	956
27	1574	1808	1626	1500	1292	980
29	1354	1536	1358	1248	1008	748
34	1492	1576	1430	1404	1016	704
37	566	476	816	784	540	508
39	1238	1188	1230	1092	732	636
46	1240	1192	1106	1036	686	580
49	1784	1688	1552	1472	906	744
53	1456	1524	1572	1448	1286	1068
57	1222	1332	1254	1316	1222	996
60	1318	1348	1252	1128	962	764
61	562	624	574	556	476	448
73	3382	3600	2708	3308	3902	4208
74	1606	1556	1650	1680	1036	860
81	1490	1532	1300	1388	1062	908
83	1202	928	1122	1076	1156	908
88	1776	1792	1604	1524	1084	728
95	540	564	458	424	298	292
100	1370	1508	1290	1264	980	776
107	1280	1372	1174	1228	910	736
126	2698	2780	2816	2484	2014	1556
131	1484	1524	1576	1376	1008	880
133	2496	2580	2556	2404	1682	1328
135	1504	1320	1372	1340	980	896
136	1470	1600	1524	1512	1014	948
141	1058	956	1024	932	862	792
143	1334	1140	1398	1192	1120	1056
144	1544	1364	1554	1356	1244	1156
146	788	684	788	664	604	584
152	566	892	678	804	716	596
158	1414	1452	1494	1172	1014	828
162	1364	1356	1448	1476	1654	1368
170	1256	1304	1216	1016	776	636
175	908	852	846	700	564	412
184	934	812	778	452	400	392
186	856	880	696	536	702	380
1a9	834	780	788	628	514	472
195	738	652	680	616	600	492
196	790	832	752	544	568	412
212	a4	36	4	0	398	260
213	1040	1072	1216	1012	690	380
219	796	848	880	868	636	368
222	784	744	850	816	542	412
224	776	680	838	792	506	428
228	910	684	860	820	504	424
233	1164	812	1094	1136	588	444
238	858	816	1004	888	518	480
243	1406	1464	1502	1476	1174	948
246	1590	1612	1524	1272	1210	1080
250	1042	932	918	1088	768	604
256	1256	1392	1334	1420	964	864
262	1750	1716	1900	1772	1340	1028
264	1944	1956	2036	1968	1462	1136
265	1824	1872	1996	1904	1562	1112
267	1468	1436	1622	1472	1336	1040

Link#	7:00	7:30	8:00	18:30	19:00	19:30
272	1516	1464	1558	1408	1206	1012
273	1752	1800	1830	1508	1214	1068
275	1650	1592	1598	1588	1362	1112
276	1538	1460	1420	1308	1106	932
278	2180	1876	1894	1828	1488	1320
281	1332	1188	1166	984	836	788
285	2562	2408	2238	2068	1572	1332
286	2328	2168	1952	1836	1406	1116
289	1266	1076	1362	1180	998	780
291	884	904	766	604	444	372
293	1434	1532	1220	1004	732	640
296	1570	1724	1346	1116	752	592
300	1154	1120	1050	784	538	472
303	878	840	752	644	390	276
305	1758	1812	1562	1392	792	596
308	1214	1188	1048	1008	598	376
312	980	1044	908	908	514	336
314	1150	1340	1324	1228	1020	780
315	894	968	752	760	500	324
317	428	364	410	440	430	244
319	684	692	572	460	358	244
325	1058	1092	1280	868	392	260
327	1090	1160	1304	908	464	304
329	1114	1120	1294	1032	548	408
330	1260	1352	1162	1156	950	712
332	1108	1140	1190	1000	608	472
339	1486	1512	1600	1332	804	512
341	980	980	1136	1120	674	528
344	932	968	1016	948	516	356
348	1480	1584	1562	1464	938	660
351	1478	1700	1630	1656	998	740
353	1672	1732	1544	1464	952	636
355	1756	1560	1532	1352	1124	900
358	1830	1864	1786	1740	1284	956
359	1974	2076	1988	1888	1398	1028
363	1658	1764	1808	1888	1418	920
366	2180	2360	2214	2216	1636	1440
368	1708	1912	1838	1876	1430	1164
369	1822	1976	2036	1892	1548	1276
386	294	316	256	236	234	172
388	692	704	616	468	352	236
390	976	900	758	592	472	340
393	1620	1580	1644	1600	1420	1272
397	966	928	758	656	502	392
398	1284	1268	1194	1156	908	708
403	1020	996	768	696	512	500
406	1108	940	1306	1196	1128	964
409	1478	1460	1298	1192	1014	692
411	412	428	398	360	268	156
429	1618	1736	1914	1652	1126	924
432	1756	1748	2004	1588	1368	1096
435	1056	1020	1288	964	592	364
440	1460	1504	1644	1268	842	492
441	890	852	932	760	526	388
445	1624	1736	1636	1604	1298	1048
450	864	860	858	936	560	392
452	1820	1672	1782	1676	1584	1132
469	384	420	302	272	196	128

