

UC San Diego

UC San Diego Electronic Theses and Dissertations

Title

Understanding the Relationship Between Prerequisite Course Proficiency and Student Success in Computer Science

Permalink

<https://escholarship.org/uc/item/9rg750h9>

Author

Krause-Levy, Sophia Louisa

Publication Date

2023

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA SAN DIEGO

Understanding the Relationship Between Prerequisite Course Proficiency and Student Success in
Computer Science

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy

in

Computer Science

by

Sophia Louisa Krause-Levy

Committee in charge:

William G. Griswold, Co-Chair
Leo Porter, Co-Chair
Christine Alvarado
Beth Simon

2023

Copyright

Sophia Louisa Krause-Levy, 2023

All rights reserved.

The Dissertation of Sophia Louisa Krause-Levy is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2023

DEDICATION

I want to thank my mother Marcia for being such an amazing, brilliant and strong woman and role model and teaching me the importance of balancing health and work.

I want to thank my father Dan for always being available to help me and for teaching me how to be creative and silly while also maintaining a good work ethic. Thank you for always indulging my inquisitive mind.

I want to thank my step-father Roy for always being excited and enthusiastic at my progress and introducing me to UCSD. Without you, I never would have applied. Your confidence in me has helped me continue to remain confident in myself.

Finally, I want to thank my partner Jeff for introducing me to computer science and believing that I was capable of succeeding, clearing up any moments of doubt I had about myself. Thank you for showing me the brilliance and creativity of computer science and always pushing me to want to learn more.

I would not be where I am today without any of you. Thank you all for your endless support and believing in my abilities.

Love,

Sophia

TABLE OF CONTENTS

Dissertation Approval Page	iii
Dedication	iv
Table of Contents	v
List of Figures	ix
List of Tables	x
Acknowledgements	xiii
Vita	xvii
Abstract of the Dissertation	xix
Chapter 1 Introduction	1
1.1 Motivation	1
1.2 The Types of Prerequisite Courses	3
1.3 The Purpose of Prerequisite Courses	4
1.4 Definitions	5
1.5 The Pitfalls of Formal Prerequisite Requirements	6
1.6 Thesis Statement	6
1.7 Acknowledgements	7
Chapter 2 Background and Related Work	8
2.1 Student Understanding in Introductory Programming Courses	8
2.2 The Role of Prerequisite Courses	9
2.2.1 The Role of Prerequisite Courses Outside of Computing	9
2.2.2 The Role of Prerequisite Courses In Computing	11
2.2.3 The Effect of Time on Prerequisite Knowledge	12
2.2.4 “Covering” Course Content is Not Enough	13
2.3 Prior Results on Prerequisites in Computing	14
2.4 Acknowledgements	15
Chapter 3 Study Context	16
3.1 Study Context for Chapters 4, 5, and 6	16
3.1.1 Course Descriptions	16
3.1.2 Course Grading	20
3.2 Study Context for Chapter 7	21
3.3 Acknowledgements	21
Chapter 4 [Causes] Exploring the Link Between Prerequisites and Performance in Advanced Data Structures	23

4.1	Study Design	24
4.1.1	Hypotheses	25
4.1.2	Population	25
4.1.3	Data Collection and Cleaning	27
4.2	Results	28
4.2.1	H-PL: Students Learn in Prerequisites	28
4.2.2	H-PT: Prerequisite Learning Fades	30
4.2.3	H-AP: Taking Alternatives to Primary Prerequisites Diminishes Student Success	30
4.2.4	H-OF: Factors Other than Prerequisite Learning Boost Student Performance	32
4.3	Limitations and Risks to Validity	35
4.4	Discussion	36
4.5	Summary	38
4.6	Acknowledgements	38
Chapter 5	[Impact] A Demographic Analysis on Prerequisite Preparation in an Advanced Data Structures Course	39
5.1	Preliminary Results	40
5.2	Our Study	40
5.3	Study Design	42
5.3.1	Research Questions	42
5.3.2	Population	42
5.3.3	Data Collection and Cleaning	43
5.4	Results	44
5.4.1	RQ1: Incoming Prerequisite Course Proficiency by Demographic	44
5.4.2	RQ2: Comparing Demographic Pairs' Prerequisite Grades	46
5.4.3	RQ3: Prerequisite Course Proficiency's Impact on Student Outcomes ..	47
5.4.4	RQ4: Comparing Prerequisite Course Proficiency by Number of Prerequisites Taken at Our University	49
5.5	Discussion	50
5.5.1	BLNPI students.	50
5.5.2	Transfer students.	50
5.5.3	Limitations and Threats to Validity	52
5.6	Summary	52
5.7	Acknowledgements	53
Chapter 6	[Extent, Impact, Cause] An Evaluation of Prerequisite Proficiency Across Multiple Computer Science Courses	54
6.1	Study Design	54
6.1.1	Hypotheses	54
6.1.2	Population	55
6.1.3	Data Collection and Cleaning	55

6.2	Results [H-Extent]: Low Incoming Prerequisite Proficiency Extends Across the Computing Curriculum	59
6.2.1	Incoming Prerequisite Proficiency	59
6.2.2	Common Misconceptions	60
6.2.3	Instructors Often Overestimate Students' Knowledge	66
6.3	Results [H-Impact]: Low Incoming Prerequisite Proficiency Impacts Student Outcomes with Unequal Outcomes for Students from Underrepresented Groups	67
6.3.1	The Relationship between Prerequisite Proficiency and Final Exam Scores	67
6.3.2	Incoming Prerequisite Course Proficiency by Demographic	67
6.3.3	Comparing Demographic Pairs' Prerequisite Grades	67
6.3.4	Differences in Student Outcomes	71
6.4	Results [H-Causes]: Possible Causes of Low Incoming Prerequisite Proficiency	71
6.4.1	Time Between Courses	71
6.4.2	Taking Alternatives to Primary Prerequisites Inside and Outside of Our University	76
6.4.3	Other Factors than Prerequisite Learning Impact Student Performance ..	76
6.5	Discussion	78
6.6	Limitations and Risks to Validity	78
6.7	Discussion	81
6.8	Summary	82
6.9	Acknowledgements	82
Chapter 7	[Cause] Instructor Perspectives on Prerequisite Courses in Computing	83
7.1	Study Design	84
7.1.1	Research Questions	85
7.1.2	Course Context	85
7.1.3	Sample Selection	85
7.1.4	Interview Method	85
7.1.5	Data Collection and Cleaning	86
7.1.6	Details of Analysis	87
7.2	Results	87
7.2.1	RQ1: Purpose of Prerequisite Courses	88
7.2.2	RQ2: Is the Purpose being Fulfilled?	90
7.3	Discussion	94
7.3.1	Comparison to Walker's Purposes	94
7.3.2	Limitations and Threats to Validity	95
7.4	Summary	95
7.5	Acknowledgements	96
Chapter 8	Discussion	97
8.1	Extent	97
8.1.1	Incoming Prerequisite Course Proficiency	97
8.2	Impact	99
8.2.1	Course Outcomes	99

8.2.2	Underrepresented Groups	99
8.3	Causes	101
8.3.1	Alternative Prerequisites Inside and Outside of Our Institution	102
8.3.2	Identifying Other Factors	102
8.3.3	The Instructor Perspective	103
8.4	A Need to Focus on (the Benefits of) Learning	105
8.5	Ideas for Instructors	107
8.6	Acknowledgements	108
Chapter 9	Conclusion	111
9.1	Acknowledgements	114
Appendix A	Preterm Survey Questions	115
A.1	ADS Pre-term Test Questions	115
A.2	CS1.5 Pre-Term Test Questions	129
A.3	CS2 Pre-Term Test Questions	136
A.4	ORG Pre-Term Test Questions	145
Bibliography		156

LIST OF FIGURES

Figure 3.1.	Prerequisite structure for courses in this dissertation.	16
Figure 3.2.	The grading breakdown for CS1.5, CS2, ORG and ADS courses. Note that 50% or more of each course grade allows for collaboration or awards effort.	20
Figure 4.1.	Scatter plot of student's score on the Pre-Term Test Score to their Prerequisite GPA and Non-Prerequisite GPA. Note that all of the data collected was normalized via z-score which accounts for the negative and positive range of values along the x and y axes.	29
Figure 4.2.	Scatter plot of student's score on the Final Exam to their Prerequisite GPA and Non-Prerequisite GPA. Note that all of the data collected was normalized via z-score which accounts for the negative and positive range of values along the x and y axes.	33
Figure 6.1.	Distribution of scores on the pre-term test including students who are and are not included in the data for this study.	60
Figure 6.2.	The percentage of students who got each question correct on the pre-term test.	60
Figure 6.3.	(a) The percentage of students to get each question correct on the pre-term test correct (Actual) and the instructors' estimation of the percentage of students who would get each question correct (Estimate). (b) The same graph highlighting the question students' struggled with the most.	66
Figure 6.4.	Scatter plot of student's score on the pre-term test to their outcomes in the course.	69
Figure 6.5.	The difference in average prerequisite course grades. + and - for the mean and median diff indicate whether women, BLNPI, FirstGen, and transfer students had higher or lower scores. A star (*) next to the course name indicates a significant correlation for $\alpha = 0.05$	72
Figure 6.6.	The average pre-term test score, final exam score and course grade broken down by course and demographic. A star (*) next to the course name indicates a significant correlation for $\alpha = 0.05$	73
Figure 8.1.	Distribution of scores on the pre-term test.	98
Figure 8.2.	The grading breakdown for CS1.5, CS2, ORG and ADS courses. Note that 50% or more of each course allows for collaboration or awards effort.	103
Figure 8.3.	Mean and median scores on the pre-term test, final exam and course grade.	110

LIST OF TABLES

Table 3.1.	Participation was solicited for instructors typically teaching following courses.	22
Table 4.1.	Majors breakdown of students as a percentage. Double majors are counted twice. Columns may not add up 100% due to rounding.	26
Table 4.2.	Non-prerequisite courses taken by ADS students.	26
Table 4.3.	Pearson correlations between precursor course measures and pre-term test score. A star (*) indicates a significant correlation for $\alpha = 0.05$	29
Table 4.4.	Williams's Test for whether the correlations of the two precursors to test score (Table 4.3, under heading "All") are different. A star (*) indicates significance for $\alpha = 0.05$	29
Table 4.5.	Pearson correlation for performance in prerequisites minus pre-term test score with time since courses taken. A star (*) indicates a significant correlation for $\alpha = 0.05$	30
Table 4.6.	Pearson correlations for performance in course with a comparison to final exam score and ADS course grade: correlation with p-values. A star (*) indicates a significant correlation for $\alpha = 0.05$	31
Table 4.7.	Multiple linear regression statistics for dependent variables of final exam score and ADS course grade. A star (*) indicates a significant correlation for $\alpha = 0.05$	31
Table 4.8.	Steigler's z-test for whether the main-track and alternate-track correlations are different. A star (*) indicates significance for $\alpha = 0.05$	32
Table 4.9.	Williams's Test for whether precursors are differently correlated to outputs. A star (*) indicates significance for $\alpha = 0.05$	33
Table 5.1.	Number of participants (N) and percentage of Women; Black/Latinx/Native American/Pacific Islander (BLNPI); Transfer; First-generation (FirstGen) students.	43
Table 5.2.	Performance details on the pre-term test. + and - for the mean and median diff indicate whether women, BLNPI, FirstGen, and transfer students have higher or lower scores. A star (*) indicates a significant correlation for $\alpha = 0.05$ according to a Holm-Bonferroni-adjusted Mann-Whitney U test. .	45

Table 5.3.	Holm-Bonferroni-adjusted Mann-Whitney U test for differences in prerequisite course grades based on transfer and BLNPI status broken down by prerequisite course. A star (*) indicates a significant correlation for $\alpha = 0.05$.	47
Table 5.4.	Spearman correlations with Holm-Bonferroni-adjusted p-values between performance on pre-term test and ADS final exam score and course grade. A star (*) indicates a significant correlation for $\alpha = 0.05$.	48
Table 5.5.	Performance details on the ADS final exam and ADS course grade by demographic. A star (*) indicates a significant correlation for $\alpha = 0.05$ according to a Holm-Bonferroni-adjusted Mann-Whitney U test.	51
Table 6.1.	Majors breakdown.	55
Table 6.2.	Number of participants (N) and percentage of Women; Black/Latinx/Native American/Pacific Islander (BLNPI); Transfer; First-generation (FirstGen) students.	56
Table 6.3.	Non-prerequisite courses taken by students.	58
Table 6.4.	Low, medium and high groups on the Pre-Term Test.	59
Table 6.5.	Question 6 on the CS1.5 Pre-Term Test.	61
Table 6.6.	Question 7 on the CS1.5 Pre-Term Test.	62
Table 6.7.	Question 7 on the CS2 Pre-Term Test.	63
Table 6.8.	The program for Questions 6-8 on the ORG Pre-Term Test.	64
Table 6.9.	Question 6 on the ORG Pre-Term Test.	65
Table 6.10.	Question 7 on the ORG Pre-Term Test.	65
Table 6.11.	Spearman correlations for pre-term test score with a comparison to final exam score and course grade: correlation with p-values. A star (*) indicates a significant correlation for $\alpha = 0.05$.	68
Table 6.12.	Mann-Whitney U test for differences in pre-term test scores by demographic. + and - for the mean and median diff indicate whether women, BLNPI, FirstGen, and transfer students had higher or lower scores. A star (*) indicates a significant correlation for $\alpha < 0.05$.	70

Table 6.13.	Mann-Whitney U test for differences in final exam scores by demographic: Women and Men; BLNPI and non-BLNPI; FirstGen and non-FirstGen; and Transfer and non-Transfer. star (*) indicates a significant correlation for $\alpha < 0.05$	74
Table 6.14.	Mann-Whitney U test for differences in course grades by demographic: Women and Men; BLNPI and non-BLNPI; FirstGen and non-FirstGen; and Transfer and non-Transfer. star (*) indicates a significant correlation for $\alpha < 0.05$	75
Table 6.15.	Pearson correlation for performance in prerequisites minus pre-term test score with time since courses taken. A star (*) indicates a significant correlation for $p = 0.05$	76
Table 6.16.	Performance details by taking the immediate predecessor at our institution. + and - for the mean and median diff indicate whether Alt-Track students had higher or lower scores. A star (*) indicates a significant correlation for $\alpha < 0.05$ according to a Mann-Whitney U test.	77
Table 6.17.	Spearman correlations for performance in course with a comparison to final exam score and course grade: correlation with p-values. A star (*) indicates a significant correlation for $p = 0.05$	79
Table 6.18.	Multiple linear regression statistics for dependent variables of final exam score and course grade. A star (*) indicates a significant correlation for $p = 0.05$	80
Table A.1.	ADS Pre-Term Test Questions	115
Table A.2.	CS1.5 Pre-Term Test Questions	129
Table A.3.	CS2 Pre-Term Test Questions	136
Table A.4.	ORG Pre-Term Test Questions	145

ACKNOWLEDGEMENTS

I want to thank my advisors, Bill Griswold and Leo Porter for the opportunity to work with them and for helping me gain my enthusiasm for computer science education research. I also want to thank Christine Alvarado, Mia Minnes, Joe Politz, Beth Simon, Sooh Nam Liao, Sander Valster and Gerald Soosai Raj for their continued passion for improving computing education and inspiring me to conduct research in the field. In addition, I want to thank Mohan Paturi for believing in me and suggesting I should apply for my masters in computer science.

I would also like to thank everyone else who I have had the pleasure of working with including, but not limited to, Rachel Lim, Adrian Salguero, Caroline Sih, Ismael Villegas Molina, Hayden McTavish, Emma Hogan, Anshul Shah, Paul Cao, and Jelena Trajkovic.

Most importantly I want to thank my parents and my partner who have always been there for me. I would not be where I am today without them.

Chapter 1, in part, is a reprint of material as they appear in three publications: 1) Proceedings of the 51st ACM Technical Symposium on Computer Science Education (SIGCSE 2020). Sophia Krause-Levy, Sander Valstar, Leo Porter, and William G. Griswold. “Exploring the Link Between Prerequisites and Performance in Advanced Data Structures”, 2) Proceedings of the 53rd ACM Technical Symposium on Computer Science Education (SIGCSE 2022). Sophia Krause-Levy, Sander Valstar, Leo Porter, and William G. Griswold. “A Demographic Analysis on Prerequisite Preparation in an Advanced Data Structures Course”, 3) Proceedings of the 54th ACM Technical Symposium on Computer Science Education (SIGCSE 2023). Sophia Krause-Levy, Adrian Salguero, Rachel S. Lim, Hayden McTavish, Jelena Trajkovic, Leo Porter, and William G. Griswold. “Instructor Perspectives on Prerequisite Courses in Computing”. The dissertation author was the primary investigator and

Chapter 2, in part, is a reprint of material as they appear in three publications: 1) Proceedings of the 51st ACM Technical Symposium on Computer Science Education (SIGCSE 2020). Sophia Krause-Levy, Sander Valstar, Leo Porter, and William G. Griswold. “Exploring the Link Between Prerequisites and Performance in Advanced Data Structures”, 2) Proceedings

of the 53rd ACM Technical Symposium on Computer Science Education (SIGCSE 2022). Sophia Krause-Levy, Sander Valstar, Leo Porter, and William G. Griswold. “A Demographic Analysis on Prerequisite Preparation in an Advanced Data Structures Course”, 3) Proceedings of the 54th ACM Technical Symposium on Computer Science Education (SIGCSE 2023). Sophia Krause-Levy, Adrian Salguero, Rachel S. Lim, Hayden McTavish, Jelena Trajkovic, Leo Porter, and William G. Griswold. “Instructor Perspectives on Prerequisite Courses in Computing”. The dissertation author was the primary investigator and author of this paper.

Chapter 3 in part, is a reprint of material as they appear in three publications: 1) Proceedings of the 51st ACM Technical Symposium on Computer Science Education (SIGCSE 2020). Sophia Krause-Levy, Sander Valstar, Leo Porter, and William G. Griswold. “Exploring the Link Between Prerequisites and Performance in Advanced Data Structures”, 2) Proceedings of the 53rd ACM Technical Symposium on Computer Science Education (SIGCSE 2022). Sophia Krause-Levy, Sander Valstar, Leo Porter, and William G. Griswold. “A Demographic Analysis on Prerequisite Preparation in an Advanced Data Structures Course”, 3) Proceedings of the 54th ACM Technical Symposium on Computer Science Education (SIGCSE 2023). Sophia Krause-Levy, Adrian Salguero, Rachel S. Lim, Hayden McTavish, Jelena Trajkovic, Leo Porter, and William G. Griswold. “Instructor Perspectives on Prerequisite Courses in Computing”. The dissertation author was the primary investigator and author of this paper.

Chapter 4, in part, is a reprint of the material as it appears in the Proceedings of the 51st ACM Technical Symposium on Computer Science Education (SIGCSE 2020). Sophia Krause-Levy, Sander Valstar, Leo Porter, and William G. Griswold. “Exploring the Link Between Prerequisites and Performance in Advanced Data Structures”. The dissertation author was the primary investigator and author of this paper.

Chapter 5, in part, is a reprint of the material as it appears in the Proceedings of the 53rd ACM Technical Symposium on Computer Science Education (SIGCSE 2022). Sophia Krause-Levy, Sander Valstar, Leo Porter, and William G. Griswold. “A Demographic Analysis on Prerequisite Preparation in an Advanced Data Structures Course”. The dissertation author

was the primary investigator and author of this paper.

Chapter 6, in part, is currently being prepared for submission for publication of the material. Sophia Krause-Levy, Adrian Salguero, Rachel S. Lim, Jelena Trajkovic, Leo Porter, and William G. Griswold. The dissertation author was the primary investigator and author of this material.

Chapter 7, in part, is a reprint of the material as it appears in the Proceedings of the 54th ACM Technical Symposium on Computer Science Education (SIGCSE 2023). Sophia Krause-Levy, Adrian Salguero, Rachel S. Lim, Hayden McTavish, Jelena Trajkovic, Leo Porter, and William G. Griswold. “Instructor Perspectives on Prerequisite Courses in Computing”. The dissertation author was the primary investigator and author of this paper.

Chapter 8 in part, is a reprint of material as they appear in three publications: 1) Proceedings of the 51st ACM Technical Symposium on Computer Science Education (SIGCSE 2020). Sophia Krause-Levy, Sander Valstar, Leo Porter, and William G. Griswold. “Exploring the Link Between Prerequisites and Performance in Advanced Data Structures”, 2) Proceedings of the 53rd ACM Technical Symposium on Computer Science Education (SIGCSE 2022). Sophia Krause-Levy, Sander Valstar, Leo Porter, and William G. Griswold. “A Demographic Analysis on Prerequisite Preparation in an Advanced Data Structures Course”, 3) Proceedings of the 54th ACM Technical Symposium on Computer Science Education (SIGCSE 2023). Sophia Krause-Levy, Adrian Salguero, Rachel S. Lim, Hayden McTavish, Jelena Trajkovic, Leo Porter, and William G. Griswold. “Instructor Perspectives on Prerequisite Courses in Computing”. The dissertation author was the primary investigator and author of this paper.

Chapter 9 in part, is a reprint of material as they appear in three publications: 1) Proceedings of the 51st ACM Technical Symposium on Computer Science Education (SIGCSE 2020). Sophia Krause-Levy, Sander Valstar, Leo Porter, and William G. Griswold. “Exploring the Link Between Prerequisites and Performance in Advanced Data Structures”, 2) Proceedings of the 53rd ACM Technical Symposium on Computer Science Education (SIGCSE 2022). Sophia Krause-Levy, Sander Valstar, Leo Porter, and William G. Griswold. “A Demographic Analysis

on Prerequisite Preparation in an Advanced Data Structures Course”, 3) Proceedings of the 54th ACM Technical Symposium on Computer Science Education (SIGCSE 2023). Sophia Krause-Levy, Adrian Salguero, Rachel S. Lim, Hayden McTavish, Jelena Trajkovic, Leo Porter, and William G. Griswold. “Instructor Perspectives on Prerequisite Courses in Computing”. The dissertation author was the primary investigator and author of this paper.

VITA

- 2017 Bachelor of Science in Cognitive Science with a specialization in Human-Computer Interaction, University of California San Diego
- 2020 Master of Science in Computer Science, University of California San Diego
- 2023 Doctor of Philosophy in Computer Science, University of California San Diego

PUBLICATIONS

Sophia Krause-Levy, Adrian Salguero, Rachel S. Lim, Hayden McTavish, Jelena Trajkovic, Leo Porter, and William G. Griswold, “Instructor Perspectives on Prerequisite Courses in Computing”, *In Proceedings of the 54th ACM Technical Symposium on Computer Science Education (SIGCSE 2023)*, 277–283, 2023.

Rachel S. Lim, Sophia Krause-Levy, Ismael Villegas Molina, and Leo Porter, “Student Expectations of Tutors in Computing Courses”, *In Proceedings of the 54th ACM Technical Symposium on Computer Science Education (SIGCSE 2023)*, 437–443, 2023.

Sophia Krause-Levy, Rachel S. Lim, Ismael Villegas Molina, Yingjun Cao, and Leo Porter, “An Exploration of Student-Tutor Interactions in Computing”, *In Proceedings of the 27th ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE 2022)*, 435–441, 2022.

Sophia Krause-Levy, Sander Valstar, Leo Porter, and William G. Griswold, “A Demographic Analysis on Prerequisite Preparation in an Advanced Data Structures Course”, *In Proceedings of the 53rd ACM Technical Symposium on Computer Science Education (SIGCSE 2022)*, 661–667, 2022.

Sophia Krause-Levy, William G. Griswold, Leo Porter, and Christine Alvarado, “The Relationship Between Sense of Belonging and Student Outcomes in CS1 and Beyond”, *In Proceedings of the 17th ACM Conference on International Computing Education Research (ICER 2021)*, 29–41, 2021.

McKenna Lewis, Zhanchong Deng, Sophia Krause-Levy, Adrian Salguero, William G. Griswold, Leo Porter, and Christine Alvarado, “Exploring Student Experiences in Early Computing Courses during Emergency Remote Teaching” *In Proceedings of the 26th ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE 2021)*, 88–94, 2021.

Sander Valstar, Sophia Krause-Levy, Adrian Salguero, Leo Porter, and William G. Griswold, “Proficiency in Basic Data Structures among Various Subpopulations of Students at Different

Stages in a CS Program”, *In Proceedings of the 26th ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE 2021)*, 429–435, 2021.

Sophia Krause-Levy, Mia Minnes, Christine Alvarado, and Leo Porter, “Experience Report: Designing Massive Open Online Computer Science Courses for Inclusion”, *In Proceedings of the 26th ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE 2021)*, 95–101, 2021.

Sander Valstar, Caroline Sih, Sophia Krause-Levy, Leo Porter, and William G. Griswold. “A Quantitative Study of Faculty Views on the Goals of an Undergraduate CS Program and Preparing Students for Industry”, *In Proceedings of the 2020 ACM Conference on International Computing Education Research (ICER 2020)*, 113–123, 2020.

Sophia Krause-Levy, Sander Valstar, Leo Porter, and William G. Griswold, “Exploring the Link Between Prerequisites and Performance in Advanced Data Structures”, *In Proceedings of the 51st ACM Technical Symposium on Computer Science Education (SIGCSE 2020)*, 386-392, 2020.

Sophia Krause-Levy, Leo Porter, Beth Simon, and Christine Alvarado, “Investigating the Impact of Employing Multiple Interventions in a CS1 Course”, *In Proceedings of the 51st ACM Technical Symposium on Computer Science Education (SIGCSE 2020)*, 1082-1088, 2020.

Sander Valstar, Sophia Krause-Levy, Alexandra Macedo, William G. Griswold, and Leo Porter, “Faculty Views on the Goals of an Undergraduate CS Education and the Academia-Industry Gap”, *In Proceedings of the 51st ACM Technical Symposium on Computer Science Education (SIGCSE 2020)*, 577-583, 2020.

ABSTRACT OF THE DISSERTATION

Understanding the Relationship Between Prerequisite Course Proficiency and Student Success in
Computer Science

by

Sophia Louisa Krause-Levy

Doctor of Philosophy in Computer Science

University of California San Diego, 2023

William G. Griswold, Co-Chair

Leo Porter, Co-Chair

Instructors and institutions tend to treat students who have fulfilled their prerequisites for a course as if they are all equally well-prepared. Recent work in computing has shown that a substantial proportion of students are entering later courses with low prerequisite course proficiency, possibly hurting student outcomes. This dissertation explores (1) the **extent** of low prerequisite course proficiency throughout our CS curriculum, (2) the **impacts** of low proficiency on student outcomes, with a focus on students typically underrepresented in computing, and (3) possible **causes** leading to low proficiency as well as possible reasons why students are able to succeed in courses despite entering underprepared.

Regarding **extent:** we find that a substantial proportion of students, across multiple courses, enter courses with low prerequisite proficiency.

Regarding **impacts:** encouragingly, we find that student grades in prior courses are predictive of prerequisite proficiency coming into a course and student success in the course. We also find a relationship between students' incoming proficiency and their performance in the course. This suggests that the low prerequisite proficiency may be negatively impacting student outcomes. In terms of subpopulations, there is a disproportionate impact on students typically underrepresented in computing.

Regarding **causes:** in contrast with previous research, we find no harmful effects on prerequisite proficiency due to the passage of time or taking alternative prerequisites. In terms of instructor expectations, we find that instructors overestimate students' incoming prerequisite proficiency. Furthermore, we find that many instructors believe prerequisites are not fulfilling their purpose. They cited various reasons including the number of conflicting pressures instructors face with respect to ensuring students learn material expected for the next course (e.g., trying not to adversely impact students' time-to-degree, being well-liked by students, and balancing the time spent on teaching with other responsibilities).

Chapter 1

Introduction

1.1 Motivation

Computer science has a history of high failure rates and low retention rates [50, 76]. There is a long line of work examining factors that may influence student success and retention in computer science, including factors such as early exposure to computing [4, 34, 35, 75], motivation [78], self-efficacy [32, 53, 78], impostor phenomenon [56], sense of belonging [29], and achievement goals [81, 82]. A majority of these factors focus on student attitudes and beliefs rather than student content knowledge or specific skill sets. One exception to this is studies on the impact of prior programming experience before entering higher-education [34, 35, 75] on student success in computing courses.

Studies on the impact of prior programming experience have shown the importance of formal or informal prior programming experience before entering university [21, 23, 75, 78, 82]. Students who have not completed any formal computing coursework before joining the major leave at higher rates [6]. This is especially important as Black and Latinx students, as well as students from lower economic status¹ and women, often do not have the same level of access to the prior programming experiences that their majority counterparts have before

¹In our country, it is common for some students to enter university after completing a 2-year college program that fulfills General Education requirements and, typically, a couple computing courses. We refer to these students as “transfer” students. Typically, transfer students are more socioeconomically diverse than those students entering university directly.

entering university [34, 35, 75]. Black, Latinx, Native American and Pacific Islander (BLNPI) students and women are traditionally underrepresented in computing compared to their majority counterparts who are White and Asian men [84]. In computing, women and BLNPI students are retained at a lower rate than their majority counterparts [83] and tend to have inequitable outcomes during their time studying computing [4, 58, 59].