Link #	17:00	17:30	18:00	18:30	19:00	19:30
470	648	652	552	480	366	364
473	990	1232	952	856	674	584
475	798	844	674	624	488	400
477	992	1000	880	816	682	540
481	772	780	742	708	488	328
485	1564	1624	1404	1404	1208	888
509	968	1112	1022	856	666	536
513	806	1048	980	656	568	416
517	616	756	860	692	536	472
519	844	908	952	792	700	440
521	772	816	880	728	594	392
524	49a	504	576	532	432	304
525	34a	368	426	444	348	168
528	982	940	836	688	454	380
531	2544	2336	2254	2100	1604	1152
533	1706	1468	1334	1120	682	524
534	996	1056	784	648	422	296
535	620	656	670	624	326	256
537	502	572	584	572	336	188
540	560	612	556	600	388	256
543	972	1028	954	688	718	560
544	338	1356	1386	1176	1174	964
546	1194	1348	1228	1076	1040	800
549	1256	1192	1154	1028	932	744
550	77a	748	770	720	574	444
551	698	580	650	516	404	396
553	490	508	466	332	346	336
558	858	788	806	724	518	456
559	1034	1100	920	876	600	496
560	942	972	770	660	424	388
564	804	868	644	600	312	244
565	898	896	844	764	484	440
566	396	376	382	356	172	192
570	1202	1244	1198	1176	716	600
571	1632	1764	1624	1492	1186	1028
573	566	672	734	608	480	416
575	912	880	814	828	612	532
578	1268	1452	1226	1120	904	776
580	1818	1912	1756	1688	1394	996
581	1326	1464	1270	1268	1018	720
583	874	932	738	680	520	372
584	974	1092	902	744	630	480
587	1178	1220	1100	1016	832	756
595	1174	1124	1256	1204	902	760
598	1124	1184	1152	1120	808	660
599	998	1056	986	948	668	596
606	230	244	228	192	130	92
613	498	416	380	288	186	100
615	562	572	616	424	358	312
616	370	368	240	164	140	120
617	932	900	748	844	1108	872
619	358	372	334	348	258	240
627	658	692	632	640	558	472
629	1652	1636	1476	1404	1238	936
631	1306	1304	1116	1136	1042	764
632	1232	1276	1008	856	626	472
633	968	1052	848	780	482	348
635	620	644	586	560	552	564

Link#	17:00	17:30	18:00	18:30	9:00	19:30
636	536	540	604	432	396	332
637	644	600	584	436	462	412
639	808	864	842	876	668	720
641	1052	976	888	748	776	568
648	1332	1380	1226	1080	812	588
650	730	696	668	576	626	580
653	1338	1372	1186	1132	810	556
656	1118	1056	1088	872	720	492
659	1232	1280	1060	1052	782	700
662	1378	1436	1224	1308	1120	976
664	1384	1160	1240	1128	1236	1084
665	1400	1340	1348	1316	932	820
673	850	912	966	808	660	660
679	504	600	522	412	334	308
680	892	980	926	760	696	620
683	2056	1960	1962	1944	1578	1296
685	1232	1372	1270	1156	1022	844
687	790	844	798	780	720	584
691	722	780	782	724	676	700
693	786	740	830	804	654	808
695	1630	1544	1848	1840	1476	1528
698	1074	1164	1174	1232	934	1008
699	882	892	894	828	854	788
718	622	540	586	528	302	220
722	346	544	404	380	264	260
729	1092	1072	1108	1124	828	776
736	1554	1572	1518	1484	1104	952
737	1688	1616	1672	1620	1190	1052
739	1762	1712	1728	1728	1224	1088
740	2136	2196	2032	2200	1750	1720
744	990	940	914	808	762	700
748	564	520	524	448	296	220
749	892	856	938	864	742	576
750	1302	1348	1346	1144	1018	852
752	1498	1532	1578	1444	1428	1272
760	1226	1208	1174	1172	1130	1020
763	1480	1412	1460	1212	1092	1036
765	1254	1360	1300	1088	940	788
768	1614	1480	1514	1356	1488	1212
772	1726	1564	1794	1632	1960	1372
773	2708	2524	2956	2608	2340	1900
776	2148	2248	2088	1940	1536	1060
780	488	428	464	504	350	248
783	1518	1548	1712	1504	1214	892
784	2048	2016	2462	2032	1902	1512
788	1322	1432	1560	1436	1492	1392
791	948	644	602	512	366	372
793	1344	1296	1370	1476	1136	1192
796	536	532	544	508	366	284
800	642	632	584	520	364	340
804	1428	1496	1348	1384	1130	828
805	1148	1128	1088	1024	900	744
806	824	852	736	764	636	528
807	1156	772	1156	1048	972	864
809	1596	1424	1636	1396	1344	896
810	1208	1188	1010	892	734	476
al i	868	836	938	864	690	488
812	2122	1640	1456	1072	900	960

Link #	17:00	17:30	18:00	18:30	19:00	19:30
816	918	1060	1011	1140	1062	800
818	794	852	914	936	1040	840
829	644	584	552	528	374	308
830	630	524	428	404	294	216
831	658	596	482	424	318	256
832	764	732	620	516	382	316
833	604	536	462	396	310	244
835	272	268	304	264	186	192
836	396	460	378	300	248	188
837	438	488	450	364	300	192
838	320	420	380	352	248	216
839	784	700	690	664	488	364
847	2134	1984	2016	1888	1738	1416
859	1524	1640	1558	1536	1486	1476
861	8	a	78	500	508	468
866	1818	1844	1862	1732	1486	1288
870	1714	1620	1682	1572	1406	1108
874	2742	2672	2338	2188	2110	1856
878	1696	1808	1710	1700	1518	1260
883	1810	1756	1552	1428	1194	1068
884	1930	2096	1884	2044	1672	1624
889	1284	1392	1210	1156	1040	1068
896	646	636	516	468	374	248
902	428	464	372	396	282	248
905	1254	1140	1112	1028	820	676
908	930	1152	1178	1252	994	700
911	2072	1944	1854	1796	1668	1952
913	2012	1832	1786	1696	1542	1176
915	1302	1200	1264	1148	1224	1096
916	1332	1312	1320	1308	1102	928
919	1882	1988	1680	1744	1620	1252
920	1638	1692	1558	1592	1366	980
921	1146	1228	1174	1084	940	660
923	1260	1352	1260	1060	940	724
931	738	756	618	584	382	272
933	1300	1316	1048	956	790	568
934	1220	1216	990	904	748	480
935	1256	1368	1088	1036	798	512
936	698	760	658	632	694	460
938	1184	1148	898	844	764	536
940	644	748	638	652	528	540
941	762	820	758	820	666	616
944	1052	980	1038	1032	938	788
945	588	588	592	620	514	436
946	676	1304	884	740	408	508
948	1310	1408	1282	1208	918	776
951	1244	1088	1098	1116	1160	812
952	1238	1152	1164	1152	1204	844
954	220	276	244	188	200	140
955	592	536	560	552	552	384
956	1084	1028	930	896	980	748
957	392	324	340	336	300	212
959	1368	1372	1246	1328	1124	820
960	1006	916	900	824	906	644
961	1608	1624	1626	1540	1416	1160
962	1636	1480	1436	1200	1038	912
964	1306	1328	1274	1176	1216	1036
968	1638	1728	1592	1432	1310	1016