In 2010, Robins defined the term “Learning Edge Momentum (LEM)”, a hypothesis for why students tend to either succeed or struggle in introductory computer science courses [55]. The argument is that students who understand a concept are more likely to understand later concepts that build on the earlier concept, with the effect compounding over time. A misunderstanding on an early concept will lead to a higher probability of misunderstandings on later concepts. With this hypothesis, it is not surprising to see that prior programming experience benefits students in introductory programming courses. If students already had a first chance to learn the material, they are building on existing knowledge and they may be less likely to develop misunderstandings that can compound over time.

With the knowledge that early exposure to computing is a factor in student success and the hypothesis of Learning Edge Momentum for introductory programming courses, the question becomes: What impact does this have on students as they progress through the computing curriculum?

Recent work outside of computing suggests that there may be, in fact, a large impact. Muralidharan et al. studied 8th grade students’ performance in mathematics and Hindi in India and found that the majority of students had inadequate prerequisite knowledge of material taught in prior grade levels [41]. These students were struggling and falling behind. Muralidharan et al. also showed that an intervention targeting deficits in prerequisite knowledge significantly improved student outcomes. These results, combined with research in computing suggesting that students don’t know as much as instructors expect, led us to wonder if we would see similar results in computer science courses in the undergraduate computing curriculum.

At our institution a grade of a D is considered to be sufficient to fulfill a prerequisite

requirement (although if a student is majoring in computer science they must receive a grade of a C- or better for it to count towards their major requirements). This means that students who received grades of A, B, C or D are all able to continue on to the next course. Given that institutions tend to treat all students who have passed a course the same and that computer science has long prerequisite chains that rely heavily on previously taught concepts, we suspect that misconceptions, or lack of knowledge, from earlier computing courses may lead to significant problems in later courses. Based on LEM, we hypothesize that lower proficiency in course content in one class leads to a higher chance of struggling in later classes. Given that prior work on early exposure to computing further suggests that this may impact students who are typically underrepresented in computing more than their majority counterparts, we further hypothesize that poor proficiency has a large impact on students from underrepresented groups [34, 35, 75].

Early results from our research group suggested that this may in fact be the case [70]. The work of my colleague Valstar et al. began to test our hypothesis through the use of pre- and post-term tests of prerequisite knowledge in an upper-division data structures course to measure the degree to which students are coming into courses with insufficient prerequisite proficiency. It was found that the degree of preparation varies widely among students and that the gap in preparation translated to students having inequitable outcomes in the course [70]. These results are concerning as they suggest students may be entering subsequent courses without the expected prerequisite course proficiency and it may be impacting their outcomes in later courses. This led to further questions regarding the **extent**, **impacts**, and **causes** of low prerequisite course proficiency, which are explored in this dissertation.

1.2 The Types of Prerequisite Courses

Here we define the two types of prerequisites. For the purposes of this dissertation we will focus on required prerequisites.

1. **Required Prerequisites.** A required prerequisite is a course that must be taken in order

to advance to the subsequent course. There are two main types of required prerequisites: (1) A single prerequisite course that must be taken in order to take a subsequent course and (2) a list of courses that can fulfill a single prerequisite in order to take a subsequent course (e.g. being required to take a mathematics course but where one of several offered mathematics courses can satisfy the prerequisite requirement).

2. **Non-Required Prerequisites.** In some cases an instructor or course may have a list of recommended prerequisites that are not required but are suggested if you lack the specific background knowledge. In other cases there are courses that are not required but that a large percentage of students may take before taking the subsequent course, such as many pre-medical students take gross anatomy and histology courses before entering medical school even though it isn't a requirement [17].

There are also hidden prerequisites. A hidden prerequisite is something that is not defined as a formal prerequisite but nevertheless impacts students' performance. This includes a variety of things such as prior programming experience, motivation, self-efficacy and achievement goals [32, 53, 78, 81]. Although these impact student performance and are a form of prerequisite, we will be focusing on formal prerequisites in this dissertation. Hidden prerequisites have a separate and large body of literature that is mostly outside of the scope of this work.

1.3 The Purpose of Prerequisite Courses

Formal course prerequisites are an important part of the structure of an undergraduate computer science curriculum. Prerequisites are beneficial as they allow students to gain the necessary background knowledge and experience for subsequent courses and allow instructors to assume all students have some level of background knowledge on their course material allowing them to focus more on new material rather than having to re-explain all material relevant to their course. The ACM Computing Curricula Guidelines describe the purpose of prerequisites as “at

universities, course prerequisites attempt to ensure that a student has the necessary knowledge needed to advance to the next course level” [25].

In reality prerequisites can fulfill multiple roles in a curriculum. A list of the most common reasons for a prerequisite course are described by Walker [74]:

1. **“Content Prerequisites:** a course at one level may build on the background a student had developed earlier.
2. **Maturity Prerequisites:** a course may build on general student experience and maturity beyond any specific detailed content.
3. **Filtering:** prerequisites may provide evidence of student commitment or may limit enrollment in popular courses.
4. **Requirement Enforcement:** a program may list specific courses as requirements, and prerequisites highlight these requirements – independent of whether the topics are needed in a particular course.
5. **Historical Requirement:** years ago, some faculty determined a prerequisite – possibly to meet a specific need.”

1.4 Definitions

We define a **prerequisite course** as a formally required course that students must take before enrolling in a subsequent course that usually imparts skills or knowledge that the subsequent course expects students to have mastered. Throughout this dissertation we will refer to a course requiring one or more prerequisites as a **subsequent course**. We define and measure **prerequisite course proficiency** as the degree to which students have mastered the concepts and skills taught in prerequisite courses.

1.5 The Pitfalls of Formal Prerequisite Requirements

While understanding that prerequisites play an important role in a curriculum, we must also understand that there are pitfalls. Formally required prerequisite courses often lead to a higher time to degree [6]. They restrict enrollment in later courses so students may be unable to take courses they are interested in until late in their degree [6, 74]. Having a large number of prerequisites can discourage students who are interested in a subject, especially if they find their interest later in their time at university. This can especially impact low-income students who cannot afford to stay at the university for additional quarters. Some of the main reasons students leave the major are due to the computing curriculum not providing enough flexibility, being too difficult or the length of the program [6].

1.6 Thesis Statement

*[Thesis Statement] Prior work primarily looks at one course, its prerequisites and its learning goals. We hypothesize the **extent** of low prerequisite course proficiency is much greater and these problems are endemic to the computing curriculum, extending across the computing curriculum and across institutions. We further hypothesize that as extent is pervasive so are the **impacts**, with both the extent and impacts being more significant for students from groups typically underrepresented in computing (women, Black, Latinx, Native American and Pacific Islander (BLNPI), first generation and transfer students). Finally, we believe there is no single source of the **cause** of low prerequisite proficiency or why students are able to make it far along into the computing curriculum with such low proficiency, it is rather many separate causes. We believe this may include the length of time between taking a course's prerequisites, taking the course itself or satisfying a course's prerequisite with an alternative to the primary identified course's prerequisites and that other factors, unrelated to subject matter knowledge, strongly affect a student's success in that course.*

These hypotheses boil down to the following research questions studied in this disserta-

tion:

- **Extent:** To what extent are students arriving in subsequent courses with low prerequisite proficiency?
- **Impacts:** How does prerequisite proficiency impact student success in subsequent courses, including measures of student learning in following courses? Moreover, does the impact of prerequisite proficiency differ for various groups of students?
- **Causes:** What are the causes of the problem (e.g., time between courses, misalignment of instructor expectations and student abilities, etc.)?

The following chapters are not organized according to extent, impacts and causes because they are often researched simultaneously. In the discussion and conclusion we will revisit these hypotheses/research questions, and summarize our key findings based on the **extent**, **impacts**, and **causes** of low prerequisite course proficiency.

1.7 Acknowledgements

Chapter 1, in part, is a reprint of material as they appear in three publications: 1) Proceedings of the 51st ACM Technical Symposium on Computer Science Education (SIGCSE 2020). Sophia Krause-Levy, Sander Valstar, Leo Porter, and William G. Griswold. “Exploring the Link Between Prerequisites and Performance in Advanced Data Structures”, 2) Proceedings of the 53rd ACM Technical Symposium on Computer Science Education (SIGCSE 2022). Sophia Krause-Levy, Sander Valstar, Leo Porter, and William G. Griswold. “A Demographic Analysis on Prerequisite Preparation in an Advanced Data Structures Course”, 3) Proceedings of the 54th ACM Technical Symposium on Computer Science Education (SIGCSE 2023). Sophia Krause-Levy, Adrian Salguero, Rachel S. Lim, Hayden McTavish, Jelena Trajkovic, Leo Porter, and William G. Griswold. “Instructor Perspectives on Prerequisite Courses in Computing”. The dissertation author was the primary investigator and author of this paper.

Chapter 2

Background and Related Work

This chapter discusses student understanding in introductory programming courses, prior work on prerequisites and the need to study the impact of prerequisite course proficiency in computing.

2.1 Student Understanding in Introductory Programming Courses

An important aspect of studying computing education is measuring how well students understand course concepts. In 1986 Soloway described the “Rainfall Problem” and general reasons for why students are struggling to learn computer science [65]. This allowed the computing education community to become more aware of the mismatch between instructor expectations and students’ actual abilities in introductory computing courses. These results have continued to replicate through today [33, 39, 46].

In 2001, McCracken et al. conducted a study at four universities to better understand the students’ level of programming competency in introductory programming courses by administering as a short lab-based assignment. The results were much lower than expected with the average score of 21% [39]. In 2011 Lister wrote a piece called “Ten Years after the McCracken Working Group” describing how the state of novice programmers has not changed [33, 39]. More recently, in 2016, Parker et al. sought to measure student understanding of course concepts through the

use of a validated instrument, finding similar results [46].

In this dissertation we build on these findings by examining the impact of low prerequisite proficiency and its alignment with instructors expectations in introductory programming courses and across three courses later in the CS curriculum (see Chapter 6).

2.2 The Role of Prerequisite Courses

Prerequisites are a necessary feature to any curriculum where concepts build from course to course and where there are desired assumptions about students' incoming knowledge and skills. Prerequisite courses are deeply ingrained in higher education, resulting in a number of studies assessing their impact. As a fundamental part of most curriculum designs, formally required prerequisite courses have been well-studied outside of computing and have just begun to be studied inside computing.

2.2.1 The Role of Prerequisite Courses Outside of Computing

The majority of prior studies outside of computing have used a quantitative approach to measure the effectiveness of prerequisites, examining the correlation between students' performance in a course and their performance in its prerequisite courses [2, 8, 24, 37, 43, 61, 73]. The majority of the studies find that prerequisites positively impact student success in the subsequent course [2, 17, 66]. In fact, many have found that prerequisite grades predict subsequent course grades [24, 36, 40, 73].

Several studies have involved examining how students who have and have not taken prerequisite courses perform in subsequent courses [17, 26, 54, 57, 60, 66, 79]. Others have explored the value of remedial courses in preparing students for subsequent courses [5] and examining the relationship between undergraduate prerequisite course grades and performance [11, 20, 40]. Furthermore, other studies have included additional measures in the correlations, trying to determine which factors have the most impact, such as gender [2, 37, 73], race [37, 73] and high school GPA [1, 17, 36, 60, 61, 79]. Many such studies have found a relationship between high

school GPA, cumulative GPA and student success in subsequent courses.

When Prerequisite are Not Required

In some cases institutions may recommend a prerequisite but not require it or update their prerequisite chain to include new course requirements. Several studies have been conducted to understand the possible impact of these prerequisite requirements on student outcomes.

Shaffer et al. classified material according to the depth of coverage in prerequisite courses and then asked questions on this knowledge in subsequent courses. Compared to students who did not take a prerequisite, students who took the prerequisite only showed superior performance on the most deeply covered material [62]. Sato et al. used this method, combined with a quantitative and qualitative analysis of student feedback, to determine the value of taking a recommended microbiology lecture class before a microbiology lab [60]. No significant difference in performance was found.

Soria and Mumpower conducted a study to try and understand the impact of requiring prerequisites for students entering an introductory composition course [66]. The university implemented mandatory prerequisites in Fall 2008 and the study compared student success before and after this requirement. It was found that required prerequisites positively affected student grades in the composition course, as well as retention rates, showing the importance of enforcing prerequisites. Interestingly, in biochemistry Wright et al. found that biochemistry course grades were not impacted by taking the recommended organic chemistry prerequisite [79].

Informal Prerequisites Across Programs

Mastery of prerequisite knowledge and preparation for courses is not strictly aligned with a course's listed prerequisites. In many cases students take courses before or in between starting a new educational program such as between high school and college. Studies have shown that students with prior formal prerequisite course experience outside of the given prerequisites also impacts student success [5, 17].

Aycaster followed the progress of developmental mathematics programs at five community colleges and found that students who took the developmental mathematics program had higher rates of retention and rates of graduation than those who did not take the developmental courses [5]. Similarly, Forester et al. studied the impact of premedical students completing a course with material they would later relearn in medical school and found students who took a histology or gross anatomy course before starting medical school were higher performers in the corresponding medical school courses [17].

When Multiple Courses Fulfill a Prerequisite

At universities, there are cases where multiple courses can satisfy a single given prerequisite for a follow-on course. This gives rise to the question, does the course students choose to use to fulfill their prerequisite requirement impact their success in the subsequent course? Studies across multiple disciplines have shown mixed results [11, 20, 24, 40, 54, 61], though the majority have found that choice of prerequisite course matters [11, 20, 24, 61].

2.2.2 The Role of Prerequisite Courses In Computing

In computer science, prerequisites can play many roles, including ensuring students have the proper background knowledge or sufficient maturity, that they display a certain degree of commitment to a program, or to convey program requirements [74]. In particular, the role of mathematics and introductory computing prerequisites and co-requisites have been well discussed [54, 74].

Computing is a particularly useful field in which to study prerequisites for several reasons. First, there are well-established curriculum guidelines from the Joint Task Force on Computing Curricula Association that provide knowledge areas that are helpful for identifying prerequisite chains [25]. Second, computing tends to have long prerequisite course chains. These chains may cause misunderstandings from an early course to have amplified effects throughout the long chain. There is also pressure to re-examine and shorten the prerequisite chains to reduce

time-to-degree, and to potentially improve the accessibility of computing.

Several studies have been conducted to measure the impact of prerequisites in computing in introductory courses. Reilly and Tomai studied the importance of students enrolling in concurrently or previously taken College Algebra before taking their first CS course (CS1) [54]. It was found that students taking a mathematics course concurrently with CS1 performed better than those not enrolled in a mathematics course. Tew et al. studied the impact of choosing between multiple introductory computing courses for the first course in a two-course introductory computing sequence [69]. Students could choose between Introduction to Computing, Computing for Engineers, and a media computation course. Tew et al. administered pre- and post-term tests in the second course of introductory computing sequence to study whether there were differences in incoming prerequisite proficiency based on the first course students chose. The results of the study showed no difference in outcomes for students who took different prerequisites, suggesting that the chosen prerequisite course didn't matter.

The work most related to this dissertation was conducted by Muralidharan et al. and Murphy and Tenenberg. Outside of computing, Muralidharan et al. studied 8th grade students' performance in mathematics and Hindi in India and found that the majority of students had inadequate prerequisite knowledge of material learned in prior grade levels [41]. These students were struggling and falling behind. Muralidharan et al. also showed that an intervention targeting prerequisite knowledge significantly improved student outcomes. In computing, Murphy and Tenenberg conducted a study that involved having students answer questions on data structures and found that students on average answered approximately half of the questions correctly [42].

2.2.3 The Effect of Time on Prerequisite Knowledge

Many factors may cause lower than expected incoming prerequisite knowledge. One such possible factor is long knowledge retention intervals, well studied in the education literature (see [12] for a literature review on this topic). Evidence points to large knowledge losses even during a short retention interval. After an extensive review of literature on knowledge retention,

Custers concluded that although different studies report vastly different results, there is support to the claim that about a third of knowledge is lost during the first year, half is lost after a few years, and loss levels off after that [12]. Custers also noted there is evidence for much worse retention for useless knowledge, citing Ebbinghaus's forgetting curve, which shows a knowledge loss of almost 80% after one month [12, 15].

In computing, a study on retention of CS1 core concepts at the start of CS2 found that students who take CS1 in fall and CS2 in spring have a knowledge loss of 4%, whereas students taking CS1 in spring and CS2 in fall have a loss of 15.1%. This difference can be explained by the summer vacation between the spring and fall terms [68]. In this dissertation we continue to explore this line of research by studying the impact of time between taking prerequisite courses and subsequent courses (see Chapter 4 and Chapter 6).

2.2.4 “Covering” Course Content is Not Enough

One of the roles of prerequisite courses is to ensure that students meet a certain baseline of knowledge before enrolling in their course [74]. This allows the instructor to build on top of students' prerequisite course proficiency in order to teach more advanced concepts. A study conducted by Shaffer et al. outside of computing, looked at the impact of prerequisites on student success in subsequent courses over the period of two quarters [62]. For this study the researchers looked at two sets of courses: 1) a genetics course which is a prerequisite of a microbiology course and 2) a human physiology course which is a prerequisite of a human anatomy course.

The researchers classified exam questions for the microbiology and human anatomy courses based on how the exam questions related to the prerequisites in terms of depth of coverage of the topics in the prerequisite courses into three categories of:

1. **“Very Familiar”**: content covered in depth in the prerequisite course
2. **“Familiar”**: content covered in the prerequisite course but not in depth
3. **“Not Familiar”**: content not covered in the prerequisite course

To do this, they looked at exam questions in the subsequent courses and grouped the questions by the above classifications. This was done by looking at the lecture slides of the prerequisite courses, as well as having students who had previously taken the class and instructors characterize a subset of the questions. The results showed that students scored higher on the “Very Familiar” questions but that there was no significant difference in performance between the questions that were deemed “Familiar” and “Not Familiar”. Shaffer et al. states that these results suggest that simply covering a topic in a course is not enough and that a level of depth is needed. It may be that some topics can simply be removed from courses to focus even more on topics deemed important and that more student-centered teaching methods should be integrated into courses.

2.3 Prior Results on Prerequisites in Computing

The role of prerequisite knowledge in student success has just begun to be explored in computer science. Our research lab conducted a preliminary study in 2019 [70]. Valstar et al. studied the prerequisite knowledge of 208 students in an upper-division data structures course through pre- and post-term tests, comparing them to final exam scores [70]. Prerequisite proficiency on entry to the course was far lower than anticipated. Nearly a third of students (29%) demonstrated low proficiency and only 27% demonstrated high proficiency. However, students improved their proficiency over the term by 8 percentage points on average, with over a quarter of the low-proficiency students improving to at least medium proficiency by the time of the final exam. A positive correlation between prerequisite course proficiency and success on the final exam was identified, indicating that students who are lagging behind in prerequisite course proficiency may continue to incur more gaps in their knowledge going forward in the CS program.

In this dissertation we expand on this work by 1) exploring possible factors for why students are able to succeed and move on to subsequent courses when they have low prerequisite

course proficiency, 2) exploring the extent and impact of low prerequisite course proficiency and its effect on various student demographics and 3) replicating this work at a larger scale across multiple courses.

2.4 Acknowledgements

Chapter 2, in part, is a reprint of material as they appear in three publications: 1) Proceedings of the 51st ACM Technical Symposium on Computer Science Education (SIGCSE 2020). Sophia Krause-Levy, Sander Valstar, Leo Porter, and William G. Griswold. “Exploring the Link Between Prerequisites and Performance in Advanced Data Structures”, 2) Proceedings of the 53rd ACM Technical Symposium on Computer Science Education (SIGCSE 2022). Sophia Krause-Levy, Sander Valstar, Leo Porter, and William G. Griswold. “A Demographic Analysis on Prerequisite Preparation in an Advanced Data Structures Course”, 3) Proceedings of the 54th ACM Technical Symposium on Computer Science Education (SIGCSE 2023). Sophia Krause-Levy, Adrian Salguero, Rachel S. Lim, Hayden McTavish, Jelena Trajkovic, Leo Porter, and William G. Griswold. “Instructor Perspectives on Prerequisite Courses in Computing”. The dissertation author was the primary investigator and author of this paper.

Chapter 3

Study Context

This chapter describes the context of the data sets used in this dissertation. Chapters 4, 5, and 6 are quantitative studies conducted at the University of California San Diego. Chapter 7 is a qualitative study conduct at the University of California San Diego and California State University, Long Beach.

3.1 Study Context for Chapters 4, 5, and 6

3.1.1 Course Descriptions

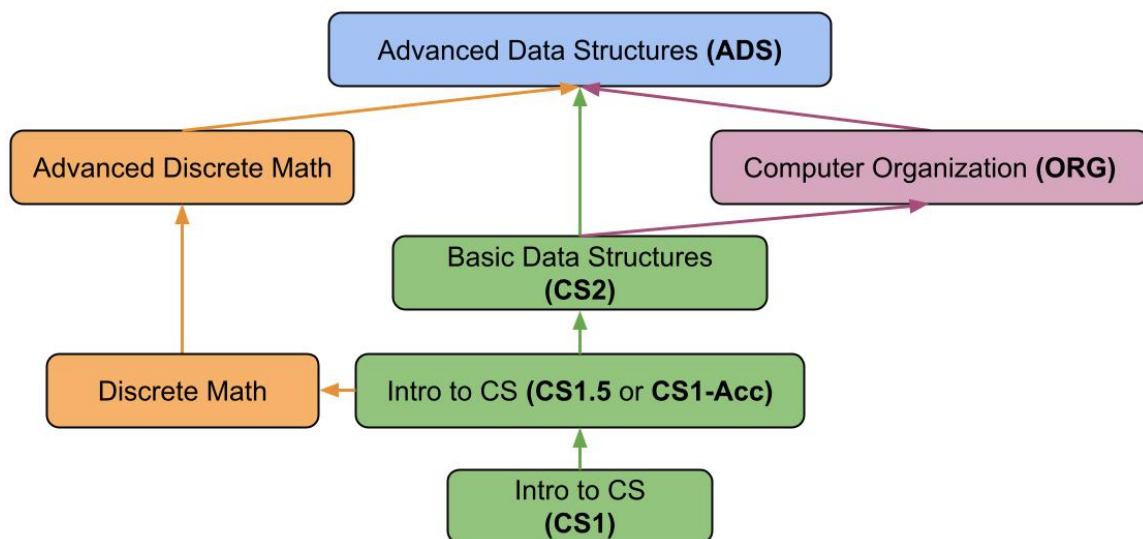


Figure 3.1. Prerequisite structure for courses in this dissertation.

These studies were carried out at the University of California San Diego, a large US

research university (R1) that operates on the quarter system. Courses meet for 10 weeks, immediately followed by a week of final exams. Figure 3.1 summarizes the lower-division primary prerequisite structure as it pertains to the courses relevant to this dissertation. Our institution's lower-division courses are fairly traditional courses such as basic data structures (CS2), discrete math and advanced discrete math. CS1 is taught as a one or two course sequence, which the students are advised to select according to their prior preparation. There are designated "alternative" courses satisfying most requirements:

- Alternatives for Discrete Math include 'Introduction to Discrete Mathematics' and 'Mathematical Reasoning' in the Mathematics department.
- Alternative courses for Advanced Discrete Math include 'Discrete Mathematics and Graph Theory' and 'Enumerative Combinatorics' in the Mathematics department.
- Equivalent courses for Computer Organization include 'Introduction to Programming I' taught in C by the CS department, 'C/C++ Programming' in the Mechanical Engineering department and 'Engineering Computation' in the Electrical Engineering department.

Below is a detailed description of each course with course descriptions from the course catalog [44].

1. **Introduction to Programming and Computational Problem-Solving I (CS1).** This is an introductory course for students who do not have prior programming experience. The current course description is: "Introductory course for students interested in computer science and programming. Basics of programming including variables, conditionals, loops, functions/methods. Structured data storage such as arrays/lists and dictionaries, including data mutation. Hands-on experience with designing, writing, hand-tracing, compiling or interpreting, executing, testing, and debugging programs. Students solve relevant computational problems using a high-level programming language." There are no prerequisites.

2. **Introduction to Programming and Computational Problem-Solving II (CS1.5).** This is the second course in a two course sequence on introductory programming (CS1). The course description is: “Introductory programming using an object-oriented approach with the Java programming language. Builds on basic programming constructs introduced in CS1 to introduce class design and use, interfaces, basic class hierarchies, recursion, event-based programming, error reporting with exceptions, and file I/O. Basics of command-line navigation for file management and running programs. Development, testing, and debugging of more complex programs.” The main prerequisite is CS1, the first course in the two-course introductory computing sequence.
3. **Introduction to Programming and Computational Problem-Solving: Accelerated Pace (CS1-Acc).** This is the accelerated one-term course for introduction to programming. The recommended preparation is that students have significant prior programming experience such as high school AP CSA). The course description is: “Accelerated introductory programming including an object-oriented approach. Covers basic programming topics from CS1 including variables, conditionals, loops, functions/methods, structured data storage, and mutation. Also covers topics from CS1.5 including the Java programming language, class design, interfaces, basic class hierarchies, recursion, event-based programming, and file I/O. Basics of command-line navigation for file management and running programs.” There are no prerequisites.
4. **Basic Data Structures and Object-Oriented Design (CS2).** The course description is: “Use and implementation of basic data structures including linked lists, stacks, and queues. Use of advanced structures such as binary trees and hash tables. Object-oriented design including interfaces, polymorphism, encapsulation, abstract data types, pre-/post-conditions. Recursion. Uses Java and Java Collections.” The main prerequisite is CS1.5 or CS1-Acc.
5. **Discrete Mathematics (Discrete Math)** The course description is: “This course will

introduce the ways logic is used in computer science: for reasoning, as a language for specifications, and as operations in computation. Concepts include sets, relations, functions, equivalence relations, partial orders, number systems, and proof methods (especially induction and recursion). Propositional and predicate logic will be introduced and applied to various computer science domains such as circuit design, databases, cryptography, and program correctness.” The main prerequisite is CS1.5 or CS1-Acc.

6. **Mathematics for Algorithms and Systems (Adv. Discrete Math)** The course description is: “This course will cover mathematical concepts used to model and analyze algorithms and computer systems. Topics include counting techniques (inclusion-exclusion; recursive counting; permutations and combinations), data representations, analysis of algorithms (order notation; time complexities; loop invariants), recurrence relations, graphs and trees (data structure representations; basic graph algorithms; special classes of graphs), and basic probability and its applications.” The main prerequisite is Discrete Math.
7. **Computer Organization and Systems Programming (ORG).** The course description is: “Introduction to organization of modern digital computers—understanding the various components of a computer and their interrelationships. Study of a specific architecture/machine with emphasis on systems programming in C and Assembly languages in a UNIX environment.” Prerequisites include CS2 and a 2-unit laboratory course on software tools and techniques. In some cases students are able to take the laboratory course concurrently with ORG.
8. **Advanced Data Structures (ADS).** The course description is: “High-performance data structures and supporting algorithms. Use and implementation of data structures like (un)balanced trees, graphs, priority queues, and hash tables. Also, memory management, pointers, recursion. Theoretical and practical performance analysis, both average case and amortized. Uses C++ and STL. ” Main prerequisites include Advanced Data Structures, CS2 and ORG.

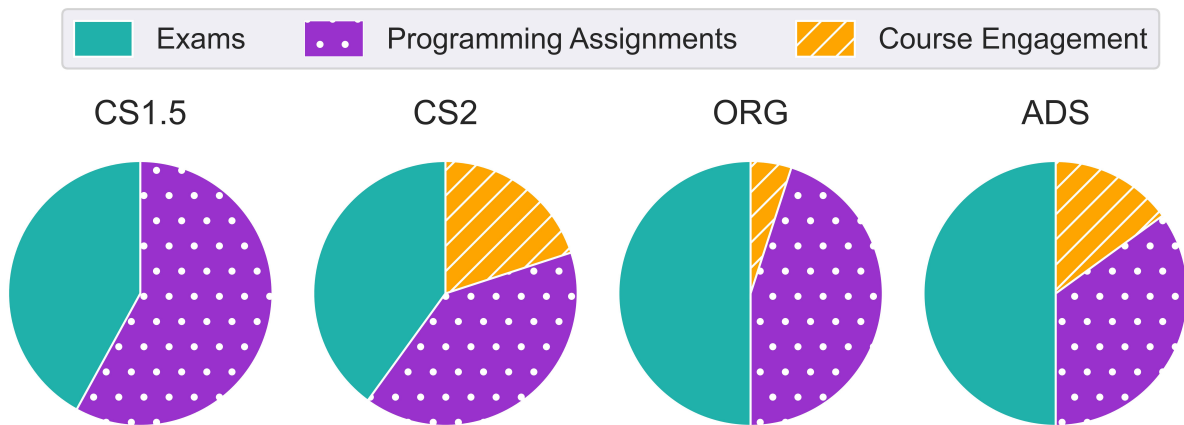


Figure 3.2. The grading breakdown for CS1.5, CS2, ORG and ADS courses. Note that 50% or more of each course grade allows for collaboration or awards effort.

3.1.2 Course Grading

The courses studied in this dissertation include CS1.5, CS2, ORG and ADS. The course assessments are broken down below and can be seen in Figure 3.2. These include the course assessments for ADS in Fall 2018 and CS1.5, CS2 and ORG in Fall 2022. We observe that at least 50% of the course grades allow for collaboration or reward effort. This proportion is not unusual in our CS program.