Link #	7:00	7:30	8:00	18:30	19:00	19:30
976	736	684	694	648	474	444
978	824	964	848	856	840	652
980	1018	1124	818	864	556	344
989	1378	1524	1370	1244	1090	800
990	1152	1144	1304	1088	592	348
992	916	956	794	856	816	668
994	818	856	818	804	608	428
996	666	604	468	420	518	480
1000	1360	1392	1272	1192	1282	1144
1012	1558	1512	1572	1564	1542	1280
1014	1374	1360	1350	1380	1356	1068
1018	1574	1800	1840	1652	1288	960
1024	1416	1432	1116	1024	796	684
1028	1248	1404	1060	924	784	632
1030	1750	1824	1542	1260	1234	916
1046	1634	1576	1366	1508	1088	836
1048	1258	1204	960	1104	778	604
1052	1176	1116	1078	1188	846	580
1054	1468	1408	1374	1252	1220	924
1057	702	720	690	636	520	448
1059	724	700	638	640	474	324
1062	1032	1004	1086	1024	714	528
1112	790	860	742	784	570	384
1117	588	604	576	572	510	392
1131	1000	1088	1142	1080	826	628
1135	812	752	668	676	828	892
1136	474	436	474	528	434	396
1176	6480	5642	5160	5027	4800	3376
1177	6480	5642	5160	0	0	0
1178	6264	5642	5160	0	0	0
1179	9194	8510	8334	0	0	0
1180	10166	9444	9258	0	0	0
1181	9936	9236	9052	0	0	0
1182	10730	10018	9814	0	0	0
1183	10490	10336	9558	0	0	0
1184	11054	10900	10156	7030	6330	3790
1185	11054	10900	10156	0	0	0
1186	11054	10900	10156	0	0	0
1187	10574	10418	9626	0	0	0
1188	10574	10418	9626	0	0	0
1189	10574	10418	9626	0	0	0
1190	10864	10778	9956	0	0	0
1191	10332	10288	9364	0	0	0
1192	10332	10288	9364	0	0	0
1193	10332	10288	9364	0	0	0
1194	10602	10548	10578	0	0	0
1195	9900	9900	9900	0	0	0
1196	10430	10402	10410	0	0	0
1197	9536	9484	9406	8660	7880	5820
1198	9536	9484	9406	0	0	0
1199	9942	9898	9870	8430	7080	5120
1200	9494	9464	9464	0	0	0
1201	9494	9464	9464	0	0	0
1202	9714	9644	9634	0	0	0
1203	8936	8894	8954	0	0	0
1204	9332	9288	9354	0	0	0
1205	8406	8396	8386	0	0	0
1206	7492	7450	7362	8210	6310	3780