1. **CS1.5.** The course assessments were homework assignments (58%), and quizzes and final exams (42%).
2. **CS2.** The course assessments were course engagement activities (weekly quizzes and surveys) (20%), programming assignments (40%) and exams (40%).
3. **ORG.** The course assessments were reading quizzes (5%), programming assignments (45%), exams (50%).
4. **ADS.** The course assessments were online reading assignments (10%), clicker participation (5%), in-class quizzes (5%), programming assignments (35%), midterm (20%), and final exam (25%).

3.2 Study Context for Chapter 7

The data set used for Chapter 7 includes interviews with 21 instructors who teach a wide range of undergraduate courses at: a public research-intensive institution (the University of California San Diego) and a Hispanic-serving public undergraduate institution (California State University, Long Beach). The courses can be seen in Table 3.1.

3.3 Acknowledgements

Chapter 3, in part, is a reprint of material as they appear in three publications: 1) Proceedings of the 51st ACM Technical Symposium on Computer Science Education (SIGCSE 2020). Sophia Krause-Levy, Sander Valstar, Leo Porter, and William G. Griswold. “Exploring the Link Between Prerequisites and Performance in Advanced Data Structures”, 2) Proceedings of the 53rd ACM Technical Symposium on Computer Science Education (SIGCSE 2022). Sophia Krause-Levy, Sander Valstar, Leo Porter, and William G. Griswold. “A Demographic Analysis on Prerequisite Preparation in an Advanced Data Structures Course”, 3) Proceedings of the 54th ACM Technical Symposium on Computer Science Education (SIGCSE 2023). Sophia Krause-Levy, Adrian Salguero, Rachel S. Lim, Hayden McTavish, Jelena Trajkovic, Leo Porter, and William G. Griswold. “Instructor Perspectives on Prerequisite Courses in Computing”. The dissertation author was the primary investigator and author of this paper.

Table 3.1. Participation was solicited for instructors typically teaching following courses.

Course	CS Area	Level	Sample Course Topics	<i>N</i> Instructors
CS1/CS1.5/CS1-Acc/OOP	Programming	Lower	variables, loops, functions/methods, testing	5
CS2/Data Structures	Programming	Lower	stacks, queues, binary search trees	3
Computer Organization	Systems Programming	Lower	assembly language, number representations	3
Advanced Discrete Math	Theory	Lower	counting principles, algorithmic analysis,	1
Advanced Data Structures	Programming	Upper	graphs, hash tables, performance analysis	2
Algorithms	Programming/Theory	Upper	greedy algorithms, divide and conquer	4
Theory of Computation	Theory	Upper	formal languages, finite automata	1
Software Engineering	Applications	Upper	process, design patterns, software testing	2

Chapter 4

[Causes] Exploring the Link Between Prerequisites and Performance in Advanced Data Structures

The study by Valstar et al. found a surprisingly low level of preparedness among many of the students, despite grades in prerequisite courses being relatively high [70]. In particular, 29% of students “failed” the pre-term test, despite fewer than 1% of their prerequisite course grades being a D. This disconnect raised questions regarding **causes** of the problem: how could such low preparedness be possible and how students could get such high grades with so little apparent preparation?

Our impression as instructors and teaching assistants is that prerequisite preparation varies widely. On the assumption that knowledge fades with time [12], we worried that the length of time between taking a prerequisite and the follow-on course could be hurting some students. We also had a concern that because our CS program allows for multiple ways to satisfy some prerequisites—most not under our program’s direct control—that some students might be disadvantaged in their prerequisite preparation.

In a different vein, it was also our impression that scores on exams tend to be lower than scores on other assessments. Collaboration on assignments could be contributing to higher scores for some, as could low-stakes assessments that measure effort more than knowledge. Moreover, we had the impression that higher performing students possess durable qualities

such as study skills, personal habits, and motivation that play a substantial role in improving performance [31]. Taken together, we hypothesized these qualities would contribute to not only knowledge acquisition, but also completing course elements with the highest possible grade.

Studying the same course as the previous study [70], we tested these hypotheses by collecting comprehensive course and grade information, and applying a variety of statistical analyses. First, we examined how prerequisite course grades correlated with the pre-term test, final exam grade, and course grade. We then did the same analysis for grade aggregates such as prerequisite grade point average (GPA) and non-prerequisite GPA (excluding other CS courses). GPA represents the average grade a student has earned in their courses over time. To gain an understanding of the relative contributions of the various inputs, we then examined the relative importance of the precursors via multiple linear regression.

We come away with two major conclusions. *Students Learn in Prerequisite Courses:* Although grades may overstate how much has been learned, we find that prerequisite grades are predictive of prerequisite knowledge on entry to the course (0.48 Pearson’s correlation). Related to this, we find no deleterious effects on prerequisite knowledge due to the passage of time or taking alternative prerequisites. *Students with Higher Outcomes are More Skilled at Taking Courses:* Prerequisite grades are even more predictive of the follow-on course grade (0.53 correlation), despite the course grade including knowledge of new content learned during the course. Somewhat surprisingly, we also find that a student’s non-prerequisite GPA, excluding other CS courses, is also highly predictive of the course grade (0.51). Combined with additional statistical tests, this supports our hypothesis that course grades reflect not just subject matter knowledge, but also other factors.

4.1 Study Design

This chapter discusses the design of our study. It will cover our hypothesis, the context and population of our study, the data we collected and our process for cleaning it.

4.1.1 Hypotheses

We formalize the hypotheses stated in the Introduction as follows:¹

H-PL: (P)rerequisite (L)earning. Students learn prerequisite subject matter in the prerequisite courses for a CS course.

H-PT: (P)rerequisite (T)ime. The length of time between taking a course's prerequisites and taking the course itself negatively affects student success in that course.

H-AP: (A)lternative (P)rerequisites. Satisfying a course's prerequisite with an alternative to the primary identified course negatively affects student success in that course.

H-OF: (O)ther (F)actors. Other factors, unrelated to subject matter knowledge, strongly affect a student's success in that course.

Although H-PL seems obvious, we include it to establish a backdrop for analyzing the other hypotheses. For example, it would be difficult to establish the relative importance of factors unrelated to prerequisite knowledge (H-OF) without understanding the contribution of learning prerequisite knowledge in prerequisite courses.

4.1.2 Population

Although our population of students comes from the same ADS course previously studied by our research group, it contains a slightly different group of students than the cohort used in the previous analysis. To enable our previous analysis, our inclusion criteria included students who took the pre-term test (see Appendix A.1), post-term test, and the final exam. Our current hypotheses do not require the post-term test, but do require other data that not all students possessed, resulting in a different cohort.

¹In hypothesis testing it is standard to formulate a negation of the hypothesis, the null hypothesis, to be falsified. For example, the null hypothesis for H-PL is that students do not learn prerequisite knowledge or even unlearn it. In this study, the null hypotheses are straightforward so we do not explicitly state them.

Table 4.1. Majors breakdown of students as a percentage. Double majors are counted twice. Columns may not add up 100% due to rounding.

	N	Math-CS	CS	CE	Other
ADS	363	32.13%	26.33%	9.42%	32.13%
Included in Study	202	36.94%	24.77%	11.26%	27.03%
Main-Track	120	24.80%	35.20%	17.60%	22.40%
Alt-Track	82	52.58%	11.34%	3.09%	32.99%

Table 4.2. Non-prerequisite courses taken by ADS students.

Course ID	Courses	
	N	%
Math	965	33.47%
Writing	423	14.67%
Physics	200	6.94%
Cog. Sci.	190	6.59%
Economics	174	6.04%
Chemistry	131	4.54%
Other	800	27.75%
Average/Student	14.27	

Many of our research questions depend on prerequisite grades and the ability to compare the grades between different offerings of a given prerequisite. This led us to exclude students who took some prerequisite courses at a community college or different university, as we did not have access to the official equivalencies of those courses nor the grade distributions for those course offerings. We hope to gain access to this information for future work. Table 4.1 summarizes many characteristics discussed below.

The pre-term test and final exam were taken by 322 students. Of those, 206 took all of their prerequisites for ADS at our university. Of those, four took only one or two non-CS courses outside of their prerequisite courses for ADS. These students were excluded from the study population as our analysis relies heavily on a student's non-prerequisite non-CS GPA, and their GPAs would not provide enough information to be reliable. The remaining 202 students form the population for the analysis in this study. Table 4.2 summarizes the kinds of non-prerequisite

non-CS courses taken by these students.

To assess our hypothesis about the effects of taking alternative courses to the primary identified prerequisite for ADS, we created two sub-populations. A student is *main-track* if they took all of the primary identified prerequisite courses for ADS (120 students). A student is *alternate-track* if they took all of the required prerequisites, but at least one of their prerequisites is a designated alternative to the primary identified prerequisite course (82 students).

4.1.3 Data Collection and Cleaning

Our study expands on the pre-term test and final exam data collected for our previous study [70]. In addition, with help from our approved human subjects protocol, we obtained the students' grades in ADS as well as grades for all courses that the students had taken at our university prior to taking ADS.

We used the students' grades to calculate the average grade for the prerequisite courses required to take ADS (*prerequisite GPA*) and the average grade for all courses taken at our university excluding their prerequisite courses and any other CS courses taken at our university (*non-prerequisite GPA*). Because the purpose of the non-prerequisite GPA is to operationalize the skills of taking courses independent of prerequisite knowledge, CS courses are excluded from the non-prerequisite GPA to lessen contributions related to (a) special enthusiasm for CS or (b) prior CS knowledge that might implicitly overlap with the prerequisites. When calculating prerequisite GPAs, if a student took more than one course that counted for a single prerequisite, we chose the course that was taken most recently, as it best reflects prerequisite knowledge when entering the ADS course. This occurred 86 times (40% alternate-track, 60% main-track), for approximately 8.5% of the prerequisite courses. Neither GPA included courses that were taken pass/fail.

At our university grades are recorded on a 4 point scale where a 4.0 represents an A and 0.0 represents an F. Using grade points to represent knowledge or accomplishment is problematic as most students who receive an F did not earn a 0% in the instructor's gradebook. To account

for this bias in grade point, we rescaled the 4.0 scale such that an F is equivalent to 50%.

All the data used in this study was normalized via z-score. For the courses outside ADS, the student grades we have are just a sample of the grades earned in those courses, not permitting a straightforward calculation of a z-score. Normalizing these is especially important because not all courses or course offerings are equally difficult. That is, an A in one course offering can mean something different than an A in another course offering. Fortunately, we were able to scrape our university's online course evaluation system for the grade distributions of all relevant courses and use them to compute the z-score for each grade (A, A-, B+, B, B-, C+, C, C-, D, F). Because we only have access to the course and term in which a student took a course, in cases where multiple sections of a course were offered the same term, we used the average of the distributions for those sections. In the cases where a distribution was not available at all, we used the average distribution for terms for which we had data. This occurred in just 10 sections across 3 different courses. In total there were 1,223 sections across 477 courses.

4.2 Results

4.2.1 H-PL: Students Learn in Prerequisites

To understand how prerequisites contribute to learning prerequisite subject matter, we calculated the Pearson correlation between prerequisite GPA and pre-term test score, which is significant with a correlation of 0.48 (Table 4.3, under the heading "All"). As expected, it is more highly correlated than the other pre-term measure, non-prerequisite GPA (0.35), see Figure 4.1. To assess whether these correlations are meaningfully different, we applied Williams's Test, which is used to test whether two correlations for the overlapping (i.e., dependent) populations are in fact different [77]. As shown in Table 4.4, the differences in the correlations are significant.² Together, these support the hypothesis that performance in prerequisite courses positively affects learning of prerequisite knowledge in CS.

²Williams's Test uses only the significance test, not the test value.

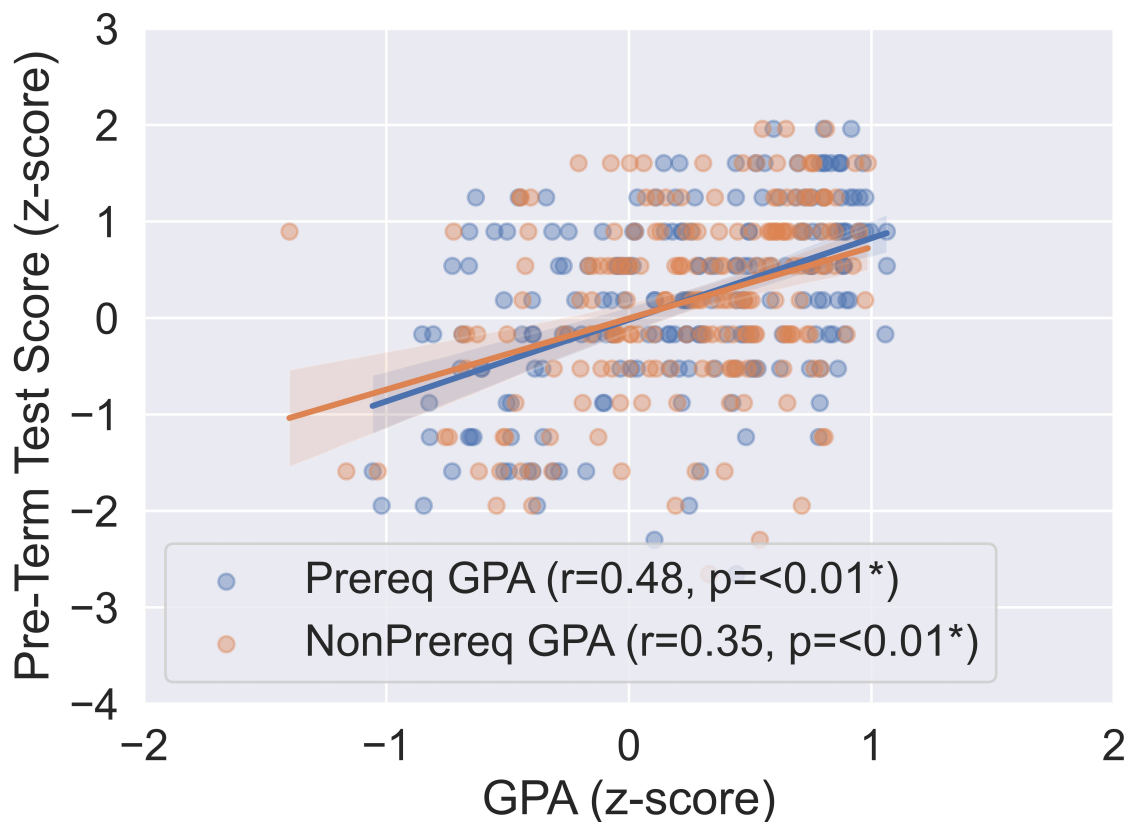


Figure 4.1. Scatter plot of student’s score on the Pre-Term Test Score to their Prerequisite GPA and Non-Prerequisite GPA. Note that all of the data collected was normalized via z-score which accounts for the negative and positive range of values along the x and y axes.

Table 4.3. Pearson correlations between precursor course measures and pre-term test score. A star (*) indicates a significant correlation for $\alpha = 0.05$.

GPA Variable	All		Main-Track		Alt-Track	
	corr	p	corr	p	corr	p
Prereq	0.48	<0.01*	0.50	<0.01*	0.46	<0.01*
NonPrereq	0.35	<0.01*	0.34	<0.01*	0.35	<0.01*

Table 4.4. Williams’s Test for whether the correlations of the two precursors to test score (Table 4.3, under heading “All”) are different. A star (*) indicates significance for $\alpha = 0.05$.

Variable 1	Variable 2	stat	p
Prereq GPA	NonPrereq GPA	3.20	<0.01*

4.2.2 H-PT: Prerequisite Learning Fades

Table 4.5. Pearson correlation for performance in prerequisites minus pre-term test score with time since courses taken. A star (*) indicates a significant correlation for $\alpha = 0.05$.

Variable	corr	p
Pre-Term Test- Main Prereq GPA	0.08	0.25

Now that we know students are learning in prerequisites, we consider whether the fading of knowledge over time could be contributing to lower performance on the pre-term test. We operationalized the passage of time as the sum of the months elapsed since taking the main prerequisites, Advanced Discrete Math, CS2, and Computer Organization (Figure 3.1). We excluded the other two prerequisites on the presumption that their immediate successors refreshed the knowledge from those courses, given the tight dependence in subject matter. In our previous study we found ADS itself provided such a refreshing effect to prerequisite knowledge [70]. To operationalize the change in knowledge over time, we subtracted a student's GPA for the aforementioned main predecessors from the pre-term test (both as z-scores). We then calculated the correlation between the variables for net time elapsed and change in knowledge (Table 4.5). It is not significant, and at 0.08, it is not in the hypothesized direction (as the time gap increases so does prerequisite knowledge), providing no support for the hypothesis.

4.2.3 H-AP: Taking Alternatives to Primary Prerequisites Diminishes Student Success

To determine whether there was a difference between the outcomes for main-track and alternate-track students, we began by calculating, for each group, the correlation for every combination of precursor measure (prerequisite GPA, non-prerequisite GPA, and pre-term test score) and the two result measures (final exam and grade). The correlations can be found on the right side of Table 4.6 under the headings Main-Track and Alt-Track. All of the correlations are significant, except for the main-track correlation between pre-term test score and ADS grade.

Table 4.6. Pearson correlations for performance in course with a comparison to final exam score and ADS course grade: correlation with p-values. A star (*) indicates a significant correlation for $\alpha = 0.05$.

GPA Variable	All						Main-Track						Alt-Track					
	Final Exam			Grade			Final Exam			Grade								
	corr	p		corr	p		corr	p		corr	p		corr	p		corr	p	
Prereq GPA	0.66	<0.01*		0.53	<0.01*		0.63	<0.01*		0.53	<0.01*		0.70	<0.01*		0.53	<0.01*	
NonPrereq GPA	0.55	<0.01*		0.51	<0.01*		0.56	<0.01*		0.51	<0.01*		0.53	<0.01*		0.51	<0.01*	
Pre-Term Test	0.30	<0.01*		0.21	<0.01*		0.35	<0.01*		0.17	0.06		0.26	0.02*		0.27	0.02*	

Table 4.7. Multiple linear regression statistics for dependent variables of final exam score and ADS course grade. A star (*) indicates a significant correlation for $\alpha = 0.05$.

Independent Variable	DV: Final Exam (r^2 : 0.437)						DV: Grade (r^2 : 0.305)					
	coeff	std. err	t-stat	p	r^2 diff	coeff	std. err	t-stat	p	r^2 diff		
Prereq GPA	1.10	0.17	6.57	<0.01*	0.1229	0.60	0.16	3.69	<0.01*	0.0479		
NonPrereq GPA	0.15	0.18	0.86	0.39	0.0021	0.41	0.17	2.34	0.02*	0.0193		
Pre-Term Test	-0.01	0.06	-0.18	0.86	0.0001	-0.04	0.06	-0.72	0.47	0.0018		

Table 4.8. Steigler’s z-test for whether the main-track and alternate-track correlations are different. A star (*) indicates significance for $\alpha = 0.05$.

	Final Exam		Grade	
	z-stat	p	z-stat	p
Prereq GPA	0.86	0.39	0.00	1.00
NonPrereq GPA	-0.29	0.77	0.00	1.00
Pre-Term Test	-0.68	0.50	0.72	0.47

Then, for each of the six combinations, we tested whether the correlation for alternate-track was significantly different than that for main-track. Because the test is between non-overlapping (i.e., independent) groups, we applied Fisher’s r-to-z transformation to the two correlations and then performed Steigler’s z-test [67] (Table 4.8). None of the correlation pairs are found to be different, as the test magnitudes are below 1.96 (and $p > 0.05$). As an example, the correlation for prerequisite GPA and ADS grade is 0.53 for main-track and 0.53 for alternate-track, with the z-test returning a result of 0.00 with $p=1.00$. Thus, we find no support for the hypothesis that alternative prerequisites negatively affect performance.

4.2.4 H-OF: Factors Other than Prerequisite Learning Boost Student Performance

If factors beyond actual subject matter knowledge contribute to student grades, we would expect to see a (strong) correlation between course grades in unrelated subjects. On the other hand, we would expect tests to more strongly correlate to subject matter knowledge since (a) collaboration is both prohibited and harder and (b) typically assess subject matter knowledge more than effort. Thus, to measure how other factors unrelated to subject matter knowledge affect student success in a course, we took the correlations between students’ non-prerequisite GPA and three measures related to prerequisites: ADS course grade, ADS final exam score, and pre-term test score (Table 4.6, under the heading “All”). All of the correlations are statistically significant. Figure 4.2 shows a scatter plot of the student pre-term test scores to final exam scores.

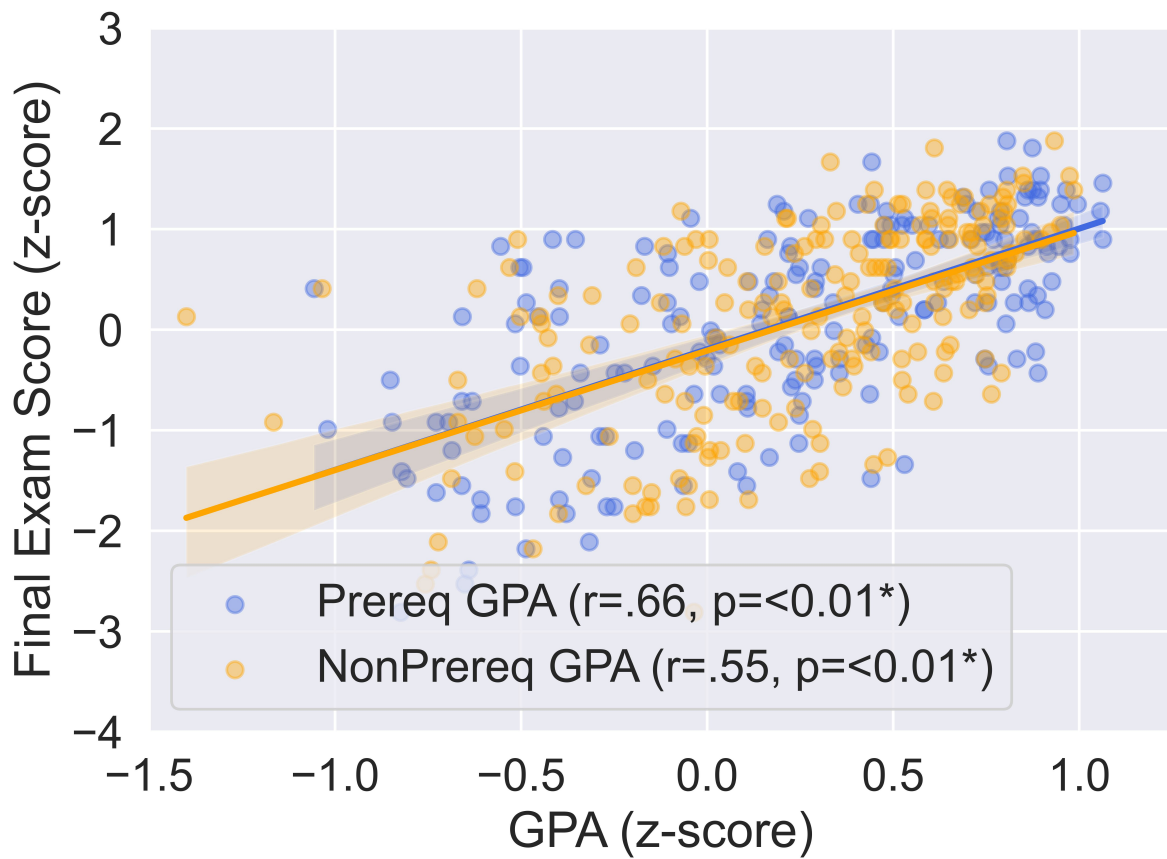


Figure 4.2. Scatter plot of student's score on the Final Exam to their Prerequisite GPA and Non-Prerequisite GPA. Note that all of the data collected was normalized via z-score which accounts for the negative and positive range of values along the x and y axes.

Table 4.9. Williams's Test for whether precursors are differently correlated to outputs. A star (*) indicates significance for $\alpha = 0.05$.

Variable 1	Variable 2	Final Exam		Grade	
		stat	p	stat	p
Prereq GPA	NonPrereq GPA	3.16	<0.01*	0.52	0.61
Prereq GPA	Pre-Term Test	6.49	<0.01*	5.17	<0.01*
NonPrereqGPA	Pre-Term Test	3.66	<0.01*	4.26	<0.01*

If non-prerequisite GPA were nearly as predictive as prerequisite GPA for success in ADS, then factors beyond subject matter knowledge must be playing a major role in success. This could be validated if the correlations between ADS grade and the two GPAs were deemed similar. Indeed, Williams's Test finds no significant difference in the correlations (Table 4.9, first

row, under the heading “ADS grade”). We would also expect Williams’s Test to find significant differences involving exams, and this is indeed the case (Tables 4.4 and 4.9 second and third rows, heading “Final Exam”).

Having established these relationships, the question of the precursors’ relative importance in outcomes arises. To this end, we performed two multiple linear regression analyses, one for the dependent variable (DV) ADS grade and the other for the ADS final exam, both with the independent variables (IVs) non-prerequisite GPA, prerequisite GPA, and pre-term test score (Table 4.7).

Since the independent variables are no-doubt interdependent, their model coefficients alone cannot determine the relative contribution of the independent variables. Nor are the coefficient p-values considered useful, as they don’t capture effect size. The relevance of an independent variable is determined instead by how it affects the model’s R-squared value. Thus we employed the method of calculating the increase in R-squared that each variable produced when it was added to a model that already contained all the other independent variables (i.e., the one other variable in this case). This is called the *R-squared difference*.

Our expectations are borne out by the data, with prerequisite GPA dominating non-prerequisite GPA for ADS final exam (coefficient 1.10 vs. 0.15 and r^2 difference 0.1229 vs. 0.0021), but with much more parity for ADS grade (coefficient 0.60 vs. 0.41 and r^2 difference 0.0479 vs. 0.0193).

In conclusion, we accept the hypothesis that factors outside subject matter knowledge are playing a large role in student success. Separately, we observe that pre-term test score is not useful in the model. We suspect that prerequisite GPA sufficiently captures prerequisite knowledge on entry to the course, since H-PL is supported and H-PT and H-AP are not.

4.3 Limitations and Risks to Validity

Our results are based on a single offering of a particular course at a single research university, limiting generalizability. However, the curriculum is quite common and informed by the ACM Curriculum guidelines [25]. Moreover, US research-intensive universities teach a large proportion of CS students on the continent, making this an interesting demographic in itself. Still, results could vary outside this context, encouraging attempts to replicate our results.

At a more specific level, our conclusions on the effects of taking alternative prerequisites only included on-campus alternatives. Variations in curricular match and instructional quality are possibly higher for courses from other campuses, limiting generalizability.

There are a few possible risks to the validity of our conclusions. For one, the pre-term test is short and not validated. This could result in over- or under-estimating prerequisite knowledge, depending on the actual shortcomings of the pre-term test. Also, beyond goodwill and pride, the students had no incentive to perform well on the test, meaning that students lacking these motivations could have under-performed. To mitigate these, we did not use the magnitude of the pre-term test score in drawing our conclusions, only the correlation, which depends on covariance. Of lesser concern is that we could not include all students who took ADS in the study, as explained in Section 4.1.2. Perhaps students who come to class are more highly motivated than those who don't, but it could also be the case that high-performing students don't feel the need to come to class every day. The motivation element increases the chance that the students who took the pre-term test would take it more seriously, mitigating the just-cited incentive risk. The additional factors for exclusion did not appear to significantly alter the population mix (Table 4.1).

A risk in distinguishing the main-track and alternate-track groups is that the groups are biased. As can be seen in Table 4.1, there are substantially more CS majors in the main-track group. One possible source of bias is that entry into the CS major is competitive, so it is possible that the main-track group would perform better in CS courses than the alternate-track

group. However, our analysis for H-AP depends on the correlation of grades to pre-term test score, and the value of a correlation does not depend on the relative magnitude of grades, but rather co-variance of the magnitudes of grades. In other words, despite a possible bias in the populations, the alternate-track group was indistinguishable from the main-track group in terms of how grades predict performance.

For main-track and alternate-track students, 8.5% of students took and passed two different courses that satisfied a prerequisite. Additionally, an unknown number of students may have taken a late drop in a course satisfying a prerequisite, or have taken a course satisfying a prerequisite at a community college. These could disguise shortcomings in the prerequisite courses that we used in our main-track/alternate-track prerequisite GPA calculations.

Our use of grade distributions to normalize grades across offerings, while controlling for the difficulty of a course offering, also potentially masks actual differences in student performance. However, the courses in our study have enough students (120 students on average) that a course offering likely represents a substantial, and representative, sample of the student body.

Finally, although we concluded that factors related to subject-matter knowledge are a large contributor to student success in a course, another possible cause is that students are acquiring ADS-relevant subject matter in non-prerequisites. We excluded CS from the non-prerequisite GPA to mitigate this risk, and it's unlikely that there would be a measurable pattern given the diverse non-prerequisite demographics we found in our population (Table 4.2). Still, this shadow-prerequisite question is deserving of future study, especially for non-prerequisite CS courses.

4.4 Discussion

This study began with two questions: how could students manifest so little prerequisite knowledge coming into a course, and how could they get such good grades with so little apparent

preparation?

The good news from our study is that students are learning in prerequisite courses, and there is no evidence for the passage of time or taking alternative prerequisites diminishing that knowledge. Regarding the lack of ill effects of the passage of time, it should be kept in mind that the time between taking prerequisites and taking ADS is relatively bounded. For those with longer breaks, this knowledge might be refreshed by intervening activities such as internships, a job, or taking other CS courses.

Also interesting is that students who do better in non-prerequisite non-CS courses also do better in ADS. More specifically, students are distinguishing themselves not just through subject matter knowledge, but also other factors. Although this study couldn't reveal what those factors are, we expect influences from pre-college privilege and other forms of preparatory privilege, study habits, excessive collaboration, and low-stakes assessments (combined with student personal habits that maximize these assessments). Future work should investigate these factors, their impact on knowledge acquisition, and why some students have these skills and habits and others do not.

Tantalizingly, students who do better in non-prerequisite non-CS courses also do better on the ADS final exam, although the effect is not as strong as on ADS course grade. We hypothesize two possibilities for this positive correlation. One is that non-prerequisite courses and ADS could be dependent on (and/or enhancing) “shadow prerequisites”, for example spatial reasoning or critical thinking skills, implying that courses outside of CS continue to improve CS skills [47]. Another explanation is that exams, like other course elements, are often gameable, e.g., by smartly guessing which topics will get the most coverage on the exam and studying accordingly, or the use of test strategies such as process-of-elimination or matching the syntax of a question to a known practice question. These test-taking skills may be due to some students having more access to resources before entering university. They may have better learned how to exist in the educational system, thus leading to higher grades. These skills fit in our category of “other factors” for success, and could be quite transferable between course subjects, explaining the

correlation. We leave the exploration of these hypotheses for future work.

4.5 Summary

This chapter focused on possible **causes** for students’ low incoming prerequisite course proficiency and possible causes for why students are able to make it into the ADS course without the expected level of prerequisite course proficiency. Interestingly, we find nothing to suggest that the passage of time between prerequisites or taking alternatives to the main prerequisites impacts student outcomes. We do however find evidence that suggests factors outside of subject matter knowledge may be allowing students progress through the computing curriculum without the expected level of prerequisite course proficiency.

4.6 Acknowledgements

This work was supported in part by NSF IUSE Grant DUE-1712508. Chapter 4, in part, is a reprint of the material as it appears in the Proceedings of the 51st ACM Technical Symposium on Computer Science Education (SIGCSE 2020). Sophia Krause-Levy, Sander Valstar, Leo Porter, and William G. Griswold. “Exploring the Link Between Prerequisites and Performance in Advanced Data Structures”. The dissertation author was the primary investigator and author of this paper.

Chapter 5

[Impact] A Demographic Analysis on Prerequisite Preparation in an Advanced Data Structures Course

In the previous chapter we explored possible **causes** for why students do not have the level of prerequisite course proficiency coming into a course that instructors expect [28, 70]. In this chapter we focus on understanding the **impact** prerequisite course proficiency may play in disparities between demographic groups.

Black, Latinx, Native American and Pacific Islander (BLNPI) students and women are traditionally underrepresented in computing compared to their majority counterparts who are White and Asian men [84]. In computing, women and BLNPI students are retained at a lower rate than their majority counterparts [83] and tend to have inequitable outcomes during their time studying computing [4, 58, 59]. Previous work in computing education has identified various factors that may impact students from groups that are typically underrepresented in computing, such as early exposure to computing [4, 34, 35, 75], impostor phenomenon [56], sense of belonging [29] and prerequisite knowledge [72]. With the knowledge that these various factors are affecting these populations of students, the question becomes: to what degree are these factors affecting student outcomes? Answering this question may identify possible interventions to help these students.

5.1 Preliminary Results

In order to better understand how well students retain knowledge they learn earlier in their computing degree, whether it improves over time, and how it varies by demographic we used the validated Basic Data Structures Inventory [52] to test students’ knowledge of basic data structures at different points throughout their degree [72]. The results showed that student performance varies widely, but that performance on the basic data structures concepts seems to improve as students move through the computing curriculum. Concerningly, we found that many students do not resolve gaps in knowledge around this early course material even as they progress through later computer science courses. In terms of demographics, the results showed that there is no difference by gender but that BLNPI students had slightly lower outcomes than their counterparts. The most concerning results were for transfer students, where there seems to be a growing gap in knowledge over time compared with non-transfer students. Given that proficiency is tied to later course success, it is all the more concerning that we found Black, Latinx, Native American and Pacific Islander (BLNPI), and transfer students to be disproportionately impacted [72].

5.2 Our Study

Motivated by current events and our recent results [28, 72], we wanted to better understand the degree to which incoming prerequisite course proficiency plays a role in the disparity of outcomes between groups typically underrepresented in computing and their majority counterparts. To shed light on this, we reanalyzed the data from our previous studies [28, 70] (Chapter 4) in view of demographic data we acquired from our institution. In this chapter we look at the relationship between prerequisite course proficiency for the different demographic groups of gender, BLNPI status, first-generation (FirstGen) status¹ and transfer status² in an Advanced

¹First-generation students are students who are members of the first-generation in their family to attend college.

²In our country, it is common for a portion of students to enter university after completing a 2-year college program that fulfills General Education requirements and, typically, a couple computing courses. We refer to these students as “transfer” students. Typically, transfer students are more socioeconomically diverse than those students entering university directly.

Data Structures (ADS) course through the use of a pre-term test used to measure students' incoming prerequisite course proficiency and students' prerequisite course grades. Our results are as follows:

- **No difference in prerequisite course proficiency by gender or first-generation status.**

We did not find anything to suggest that there is a difference in prerequisite course proficiency by gender or first-generation status when analyzing differences in the pre-term test.

- **Transfer and BLNPI students come in with less prerequisite course proficiency.**

We find the biggest gap in prerequisite course proficiency by transfer status with a statistically significant difference in pre-term test scores. Although there is not a statistically significant difference by BLNPI status, a substantial difference is seen that is similar to the difference for transfer status. The lack of significance may be due to the small sample size of BLNPI students.

- **Unequal (or inequitable) outcomes for Transfer and BLNPI students in courses leading into the ADS course.**

BLNPI students' outcomes are significantly lower in all of the prerequisite courses of the ADS course compared to Non-BLNPI students. Transfer students' outcomes are significantly lower than non-transfer students in the two immediate predecessors of the ADS course: Advanced Discrete Math and Computer Organization.

- **Prerequisite course proficiency is correlated with student outcomes.**

We find prerequisite course proficiency (the pre-term test) to be statistically significantly correlated with both the ADS final exam score and ADS course grade for all populations except BLNPI students (which may be due to the small sample size).

- **Transfer students' prerequisite course proficiency is not correlated with the number of prerequisites they take at our institution with the ADS course.**

There was not a

statistically significant correlation between the number of courses taken at our institution and the pre-term test score.

These results show a disparity in who has prerequisite course proficiency, with BLNPI and transfer students having unequal outcomes on measures of prerequisite knowledge (pre-term test scores and prerequisite course grades). Further, these results suggest that institutions are not currently meeting the needs of these students. More work needs to be done to identify how we can best serve these populations.

5.3 Study Design

5.3.1 Research Questions

RQ1: Does incoming prerequisite course proficiency to the ADS course vary by gender, BLNPI, first-generation and transfer student status?

RQ2: Is there a consistent gap in performance in the prerequisite courses to the ADS course by gender, BLNPI, first-generation and transfer student status?

RQ3: Is prerequisite course proficiency correlated with student outcomes in the ADS course and is there a difference in student outcomes by gender, BLNPI, first-generation and transfer student status?

RQ4: Do students who take more of their prerequisites at our institution, the same institution as the ADS course, have higher prerequisite course proficiency?

5.3.2 Population

Our population of students comes from the same course offering of the ADS course previously studied by our research group [28, 70]. For the purposes of our research, we could include all students who took the pre-term test that measured students' prerequisite course proficiency, which gave us 320 students. (See [70] for more details on the pre-term test).

Table 5.1. Number of participants (N) and percentage of Women; Black/Latinx/Native American/Pacific Islander (BLNPI); Transfer; First-generation (FirstGen) students.

	N	Women	BLNPI	Transfer	FirstGen
Women	102	–	5.9%	17.7%	17.7%
Men	218	–	6.9%	31.2%	20.6%
BLNPI	21	28.6%	–	14.3%	47.6%
Non-BLNPI	299	32.1%	–	27.8%	17.7%
Transfer	86	20.9%	3.5%	–	30.2%
Non-Transfer	230	36.1%	7.8%	–	16.1%
FirstGen	63	28.6%	15.9%	41.3%	–
Non-FirstGen	257	32.7%	4.3%	23.4%	–

We chose to look at the demographics of gender, BLNPI, transfer, and first-generation status. The demographic breakdowns can be seen in Table 5.1.³

5.3.3 Data Collection and Cleaning

Our study expands on the pre-term test data, final exam scores, overall course grades, and prerequisite course grades collected by our research group [28, 70].

At our university, a 4.0 scale is used for course grades where a 4.0 represents an A and a 0.0 represents an F. Looking at the structure of our courses, a student who receives a grade of an F in a course (0.0/4.0) usually receives closer to a 50% than a 0% in terms of their actual course grade.⁴ To account for this, we re-scaled course grades such that a 0.0 on the 4.0 scale is equivalent to 50 points on the 100 point scale. This was done for both the ADS course grade and all prerequisite course grades.

Additionally, all data used in this study was normalized using z-scores. Normalizing this data was important, as not all course offerings and courses are equal in terms of difficulty (an A

³In the case of transfer and non-transfer students the number of participants do not add up to 320 due to four students not being included because they were neither freshman admits nor transfer students(e.g., exchange students or professionals attending a single class).

⁴This could be due to the fact that most students who are performing especially poorly at the beginning of the course drop it in order to avoid failing the course. Also, as discussed in Section 5.5, grades at our institution are generally high.

in one course may not be equivalent to an A in another course). In the case of the prerequisite course grades, we used the z-scores obtained from our previous work where we scraped our universities' online evaluation system for the course grade distributions for the relevant courses (for more details see [28]).

Finally, under our approved human-subjects protocol, we were able to obtain demographic information including gender, race/ethnicity, first-generation college status and whether students transferred from another college or university. For demographic information, we were only provided with a binary measure for whether students identified as Black, Latinx, Native American or Pacific Islander (BLNPI) versus students who identified as White or Asian (Non-BLNPI) to avoid accidentally deanonymizing students. For the case of gender, we were provided with a binary measure of whether students identified as women or men due to our university only collecting a binary Male/Female Category.

5.4 Results

Using normality tests, we found that the data was not normally distributed and therefore we applied nonparametric tests for our analysis. This includes using Mann-Whitney U tests instead of independent sample t-tests, Spearman correlations instead of Pearson correlations, and Kruskal-Wallis H-Tests instead of ANOVAs. Due to the large number of statistical tests run during our analysis, we adjust our p-values to account for the increased chance of Type I errors. We used the Holm-Bonferroni adjustment method, as it reduces the probability of introducing Type II errors while not excessively diluting statistical power [3].

5.4.1 RQ1: Incoming Prerequisite Course Proficiency by Demographic

In order to understand whether there is a difference in students' incoming prerequisite course proficiency by demographic pair, we looked at the differences in students' scores on the pre-term test. The differences in mean and median pre-term test scores broken down by demographics as a percentage can be seen in Table 5.2.

Table 5.2. Performance details on the pre-term test. + and - for the mean and median diff indicate whether women, BLNPI, FirstGen, and transfer students have higher or lower scores. A star (*) indicates a significant correlation for $\alpha = 0.05$ according to a Holm-Bonferroni-adjusted Mann-Whitney U test.

	Group	N	Mean	Diff	Median	Diff	std dev	U	p
Gender	Women	102	56.3%	-0.8%	60.0%	+6.7%	18.9%	10983.5	0.86
	Men	218	57.1%		53.3%		18.7%		
BLNPI Status	BLNPI	21	52.1%	-5.1%	53.3%	-6.7%	15.7%	2622.0	0.31
	Non-BLNPI	299	57.2%		60.0%		18.9%		
Transfer Status	Transfer	86	51.2%	-7.6%	53.3%	-6.7%	19.4%	7816.0	<0.01*
	Non-Transfer	230	58.8%		60.0%		18.0%		
FirstGen Status	FirstGen	63	57.4%	+0.7%	60.0%	0.0%	20.1%	8248.5	0.86
	Non-FirstGen	257	56.7%		60.0%		18.4%		

To determine whether there was a statistically significant difference between demographic pairs, we ran a Holm-Bonferroni-adjusted Mann-Whitney U test for each pair (see Table 5.2). The only significant difference was for transfer status where transfer students had statistically significantly lower scores than non-transfer students. However, we do see a similarly large difference in pre-term test scores for BLNPI students. It may be that the lack of statistical significance is due to the small sample size ($N=21$).

5.4.2 RQ2: Comparing Demographic Pairs' Prerequisite Grades

Our previous work on this data showed that the pre-term test is correlated with prerequisite course grades [28]. This led us to the question of whether we would find similar results to RQ1 when looking at students' prerequisite course grades instead of pre-term test scores. We expect the pre-term test to be an accurate measure of students' prerequisite course proficiency because it is given at the beginning of the quarter and is specifically focused on prerequisite course content, although it is a short instrument that only takes 20 minutes to complete. Prerequisite course grades each represent 10 weeks of knowledge and may capture nuances that the pre-term test does not.

To measure this we ran Holm-Bonferroni-adjusted Mann-Whitney U tests comparing each demographic pairs' course grades for each of the five prerequisite courses of the ADS course (CS1.5/CS1-Acc, CS2, Discrete Math, Advanced Discrete Math and Computer Organization). Table 5.3 shows the results of these tests, as well as, the mean and median course grades as z-scores (e.g., in the Computer Organization course, transfer students were 0.16 standard deviations below the mean course grade and non-transfer students were 0.20 standard deviations above). Due to students' ability to use AP credit towards course requirements, take summer school and transfer credits during their time at our university, the number of students does not always add up to the same number in these comparisons.

Similarly to the results of RQ1, we see no statistically significant differences by gender. By first-generation status, we only see a statistically significant difference in the Advanced

Table 5.3. Holm-Bonferroni-adjusted Mann-Whitney U test for differences in prerequisite course grades based on transfer and BLNPI status broken down by prerequisite course. A star (*) indicates a significant correlation for $\alpha = 0.05$.

Transfer Status (Transfer / Non-Transfer)					
Course	N	Mean	Median	U	p
CS1.5/CS1-Acc	31 / 229	0.27 / 0.34	0.41 / 0.56	3156.0	0.43
CS2	55 / 230	0.30 / 0.29	0.55 / 0.33	6459.5	0.60
Discrete Math	51 / 207	0.02 / 0.16	0.07 / 0.14	4773.0	0.43
Adv. Discrete Math	85 / 225	-0.11 / 0.19	0.07 / 0.43	7384.5	<0.01*
ORG	59 / 202	-0.16 / 0.20	-0.05 / 0.39	4257.5	<0.01*

BLNPI Status (BLNPI/ Non-BLNPI)					
Course	N	Mean	Median	U	p
CS1.5/CS1-Acc	18 / 242	0.06 / 0.35	0.04 / 0.56	1557.0	0.04*
CS2	19 / 266	0.01 / 0.31	0.08 / 0.46	1635.5	0.02*
Discrete Math	20 / 240	-0.31 / 0.17	-0.32 / 0.19	1621.0	<0.01*
Adv. Discrete Math	19 / 295	-0.67 / 0.16	-0.56 / 0.25	1285.0	<0.01*
ORG	17 / 247	-0.13 / 0.14	-0.01 / 0.28	1574.5	0.04*

Discrete Math course ($U = 5928, \alpha = 0.01$), with first-generation students having lower grades. For BLNPI students, we see a statistically significant difference in all of the prerequisite course grades with BLNPI students having lower prerequisite course grades. We find that transfer students have statistically significantly lower prerequisite course grades than non-transfer students in Advanced Discrete Math and Computer Organization. This suggests that the gap in knowledge for BLNPI and transfer status begins earlier in the program, manifests itself as low prerequisite course proficiency, and that low prerequisite course proficiency correlates with student outcomes in later courses.

5.4.3 RQ3: Prerequisite Course Proficiency's Impact on Student Outcomes

Our prior work on this data set has shown that prerequisite course proficiency is correlated with student success [70] and that students' incoming prerequisite course proficiency is

correlated with their performance in the course, including their ADS course grades and final exam scores [28]. We wanted to see if (1) prerequisite course proficiency was still correlated with student outcomes when looking at the different demographic groups and (2) if there were differences in student outcomes between demographic pairs.

Table 5.4. Spearman correlations with Holm-Bonferroni-adjusted p-values between performance on pre-term test and ADS final exam score and course grade. A star (*) indicates a significant correlation for $\alpha = 0.05$.

	Final Exam		Grade	
	r	p	r	p
Women	0.47	<0.01*	0.47	<0.01*
Men	0.31	<0.01*	0.26	<0.01*
BLNPI	0.16	0.48	0.33	0.14
Non-BLNPI	0.37	<0.01*	0.33	<0.01*
Transfer	0.42	<0.01*	0.40	<0.01*
Non-Transfer	0.32	<0.01*	0.29	<0.01*
FirstGen	0.43	<0.01*	0.38	<0.01*
Non-FirstGen	0.34	<0.01*	0.32	<0.01*

First, we used Spearman correlations with Holm-Bonferroni-adjusted p-values to examine the relationship between each demographics' pre-term test scores and their ADS course grades, and final exam scores (see Table 5.4). For all groups except for BLNPI students⁵ we found prerequisite course proficiency to be statistically significantly correlated with students' ADS course outcomes.

After seeing that all demographics have a strong and statistically significant correlation between prerequisite course proficiency and course outcomes (aside from BLNPI students), we wanted to know if there is a statistically significant difference between demographics pairs' ADS course outcomes. To do this, we ran a Holm-Bonferroni-adjusted Mann-Whitney U test comparing each demographic pairs' course grades and final exam scores (see Table 5.5).

Here we find no statistically significant difference in course outcomes by gender, first-

⁵The lack of statistical significance for BLNPI students may be due to the small sample.

generation, or BLNPI status. We do find a statistically significant difference for the ADS final exam scores by transfer status, with transfer students having unequal outcomes. The results for BLNPI students are not statistically significant, but the mean and median differences in the final exam score are similar to that of transfer and non-transfer students leading us to believe this may be due to the small sample size of BLNPI students.

When we look at the ADS course grades we find that they are very similar for the entire population of students seen in Table 5.5. This is likely due to the grading scheme of this course, which is representative of the typical grading scheme at our university, where a large proportion of the course grade rewards effort or may indirectly allow students to collaborate⁶ (often 50% of the course grade). The average and median grades for the ADS course are 93.5% and 96%. It is extremely unlikely the majority of the students in the course have proficiency of the material at the level equivalent to receiving an A in the course. Unfortunately, this type of grading breakdown can inflate students' grades, allowing institutions to ignore students' low proficiency of course material in the short term, while leading to long-term issues.

5.4.4 RQ4: Comparing Prerequisite Course Proficiency by Number of Prerequisites Taken at Our University

Transfer students enter our institution with different numbers of prerequisite courses for ADS already completed. Given their lower average score on the pre-term test, we next ask whether taking more of the prerequisite courses at our institution could result in better performance, perhaps because of better alignment of learning outcomes. To measure this, we grouped the transfer students based on how many prerequisites they took at our university (no transfer students took all of their prerequisites at another university). We then ran a Kruskal-Wallis H-Test, which is a non-parametric equivalent of an ANOVA, to determine whether there was a difference between transfer students' pre-term test scores based on how many courses they took at our university, ranging from one to five courses. The results were statistically

⁶The ADS course grade breakdown is online reading assignments (10%), clicker participation (5%), programming assignments (35%), midterm (20%), in-class quizzes (5%), and final exam (25%)

significant with the statistic of $stat = 10.19$ and a p-value of $\alpha = 0.04$. To determine which groups of transfer students differed, we then performed a Post-Hoc Dunn's test using a Bonferroni correction to correct for running multiple statistical tests. The results showed that there was no statistically significant difference between transfer students no matter how many prerequisites they took at our university.

These results suggest that the number of prerequisites a transfer student takes at our university may not impact their prerequisite course proficiency. The reason we see a global difference but not a difference in the Post-Hoc Dunn's is due to correcting for running multiple statistical tests. It may be that with a larger number of participants we would see a statistically significant difference between groups and be able to identify where these differences are.

5.5 Discussion

These results suggest that we may need to look for other factors that are leading to disparities in student outcomes for women and first-generation students. We caution that prerequisite course proficiency may still be an issue for these groups, as it's possible that the women and first-generation students who might have struggled have already self-selected out of computing before they reach the ADS course.

5.5.1 BLNPI students.

It appears that prerequisite course proficiency may be playing a role in the outcome disparities between BLNPI and Non-BLNPI students. We see that there is a gap between BLNPI and Non-BLNPI students' grades in all of their prerequisite courses leading into the ADS course, as well as in their pre-term test scores.

5.5.2 Transfer students.

For transfer students, we see the most sizable results in terms of differences in pre-term test scores and ADS final exam scores. We also find that pre-term test scores are correlated with

Table 5.5. Performance details on the ADS final exam and ADS course grade by demographic. A star (*) indicates a significant correlation for $\alpha = 0.05$ according to a Holm-Bonferroni-adjusted Mann-Whitney U test.

		Final Exam						Grade			
	Group	N	Mean	Med.	std. dev	U	p	Mean	Med.	std. dev	p
Gender	Women	102	81.2%	83.3%	10.2%	10780.0	0.44	93.1%	96.0%	10.2%	0.53
	Men	218	81.8%	84.0%	9.4%			93.6%	96.0%	8.8%	
BLNPI Status	BLNPI	21	78.8%	80.7%	10.8%	2618.0	0.31	92.5%	96.0%	7.3%	0.28
	Non-BLNPI	299	81.8%	84.0%	9.5%			93.5%	96.0%	9.4%	
Transfer Status	Transfer	86	79.3%	81.0%	10.1%	8079.0	0.02*	92.5%	96.0%	9.6%	0.28
	Non-Transfer	230	82.4%	84.0%	9.4%			93.8%	96.0%	9.2%	
FG Status	FirstGen	63	81.0%	81.3%	10.0%	7589.5	0.44	92.7%	96.0%	10.3%	0.39
	Non-FirstGen	257	81.8%	84.0%	9.5%			93.7%	96.0%	9.0%	

outcomes in the ADS course. Interestingly, we do not find strong evidence to suggest the gap by transfer status is related to the number of courses transfer students take at our university. This suggests a solution to the problem is not as simple as requiring transfer students to take all of their ADS prerequisites at our university. However, we caution that due to the small sample size of transfer students in each group, we may have seen statistically significant differences between groups if we had a larger population.

5.5.3 Limitations and Threats to Validity

The results of this study are limited in that they come from a single course offering of an Advanced Data Structures course at a research intensive university. However, our university curriculum does follow the ACM Curriculum Guidelines [25], which leads us to believe that the findings from this study may be similar to other institutions following the same guidelines.

Additionally, the pre-term test is not a validated instrument, which could lead to an incorrect estimation in students' prerequisite course proficiency. Students were also not provided with an incentive to complete the pre-term test and therefore may not have had the proper motivation to try and perform well. The pre-term test was also given in class and therefore students who did not show up were not included in the study, which may impact the results.

In terms of demographics, we had a small number of students for many of the populations we studied. This may have impacted our results in terms of statistical significance.

5.6 Summary

Prior work has shown that prerequisite course proficiency matters [28, 70]. This chapter extends our understanding of the importance and **impact** of prerequisite course proficiency by showing that it varies by demographic groups and how it relates to disparities in student outcomes. We did not see a substantial or statistically significant difference in prerequisite course proficiency for gender or first-generation status. However, we did find substantial disparities for BLNPI and transfer students. This is statistically significant for transfer students, but not

BLNPI students, likely due to their small numbers in our study. We find that prerequisite course proficiency is correlated with students' ADS course grades and final exam scores, and we find that transfer students have lower scores in the ADS course compared to non-transfer students. We see similarly substantial, but non-significant, results for BLNPI students. When looking at students' prerequisite course grades, we find that BLNPI and transfer students have statistically significantly lower grades compared to their majority counterparts. This suggests that the gap between BLNPI and Non-BLNPI and transfer and non-transfer students' knowledge begins earlier in the program, manifests itself as poor prerequisite course proficiency, and that poor prerequisite course proficiency correlates with their outcomes in later courses. Overall our study shows a disparity in prerequisite course proficiency between demographic groups, specifically for BLNPI and transfer students who are experiencing the worst effects.

5.7 Acknowledgements

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE-1650112, NSF Award #2121592, NSF Award #1712508 as well as a program assessment grant from the office of UC San Diego's Dean of Undergraduate Education.

Chapter 5, in part, is a reprint of the material as it appears in the Proceedings of the 53rd ACM Technical Symposium on Computer Science Education (SIGCSE 2022). Sophia Krause-Levy, Sander Valstar, Leo Porter, and William G. Griswold. "A Demographic Analysis on Prerequisite Preparation in an Advanced Data Structures Course". The dissertation author was the primary investigator and author of this paper.

Chapter 6

[Extent, Impact, Cause] An Evaluation of Prerequisite Proficiency Across Multiple Computer Science Courses

In the previous chapters we looked at impact and possible causes of low prerequisite course proficiency in both (1) a single course (ADS) and its prerequisites (Chapter 4, 5) and (2) a single key prerequisite (CS2 knowledge) across multiple courses (Chapter 5). In this chapter we expand this previous work using pre-term tests in three lower-division courses at our institution, finding that the results replicate.

6.1 Study Design

6.1.1 Hypotheses

Our study begins with the following hypotheses:

H-Extent: Low incoming prerequisite proficiency extends across the computing curriculum.

H-Impact: Low incoming prerequisite course proficiency impacts student outcomes and is greater for groups typically underrepresented in computing.

H-Cause: Causes of low incoming prerequisite proficiency include: (1) time between courses, (2) taking alternative prerequisites, (3) other factors.

Table 6.1. Majors breakdown.

Major	CS1.5		CS2		ORG	
	N	%	N	%	N	%
Math-Computer Science	23	29.11%	69	31.94%	136	41.98%
Computer Science	8	10.13%	42	19.44%	79	24.38%
Computer Engineering	3	3.80%	20	9.26%	39	12.04%
Bioinformatics	9	11.39%	14	6.48%	9	2.78%
Cognitive Science	11	13.92%	17	7.87%	18	5.56%
Other	25	31.65%	54	25.00%	43	13.27%

6.1.2 Population

We ran our study in CS1.5, CS2 and ORG in Fall 2022. Our inclusion criteria for students from each course includes students who consented to participate in the study and who took the pre-term test, the final exam and received a letter grade in the course. We excluded students who dropped the course, received the grade of a W or Pass/No Pass. This was done with the approval of our human subjects protocol. Table 6.1 shows the breakdown of our study population. Table 6.2 shows the demographics of the study population.

6.1.3 Data Collection and Cleaning

We obtained students' course grades and final exam scores in the courses (CS1.5, CS2, ORG), as well as all of the students' previous course grades for courses taken at our institution and student demographic information. All data was collected with approval of our human subjects protocol.

Pre-Term Test

In order to study students' level of incoming prerequisite knowledge we had students in CS1.5, CS2 and ORG complete a 15 minute pre-term test in-person in the first week of the course. Students' received a small amount of course credit for participating. The pre-term tests consisted of seven questions for CS1.5 and CS2 and nine questions for ORG.

Table 6.2. Number of participants (N) and percentage of Women; Black/Latinx/Native American/Pacific Islander (BLNPI); Transfer; First-generation (FirstGen) students.

CS1.5						
Demographic	Group	N	BLNPI	FirstGen	Women	Transfer
BLNPI Status	BLNPI	18	—	61%	39%	0%
	Non-BLNPI	61	—	15%	30%	16%
FirstGen Status	FirstGen	20	55%	—	40%	20%
	Non-FirstGen	54	11%	—	31%	9%
Gender	Women	25	28%	32%	—	8%
	Men	52	21%	23%	—	13%
Transfer Status	Transfer	10	0%	40%	20%	—
	Non-Transfer	68	26%	24%	34%	—
CS2						
Demographic	Group	N	BLNPI	FirstGen	Women	Transfer
BLNPI Status	BLNPI	19	—	37%	37%	26%
	Non-BLNPI	197	—	17%	29%	17%
FirstGen Status	FirstGen	41	17%	—	34%	29%
	Non-FirstGen	159	6%	—	30%	12%
Gender	Women	64	11%	22%	—	9%
	Men	147	8%	18%	—	21%
Transfer Status	Transfer	39	13%	31%	15%	—
	Non-Transfer	173	8%	17%	34%	—
ORG						
Demographic	Group	N	BLNPI	FirstGen	Women	Transfer
BLNPI Status	BLNPI	36	—	56%	39%	8%
	Non-BLNPI	288	—	25%	20%	7%
FirstGen Status	FirstGen	93	22%	—	24%	13%
	Non-FirstGen	216	6%	—	23%	3%
Gender	Women	73	19%	30%	—	4%
	Men	245	9%	29%	—	8%
Transfer Status	Transfer	22	14%	55%	14%	—
	Non-Transfer	295	11%	27%	23%	—

The questions on the pre-term test are questions on prerequisite course content instructors expect students to know coming into their course. In order to create these questions we first interviewed instructors who typically teach each of the courses (CS1.5, CS2, ORG) and instructors who teach the prerequisites to gain an understanding of what students are expected to know at the beginning of each course and the end of each prerequisite. From the interviews we created a list of key topics students were expected to know. We shared these topics with the instructors teaching the courses in Fall 2022 to confirm that the topics matched their expectations. Once confirmed, we selected questions for the pre-term test from prior research where possible. In cases where no such questions were available we worked with instructors who had previously taught the course to create the questions. Once the pre-term tests were created, we went back to the instructors teaching the courses in Fall 2022 to get their input and confirm the questions matched what they expected students to know. All of the pre-term test questions and a summary of where they came from can be found in the Appendix.