Link #	17:00	17:30	18:00	18:30	19:00	19:30
1207	7492	7284	7362	0	0	
1208	8028	7828	7936	0	0	0
1209	8714	8722	8834	0	0	0
1210	8084	8082	8086	0	0	0
1211	9250	9260	9222	0	0	0
1212	8400	8400	8400	8840	7430	4280
1213	8400	8400	8400	0	0	0
1214	7310	7300	7278	0	0	0
1215	8448	8482	8574	0	0	0
1216	6654	6760	7092	0	0	0
1217	4452	4478	4584	0	0	0
1218	4452	4478	4584	4280	3659	2898
1219	4992	4734	4204	3617	3356	3049
1220	4992	4734	4204	0	0	0
1221	8936	8526	7840	0	0	0
1222	7728	7318	6712	0	0	0
1223	8838	8434	7798	8310	7220	4200
1224	8838	8434	7798	0	0	0
1225	9390	9014	8338	0	0	0
1226	8396	8076	7498	7970	6860	3930
1227	8396	8076	7498	0	0	0
1228	9046	8686	8178	0	0	0
1229	8874	8536	8008	0	0	0
1230	9532	9106	8688	0	0	0
1231	8876	8470	8082	0	0	0
1232	8054	7720	7340	8220	6790	3940
1233	8054	7720	7340	0	0	0
1234	8806	8300	7820	0	0	0
1235	9858	9416	8896	0	0	0
1236	9448	9030	8584	0	0	0
1237	9960	9612	9092	0	0	0
1238	9452	9064	8504	8200	6930	3900
1239	9452	9490	8504	0	0	0
1240	10200	10200	9124	0	0	0
1241	9388	9380	8452	9390	7820	4410
1242	9388	9380	8452	0	0	0
1243	10268	10240	9254	0	0	0
1244	9742	9716	8796	0	0	0
1245	10158	10132	9208	0	0	0
1246	9718	9696	8820	0	0	0
1247	9718	9696	8820	0	0	0
1248	10268	10236	9350	0	0	0
1249	9794	9768	8942	0	0	0
1250	9794	9768	8942	0	0	0
1251	10236	10216	9398	0	0	0
1252	10236	10216	9398	0	0	0
1253	9614	9712	8850	6990	6970	4150
1254	9614	9712	8850	0	0	0
1255	10224	10212	9520	0	0	0
1256	9116	9182	8564	0	0	0
1257	9636	9718	9090	0	0	0
1258	8128	8220	7708	0	0	0
1259	6040	6132	5498	0	0	0
1260	6040	6132	5498	4776	4898	4191
1262	228	211	211	234	286	327
1264	236	221	262	291	339	380
1266	473	492	545	545	496	463
1268	524	500	607	611	548	506

Link #	17:00	17:30	18:00	18:30	19:00	19:30
1270	691	661	695	631	490	376
1272	900	888	938	890	761	659
1274	450	420	380	540	460	360
1275	782	727	638	591	579	555
1277	930	868	773	763	827	866
1279	918	920	968	992	1008	1026
1281	631	627	715	707	627	572
1283	857	851	794	754	709	668
1286	1115	1107	1103	1046	924	825
1288	1830	1731	1466	1524	1890	2153
1289	2241	2294	2478	2517	2444	2406
1291	1221	1239	1171	1091	974	866
1292	991	961	872	798	719	640
1293	173	154	177	163	122	91
1294	654	652	628	597	551	510
1296	820	770	770	650	560	320
1298	404	394	325	276	223	173
1299	500	560	610	630	490	370
1301	807	807	697	630	569	504
1303	524	516	475	423	349	280
1305	439	431	402	390	390	388
1307	471	461	425	404	394	378
1309	620	500	570	650	520	230
1311	1006	1024	992	868	641	4 4 6
1312	1508	1492	1432	1412	1416	1412
1313	2085	2077	2296	2420	2515	2623
1314	2928	2868	3171	2348	2300	2200
1316	974	932	924	916	920	920
1318	792	782	762	717	640	573
1320	563	567	599	610	610	612
1322	290	360	330	360	270	190
1324	270	260	280	320	310	230
1326	530	500	510	380	350	200
1328	406	414	465	469	437	414
1330	220	180	170	140	100	80
1332	394	392	400	419	455	488
1334	534	543	575	555	586	433
1336	882	894	898	876	819	772
1338	1166	1178	1136	1032	851	697
1340	1139	1181	1297	1303	1213	1155
1342	1305	1307	1480	1109	1000	900
1344	2597	2482	2155	1468	1300	1200
1346	3728	3659	3485	3499	3679	3804
1347	1113	1115	1087	1010	874	757
1349	550	580	540	520	420	250
1351	650	610	680	640	550	320
1353	660	570	680	630	520	300
1355	591	579	479	410	343	2 7 6
1357	1129	1117	1074	1034	988	946
1359	510	580	510	450	470	230
1361	750	710	620	660	610	350
1363	880	860	800	850	770	430
1365	418	416	412	398	378	360
1367	550	540	530	510	580	370
1369	441	449	457	465	467	473
1371	610	500	670	700	730	420
1373	520	536	524	526	540	551
1386	692	836	708	696	634	448