In addition to working with instructors to create the questions, we had the instructors complete a form where they were asked to estimate how well, on average, they believed students would do on each question on the pre-term test. This was also done with approval of our human subjects protocol.

Students' Previous Course Grades

Similarly to our previous study (Chapter 4), we used students' grades to calculate the average grade for prerequisite courses required to take the CS1.5, CS2 and ORG (*prerequisite GPA*) and the average grade for all courses taken at our institution excluding their prerequisite courses and any other CS courses taken at our institution (*non-prerequisite GPA*).

Before calculating the two measures we scraped our university's online course evaluating system for all relevant courses and used them to compute the z-score for each of students' previous course grades (A, B, C, D, F). We had access to the specific term and section of each course a student took. In some cases the distributions were not available at all for a course

Table 6.3. Non-prerequisite courses taken by students.

Course ID	CS1.5		CS2		ORG	
	N	%	N	%	N	%
MATH	66	20.37%	215	22.33%	480	28.5%
COGS	26	8.02%	56	5.82%	86	5.11%
PHYS	11	3.4%	57	5.92%	83	4.93%
CHEM	29	8.95%	63	6.54%	68	4.04%
BILD	19	5.86%	51	5.3%	60	3.56%
ECON	7	2.16%	36	3.74%	75	4.45%
Writing	56	17.28%	182	18.9%	384	22.8%
Other	95	29.32%	279	28.97%	403	23.93%
Average/Student	6	—	6	—	5	—
Median	5	—	6	—	5	—
Max	21	—	19	—	21	—
Min	1	—	1	—	1	—

because too few students were enrolled for the information to be available or the course was not eligible to be in the evaluation system. This occurred for 251 (4.9%) sections across 99 (12.1%) courses. We did not include courses where all students received the same grade. This occurred for 44 (0.01%) sections across 33 (4.04%) courses where all students received an A. In total we have 4,815 (94.23%) sections across 700 (85.68%) courses. Table 6.3 summarizes the kinds of non-prerequisite non-CS courses taken by these students.

Students' Demographic Information

Through our human-subjects protocol we were able to obtain demographic information for the students in our study including gender, race/ethnicity, first-generation college status and whether students transferred from another college or university. In some cases students' demographic information for a specific category was unknown.

We were only provided with a binary measure for students identified as Black, Latinx, Native American or Pacific Islander (BLNPI) versus students who identified as White or Asian (Non-BLNPI) to avoid accidentally deanonymizing students. For the case of gender, we were provided with the categories of women, men and non-binary. The number of non-binary students

was less than 5 in each course so they were excluded in the demographic analysis to avoid accidentally deanonymizing the students. For transfer status, there was a small population of students who were studying abroad at our institution or who were readmits. These students were also excluded from the demographic analysis in this chapter.

6.2 Results [H-Extent]: Low Incoming Prerequisite Proficiency Extends Across the Computing Curriculum

6.2.1 Incoming Prerequisite Proficiency

Table 6.4. Low, medium and high groups on the Pre-Term Test.

		CS1.5		CS2		ORG	
Group	Threshold	N	%	N	%	N	%
low	$0 \leq \text{score} < 50\%$	4	5.06	13	6.02	71	21.91
med	$50\% \leq \text{score} \leq 70\%$	14	17.72	31	14.35	187	57.72
high	$70\% < \text{score} \leq 100\%$	61	77.22	172	79.63	66	20.37

On the pre-term test students scored an average of 5.57 / 7 points (78.12%) in CS1.5, 5.42 / 7 points (77.45%) in CS2 and 5.52 / 9 points (61.37%) in ORG with the standard deviations of 1.21 (17.25%), 1.17 (16.78%) and 1.38 (15.28%) respectively. In order to provide a comparison to the work of Valstar et al. Table 6.4 shows student performance broken into the categories of low, medium and high performance [70]. We use the definitions provided by Valstar et al. where low performance is below 50% and high performance is above 70%. As shown in Table 6.4 the number of students that fall into the low and medium categories increases in courses farther into the curriculum (CS1.5 $\rightarrow$ CS2 $\rightarrow$ ORG).

Figure 6.1 shows the distribution of scores for each of the courses. It is a stacked bar chart including who did (*included*) and did not (*excluded*) meet the inclusion criteria for our study (taking the pre-term test, the final exam and receiving a letter grade in the course). Overall students are entering the courses with lower than expected incoming prerequisite proficiency, given that the questions on the pre-term test were created with the expectation that all students

should get the questions correct. The distributions on the pre-term test become larger as we look at courses later in the curriculum.

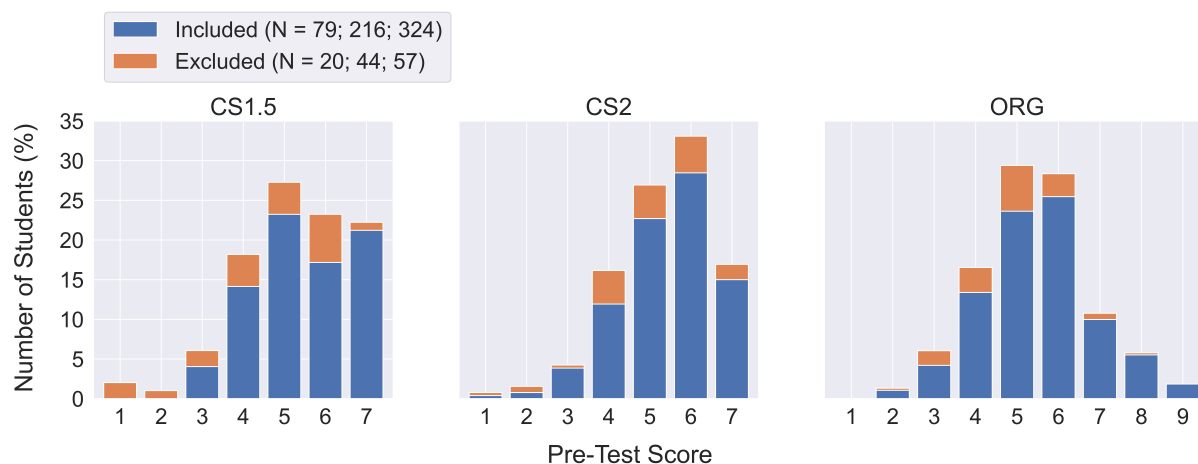


Figure 6.1. Distribution of scores on the pre-term test including students who are and are not included in the data for this study.

6.2.2 Common Misconceptions

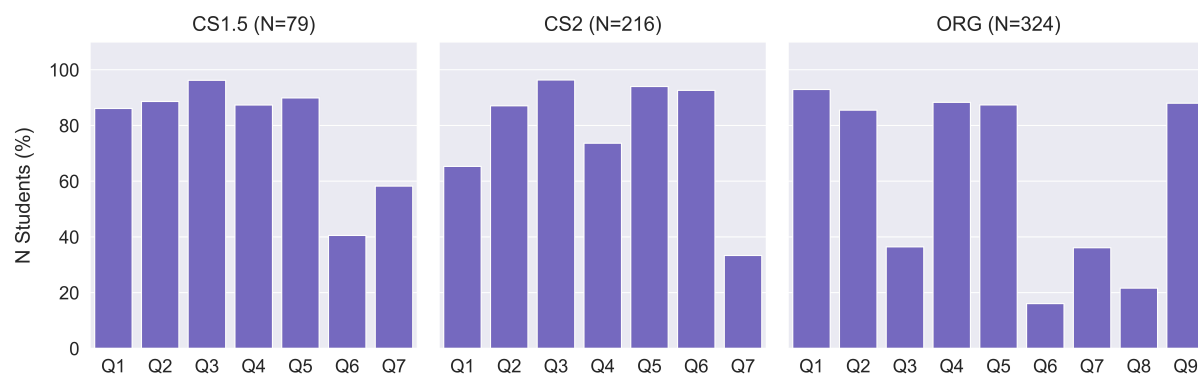


Figure 6.2. The percentage of students who got each question correct on the pre-term test.

In addition to looking at the pre-term test scores, we wanted to gain insight into which concepts students are struggling with the most. To do this we looked at the percentage of students that got each question correct (see Figure 6.2). Overall, students are performing well on the majority of the questions, though we continue to see the trend of students having lower scores as we get farther into the curriculum.

Once we identified the questions on which students were struggling the most, we determined what the most common incorrect answers are. A complete list of the questions and the percentage of students who selected each answer can be found in the Appendix. In some cases the numbers may not add up to 100% due to some students leaving the answer blank.

CS1.5

On the CS1.5 pre-term test, the questions students struggled with the most were question 6 (see Table 6.5) and question 7 (see Table 6.6). Question 6 and 7 both rely on a students' ability to trace through a segment of code. Question 6 asks students to trace through a segment of code and identify its purpose (printing out an array in reverse order). Only 41% of students got this question correct with the most common incorrect answers being off-by-one errors. Question 7 provides a segment of code with a code block missing. It asks students to select the code block that, if inserted, will make the code print the minimum value in an array. 58.2% of students got this question correct, with 24.1% of students selecting an answer where the wrong variable is updated with the minimum value.

Table 6.5. Question 6 on the CS1.5 Pre-Term Test.

Question 6. What does executing the following code do?

```
i = len(array)
while (i > 0):
    print(array[i-1])
    i = i - 1
```

Select **ONE**:

- ☐ A. Prints out the array in reverse order (40.5%) → CORRECT
- ☐ B. Prints out the array in order (3.8%)
- ☐ C. Prints out the array in reverse order but errors out at the last index (8.86%)
- ☐ D. Prints out the array in reverse order except the last element (20.3%)
- ☐ E. Prints out the array in reverse order except the first element (26.6%)

Table 6.6. Question 7 on the CS1.5 Pre-Term Test.

Question 7. The following code finds the **minimum** element in an integer array and prints out that minimum value. Assume that the array is guaranteed to have at least one element. The solution should work on any non-empty integer array of size 1 or greater.

```
minimum = array[0]
for value in array:
    # Insert code here
print(minimum)
```

What is the missing code that goes in the space above?

Select **ONE**:

- ☐ A.

```
if (value <= minimum):
    value = array[value]
```

 (24.1%)
- ☐ B.

```
if (value != minimum):
    minimum = value
```

 (7.6%)
- ☐ C.

```
if (value < minimum):
    minimum = value
```

 (58.2%) → CORRECT
- ☐ D.

```
if (value > minimum):
    minimum = value
```

 (10.1%)

CS2

On the CS2 pre-term test, the question students struggled with the most was question 7 (see Table 6.7). Question 7 is a false-swap question. Two variables are created in the main method and then passed to the method func where the variables' values are swapped. The method is void and does not return anything. Once the method returns, the main method prints out the values stored in the variables. In Java values are passed by value so the variables in the main method are not updated by the call to g and still store their original values. Only 33.3% of students got this question correct, with 59.3% believing the values stored in the variables in the

main method have been swapped. This shows a misunderstanding of pass-by-value. Students in CS2 enter the course having taken at least two computing courses, with at least one previous course in Java (CS1.5 or CS1-Acc).

Table 6.7. Question 7 on the CS2 Pre-Term Test.

Question 7. Consider the following code:

```
public static void main(String[] args)
{
    int a = 8;
    int b = 25;
    funct(a, b);
    System.out.print(a + ", " + b);
}

public static void funct(int a, int b)
{
    int temp = a;
    a = b;
    b = temp;
}
```

What is the output of the code above?

Select **ONE**:

- ☐ A. (33.3%) → CORRECT
- ☐ B. (59.3%)
- ☐ C. Neither A nor B is correct (6.9%)

ORG

On the ORG pre-term test, the questions students struggled with the most were questions 6 and 7 (see Table 6.9 and Table 6.10). Questions 6 and 7 refer to the same program written in Java (see Table 6.8). The purpose of these questions is to test students' understanding of memory management and pass-by-value. The program contains two classes: Car and Q6. The Car class

Table 6.8. The program for Questions 6-8 on the ORG Pre-Term Test.

Consider the following program for questions 6-8:

```
1.  class Car {
2.      public String color;
3.      public Car(String color) {
4.          this.color = color;
5.      }
6.  }
7.
8.  class Q6 {
9.      public static void g(Car c1, Car c2) {
10.         c1 = c2;
11.         c1.color = "blue";
12.     }
13.
14.     public static void main(String[] args) {
15.         Car car1 = new Car("orange");
16.         Car car2 = new Car("green");
17.         g(car1, car2);
18.         System.out.println(car1.color + ", " + car2.color);
19.     }
20. }
```

contains the field `color`, for the color of the car and a one argument constructor to set the initial color of the `Car` object. The class `Q6` contains a `main` method that creates two `Car` objects and stores the references in the variables `car1` and `car2`. The `main` method then calls the method `g` and once the method returns, it prints the colors of both `Car` objects.

The method `g` has the parameters of two `Car` objects, `c1` and `c2`. Inside of the method the reference `c1` is updated to store the reference in `c2` (`c1 = c2;`). Then the field `color` for the reference stored in `c1` is set to ‘blue’. Only 16.1% of students got the question correct. 48.2% of students chose the most common incorrect answer which was that both `Car` objects color was now ‘blue’. The correct answer was that the call to `g` only changes the color of the object referred to by `c2` in the `main` method.

Table 6.9. Question 6 on the ORG Pre-Term Test.

Question 6. What does the program print?

Select **ONE**:

☐ A. orange , green (15.1%)

☐ B. blue , blue (48.2%)

☐ C. blue , orange (18.5%)

☐ D. green , orange (1.54%)

☐ E. orange , blue (16.1%) → CORRECT

Table 6.10. Question 7 on the ORG Pre-Term Test.

Question 7. If line 10 was changed to `c2 = c1;`, what would the program print?

Select **ONE**:

☐ A. blue , green (36.1%) → CORRECT

☐ B. blue , blue (42.9%)

☐ C. orange , orange (5.6%)

☐ D. green , orange (3.1%)

☐ E. orange , blue (12.0%)

Question 7 refers to the same program but the line `c1 = c2;` in the method `g` is changed to `c2 = c1;`. In this case 36.1% of students got the question correct with 42.9% of students still

believing both objects having their color updated to “blue”. These results indicate that students are struggling with understanding memory management and have a misunderstanding of Java references and objects. Students entering ORG should have taken at least two courses in Java by this point.

6.2.3 Instructors Often Overestimate Students’ Knowledge

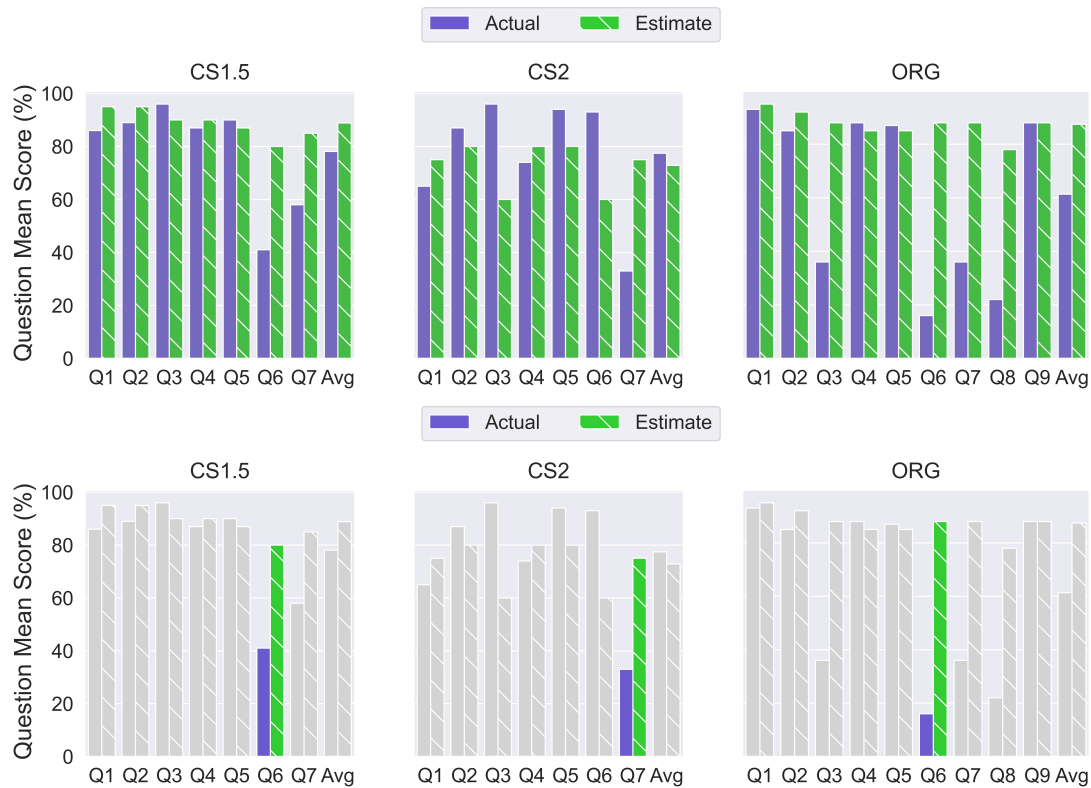


Figure 6.3. (a) The percentage of students to get each question correct on the pre-term test correct (Actual) and the instructors’ estimation of the percentage of students who would get each question correct (Estimate). (b) The same graph highlighting the question students’ struggled with the most.

Figure 6.3 is the same graphs as Figure 6.2 with the addition of the instructor’s estimations of how well students’ would do on each question and the average score on the pre-term test. There are some questions for which instructors are making good estimates on how well students’ will perform, but there are others for which instructors are greatly overestimating students’

abilities. This is especially true for the questions students' struggled the most on. This may create challenges for the students, as they are being treated as if they know these topics.

6.3 Results [H-Impact]: Low Incoming Prerequisite Proficiency Impacts Student Outcomes with Unequal Outcomes for Students from Underrepresented Groups

6.3.1 The Relationship between Prerequisite Proficiency and Final Exam Scores

To understand whether there is a relationship between incoming prerequisite proficiency and student outcomes in the courses, we calculated the Spearman correlations between the course outcomes (final exam score and course grade) and the pre-term test score for each course (see Table 6.11). Figure 6.4 shows a scatter plot plotting the two outcome measures against the pre-term test for each course. Students' final exam scores and course grades are statistically significantly correlated with the pre-term test.

6.3.2 Incoming Prerequisite Course Proficiency by Demographic

In order to understand whether there is a difference in student's incoming prerequisite course proficiency by demographic pair we looked at the differences in student scores on the pre-term tests. To determine if the differences are statistically significant, we ran a Mann-Whitney U test for each pair (see Table 6.12). In all cases the underrepresented groups had lower average scores on the pre-term test, though not all of the differences are statistically significant. We see the biggest differences by first-generation status.

6.3.3 Comparing Demographic Pairs' Prerequisite Grades

To see if there were differences by demographic pairs in students' prerequisite grades, we ran a Mann-Whitney U test comparing each demographic pairs' course grades for each of the prerequisites of the courses. For CS1.5 this includes CS1, for CS2 it includes CS1 and CS1.5

Table 6.11. Spearman correlations for pre-term test score with a comparison to final exam score and course grade: correlation with p-values. A star (*) indicates a significant correlation for $\alpha = 0.05$.

		CS1.5				CS2				ORG			
		Final Exam		Grade		Final Exam		Grade		Final Exam		Grade	
Comparison		r	p	r	p	r	p	r	p	r	p	r	p
Pre-Term Test Score		0.35	<0.01*	0.37	<0.01*	0.42	<0.01*	0.29	<0.01*	0.23	<0.01*	0.14	0.01*



Figure 6.4. Scatter plot of student's score on the pre-term test to their outcomes in the course.

Table 6.12. Mann-Whitney U test for differences in pre-term test scores by demographic. + and - for the mean and median diff indicate whether women, BLNPI, FirstGen, and transfer students had higher or lower scores. A star (*) indicates a significant correlation for $\alpha < 0.05$.

Women and Men					
	N	Pre-Term Test Scores		Mann-Whitney U	
		Mean diff	Median diff	U	p
CS1.5	25 / 52	-7.6	-14.3	486.5	0.03*
CS2	64 / 147	-1.1	0.0	4506.5	0.31
ORG	73 / 245	-6.3	-11.1	7193.0	<0.01*

BLNPI and Non-BLNPI					
	N	Pre-Term Test Scores		Mann-Whitney U	
		Mean diff	Median diff	U	p
CS1.5	18 / 61	-5.6	-14.3	455.0	0.13
CS2	19 / 197	-3.3	-14.3	1661.5	0.20
ORG	36 / 288	-6.9	-11.1	4037.0	0.01*

FirstGen and Non-FirstGen					
	N	Pre-Term Test Scores		Mann-Whitney U	
		Mean diff	Median diff	U	p
CS1.5	20 / 54	-5.9	0.0	452.5	0.14
CS2	41 / 159	-12.5	-14.3	2087.5	<0.01*
ORG	93 / 216	-4.6	-11.1	8321.5	0.01*

Transfer / Non-Transfer					
	N	Pre-Term Test Scores		Mann-Whitney U	
		Mean diff	Median diff	U	p
CS1.5	10 / 68	-1.4	7.1	329.5	0.44
CS2	39 / 173	-1.8	0.0	3351.0	0.47
ORG	22 / 295	-4.7	-5.6	2852.0	0.17

and for ORG it includes CS1, CS1.5 and CS2. Figure 6.5 shows the differences in pre-term test scores for each demographic pair. We see the largest differences in outcomes by BLNPI and first-generation status, with BLNPI and first-generation students having lower grades. Due to the small number of transfer students, and even smaller population who took their prerequisites at our institution, we chose not to include them in this analysis.

6.3.4 Differences in Student Outcomes

Next we wanted to know to what extent there were difference between demographic pairs outcomes in the courses. To do this, we ran a Mann-Whitney U test comparing each demographic pairs' course grades and final exam scores. We find that the underrepresented groups had unequal outcomes across all courses, with the differences being largest for BLNPI and FirstGen students. Table 6.13 for final exam scores and Table 6.14 for course grades. Figure 6.6 shows demographic pairs mean pre-term test scores, final exam scores and course grades.

6.4 Results [H-Causes]: Possible Causes of Low Incoming Prerequisite Proficiency

6.4.1 Time Between Courses

Similar to our previous study, we operationalized the passage of time as the sum of the months elapsed since taking the main prerequisite predecessors of each course (CS1 for CS1.5, CS1.5 for CS2 and CS2 for ORG). We again excluded the other prerequisites on the presumption that their immediate successors refreshed the knowledge from those courses, given the tight dependence in subject matter (see Table 6.15). The results are only significant for CS2 and not in the hypothesized direction for any of the courses (as the time gap increases so does the gap in prerequisite knowledge).

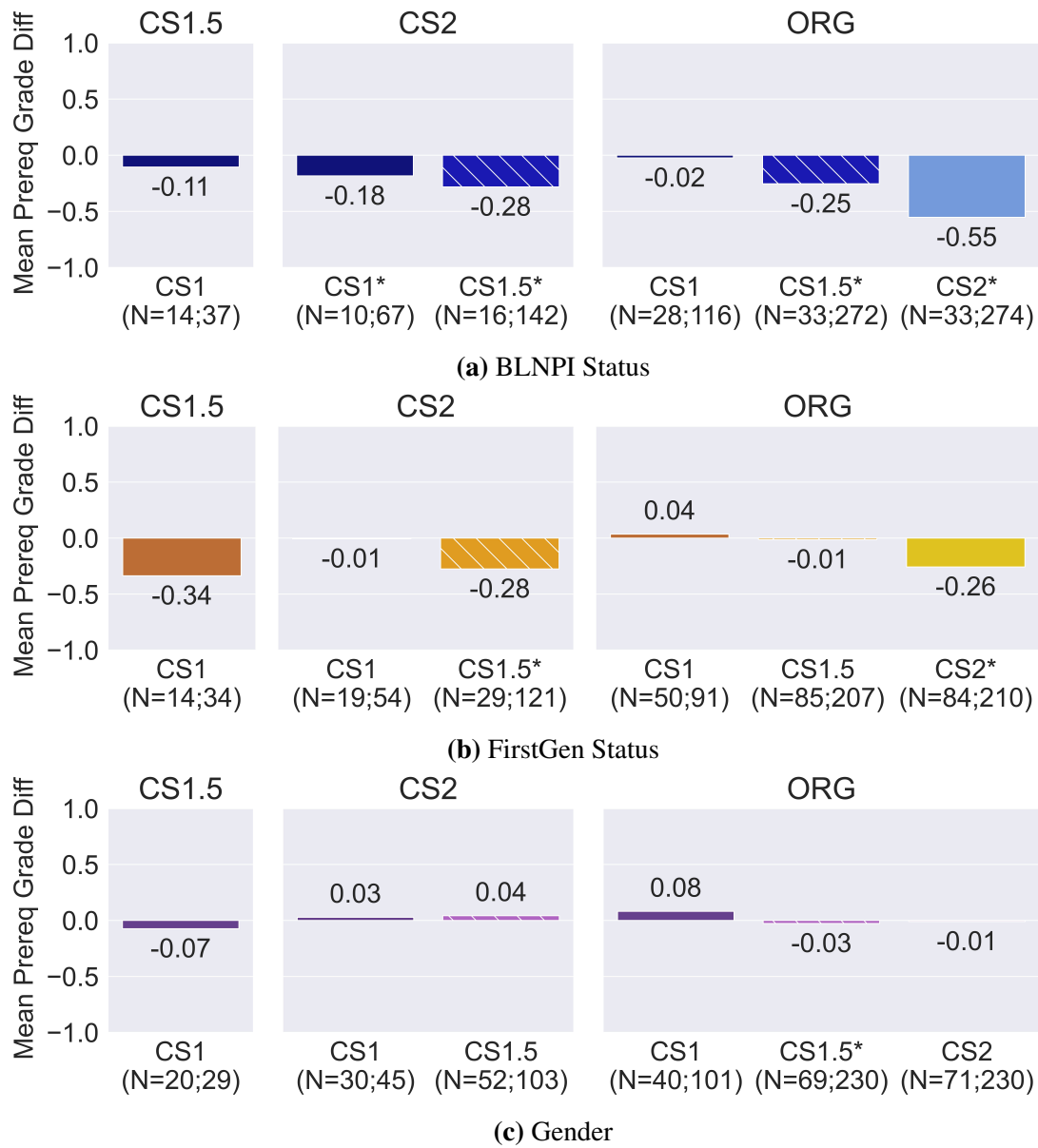


Figure 6.5. The difference in average prerequisite course grades. + and - for the mean and median diff indicate whether women, BLNPI, FirstGen, and transfer students had higher or lower scores. A star (*) next to the course name indicates a significant correlation for $\alpha = 0.05$.

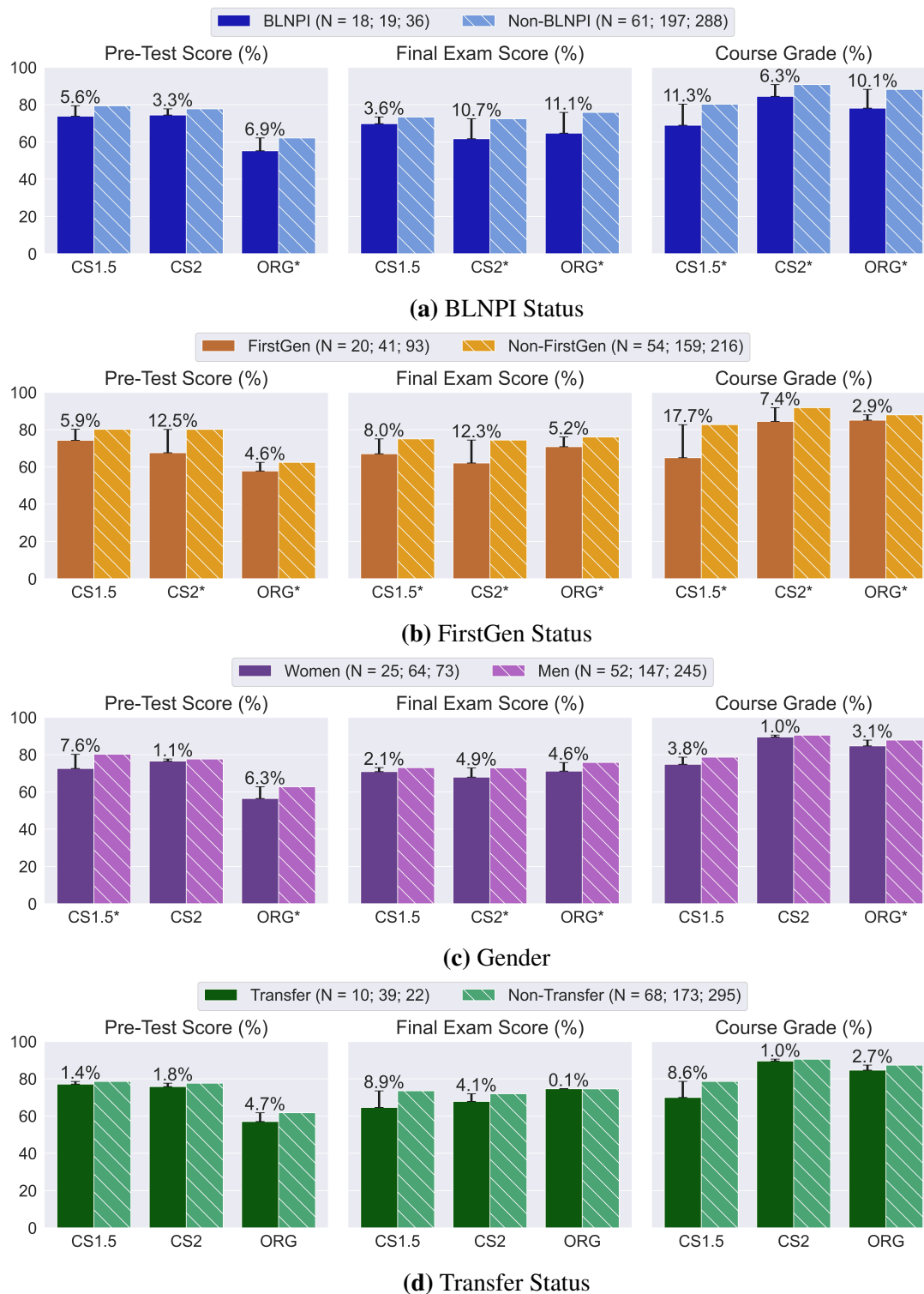


Figure 6.6. The average pre-term test score, final exam score and course grade broken down by course and demographic. A star (*) next to the course name indicates a significant correlation for $\alpha = 0.05$.

Table 6.13. Mann-Whitney U test for differences in final exam scores by demographic: Women and Men; BLNPI and non-BLNPI; FirstGen and non-FirstGen; and Transfer and non-Transfer. star (*) indicates a significant correlation for $\alpha < 0.05$.

Women and Men					
	N	Final Exam Scores		Mann-Whitney U	
		Mean diff	Median diff	U	p
CS1.5	25 / 52	-2.1	-3.9	557.5	0.16
CS2	64 / 147	-4.9	-9.4	3941.0	0.03*
ORG	73 / 245	-4.6	-4.9	7285.5	0.01*

BLNPI and Non-BLNPI					
	N	Final Exam Scores		Mann-Whitney U	
		Mean diff	Median diff	U	p
CS1.5	18 / 61	-3.6	-3.2	445.0	0.11
CS2	19 / 197	-10.7	-17.5	1335.0	0.02*
ORG	36 / 288	-11.1	-14.7	3219.0	<0.01*

FirstGen and Non-FirstGen					
	N	Final Exam Scores		Mann-Whitney U	
		Mean diff	Median diff	U	p
CS1.5	20 / 54	-8.0	-5.8	341.5	0.01*
CS2	41 / 159	-12.3	-10.0	2230.5	<0.01*
ORG	93 / 216	-5.2	-5.9	8214.5	0.01*

Transfer / Non-Transfer					
	N	Final Exam Scores		Mann-Whitney U	
		Mean diff	Median diff	U	p
CS1.5	10 / 68	-8.9	-3.0	239.0	0.07
CS2	39 / 173	-4.1	-5.0	3019.5	0.15
ORG	22 / 295	0.1	-2.9	3222.0	0.48

Table 6.14. Mann-Whitney U test for differences in course grades by demographic: Women and Men; BLNPI and non-BLNPI; FirstGen and non-FirstGen; and Transfer and non-Transfer. star (*) indicates a significant correlation for $\alpha < 0.05$.

Women and Men					
	N	Course Grade		Mann-Whitney U	
		Mean diff	Median diff	U	p
CS1.5	25 / 52	-3.8	-4.0	534.0	0.10
CS2	64 / 147	-1.0	0.0	4450.0	0.24
ORG	73 / 245	-3.1	-3.0	7676.0	0.03*

BLNPI and Non-BLNPI					
	N	Course Grade		Mann-Whitney U	
		Mean diff	Median diff	U	p
CS1.5	18 / 61	-11.3	-20.0	344.0	0.01*
CS2	19 / 197	-6.3	-7.0	1190.5	<0.01*
ORG	36 / 288	-10.1	-10.0	3064.0	<0.01*

FirstGen and Non-FirstGen					
	N	Course Grade		Mann-Whitney U	
		Mean diff	Median diff	U	p
CS1.5	20 / 54	-17.7	-26.0	227.0	<0.01*
CS2	41 / 159	-7.4	-7.0	2163.5	<0.01*
ORG	93 / 216	-2.9	-3.0	8528.5	0.01*

Transfer and Non-Transfer					
	N	Course Grade		Mann-Whitney U	
		Mean diff	Median diff	U	p
CS1.5	10 / 68	-8.6	-10.0	228.5	0.05*
CS2	39 / 173	-1.0	0.0	2964.0	0.09
ORG	22 / 295	-2.7	-3.0	2997.5	0.27

Table 6.15. Pearson correlation for performance in prerequisites minus pre-term test score with time since courses taken. A star (*) indicates a significant correlation for $p = 0.05$.

Comparison	CS1.5			CS2			ORG		
	N	r	p	N	r	p	N	r	p
Pre-Term Test Score - Main Prereq GPA	51	0.04	0.78	158	0.18	0.02*	307	0.08	0.17

6.4.2 Taking Alternatives to Primary Prerequisites Inside and Outside of Our University

Because we are looking at courses earlier in the curriculum there are no options for students to take courses other than the main prerequisites (alternative prerequisites) at our institution. However, many of the students entering these earlier courses either have Advanced Placement (AP) credit or have transfer credit from taking an equivalent course at another institution. To determine whether there is a difference in outcomes between students who took their prerequisites at our institution compared to those who did not, we ran Mann-Whitney U tests for pre-term test, final exam scores and course grades comparing the two groups based on whether they took the immediate predecessor course at our institution. We again excluded the other prerequisites on the presumption that their immediate successors refreshed the knowledge from those courses (see Chapter 4 and [72]). The results can be seen in Table 6.16. None of the results were statistically significant and therefore do not suggest that taking courses outside of our institution impacts students' incoming prerequisite proficiency or success in the courses.

6.4.3 Other Factors than Prerequisite Learning Impact Student Performance

To determine whether other factors beyond subject matter knowledge impact student success in a course we looked at the correlation between students' non-prerequisite GPA and three measures related to prerequisites: course grade, final exam score and pre-term test. Table 6.17 shows the Spearman correlations for each measure. In all cases the results were strong and

Table 6.16. Performance details by taking the immediate predecessor at our institution. + and - for the mean and median diff indicate whether Alt-Track students had higher or lower scores. A star (*) indicates a significant correlation for $\alpha < 0.05$ according to a Mann-Whitney U test.

Pre-Term Test Alt-Track and Main-Track					
	N	pre-term test		Mann-Whitney U	
	Alt / Main	Mean diff	Median diff	U	p
CS1.5	27 / 52	6.7	14.3	868.0	0.96
CS2	55 / 161	4.1	14.3	5131.0	0.97
ORG	13 / 311	1.1	11.1	2196.5	0.71
Final Exam Score Alt-Track and Main-Track					
	N	Final Exam Score		Mann-Whitney U	
	Alt / Main	Mean diff	Median diff	U	p
CS1.5	27 / 52	4.2	6.4	868.5	0.96
CS2	55 / 161	2.6	5.0	4830.0	0.84
ORG	13 / 311	6.2	6.9	2600.5	0.96
Course Grade Alt-Track and Main-Track					
	N	Course Grade		Mann-Whitney U	
	Alt / Main	Mean diff	Median diff	U	p
CS1.5	27 / 52	7.5	10.0	927.0	0.99
CS2	55 / 161	1.4	0.0	4677.5	0.77
ORG	13 / 311	1.8	4.0	2388.5	0.88

statistically significant. We do see that the correlations for the prerequisite GPA are larger than those of the non-prerequisite GPA.

To understand the relative importance of each measure (pre-term test, prerequisite GPA, and non-prerequisite GPA) we ran two multiple linear regression analyses for each course. For this part of the analysis we only included students that had both GPA metrics. One for the dependent variable (DV) course grade and the other for final exam score, both with the independent variables (IVs) non-prerequisite GPA, prerequisite GPA, and pre-term test (Table 6.18, when understanding the coefficients, keep in mind that the GPA measures are z-scored). Similar to Chapter 4, we employed the method of calculating the increase in R-squared that each variable produced when it was added to the model that already contained all other independent variables (the r-squared difference). Prerequisite GPA dominates non-prerequisite GPA and the pre-term test. The only exception is CS2 for the dependent variable of the final exam where the pre-term test r-squared difference is largest.

6.5 Discussion

6.6 Limitations and Risks to Validity

Our results are based on a single offering of CS1.5, CS2 and ORG at one institution limiting generalizability. However, as previously stated, the curriculum is quite common and informed by the ACM Curriculum guidelines [25].

The same risks to validity from Chapter 4 are true here as well. The pre-term tests are short and not validated. This could result in over- or under-estimating prerequisite knowledge, depending on the actual shortcomings of the pre-term test. In this case students were given a small amount of credit to participate with the hope that they would be more likely to take the pre-term test seriously. Still, students may have lacked sufficient motivation given that the credit was not tied to their score.

In the ORG pre-term test there was an issue with two of the questions. Question 3's

Table 6.17. Spearman correlations for performance in course with a comparison to final exam score and course grade: correlation with p-values. A star (*) indicates a significant correlation for $p = 0.05$.

Comparison	CS1.5						CS2						ORG					
	Final Exam			Grade			Final Exam			Grade			Final Exam			Grade		
	r	p		r	p		r	p		r	p		r	p		r	p	
Prereq GPA	0.53	<0.01*		0.58	<0.01*		0.33	<0.01*		0.47	<0.01*		0.47	<0.01*		0.55	<0.01*	
Non-Prereq GPA	0.47	<0.01*		0.63	<0.01*		0.30	<0.01*		0.36	<0.01*		0.45	<0.01*		0.54	<0.01*	
Pre-Term Test	0.35	<0.01*		0.37	<0.01*		0.42	<0.01*		0.29	<0.01*		0.23	<0.01*		0.14	0.01*	

Table 6.18. Multiple linear regression statistics for dependent variables of final exam score and course grade. A star (*) indicates a significant correlation for $p = 0.05$.

IV	DV: Final Exam						DV: Grade					
	r^2	coeff	std. err	t-stat	p	r^2 diff	r^2	coeff	std. err	t-stat	p	r^2 diff
CS1.5	Prereq	11.13	3.41	3.27	<0.01*	0.164		15.58	4.15	3.75	<0.01*	0.191
	NonPrereq	4.89	2.64	1.85	0.07	0.053	0.389	6.67	3.22	2.07	0.04*	0.058
	Pre-Term Test	-1.08	1.36	-0.79	0.43	0.010		0.50	1.66	0.30	0.76	0.001
CS2	Prereq	11.78	3.62	3.25	<0.01*	0.060		11.08	1.96	5.67	<0.01*	0.163
	NonPrereq	-0.04	2.93	-0.01	0.99	0.000	0.290	0.07	1.58	0.04	0.97	0.000
	Pre-Term Test	5.19	1.29	4.01	<0.01*	0.092		1.92	0.70	2.74	0.01*	0.038
ORG	Prereq	10.54	1.89	5.59	<0.01*	0.078		12.49	1.42	8.81	<0.01*	0.173
	NonPrereq	4.14	1.46	2.83	<0.01*	0.020	0.319	3.51	1.10	3.19	<0.01*	0.023
	Pre-Term Test	2.50	0.55	4.52	<0.01*	0.051		0.99	0.42	2.38	0.02*	0.013

method had the wrong return type (int instead of String) in the method signature. This did not impact the correctness of the question, which asked students' to choose the correct segment of code to complete the method. Question 5 had two correct answers which were selected by the majority of students (87.4% of students). All students completed the pre-term test before the end of 15 minutes, with many leaving early. This suggests that these issues did not impact their ability to complete the pre-term test.

6.7 Discussion

We find that our results are consistent with the results of the ADS course in Chapter 4. In addition, students do not perform as well as instructors expect on the pre-term test. The distribution of scores on the pre-term test increases in later courses, suggesting that there is a larger spread in students' incoming prerequisite knowledge as we get farther into the curriculum. While instructors are good at estimating students' incoming knowledge, there is a large discrepancy in instructor expectations and estimations when it comes to the concepts with which students are struggling. Using pre-term tests allows instructors to gain a better understanding of what students know upon arrival to their course and may allow them to spend time better preparing students.

There is a strong correlation between pre-term test scores and student success in the course, suggesting that students' incoming prerequisite proficiency is impacting their success in the course. There is also a trend across all outcome measures that there are unequal outcomes for students from underrepresented groups. This is a problem, as it suggests we, as instructors, are not adequately supporting these students, Chapter 8 discusses this in more detail.

In terms of causes, we see nothing to suggest that the time between taking prerequisites and the subsequent course impacts student outcomes or taking prerequisites outside of our institution. We do see a relationship between students' non-prerequisite GPA and their success in subsequent courses suggesting that other factors are also at play in students' success.

6.8 Summary

The results of this study suggest that instructors overestimate students' incoming prerequisite proficiency. Students' incoming prerequisite proficiency is correlated with their outcomes in the course, suggesting that low prerequisite proficiency may be impacting student outcomes. We do not see anything to suggest that taking courses outside of our institution or the passage of time is affecting students' prerequisite proficiency or outcomes in the course. It does seem like other factors are playing a role in students' course grades and may be a part of why we see students succeeding despite their low incoming prerequisite proficiency.

6.9 Acknowledgements

This material is based upon work supported by the NSF GRFP under Grant No. DGE-1650112, NSF Award #2121592, a UCSD Sloan Scholar Fellowship, a UCSD STARS Fellowship, and a Gates Millennium Scholarship.

Chapter 6, in part, is currently being prepared for submission for publication of the material. Sophia Krause-Levy, Adrian Salguero, Rachel S. Lim, Jelena Trajkovic, Leo Porter, and William G. Griswold. The dissertation author was the primary investigator and author of this material.

Chapter 7

[Cause] Instructor Perspectives on Prerequisite Courses in Computing

In the previous chapters we found that student performance on prerequisite course content varies widely, even when students continue to progress further into the computing curriculum [28, 70, 72]. Given that mastering prior course content is tied to student success and may be an issue impacting the diversity of the field, this chapter seeks to gain a better understanding of the problem. Specifically, prior work both inside and outside computing has not examined the perspectives of a key stakeholder who controls the curriculum and establishes course prerequisites in the first place: the instructor. As such, this work asks the questions: 1) what are instructor views about the purpose of prerequisite courses and 2) whether they believe this purpose is presently being fulfilled? Answers to these questions could inform remediation of the problem.

In this study, we interviewed twenty-one instructors from two institutions and conducted a phenomenographic analysis on the interviews to understand the range of instructor perspectives. We find that instructors report prerequisites serving varying purposes. These purposes include ensuring students enter later courses with a certain level of proficiency, helping to avoid repeated teaching of concepts, and helping students by providing an efficient path through the curriculum. Instructors also vary in their beliefs about whether prerequisites are currently fulfilling their purpose with significant differences in views about who is responsible (among students, instruc-

tors, and the institution) for ensuring students are learning what is expected. In addition, there are varying ideas for why prerequisites may not be fulfilling their purpose, such as academic integrity issues, grade inflation, the COVID-19 pandemic, pressure to graduate students within a certain number of years, and student evaluations impacting instructor motives.

In view of these findings, we discuss how instructors' perspectives relate to issues surrounding who is responsible for prerequisite courses fulfilling their purpose, as well as the many conflicting pressures on instructors and relevant theories of learning.

The work most similar to our own is a multi-methods study that investigated the effects of making a lecture prerequisite optional for a microbiology lab course by Sato et al. [60]. While not the focus of this work, the authors describe students' perspectives as they more broadly relate to prerequisite courses, providing some insight into how students view the purpose of prerequisite courses.

In computing, Walker described the most common purposes of a prerequisite course as 1) content - allowing another course to build on its foundation; 2) maturity - allowing a course to build on previous experiences and maturity outside of the specific content; 3) filtering - filtering out students who are not committed and limiting enrollment in courses; 4) requirement enforcement - using prerequisites to show students a path through the curriculum, irrespective of required course content; and 5) historical - a long time ago a course was determined to be a prerequisite [74]. However, this may not include the larger instructor perspective. In Section 7.3.1, we will compare the purposes identified by Walker and touch on the student perspective from Sato et al. with those identified in this paper [60, 74].

7.1 Study Design

The motivation of our study was to better understand the range of perspectives held by CS instructors about prerequisites.

7.1.1 Research Questions

Our research questions are as follows:

RQ-1: What are instructor views on the purpose of prerequisite courses?

RQ-2: What are instructor views on whether the purpose of prerequisite courses is being fulfilled?

7.1.2 Course Context

The course context is summarized in Section 3.2. We interviewed 21 instructors who teach a wide range of undergraduate courses at the University of California San Diego and California State University, Long Beach. To obtain the full range of instructor perspectives, we chose to interview instructors who teach a wide range of computing courses (e.g., theory, systems, applications and both lower- and upper-division courses). The majority of the instructors interviewed teach multiple courses, but we recruited each instructor with a specific course in mind.

7.1.3 Sample Selection

Before conducting any interviews, we first compiled a list of instructors across both of the institutions who either recently taught or were currently teaching the computer science courses in which we were interested. We reached out to each instructor by email to ask them to participate in our study. This was done with the approval of our human subjects protocol. The interviews lasted one hour, and instructors were given a \$50 gift card in compensation for their time. In total, we recruited 26 instructors and 21 participated in our study. Fifteen of the instructors were from the research-intensive institution and six were from the Hispanic-serving institution.

7.1.4 Interview Method

In order to make sure our research questions were addressed and that we were able to get complete responses from the instructors, we chose to use semi-structured interviews. The

interviews were conducted over Zoom and recorded. After the interviews were completed, they were transcribed and anonymized.

The interviews were semi-structured where the interviewer would ask prepared questions and additional follow up questions when necessary. The follow up questions generally revolved around asking the interviewee to provide further detail to their response of a previous question. At each interview two members of the research team were present. The primary interviewer was the same for all interviews. The secondary team member would introduce themselves at the beginning of the interview, explain their role and then observe the interview with their camera turned off. They would ensure the interviewer did not miss any questions and ask follow-up questions after all prepared questions were answered.

Before conducting our interviews, we developed an initial set of interview questions that was reviewed by the research team. We reused a general question on prerequisites from the work of Sato et al.. This question, broken into two parts, can be found in Section 4.1.3. These interview questions were then piloted with three instructors who were not going to be selected for our analysis. After each of these interviews, we asked for feedback from the interviewees and updated the questions as needed.

During the interviews, we asked the participants about their views on prerequisite courses in general and within the context of a course they teach. We did not provide a definition of prerequisite courses to allow us to gain insight into how instructors think about prerequisite courses and what they believe the term encompasses. Depending on the individual instructor, the definition may or may not have included prerequisite courses outside of computing.

7.1.5 Data Collection and Cleaning

All interviews were conducted through Zoom using its video- and audio-recording features. When an instructor was scheduled for an interview, they were given an online consent form to confirm they were willing to have their video and audio recorded per our approved human subjects protocol. In the event a stakeholder was not willing to have their video recorded,

they were asked to turn off their camera for the duration of the interview. Once the interviews were completed, the recordings were transcribed for analysis and the original recordings were deleted.

7.1.6 Details of Analysis

Phenomenography is a method used in qualitative research that focuses on understanding the variation in how people experience a phenomena [27]. We chose to use phenomenography as it allowed us to understand the range of perspectives of instructors on prerequisites. This means our results encompass the breadth of perspectives and are not meant to provide a direct comparison between individual responses. The analysis was performed by three of the researchers of this study [30]. The steps of analysis were derived by Carbone et al. [10, 71].

Step 0: Reading the Interviews. The authors examined each of the transcripts, reading them multiple times to gain an understanding of scope of the interviews and the instructor perspectives.

Step 1: Individually Labeling. Each author extracted excerpts associated with each research question from the transcripts and created and applied their own labels.

Step 2: Extracting Categories of Variation. The labeled excerpts were discussed and categorized to organize the variation in instructor's perspectives. These categorizations were then mapped back onto the instructors from whom the excerpts were taken.

Step 3: Constructing Categories of Description. The extracted categories of variation (from Step 2) were then used to create a hierarchical structure of categories and subcategories. This was done through discussion of the authors. Any disagreements were discussed until there was a consensus.

7.2 Results

In this section we will discuss the results of our phenomenographic analysis in terms of the hierarchical structure of categories of description and subcategories identified per research

question. The categories are presented in bold and italic, followed by subcategories in bold. We provide quotes where relevant, with an anonymous instructor identifier included (I1-I21). These responses encompass the range of views of instructors and therefore it is important to point out that not all instructors share all of these viewpoints.

7.2.1 RQ1: Purpose of Prerequisite Courses

We present hierarchical categories for instructor's beliefs on the purpose of prerequisite courses. While this analysis was informed by the complete instructor interview, it focused on responses to the question "In general, what do you think the purpose of prerequisite courses is?" This is the same question asked by Sato et al. [60].

Student Preparation.

Baseline and/or Solid Foundation. Instructors believe the role of prerequisite courses is to provide students with knowledge and skills to build upon. The exact wording of answers varied in terminology, however the wording matters as it offers insight into the perspectives of the instructors. When defining the purpose in relation to course content, instructors used the phrases: "mastered," "have seen," "have the necessary skill and knowledge." In relation to how well the students should be prepared instructors used the phrases: "succeed at the highest level," "minimize struggle," "success," "ensure preparation," and "competency." The minimal requirement of having "seen" a topic can be understood as a useful precursor to increasing mastery of it later.

Preventing Moving Forward without Required Knowledge. Instead of preparing students for the following course, other instructors framed the purpose as a way of preventing students from moving forward without the required knowledge.

"It's also a place where we need to stop them if we know they're going to be unsuccessful" - I11

Instructor Preparation.

Assumptions of Student Knowledge. Instructors can assume some level of knowledge of students entering the course.

“So that the instructor in the course for which it’s a prerequisite can assume certain knowledge of his or her students.” - I20

Not Reteaching Material. Some instructors specifically mentioned that the purpose of prerequisites is to allow them to cover the current course material without having to review previous material, although many instructors mentioned they do still review previous material at the beginning of the term.

“so that you don’t have to review material or teach material from the start of every course.” - I1

Implicit Requirements.

These were frequently listed when instructors were asked about their own courses.

Maturity. Instructors believe prerequisites allow them to assume some level of maturity when students enter a course and specifically skill maturity as it related to computing. Instructors specifically mention the term “maturity” in their descriptions as it related to mathematical maturity and ability to reason about larger problems.

“It’s not just learning some stuff, not just learning knowledge, but it’s really learning the knowledge, assimilating it, understanding a step in their maturing, all of those things make it so that they can be successful in these other courses.” - I19

Critical Thinking and Reasoning Skills. Instructors believe that prerequisites need to prepare students to think critically and reason about computer science related problems. Students should be able to apply their knowledge to program effectively and understand the trade-offs of what they are creating.

Soft Skills. Instructors mentioned students learn to communicate and work in groups in prerequisites.

Guidance through the Curriculum

Guiding Student Choice. Instructors said prerequisites give students the knowledge necessary to help them select courses when they are given multiple options, such as electives. The prerequisite courses also provide them with a breadth of knowledge to help determine what they are interested in pursuing as a career.

“To try and give them enough material and a broad enough scope so that they can better select their electives.” - I2

Administrative Guidance through the Curriculum. Instructors stated that prerequisites provide administrative guidance for the department, instructors and students.

“Prerequisites also help from an administrative perspective, on how the student can progress from one step to another, in a formal way. That gives guidelines, which anybody can follow.” - I8

Historical Requirements. One instructor mentioned historical requirements which is best described in the quote itself. *“I think what doesn’t always work well is that there are courses that were introduced a while ago and have migrated from their original intention or changed as technology progressed. The prerequisites that made sense when they were adopted, or it could be the course, that course is the same, but the prerequisite has changed a lot.” - I1*

7.2.2 RQ2: Is the Purpose being Fulfilled?

The quotes in this section are all responses to the question “Is this purpose being fulfilled?” a question as a follow-up to “In general, what do you think the purpose of prerequisite courses is?” Our high-level categories of description are “Yes,” “Sometimes,” and “No.” and are explained in order below. This follow-up question is from [60].

Yes, prerequisites are fulfilling their purpose.

This category reflects responses that prerequisite courses are fulfilling their purpose as much as possible. Responses categorized as yes started off as follows: “Probably, as well as it can be.”, “I think so”, and “Oh... Yeah. I tend to be an optimist about that kind of thing.” In

most cases where instructors responded saying yes, this did not mean they felt students entered courses with the required knowledge and skill, but rather that the problem was not caused by the prerequisite courses. *“Do students know every single thing they’re supposed to know from the prerequisite course? No, of course not. But do they know something? Yeah, definitely. Otherwise, I’d be starting over... every single time I taught any class.”* - I15

Interest/Motivation. Instructors believe students do not always have the necessary interest or motivation to learn the content.

“I feel yes, it has been fulfilled, but no matter what you do, there will be some students who had no interest in the prerequisite, so they just try to get it over with.” - I18

Asking for help. Students are unwilling to admit that they don’t understand and do not reach out for help.

“And there will be still some students who just don’t want to admit that they are not understanding the problem and they don’t want me to help them so I can recognize them, but no matter how hard I work to try to reach out to those students, if they don’t want me to help them, I can’t.” - I18

Students Lack Confidence in Themselves. Instructors believe that the students do not lack knowledge, but self-confidence.

“A lot of times, it’s just confidence, believing that they have the problem-solving skills or knowledge to break a problem down and think of it all more in a mathematical way.” - I19

Transfer Students. Transfer students came up as a population of students that is not prepared for courses¹.

Prerequisites are sometimes fulfilling their purpose.

The majority of instructors did not answer this question solely with a Yes/No answer and rather specified where they think issues may and may not lie. They started off their answers

¹Transfer students are students who start their undergraduate degree after completing a 2-year college program that fulfills General Education requirements and normally a few computing courses.

with wording such as “For the courses that I teach...”, “In large part yes...”, and “I would say it depends on the class and it also depends on who teaches it...” The same categories of description were mentioned as in the previous section, as well as additional categories shown below.

Variability of Course offerings. Due to variability across prerequisite course offerings, some instructors feel they need to repeat material and therefore end up covering less material in their own courses, furthering the issues through the prerequisite chains. A couple of possible reasons instructors described for this issue were that some instructors may do a bad job of teaching the material or simply did not teach all of the material.

“I would say it depends on the class and it also depends on who teach it, because I don’t think we have a very uniform requirement of what a class should cover.” - I20

Courses Outside of the Department. While transfer students struggling has already been mentioned, here the idea of Advanced Placement (AP) credit² was introduced, and the idea that allowing students to take equivalent courses outside of the department may allow students to progress through the computing curriculum without the necessary knowledge.

Design of the Course. Instructors mentioned the importance, and sometimes lack, of aligning courses in prerequisite chains.

Grade Inflation. Grade inflation allows students to progress in the curriculum without the required knowledge. This was described inside and outside of the context of the COVID-19 pandemic. The pandemic worsened this issue as students could take courses with a "pass/no-pass" grade that would normally require a letter grade.

“In general, I think most of the prerequisites should be covered if a student did well in the previous class. But because of grade inflation, I think there may be issues where a student was able to pass a certain class with a very low grade, but they got the credit, and they will struggle in the next class.” - I5

²AP credit is college credit that can be earned by students in high school who take AP courses and pass an exam on the course material.

Prerequisites are not fulfilling their purpose

The categories of description are most detailed for this section as the instructors who said prerequisite courses were not fulfilling their purpose tended to give longer explanations. These responses covered some previous categories, as well as introducing more nuanced issues.

Individual Student Abilities and Motivations. Instructors provide additional context for this category. One instructor specifies the impact of impostor phenomenon [56] and how when a student feels like an impostor “...the student will often build on a shoddy foundation, because they’re not willing to reveal their lack of understanding of a base idea that they should have learned maybe, potentially, months or years ago.” - I12

Retention of Knowledge Cannot be Assumed. Instructors believe retention of knowledge can no longer be assumed to the extent that used to be possible. One instructor suggests this is due to students being assessed too much so that the focus shifts from understanding to getting a grade. “And the other thing is that I feel that we focus more on getting the right answer rather than teaching them how to get to a point where they’re able to...expand on what they’re looking for.” - I21

Teaching to a Diverse Population. Instructors believe it is hard to teach material in a way that is salient enough for a more diverse population of students.