Link #	17:00	17:30	18:00	18:30	19:00	19:30
1388	982	932	988	864	832	576
1396	418	332	390	384	242	208
1429	1352	1324	1156	1080	920	892
1431	232	232	252	232	146	128

Appendix T: The TO_QOD5 Program

This program reads the demand data files for any given set of half hour time slices and produces input file number five for the QUEENSOD program. The user provides an input file that specifies which time slices files are to be used in the construction of the input file. Thus, the user may create input files for multiple time slice QUEENSOD simulation runs.

The C++ source code for the program is given below.

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>
#include <io.h>

#define MAXNUMOFLINKS 3500
// #define MAXNUMOFFLOWS 500

void gettsdata();
void getlinkfile();
void getlinkdata();
int getflowdata(int j, int k);
//void getFlowFileData(int k);
void putoutline(int k);
void getoutfile();
int getduration();
char dummystring[121], slicefile[40][13];
char linkfile[13];
char outfile[15];
int sliceFlow[MAXNUMOFLINKS];
int j, k, numofslices, upnode, downnode, freeflow, capacity, numoflanes;
int flow, numoflinks, linknum, slicelength;
//int theFlows[MAXNUMOFFLOWS];
float length;
FILE *out, *linkin, *flowin;
struct linkData {
    int aLinkNum;
    int totCap;
    float FFS;
}; struct linkData theLinks[MAXNUMOFLINKS];

void main() // MAIN      MAIN      MAIN      MAIN      MAIN      MAIN
{
    long int sliceTime;
    gettsdata();
    slicelength = getduration();
    getlinkfile();
    getlinkdata();
    getoutfile();
    for (j = 0; j < numofslices; j++) {
        out = fopen(outfile, "a");
        sliceTime = ((j + 1) * slicelength);
        fprintf(out, "%ld %d\n", sliceTime, (j + 1));
        for (k = 0; k < numoflinks; k++) {
            if (k % 10 == 0) {
                clrscr();
                printf("\n\nProcessing Time Slice File %s.", slicefile[j]);
                printf("\n\nProcessing Link Number %d.", k);
            }
            flow = getflowdata(j, k);
            fprintf(out, "%5d %5d %6d %7.0f %7.0f\n", theLinks[k].aLinkNum, \
                flow, theLinks[k].totCap, theLinks[k].FFS, theLinks[k].FFS);
        }
    }
}
```

```

        }
        fclose(out);
    }
    clrscr();
    printf("\n\nThe output for this run will be found in %s.", outfile);
} // END OF MAIN          END OF MAIN          END OF MAIN          END OF MAIN

int getduration()
{
    printf("\nEnter the length of the time slices: ");
    scanf("%d", &slicelength);
    return slicelength;
}

void gettsdata(void)
{
    char tsfilename[12];
    FILE *tsnamein;
    int i;