“Somehow the material may not be absorbed or learned in a sticky enough way, in a retained way. I think there’s many factors that go into that, having to do with the difficulty of teaching a diverse student population who come into the institution with a diverse set of backgrounds and preparations and then shepherding that diverse cohort through the curriculum with adequate success and progress through the degree program, right?” - I9

Pressure on Instructors. Instructors mentioned different pressures they have experienced. The pressures mentioned included the pressure to pass students and help them complete their degree, not having enough time to teach everything in a term, and social pressures to be well-liked and maintain good evaluations.

“Students simply have to move on, in other words. And I think instructors, for whatever reason, are sometimes motivated to help students move on.” - I9

7.3 Discussion

Section 7.2.1 enumerates several categories and subcategories of purposes for prerequisites, highlighting their richness and potential complexity as seen from the instructor perspective. This richness and complexity plays out further with regards to the fulfillment of those purposes. This section focuses on a comparison to Walker’s purposes of prerequisite courses. A more detailed discussion of the instructor perspective is in Chapter 8 under Subsection 8.3.3.

7.3.1 Comparison to Walker’s Purposes

Walker’s description of the most common purposes of a prerequisite course are provided in Section 1.4 [74]. We first note that filtering was not mentioned by the instructors we interviewed. Certainly, the concept of a “weeder course” is well known. However, being that current priorities in computing education are focused on recruitment and retention, instructors could hold the view that filtering is not a legitimate purpose. Interestingly, a few students (positively) mentioned filtering in the work by Sato et al., for example a prerequisite reducing competition for senior students to get in a subsequent course [60].

Requirement enforcement, as defined by Walker, was not mentioned by instructors. We take this to be a reflection of the instructor perspective: such a prerequisite would be in name only—a program requirement masquerading as a course prerequisite—a bureaucratic maneuver of sorts.

Walker did not list administrative guidance (Section 7.2.1), a major category for our instructors. This is a role prerequisites play in terms of helping students plan their path through a curriculum. Instructors describe it as allowing students to manage their time-to-degree and taking the right prerequisites for later desired electives (e.g., as they relate to their desired career path). Sato et al. shows that students also appreciate this role of prerequisites [60].

Finally, we note an important difference in the way prerequisite purposes are framed. Walker does not list instructor preparation as a purpose, but later mentions it as a consequence, saying “a prerequisite allows an instructor to assume student knowledge of specific content.” Instructor preparation is a major purpose brought up by instructors (Section 7.2.1). That is, from the instructor viewpoint, instructor preparation is a primary rather than derivative purpose.

7.3.2 Limitations and Threats to Validity

Our results are based on interviews with instructors from the University of California San Diego and California State University, Long Beach which may limit the generalizability of our findings. However, both of these programs follow the curriculum guidelines from the Joint Task Force on Computing Curricula Association which are often used in curricular design [25] and therefore the instructors’ perspectives are likely similar to those of instructors at similar institutions. We did not interview instructors from liberal arts or community colleges and therefore may not have encompassed all of their perspectives in our results.

While our instructors came from two institutions, due to the differences in sizes of the programs we did not have an even number of instructors from both institutions, with the majority, 15 instructors, coming from one of the two institutions. Additionally, one possible threat to validity is that it is possible that instructors forgot to mention a purpose of prerequisites or stopped after explaining several purposes in order to shorten the interaction. We believe that interviewing 21 instructors likely mitigated this issue.

7.4 Summary

Prerequisite courses are a prominent feature of CS curricula. Yet, little was known about what instructors’ views are on prerequisites. We interviewed 21 CS instructors at our two institutions who taught key courses in the typical CS curriculum. We then employed phenomenography on the resulting transcripts to uncover the variety of views instructors have on the purposes of prerequisites and whether those purposes are being fulfilled.

The instructors articulated a surprising range of purposes. Instructors also varied in their views on the fulfillment of those purposes. This exposed a range of views on who had the *responsibility* for fulfilling the purposes of prerequisites, that is, the degree to which the responsibility fell to instructors or students. Additionally, instructors noted that their efforts to fulfill the purposes of prerequisites were challenged by a wide range of pressures from students, administrators, and fellow instructors.

It is apparent that prerequisite courses, as far as instructors are concerned, have a complex and central role in their curricula. Moreover, some instructors view the fulfillment of prerequisite purposes as contested territory, variously occupied by instructors, students, and administrators. Future research should quantify the degree to which instructors hold these different views and administrators would be advised to consider how the results presented here might inform the design and administration of prerequisite courses, such as the use of mastery learning and achievement goal interventions.

7.5 Acknowledgements

This material is based upon work supported by the NSF GRFP under Grant No. DGE-1650112, NSF Award #2121592, a UCSD Sloan Scholar Fellowship, a UCSD STARS Fellowship, and a Gates Millennium Scholarship.

Chapter 7, in part, is a reprint of the material as it appears in the Proceedings of the 54th ACM Technical Symposium on Computer Science Education (SIGCSE 2023). Sophia Krause-Levy, Adrian Salguero, Rachel S. Lim, Hayden McTavish, Jelena Trajkovic, Leo Porter, and William G. Griswold. “Instructor Perspectives on Prerequisite Courses in Computing”. The dissertation author was the primary investigator and author of this paper.

Chapter 8

Discussion

This dissertation began with the following research questions:

- **Extent:** To what extent are students arriving in subsequent courses with low prerequisite proficiency?
- **Impacts:** How does prerequisite proficiency impact student success in subsequent courses, including measures of student learning in following courses? Moreover, does the impact of prerequisite proficiency differ for various groups of students?
- **Causes:** What are the causes of the problem (e.g., time between courses, misalignment of instructor expectations and student abilities, etc.)?

8.1 Extent

To what extent are students arriving in subsequent courses with low prerequisite proficiency?

8.1.1 Incoming Prerequisite Course Proficiency

We find that there is a misalignment between instructor's expectations of students' incoming prerequisite course proficiency and students' actual proficiency. Not all students' scored 100% on the pre-term tests, tests created with the expectation that all students would get all questions correct. We see a decline in pre-term test scores as we look at courses further into

the computing curriculum, with the distribution of scores getting wider. This suggests that the problem of low prerequisite course proficiency grows over time, aligning with our hypothesis that lower proficiency of course content in one class leads to a higher chance of struggling in later classes (see Figure 8.1).

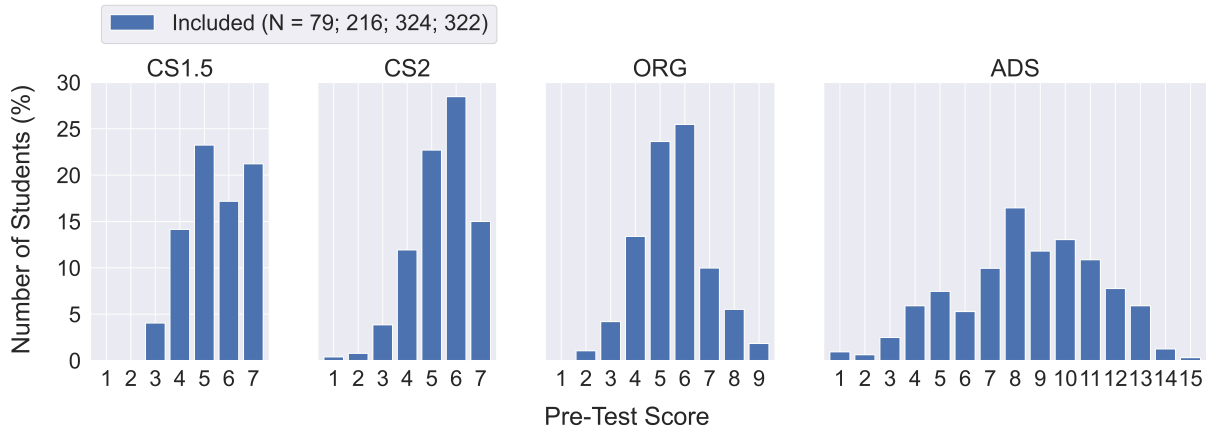


Figure 8.1. Distribution of scores on the pre-term test.

By looking at questions on the pre-term test that most students' got incorrect, we identified concepts students are struggling with in different stages of the curriculum. These include code tracing, and understanding memory management and references. This alone is not a problem, as these concepts are difficult to learn and take time to learn. Unfortunately, instructors greatly overestimated students' abilities on these concepts. This shows instructors do not know students' level of knowledge when entering the course. While in some cases students may be able to get up to speed quickly, in others, students may not have the time to brush up on their knowledge while also trying to succeed in the courses they are enrolled in. Work by Salguero et al. found that students with worse outcomes in courses report higher levels of personal obligations (other course requirements, illness, family obligations, work obligations and social/personal life issues) [59]. This means students who have other obligations are at a disadvantage if they are expected to catch up on prerequisite course content while making progress in their current courses.

Pre-term tests can help instructors gain insight into what their students are struggling with and it can allow them to know what topics to refresh or spend more time on at the beginning

of a course. By treating students as if they are all the same in their prerequisite knowledge we are likely leaving students behind.

8.2 Impact

How does prerequisite proficiency impact student success in subsequent courses, including measures of student learning in following courses? Moreover, does the impact of prerequisite proficiency differ for various groups of students?

8.2.1 Course Outcomes

Our results from Chapter 4 and 5 showed that there is a strong correlation between students' pre-term test score and their outcomes in the course. We also see a strong relationship between their prerequisite grades (prerequisite GPA) and their outcomes. This suggests that students' incoming prerequisite proficiency and their prerequisite course grades predict their outcomes in later courses.

8.2.2 Underrepresented Groups

We see a trend in inequitable outcomes for women, BLNPI, first-generation and transfer students. It appears that prerequisite course proficiency may be playing a role in the outcome disparities for these students. Work by Alvarado et al. found statistically significant differences in course grades throughout students' undergraduate computing careers based on whether students had computing experiences prior to entering university [4]. Research also found that BLNPI students, women and students from lower economic status often do not have access to the prior programming experiences [34, 35, 75]. This suggests that the gap we see for these students in the prerequisite course grades and pre-term test may be due to lacking prior experiences before entering university.

Additionally, previous research has identified that these students may have higher levels of personal obligations outside of the course, in-class confusion, lack of sense of belonging

and lack of confidence [59]. This may lead to these students having less time outside of class to spend on topics they are struggling with that instructors expect them to know, while their counterparts have time to catch up.

Women. Women have inequitable outcomes on the pre-term test for CS1.5 and ORG, though only minimal differences in their prerequisite grades. In terms of outcomes women have lower average final exam scores for CS2 and ORG, and lower average course grades for ORG. We see no differences in outcomes for ADS. We caution that while we don't see differences in outcomes in ADS, prerequisite course proficiency may still be an issue for women, as it's possible that the women who struggled have already self-selected out of computing before they reach the ADS course.

First-Generation Students. For first-generation students we see differences across all outcome measures in CS1.5, CS2 and ORG with the exception of the CS1.5 pre-term test. We see differences in the immediate predecessor prerequisites to each of the courses with the exception of ORG as a prerequisite for ADS. In terms of course grade, in CS1.5 first-generation students, on average, receive a course grade that is two letter grades lower than non-first-generation students. In CS2 the difference is less than a letter grade and in ORG it is one-third of a letter grade. This may mean that the gap between these populations is closing or it may be that first-generation students who are struggling are self-selecting out of computing.

BLNPI students. BLNPI students have inequitable outcomes across all measures, though not all are statistically significant. We see that there is a gap between BLNPI and Non-BLNPI students' grades in all of their prerequisite courses leading into each course (CS1.5, CS2, ORG, ADS), as well as in their pre-term test scores. In terms of course outcomes, there is a trend of BLNPI students receiving scores that are on average one letter grade lower than their majority counterparts.

Transfer students. For transfer students, we see the most sizable differences in outcomes in the ADS course, with small differences in CS1.5, CS2 and ORG. The ADS course is the first course that all transfer students must take, as it is not possible to take an alternative at or outside

of our institution. We believe this may, in part, account for the increase in lower outcomes for transfer students in ADS. In future work we want to look at courses later in the curriculum to see if the gap continues to grow as transfer students continue on to upper-division courses and rerun our study in the ADS course.

Intersectionality.

More work needs to be done to understand the possible effects of intersectionality. It is unlikely our results are strongly influenced by cross-membership in most of these groups, as each is represented in the other in small percentages. The only two cases where the cross-membership is 50% or more is in CS1.5 and ORG by BLNPI status and first-generation status. We reanalyzed our data on outcome measures looking at students who are part of both groups (intersectional: first-generation and BLNPI) and those who are only a part of one group (non-intersectional: first-generation or BLNPI). We found a statistically significant difference in course grades for CS2 and ORG, with the intersectional group having lower course grades than the non-intersectional group. We reanalyzed the outcome measures for students from the non-intersectional group and their majority counterparts, finding that the inequality in course outcomes still exists for these students (students who identify as BLNPI or first-generation but not both).

There could also be intersectionality effects for individual students, even with groups that did not show lower prerequisite course proficiency or course outcomes. The small sample sizes of the populations we examined in this dissertation necessitated leaving the majority of this analysis for future work. Institutions need to find ways to support these students. Our findings necessitate more studies focused on better understanding the role prerequisite preparation may have on the outcomes of traditionally underrepresented groups in computing.

8.3 Causes

What are the causes of the problem (e.g., time between courses, misalignment of instructor expectations and student abilities, etc.)?

8.3.1 Alternative Prerequisites Inside and Outside of Our Institution

We did not see any evidence for taking alternative prerequisites inside or outside of our institution affecting students' prerequisite knowledge and subsequent course grades. For CS1.5, CS2 and ORG we looked at students who did or did not take their immediate predecessor prerequisite at our institution. For ADS we only looked at students who took their prerequisites at our institution, differentiating by whether or not they took their prerequisites in the computer science department. In both of these analyses we did not see anything to suggest that students' who took their prerequisites at another institution or department had worse outcomes.

For ADS we were unable to include students that took their prerequisites at other universities when doing our analysis. While we did not see a negative effect for students taking alternative prerequisites at our university, it may be that there is a negative effect for students that took their prerequisites at other universities, for example because the courses may not cover the exact same material. In the future we would like to include these students and see if our hypothesis is still invalidated.

Excluding these students may have an impact on our results for prerequisite learning fading with time, as we don't have results for these students. Students who took their prerequisites at another university may have a large gap in time between taking the prerequisite and follow-on course.

8.3.2 Identifying Other Factors

We found that "other factors" are positively correlated with course grades but we were unable to identify what those "other factors" are. Future work should aim to identify these factors and see how much they contribute to student success. Figure 8.2 shows a simplified version of the grading breakdowns and highlights the fact that a large portion of each course grade (50% or more) awards points based on effort or for work performed outside of class, allowing students' access to resources outside of the control of instructors.

This suggests something needs to be changed in how we run our courses and how we grade our students. We need to provide students with a more accurate measure of their understanding of course material. Merely passing a course is not enough to determine course proficiency and, as we see in the course grades, even passing the course with a high mark is not always representative of having proficiency in the course material (see Figure 8.3).

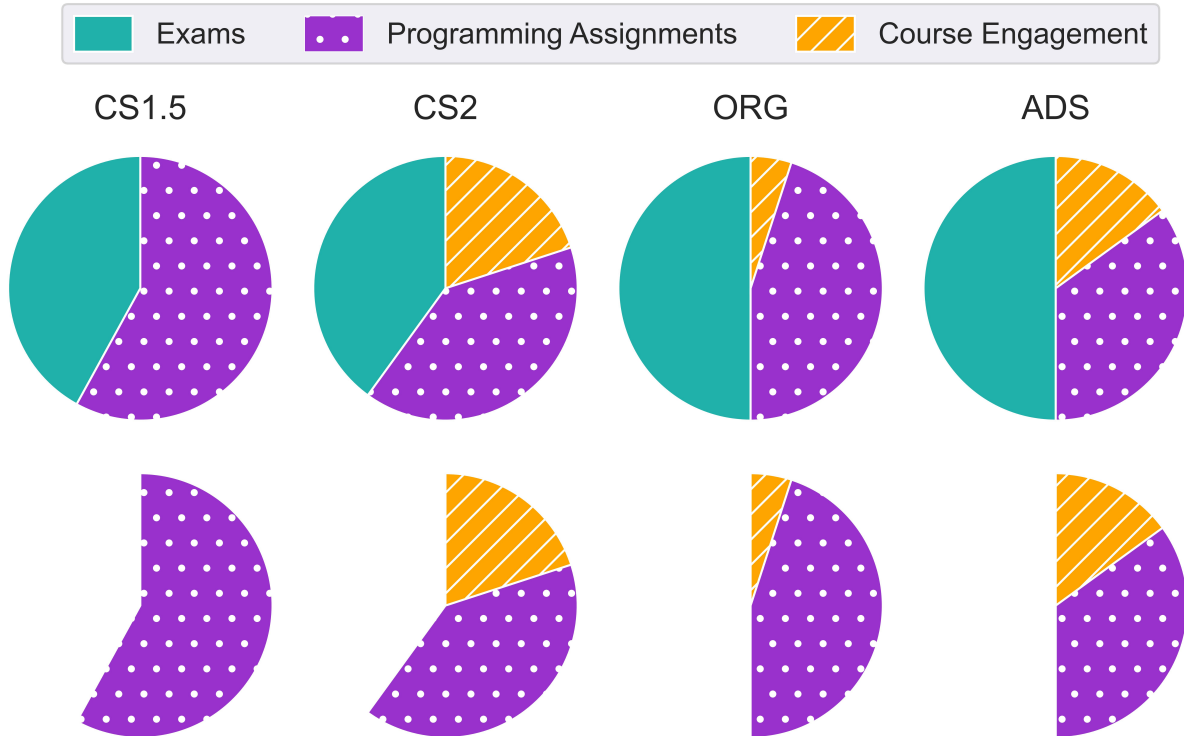


Figure 8.2. The grading breakdown for CS1.5, CS2, ORG and ADS courses. Note that 50% or more of each course allows for collaboration or awards effort.

8.3.3 The Instructor Perspective

Through the instructor interviews in Chapter 7 we were able to gain insight into additional possible causes of the problem of low prerequisite course proficiency and the number of different people and groups who are responsible for making sure we can support our students.

Emergent Concept: Responsibility

Responsibility is a concept that emerged through further analysis of the interviews. Instructors view prerequisite courses' fulfillment of their purposes as a network of responsibilities among instructors, students, the department and the institution. Here we describe the list of responsibilities.

At a high level, instructors talked about administrative guidelines (Section 7.2.1) that are the responsibility of the institution and department. Calibrating course offerings and alignment across prerequisite chains is the responsibility of the instructor, as well as the department administration (Section 7.2.2). The responsibility of student preparation lies with the students themselves to put in effort to learn the material, but also with the department and instructors to stop students from entering a course if the students do not meet the requirements (e.g., transfer or AP credit, and petitions; Section 7.2.1). During a course, instructors are responsible for teaching the required content in a way that is salient (Section 7.2.2). They must also fail students who have not learned the material, which also includes identifying any cheating and enforcing consequences, an additional point brought up by instructors (Section 7.2.1).

A responsibility that has become more prominent in the past few years due to the COVID-19 pandemic is adjusting courses based on world events and individual student needs (Section 7.2.2). The institution and department play a role in this responsibility, but it ultimately falls on instructors. Instructors mentioned that adjusting to world events sometimes caused conflicts with their responsibility to ensure students progressed through the program with ensuring student learning outcomes.

Many Conflicting Pressures on Instructors

The discussion of responsibility highlights the pressures on instructors regarding prerequisite courses, many of which are conflicting. Instructors described the pressures they face from students, the department, and the institution, such as not to adversely impact students' time-to-degree and to be well-liked by students (Section 7.2.2). This pressure has been amplified

by the COVID-19 pandemic.

Given the limited resources available, instructors must make tough choices. One such resource raised by instructors is time: time available in a given day, week, or term. Instructors need to balance time spent on teaching with other responsibilities such as identifying and reporting academic integrity violations.

As seen in the interviews, instructors vary in how they cope with conflicting pressures. One such example is the pressure of trying to accommodate students who do not have the required incoming prerequisite knowledge and the pressure to cover all required course content so students are prepared for subsequent courses. For some instructors this meant cutting back on new content to ensure students have a solid foundation of previous content, while others focus on covering their course content—at least one instructor mentioned they do not review any required prerequisite content. Cutting back on new content will help students build a more solid foundation of previous content but it may cause issues further down the line.

8.4 A Need to Focus on (the Benefits of) Learning

Many instructors shared their belief that students are entering their courses without the required knowledge and expected skills. As discussed in Section 7.2.2, regardless of whether instructors say that the purposes of prerequisites are being fulfilled, they cited several factors over which instructors (we believe) have some control, albeit while experiencing pressure from all sides. These include: lack of student interest (some students “just try to get it over with” (Section 7.2.2)), grade inflation (Section 7.2.2), and a focus on grades over learning (Section 7.2.2).¹

It is important that instructors align student learning outcomes with their actual knowledge at the end of the course. There are many promising techniques that may help with this problem, such as active learning for frequent formative assessment [51], standardized exams to provide

¹Education and psychology literature provides ample evidence that having the wrong achievement goals reduces learning. A starting point is [82].

comparisons with other populations in similar courses [52], and equity grading [16] to ensure grades are based on individual performance. As we already use active learning in many of the courses at our institutions, we suspect that active learning alone is not a solution to this problem.

Another possible solution is to embrace mastery learning, as it focuses entirely on student learning outcomes. Mastery learning is the idea of organizing learning goals into well-defined units of concepts such that students must master a unit before moving on to the next one [7, 38]. The mastery learning perspective claims that the conveyor-belt approach of passing students without mastery of the current course material is a major flaw in our current educational systems. Mastery learning cannot be fully implemented in universities for many reasons including increasing students' time to degree, but concepts from mastery learning can be applied to how we structure our courses. We can test our students on concepts more regularly so they are able to gauge their understanding, and if a student does not perform well, we can give them resources to study and allow them to be retested until they understand the concepts. Some researchers in CS education have already begun to look into this, as seen in the literature review conducted by Garner et al. [18] and more recently by Ott et al. [45]. Lack of past prerequisite course proficiency is statistically significantly correlated with lack of future course proficiency, and this issue falls disproportionately on students from underrepresented groups in computing. Not requiring a higher level of proficiency in course material before allowing students to move on to the next course is doing a disservice to the students in populations that are disproportionately affected.

Although mastery learning puts a strong focus on learning, it does not, *per se*, provide internal motivation for the student to *want* to learn. It would need to be complemented by achievement goal interventions such as messaging from the instructor on the importance of the material for the next course or their careers [14]. As cited by our instructors, the issue of students not coming forward for help also needs to be addressed.

8.5 Ideas for Instructors

In addition to aligning student learning outcomes with their actual knowledge at the end of the course, we believe there are other ways instructors can address students' low incoming prerequisite proficiency.

- **Using Pre-Term Tests.** We encourage instructors to create pre-term tests that represent concepts they expect students to know coming into their course. The pre-term test should be given in the first week of the course so that instructors can get a sense of students' incoming prerequisite course proficiency. We recommend instructors estimate how well they think students will do on each question so that instructors can easily compare their expectations to students' actual abilities. Giving students some form of participation credit may help students take the test more seriously, though mastery learning recommends against giving credit based correctness. The purpose of administering the pre-term test is to allow instructors to gain insight into the concepts students are struggling with. Understanding which concepts students are struggling with will allow instructors to plan their lessons in a way that refreshes students' knowledge of these concepts. Instructors may also decide to provide additional review sessions, resources on the concepts, or a homework 0 at the beginning of the course focused on the topics students are still struggling with.
- **Understanding Low Prerequisite Course Proficiency.** Instructors should be aware that students come in with different levels of background knowledge and be prepared to help these students. Instructors can gain a better understanding of the mismatch between their expectations and students' actual abilities through the use of pre-term tests.
- **Telling Students which Concepts They are Struggling With.** Students can pass a course with a wide range of knowledge. It may be beneficial for students to know what topics they are still struggling with at the end of a course. Instructors could share this information through something such as labeling final exam questions based on concepts and by telling

students they should review the concepts related to questions they got wrong on the test, pointing them to resources. Another option for while the course is running, is to provide just-in-time prerequisite review. Zhang and Allin recently presented the idea of having students complete 2-4 multiple choice questions reviewing prerequisite concepts before each lecture in a machine learning course [80]. A 3-10 minute video was provided on each questions' concept so that students could review content on any question they were unable to complete on their own. Anecdotally, students appreciated this format.

- **Working with Colleagues.** Instructors should collaborate with the instructors who teach the prerequisites of their course and those who teach courses for which their course is a prerequisite. This will allow instructors to get a better sense of each others' expectations and help alignment across courses. It may also be beneficial to let each other know if a specific cohort of students seems to be struggling more with a concept so that the instructor of the subsequent course can prepare.
- **Talking about Responsibility and Pressures.** Instructors should talk to each other and their institutions about the conflicting pressures that they face. While it may be difficult to make changes, the first step is awareness of these issues.

8.6 Acknowledgements

Chapter 8, in part, is a reprint of material as they appear in three publications: 1) Proceedings of the 51st ACM Technical Symposium on Computer Science Education (SIGCSE 2020). Sophia Krause-Levy, Sander Valstar, Leo Porter, and William G. Griswold. "Exploring the Link Between Prerequisites and Performance in Advanced Data Structures", 2) Proceedings of the 53rd ACM Technical Symposium on Computer Science Education (SIGCSE 2022). Sophia Krause-Levy, Sander Valstar, Leo Porter, and William G. Griswold. "A Demographic Analysis on Prerequisite Preparation in an Advanced Data Structures Course", 3) Proceedings of the 54th ACM Technical Symposium on Computer Science Education (SIGCSE 2023). Sophia

Krause-Levy, Adrian Salguero, Rachel S. Lim, Hayden McTavish, Jelena Trajkovic, Leo Porter, and William G. Griswold. “Instructor Perspectives on Prerequisite Courses in Computing”. The dissertation author was the primary investigator and author of this paper.

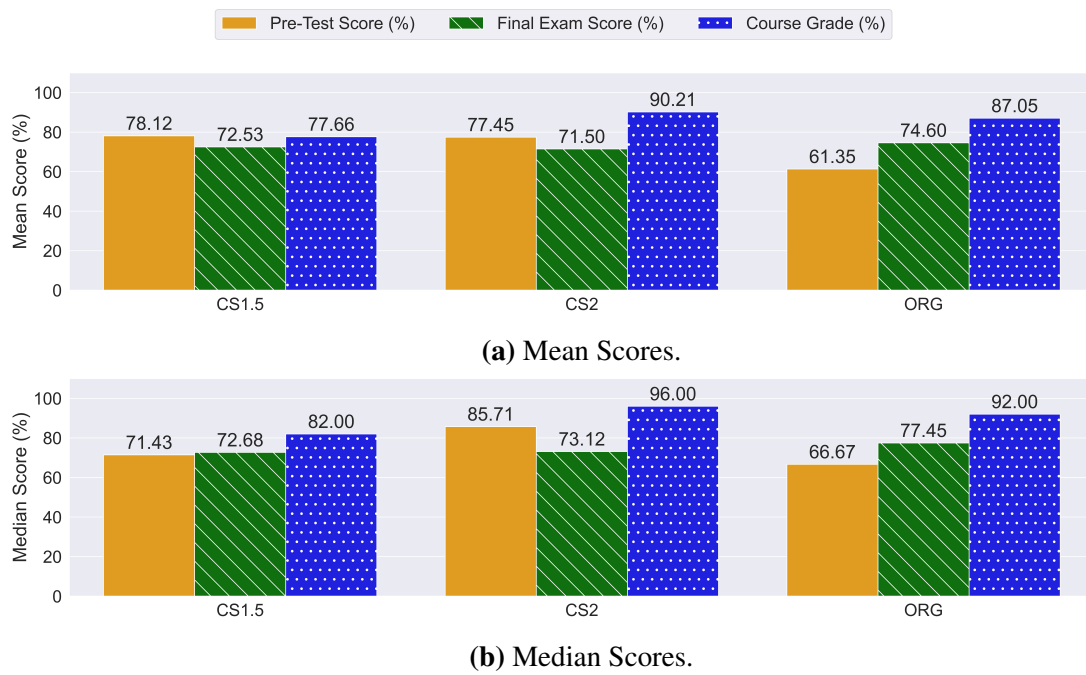


Figure 8.3. Mean and median scores on the pre-term test, final exam and course grade.

Chapter 9

Conclusion

There are high rates of failure and low rates of retention in computer science [50, 76]. Prior programming experience to entering university has been identified as a factor in student success in computing [4, 23, 75, 78, 82]. This is especially important as students from groups typically underrepresented in computing have less access to these experiences [34, 35, 75], are retained at a lower rate [83] and often have inequitable outcomes during their undergraduate computing careers [4, 58, 59].

Given the hypothesis of “Learning Edge Momentum (LEM)”, a hypothesis for why students tend to either succeed or struggle in introductory computer science courses, it is not surprising that students with prior programming experience are set up for success in introductory programming courses [55]. Prerequisites play a central role in the undergraduate computing curriculum, where there are long prerequisite chains with concepts building from one course to the next. Given the hypothesis of LEM, we hypothesized that lower proficiency in course content in one class leads to a higher chance of struggling in later classes and that there is an inequitable impact on students from underrepresented groups. The results of this dissertation suggest that our hypothesis is correct.