    clrscr();
    printf("\nEnter the name of the file with the time slice files: ");
    scanf("%s", tsfilename);
    if ((tsnamein = fopen(tsfilename, "rt")) == NULL) {
        printf ("Time slice file list file not opened!");
        exit(1);
    }
    fscanf(tsnamein, "%d", &numofslices);
    if (numofslices > 40) {
        printf("\n\nERROR - The maximum number of time slices is 40");
        exit(1);
    }
    fgets(dummystring, 80, tsnamein);
    for(i = 0; i < numofslices; i ++){
        fscanf(tsnamein, "%s %s" , dummystring, slicefile[i]);
        fgets(dummystring, 80, tsnamein);
    }
    fclose(tsnamein);
} // END OF GETTSDATA          END OF GETTSDATA          END OF GETTSDATA

void getoutfile(void)
{
    clrscr();
    printf("\nEnter the desired name of the output file. ");
    scanf("%s", outfile);
    if ((out = fopen(outfile, "w")) == NULL) {
        printf ("\n\nOutput file not opened!");
        exit(1);
    }
    fprintf(out, "Link Flow file for QUEENSOD\n");
    fprintf(out, " %d %d %d %d\n", numofslices, slicelength,\
        numoflinks, numoflinks);
    fclose(out);
}

void getlinkfile(void)
{
    int k;
    clrscr();
    printf("\nEnter the INTEGRATION file name with the link data. ");
    scanf("%s", linkfile);
    if ((linkin = fopen(linkfile, "rt")) == NULL) {
        printf ("\n\nLink data input file not opened!");
        exit(1);
    }
    fgets(dummystring, 80, linkin);
}

```

```

    fscanf(linkin, "%d", &numoflinks);
    if (numoflinks > MAXNUMOFLINKS) {
        printf("You may only have %d links", MAXNUMOFLINKS);
        exit(0);
    }
    fclose(linkin);
}

void getlinkdata()
{
    char ch2;
    int m;
    linkin = fopen(linkfile, "r");
    fgets(dummystring, 80, linkin);
    fgets(dummystring, 80, linkin);
    linknum = 0;
    for (m = 0; m < numoflinks; m++) {
        fscanf(linkin, "%d %d %d %f %d %d %d", &linknum, &upnode, \
            &downnode, &length, &freeflow, &capacity, &numoflanes);
        ch2 = 0; while(ch2 != '\n') ch2 = fgetc(linkin);
        if (freeflow == -1) freeflow = 10;
        theLinks[m].aLinkNum = linknum;
        theLinks[m].totCap = capacity * numoflanes;
        theLinks[m].FFS = ((length/freeflow) * 3600);
        if (theLinks[m].FFS == 0) theLinks[m].FFS = 1;
    }
    fclose(linkin);
}

int getflowdata(int j, int k)
{
    int flowlinknum, flowfromfile;

    if ((flowin = fopen(slicefile[j], "rt")) == NULL) {
        printf("Flow data file %s not opened.", slicefile[j]);
        exit (1);
    }
    fgets(dummystring, 80, flowin);
    while (!feof(flowin)) {
        fscanf(flowin, " %d %d ", &flowlinknum, &flowfromfile);
        if (flowlinknum == theLinks[k].aLinkNum) {
            fclose(flowin);
            if (flowfromfile == 0) return -1;
            return(flowfromfile);
        }
    }
    fclose(flowin);
    return (-1);
}

```

Appendix U: The TO-INT3 Program

This program reads the signal control data file (see Appendix N) that contains the signal timing plans for all of the signalized intersections in the network. The output of the program is input file number three of the INTEGRATION program. The program contains the parameters of what signal timing plan is used during each period of the time slice. The program outputs a file that contains a total of 14 signal timing plans, one for each hour of the 14 hour simulation that was planned. Many of these signal timing plans are the same since the plans do not necessarily change of the hour.

The C++ source code for the program is given below.

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <conio.h>

#define LOSTTIME 3
#define SIGOPTIME 3600

int getplannum(int);
void openfiles(void);
int calcmintime(int * mintime);
void getsigdata(void);
void writedata(void);

char sigtimfile[12], sigstring[] = "Signal", dummystring[81];
char startline[81], outfile[12];
int hour, plannum, signum, cyclength, totmintime, offset, numofphases;
int i, j, phase[5], greentime[5], mintime[5], currplan;
FILE *signin, *out;

void main(void)

    openfiles();
    for (hour = 1; hour < 14; hour ++ ) {
        out = fopen(outfile, "a");
        fprintf(out, "%d\n" , hour);
        fclose(out);
        clrscr();
        printf("\nProcessing hour number %2d of 14", hour);
        plannum = getplannum(hour);
        signin = fopen(sigtimfile, "rt");
        while (!feof(signin)) {
            fscanf(signin, "%s", startline);
            if(strcmp(startline, sigstring) == 0) {
                fscanf(signin, "%d", &signum);
                fgets(dummystring, 80, signin);
                for (i = 0; i < 4; i ++ ) {
                    getsigdata();
                }
                totmintime = calcmintime(mintime);
                if (feof(signin)) break;
                writedata();
            }
            else fgets(dummystring, 80, signin);
        }
        fclose(signin);
    }
```



```

        clrscr();
        printf("\n\n\n\nThe output file %s has been created.', outfile);
    }

void openfiles(void)
{
    clrscr();
    printf("\n\nEnter the name of the file with the signal timing plans: ");
    scanf("%s", sigtimfile);
    if ((sigin = fopen(sigtimfile, "rt")) == NULL) {
        printf ("Signal timing file not opened!");
        exit(1);
    }
    fclose (sigin);

    clrscr();
    printf("\n\nEnter the name of the desired output file: ");
    scanf("%s", outfile);
    if ((out = fopen(outfile, "w")) == NULL) {
        printf ("Output file not opened!");
        exit(1) ;
    }
    fprintf(out, "Signal timing plans\n");
    fprintf(out, "\num of signals\    14 3600\n");
    fclose (out);
}

void getsigdata(void)
{
    char plan[6];
    fscanf(sigin, "%s %d %d", plan, &currplan,\
        &numofphases);
    if (currplan == plannum) {
        fscanf(sigin, " %d %d", &cyclength, &offset);
        for (j = 0; j < numofphases; j ++){
            fscanf(sigin, "%d %d %d", &phase[j],
                &greentime[j], &mintime[j]);
        }
        fgets(dummystring, 80, sigin);
        return;
    }
    else fgets(dummystring, 80, sigin);
}

int calcmintime(int * mintime)
{
    int m, total = 0;
    for (m = 0; m < numofphases; m ++){
        total += *(mintime + m);
    }
    return total;
}

int getplannum(int hour)
{
    int plannumb;
    if (hour == 1) plannumb = 2;
    else if (hour == 2) plannumb = 2;
    else if (hour == 3) plannumb = 2;
    else if (hour == 4) plannumb = 2;
    else if (hour == 5) plannumb = 3;
    else if (hour == 6) plannumb = 3;
    else if (hour == 7) plannumb = 3;
    else if (hour == 8) plannumb = 3;
    else if (hour == 9) plannumb = 3;
    else if (hour == 10) plannumb = 4;
}

```

```

else if (hour == 11) plannumb = 4;
else if (hour == 12) plannumb = 4;
else if (hour == 13) plannumb = 4;
else if (hour == 14) plannumb = 3;
else (printf("\n\n\nERROR 1"); exit(1) ;)
return plannumb;
}

void writedata(void)
{
    out = fopen(outfile, "a");
    fprintf(out, "%5d %4d %4d 120 %3d %3d", Signum, cyclength,\
        totmintime, offset, numofphases);
    for (j = 0; j < numofphases; j ++){
        fprintf(out, " %3d %3d", \
            greentime[j] - LOSTTIME, LOSTTIME);
    }
    fprintf(out, " %d \n", SIGOPTIME);
    fclose(out);
}

```