Throughout this dissertation we focused on answering questions regarding the extent, impacts and causes of low prerequisite course proficiency. More specifically we sought to gain answers to the following questions: **(1) extent:** To what extent are students arriving in

subsequent courses with low prerequisite proficiency? **(2) impacts:** How does prerequisite proficiency impact student success in subsequent courses, including measures of student learning in following courses? Moreover, does the impact of prerequisite proficiency differ for various groups of students? **(3) causes:** What are the causes of the problem (e.g., time between courses, misalignment of instructor expectations and student abilities, etc.)?

In Chapter 4, we discussed possible causes of low prerequisite course proficiency in the ADS course. We find no evidence for the passage of time or taking alternative prerequisites diminishing student success. We do find a strong relationship between prerequisite course grades and student success in the ADS course ($r=0.66$ for the final exam, $r=0.53$ for the ADS course grade). Interestingly we also find a strong relationship between students' non-prerequisite GPA and their success in ADS ($r=0.55$ for the final exam, $r=0.51$ for the ADS course grade), suggesting that other factors are also at play in student success.

In Chapter 5, we studied possible disparities in outcomes by demographic. We find that BLNPI and transfer students have lower scores on the pre-term test and inequitable outcomes in their prerequisite course grades. We see a strong correlation between pre-term test and student outcomes suggesting that students' incoming prerequisite proficiency may be impacting their outcomes.

In Chapter 6, we replicated the work of Valstar et al. and the studies in Chapter 4 and 5 across three lower-division courses in our curriculum. We find that students are not coming in with the prerequisite proficiency instructors expect. The distribution of scores on the pre-term test widens as we get farther into the computing curriculum. The good news from this work is that using the groupings of low, medium and high defined by Valstar et al., we find that in CS1.5 and CS2 few students are in the low-proficiency category (5% and 6%), with the majority of the students in the high-performing category (77% and 80%). In ORG we see a decline in incoming prerequisite proficiency with 22% of students in the low proficiency category and only 20% of students in the high-performing category. In ADS with 29% of students falling into the low category and 27% falling into the high category. In courses farther into the curriculum there is a

wider distribution in incoming proficiency. Unfortunately, given that the pre-term test consists of questions we expect all students to get correct, these categories are not representative of our expectations. The high-performing category is defined as students who received a score of > 70%, whereas the pre-term tests were created with the expectation that all students should get the questions correct. The questions are not meant to be difficult but rather representative of what students should know entering the course.

In terms of impact, we see a strong relationship between the pre-term test and student outcomes in the courses (for the final exam $r=0.35, 0.42, 0.23$ for CS1.5, CS2 and ORG). This suggests that students' incoming prerequisite course proficiency may have a large impact on their outcomes in the course. We continue to see a trend of inequitable outcomes for women, first-generation, BLNPI and transfer students. With a trend in these students having outcomes lower by one-third to one letter grade.

In terms of causes, our results confirm the results seen in Chapter 4. We don't see anything to suggest that time between prerequisites or taking alternative prerequisites impacts student outcomes.

In Chapter 7, we identified new possible causes of low prerequisite proficiency through a phenomenographic analysis of 21 interviews conducted with instructors. Instructors cited various reasons for why prerequisites may not be fulfilling their purpose, including the number of conflicting pressures instructors face with respect to ensuring students learn material expected for the next course (e.g., trying not to adversely impact students' time-to-degree, being well-liked by students, and balancing the time spent on teaching with other responsibilities).

In summary, we find that low incoming prerequisite course proficiency extends across the lower division curriculum, with the gap in knowledge increasing farther into the curriculum. Low incoming prerequisite course proficiency is correlated with student outcomes and has a larger impact on students from groups typically underrepresented in computing. In terms of causes, we believe a part of the reason students are able to continue on to subsequent courses without the expected level of knowledge is due to other factors beyond subject matter knowledge

playing a large role in students' course grades.

In the future we plan to continue this line of work by expanding it into upper-division courses at the University of California San Diego and at other institutions.

9.1 Acknowledgements

Chapter 9, in part, is a reprint of material as they appear in three publications: 1) Proceedings of the 51st ACM Technical Symposium on Computer Science Education (SIGCSE 2020). Sophia Krause-Levy, Sander Valstar, Leo Porter, and William G. Griswold. “Exploring the Link Between Prerequisites and Performance in Advanced Data Structures”, 2) Proceedings of the 53rd ACM Technical Symposium on Computer Science Education (SIGCSE 2022). Sophia Krause-Levy, Sander Valstar, Leo Porter, and William G. Griswold. “A Demographic Analysis on Prerequisite Preparation in an Advanced Data Structures Course”, 3) Proceedings of the 54th ACM Technical Symposium on Computer Science Education (SIGCSE 2023). Sophia Krause-Levy, Adrian Salguero, Rachel S. Lim, Hayden McTavish, Jelena Trajkovic, Leo Porter, and William G. Griswold. “Instructor Perspectives on Prerequisite Courses in Computing”. The dissertation author was the primary investigator and author of this paper.

Appendix A

Preterm Survey Questions

A.1 ADS Pre-term Test Questions

Below are the 15 questions given on the pre-term test. The sources of the 15 questions are as follows:

Question 1: Based on the work of Sheard et al. [63].

Questions 2-5 & 8: From the work of Danielsiek et al. [13].

Question 9: From the work of Götschi et al. [19].

Questions 10-12: From ORG taught by Joe Gibbs Politz in 2017 [48].

Question 15: From the work of Hamouda et al. [22].

All Other Questions: Custom made for the test by instructors.

Table A.1. ADS Pre-Term Test Questions

#	Question
---	----------

Table A.1, Continued

#	Question
1	Given the Java code below, what is the output? [63] <pre data-bbox="345 451 1339 1123"> public void swap(int a, int b) { int temp = a; a = b; b = temp; } public static void main(String[] args) { int x = 10; int y = 20; swap(x, y); System.out.println(x + ", " + y); } </pre> Select one (fill in the bubble below): ☐ 10, 20 ☐ 20, 10 ☐ 10, 10 ☐ 20, 20

Table A.1, Continued

#	Question
2	a) b) c) d) Figure B.1. Figures for questions 2 to 5. Is the tree in figure (a) a valid min-heap, or binary search tree? (see Figure A.1) [13] Select one (fill in the bubble below): ☐ Valid min-heap ☐ Valid binary search tree ☐ Both a valid min-heap and a valid binary search tree ☐ Neither a valid min-heap nor a valid binary search tree

Table A.1, Continued

#	Question
3	Is the tree in figure (b) a valid min-heap, or binary search tree?(see Figure A.1) [13] Select one (fill in the bubble below): ☐ Valid min-heap ☐ Valid binary search tree ☐ Both a valid min-heap and a valid binary search tree ☐ Neither a valid min-heap nor a valid binary search tree
4	Is the tree in figure (b) a valid min-heap, or binary search tree?(see Figure A.1) [13] Select one (fill in the bubble below): ☐ Valid min-heap ☐ Valid binary search tree ☐ Both a valid min-heap and a valid binary search tree ☐ Neither a valid min-heap nor a valid binary search tree
5	Is the tree in figure (b) a valid min-heap, or binary search tree?(see Figure A.1) [13] Select one (fill in the bubble below): ☐ Valid min-heap ☐ Valid binary search tree ☐ Both a valid min-heap and a valid binary search tree ☐ Neither a valid min-heap nor a valid binary search tree

Table A.1, Continued

#	Question
6	Suppose you are trying to decide which data structure to use for a List implementation. Your code frequently adds elements to either the beginning or end of the list, but only removes elements from the end of the list. Which data structure List implementation would offer you the best performance? Select one (fill in the bubble below): ☐ A Singly-Linked List with a pointer to the first node ☐ A Doubly-Linked List with a pointer to the first and last node ☐ An ArrayList ☐ Both A and B would offer similarly “best” performance ☐ Both A and C would offer similarly “best” performance ☐ Both B and C would offer similarly “best” performance

Table A.1, Continued

#	Question
7	Suppose you are given the code below: <pre data-bbox="345 510 1339 945"> for (int i = 0; i < n; i++) \{ for (int j = i; j < n / 8; j++) { for (int k = 0; k < 10000; k++) { // perform some constant time task } } } </pre> What is the tightest upper-bound for the time complexity of the code above (in Big-O notation)? Select one (fill in the bubble below): ☐ $O(1)$ ☐ $O(n)$ ☐ $O(n \log n)$ ☐ $O(n^2)$ ☐ $O(n^2 * \log n)$ ☐ $O(n^3)$

Table A.1, Continued

#	Question
8	Input: A natural number $n \in \mathbb{N}$ Output: $n \cdot (n + 1)/2$ <pre> 1 : $sum \leftarrow 0$ 2: for $i \leftarrow 1$ to n do 3. $sum \leftarrow sum + i$ 4: return sum </pre> 1. Let $1 \leq j \leq i$ be the number of executed iterations of the for-loop. The invariant $A(j)$ states that $sum = j \cdot (j + 1)/2$ after the j-th iteration. 2. Let $1 \leq j \leq n$ be the number of executed iterations of the for-loop. The invariant $A(j)$ states that $sum = j \cdot (j + 1)/2$ after the j-th iteration. 3. Let $i \leq j \leq n$ be the number of executed iterations of the for-loop. The invariant $A(j)$ states that $sum = j \cdot (j + 1)/2$ after the j-th iteration. 4. Let $0 \leq j \leq n$ be the number of executed iterations of the for-loop. The invariant $A(j)$ states that $sum = j \cdot (j + 1)/2$ after the j-th iteration. Which of the invariants on the right are correct for the code above? [13] Select ALL that are correct (fill in the bubble(s) below): ☐ 1 ☐ 2 ☐ 3 ☐ 4

Table A.1, Continued

#	Question
9	<pre> public int funct(int n) { if (n == 1) { return 1; } else { return 2 * funct(n / 2) + 1; } } </pre> What is returned if you call funct passing it the argument 8 as in funct(8)? [19] Select one (fill in the bubble below): ☐ 1 ☐ 3 ☐ 7 ☐ 9 ☐ 10 ☐ 11 ☐ 14 ☐ 15 ☐ 19 ☐ 22 ☐ 31 ☐ 255

Table A.1, Continued

#	Question
	Given this C code below: <pre> int length = 5; int arr[5] = {20, 30, 40, 50, 60}; int i; for (i = 0; i < 5; i++) { arr[i] = arr[i] + 10; } </pre> Below is pseudoassembly which corresponds to the C-code above. R is for Register. R1 contains the base address of array "arr". R2 $\leftarrow$ R1 # Register 2 is given the value in Register 1 R3 $\leftarrow$ R1 + 20 # Register 3 is given the value in Register 1 + 20 10 loop: R4 $\leftarrow$ MEM[R2] # R4 is given the value in memory at address R2 R4 $\leftarrow$ R4 + 10 MEM[R2] $\leftarrow$ R4 # MISSING INSTRUCTION BNE R2, R3, loop # If R2 $\neq$ R3, goto loop, # otherwise advance to the next instruction Fill in the # <i>MISSING INSTRUCTION</i> [48] Select one (fill in the bubble below): ☐ R2 $\leftarrow$ R1+1 ☐ R2 $\leftarrow$ R1+4 ☐ R2 $\leftarrow$ R2+1 ☐ R2 $\leftarrow$ R2+4 ☐ None of the above.

Table A.1, Continued

#	Question
11	Imagine we have a 32-bit value with unknown bits, and would like to know if the 4 least-significant bits are 1001. Which of the following operations would tell us? [48] Select one (fill in the bubble below): ☐ Perform a bitwise AND with 0x0000000F and check if the resulting value is equal to 0x00000009 ☐ Perform a bitwise OR with 0xFFFFFFF6 and check if the resulting value is equal to 0x00000009 ☐ Perform a bitwise AND with 0x00000009 and check if the resulting value is equal to 0x00000009 ☐ Perform a bitwise XOR with 0x00000009 and check if the resulting value is equal to 0x00000009 ☐ Perform a bitwise OR with 0x00000009 and check if the resulting value is equal to 0x00000009 ☐ None of the above

Table A.1, Continued

#	Question																
12	Address Value 0x10000 0x65646362 0x10004 0x68006766 0x10008 0x6B006A69 Assume a little endian layout for byte addressing, so address 0x10005 contains 0x67, for example. What string would print if we ran: <code>printf("%s", x);</code> where <code>x</code> holds the address 0x10008? Hint, 0x61 in hex translates to the ascii character 'a'.) [48] Select one (fill in the bubble below): <table> <tr> <td>☐ k0ji</td><td>☐ bcde0</td></tr> <tr> <td>☐ kji</td><td>☐ lki</td></tr> <tr> <td>☐ ij0k</td><td>☐ 10ki</td></tr> <tr> <td>☐ ik0l</td><td>☐ ij</td></tr> <tr> <td>☐ ji</td><td>☐ ik</td></tr> <tr> <td>☐ kl</td><td>☐ lk</td></tr> <tr> <td>☐ k</td><td>☐ l</td></tr> <tr> <td>☐ i</td><td>☐ None of the above</td></tr> </table>	☐ k0ji	☐ bcde0	☐ kji	☐ lki	☐ ij0k	☐ 10ki	☐ ik0l	☐ ij	☐ ji	☐ ik	☐ kl	☐ lk	☐ k	☐ l	☐ i	☐ None of the above
☐ k0ji	☐ bcde0																
☐ kji	☐ lki																
☐ ij0k	☐ 10ki																
☐ ik0l	☐ ij																
☐ ji	☐ ik																
☐ kl	☐ lk																
☐ k	☐ l																
☐ i	☐ None of the above																

Table A.1, Continued

#	Question
13	Suppose you have the following code: <pre data-bbox="391 478 656 569">int x = 5; int * ptr = &x;</pre> Which of the following would correctly change x to be the value 10 (select all that apply)? Select ALL that apply (fill in the bubble(s) below): <input data-bbox="383 869 412 907" type="radio"/> ptr = 10; <input data-bbox="383 928 412 966" type="radio"/> *ptr = 10; <input data-bbox="383 987 412 1024" type="radio"/> &ptr = 10; <input data-bbox="383 1045 412 1083" type="radio"/> *x = 10; <input data-bbox="383 1104 412 1142" type="radio"/> &x = 10;

Table A.1, Continued

#	Question
14	Given a bag with 3 red and 2 blue marbles. One trial means picking 1 marble from the bag. Marbles are not placed back after a trial. If we do 2 trials, what is the probability that we drew at least 1 red marble from the bag. Select one (fill in the bubble below): ☐ 0.9 ☐ 0.75 ☐ 0.84 ☐ 0.95 ☐ 0.6 ☐ 0.5

Table A.1, Continued

#	Question
15	Given the following code: <pre data-bbox="344 451 1339 1003"> int function(int y) { if (y == 1) { return 5; } else { function(y - 1); y = y + 1; return 83; } } </pre> What will be returned when <code>function(2)</code> is executed? Select a number, or select “infinite recursion” if you think that this call will lead to infinite recursion. [22] Select one (fill in the bubble below): ☐ 5 ☐ 83 ☐ 6 ☐ 583 ☐ Infinite recursion ☐ Other

A.2 CS1.5 Pre-Term Test Questions

Below are the 7 questions given on the pre-term test. The sources of the 7 questions are as follows:

Question 1: Similar to the work of Sheard et al. [63].

Questions 2-3: From the work of Simon et al. [64].

Question 5: Similar to the work of Caceffo et al. [9].

All Other Questions: Custom made for the test by or with instructors. See Section 6.1.3 for more details on how the questions were created).

Table A.2. CS1.5 Pre-Term Test Questions

#	Question
1	What is the value of b after the following code is run? <pre> a = 3 b = a a = 7 </pre> <u>Answers</u> Fill in the blank: 3 (86.1%) → CORRECT 7 (12.7%) The value of b after the code is run is _____ a (1.3%)

Table A.2, Continued

#	Question
2	Consider the following code: <pre> girls = 0 boys = 0 children = 0 children = girls + boys girls = 15 boys = 12 </pre> What are the values of girls, boys, and children after the following code has been executed? Select ONE: ☐ A. <pre> girls = 0 boys = 0 children = 0 </pre> (6.3%) ☐ B. <pre> girls = 0 boys = 0 children = 27 </pre> (1.3%) ☐ C. <pre> girls = 15 boys = 12 children = 0 </pre> (88.6%) → CORRECT ☐ D. <pre> girls = 15 boys = 12 children = 27 </pre> (3.8%)

Table A.2, Continued

#	Question
3	Consider the following code: <pre data-bbox="285 451 1279 1003"> x = 1 y = 2 z = 3 if (x < y): if(y > 4): z = 5 else: z = 6 </pre> What is the value of the variable z after the code above is run? Select ONE: ☐ A. 1 ☐ B. 3 (3.8%) ☐ C. 5 ☐ D. 6 (96.2%) → CORRECT

Table A.2, Continued

#	Question
4	Consider the following code: <pre data-bbox="285 451 1279 705"> x = 3 while (x < 5): print(x) x = x + 1 </pre> What is the output of this code? Select ONE: ☐ A. 3 (5.1%) ☐ B. 3 4 (87.3%) → CORRECT ☐ C. 3 4 5 (6.3%) ☐ D. 1 2 3 4 5 (1.3%)

Table A.2, Continued

#	Question
5	Given a function <code>multiplyBy3</code> that multiplies its input by 3 and returns the result, what is the output of the following code? <pre data-bbox="329 541 649 804">a = 5 a = multiplyBy3(a) b = multiplyBy3(a) a = b + 3 print(a)</pre> Select ONE: ☐ A. 8 (5.1%) ☐ B. 15 (3.8%) ☐ C. 45 (1.3%) ☐ D. 48 (89.9%) → CORRECT

Table A.2, Continued

#	Question
6	What does executing the following code do? <pre data-bbox="285 451 1279 705"> i = len(array) while (i > 0): print(array[i-1]) i = i - 1 </pre> Select ONE: ☐ A. Prints out the array in reverse order (40.5%) → CORRECT ☐ B. Prints out the array in order (3.8%) ☐ C. Prints out the array in reverse order but errors out at the last index (8.9%) ☐ D. Prints out the array in reverse order except the last element (20.3%) ☐ E. Prints out the array in reverse order except the first element (26.6%)

Table A.2, Continued

#	Question
7	The following code finds the minimum element in an integer array and prints out that minimum value. Assume that the array is guaranteed to have at least one element. The solution should work on any non-empty integer array of size 1 or greater. <pre> minimum = array[0] for value in array: # Insert code here print(minimum) </pre> What is the missing code that goes in the space above? Select ONE: ☐ A. <pre>if (value <= minimum): value = array[value]</pre> (24.1%) ☐ B. <pre>if (value != minimum): minimum = value</pre> (7.6%) ☐ C. <pre>if (value < minimum): minimum = value</pre> (58.2%) → CORRECT ☐ D. <pre>if (value > minimum): minimum = value</pre> (10.1%)

A.3 CS2 Pre-Term Test Questions

Below are the 7 questions given on the pre-term test. The sources of the 7 questions are as follows:

Questions 2-3, 5: From the work of Simon et al. [64].

Questions 1, 4, 6: From previous instructor's tests.

All Other Questions: Custom made for the test by or with instructors. See Section 6.1.3 for more details on how the questions were created).

Table A.3. CS2 Pre-Term Test Questions

#	Question
---	----------

Table A.3, Continued

#	Question
1	The following method consecutiveRepeats should return true if any two consecutive elements in the array x are identical (and return false otherwise). <pre> public static boolean consecutiveRepeats (int[] x){ // missing code goes here } </pre> Which of the following is the missing code from consecutiveRepeats? Select ONE: ☐ A. <pre> boolean b = true; for(int i = 0; i < x.length-1; i++) { if(x[i] == x[i+1]) { b = true; } else { b = false; } } return b; </pre> (22.7%) ☐ B. <pre> for(int i = 0; i < x.length-1; i++) { if(x[i] == x[i+1]) { return true; } } return false; </pre> (65.3%) → CORRECT ☐ C. <pre> boolean b = true; for (int i = 0; i < x.length-1; i++) { if(x[i] == x[i+1]) { b = true; } } return b; </pre> (2.3%) ☐ D. <pre> boolean b = true; for(int i = 0; i < x.length-1; i++) { if (x[i] != x[i+1]) { b = false; } } return b; </pre> (8.8%) ☐ E. <pre> for (int i = 0; i < x.length-1; i++) { if (x[i] != x[i+1]) { return false; } } return true; </pre> (0.9%)

Table A.3, Continued

#	Question
2	What will be the value of the variable result after the following code statements are executed? <pre data-bbox="285 510 1279 1052"> int[] nums1 = { 1, -5, 2, 0, 4, 2, -3 }; int[] nums2 = { 1, -5, 2, 4, 4, 2, 7 }; int result = 0; int j = 0; while (j < nums1.length) { if (nums1[j] != nums2[j]) { result = result + 1; } j = j + 1; } </pre> Select ONE: ☐ A. 0 (2.3%) ☐ B. 2 (87.0%) → CORRECT ☐ C. 3 (2.3%) ☐ D. 5 (7.9%) ☐ E. 7 (0.5%)

Table A.3, Continued

#	Question
3	What is the outcome or likely purpose of the following code? <pre data-bbox="285 451 1279 825"> int result = 0; for (int j = 0; j < number.length; j++) { if (number[j] < 0) { result = result + 1; } } </pre> Select ONE: ☐ A. to find the smallest number in the array ☐ B. to count the negative numbers in the array (96.3%) → CORRECT ☐ C. to sum the negative numbers in the array (2.3%) ☐ D. to add 1 to each of the negative numbers in the array (0.9%) ☐ E. to find the index of the first negative number in the array (0.5%)

Table A.3, Continued

#	Question
4	Which of the following is equivalent to and has the same effect as: <pre>b = (b == false);</pre> Select ONE: ☐ A. <pre>b = (b != b);</pre> (1.9%) ☐ B. <pre>b = (b == true);</pre> (1.4%) ☐ C. <pre>b = (b != false);</pre> (1.4%) ☐ D. <pre>b = (b != true);</pre> (73.6%) → CORRECT ☐ E. <pre>b = (b == b);</pre> (0.5%) ☐ F. More than one of the above statements is equivalent (20.8%)

Table A.3, Continued

#	Question
5	If a dependent child is a person under 18 years of age who does not earn \$10000 or more a year, which expression would define a dependent child? Select ONE: ☐ A. age < 18 && salary < 10000 (94.0%) → CORRECT ☐ B. age < 18 salary < 10000 (1.4%) ☐ C. age <= 18 && salary <= 10000 (4.6%) ☐ D. age <= 18 salary <=10000

Table A.3, Continued

#	Question
6	Consider the following code: <pre> public void Q14 (int a, int b) { if ((a > 0) && (b > 0)) { if (a < b) { System.out.println("A"); } else { System.out.println("B"); } } else if ((a < 0) (b < 0)) { System.out.println("C"); } else { System.out.println("D"); } } </pre>

Table A.3, Continued

#	Question
6	What gets printed as a result of the following call: Q14 (1, 3) Select ONE: <input data-bbox="337 667 370 699" type="radio"/> A. B D (4.2%) <input data-bbox="337 835 370 867" type="radio"/> B. D (0.5%) <input data-bbox="337 972 370 1003" type="radio"/> C. A (92.6%) → CORRECT <input data-bbox="337 1140 370 1171" type="radio"/> D. C B (0.5%) <input data-bbox="337 1339 370 1371" type="radio"/> E. B C (2.3%)

Table A.3, Continued

#	Question
7	Consider the following code: <pre data-bbox="285 451 1279 1201"> public static void main(String[] args) { int a = 8; int b = 25; funct(a, b); System.out.print(a + ", " + b); } public static void funct(int a, int b) { int temp = a; a = b; b = temp; } </pre> What is the output of the code above? Select ONE: ☐ A. 8, 25 (33.3%) → CORRECT ☐ B. 25, 8 (59.3%) ☐ C. Neither A nor B is correct (6.9%)

A.4 ORG Pre-Term Test Questions

Below are the 9 questions given on the pre-term test. The sources of the 9 questions are as follows:

Questions 6-7: From a Data Structures (CS2) course taught by Joe Gibbs Politz in 2017 [49].

Question 9: From the work of Simon et al. [64].

All Other Questions: Custom made for the test by or with instructors. See Section 6.1.3 for more details on how the questions were created).

Table A.4. ORG Pre-Term Test Questions

#	Question
---	----------

Table A.4, Continued

#	Question
1	The following method should return a student's letter grade, however there is a bug in the method. The bug is not a syntax error and the code compiles. Which line is the bug at? <pre> 1. public static String buggyLetterGrade(int grade) { 2. String letter; 3. if(grade >= 90) { letter = "A"; } 4. if(grade >= 80) { letter = "B"; } 5. else if(grade >= 70) { letter = "C"; } 6. else if(grade >= 60) { letter = "D"; } 7. else { letter = "F"; } 8. return letter; 9. }</pre> Select ONE: ☐ A. 3 (2.5%) ☐ B. 4 (92.9%) → CORRECT ☐ C. 5 (0.9%) ☐ D. 6 (1.9%) ☐ E. 7 (1.9%)

Table A.4, Continued

#	Question
2	What does the mystery method return when index is greater than the length of the array, arr? <pre> static int mystery(int[] arr, int index){ if(index < arr.length && arr[index] > 0){ return arr[index]; } else if(index > arr.length){ return -1; } else { return 0; } } </pre> Select ONE: ☐ A. -1 (85.5%) → CORRECT ☐ B. 0 (0.3%) ☐ C. ArrayIndexOutOfBoundsException is thrown (13.9%) ☐ D. None of the above (0.3%)

Table A.4, Continued

#	Question
3	The following method should return the string <code>s</code> with <code>". . ."</code> appended to the end if <code>s</code> has a length greater than 0, otherwise it should return an empty string. Select the code to complete the method by replacing the line <code>// FILL IN HERE</code> with the correct code. Assume that <code>s</code> will not be null. <pre data-bbox="337 630 1333 896"> static int someMethod(String s) { if(s.length() > 0) { return s + "..."; } // FILL IN HERE } </pre> Select ONE: ☐ A. (No code needs to be added) (1.23%) ☐ B. <code>return s;</code> (35.8%) → CORRECT ☐ C. <pre>else if(s.length() < 0) { return ""; }</pre> (0.6%) ☐ D. <pre>else if(s.length() <= 0) { return ""; }</pre> (60.8%) ☐ E. <pre>else if(s.length() < 0) { return s; } else { return null; }</pre> (0.9%)

Table A.4, Continued

#	Question
4	The following method should return the element in <code>strArr</code> at index <code>index</code>, and if <code>index</code> is out of bounds, return the empty string. However, this implementation of the method has a bug. The code compiles and it is not a syntax error. Which line is the bug at? Assume the index passed in will not be negative. <pre> 1. String getElementAtIndex(String[] strArr, int index) { 2. if(index > strArr.length) { 3. return ""; 4. } 5. else { 6. return strArr[index]; 7. } 8. }</pre> Select ONE: ☐ A. 1 (3.7%) ☐ B. 2 (88.3%) → CORRECT ☐ C. 3 (1.2%) ☐ D. 5 (1.5%) ☐ E. 6 (5.3%)

Table A.4, Continued

#	Question
5	Fill in the following code to make the method return an array with the same elements as the parameter arr but in reverse order. <pre data-bbox="337 510 1333 816"> int[] getReversedElements(int[] arr) { int[] arrReversed = new int[arr.length]; // FILL IN HERE return arrReversed; } </pre> Select ONE: ☐ A. <pre data-bbox="503 905 1271 1047"> for(int i = 0; i < arr.length; i++) { arrReversed[i] = arr[i]; } </pre> ☐ B. <pre data-bbox="503 1110 1271 1253"> for(int i = 0; i < arr.length; i++) { arrReversed[i] = arr[arr.length - i - 1]; } </pre> (73.2%) → CORRECT ☐ C. <pre data-bbox="503 1373 1271 1516"> for(int i = arr.length - 1; i >= 0; i--) { arrReversed[i] = arr[arr.length - i - 1]; } </pre> (14.2%) → CORRECT ☐ D. <pre data-bbox="503 1638 1271 1780"> for(int i = arr.length - 1; i >= 0; i--) { arrReversed[arr.length - i] = arr[i]; } </pre> (12.7%)

Table A.4, Continued

#	Question
6	Consider the following program for questions 6-8: <pre> 1. class Car { 2. public String color; 3. public Car(String color) { 4. this.color = color; 5. } 6. } 7. 8. class Q6 { 9. public static void g(Car c1, Car c2) { 10. c1 = c2; 11. c1.color = "blue"; 12. } 13. 14. public static void main(String[] args) { 15. Car car1 = new Car("orange"); 16. Car car2 = new Car("green"); 17. g(car1, car2); 18. System.out.println(car1.color + ", " + car2.color); 19. } 20. }</pre>

Table A.4, Continued

#	Question
6	What does the program print? Select ONE: ☐ A. orange , green (15.1%) ☐ B. blue , blue (48.2%) ☐ C. blue , orange (18.5%) ☐ D. green , orange (1.54%) ☐ E. orange , blue (16.1%) → CORRECT

Table A.4, Continued

#	Question
7	If line 10 was changed to <code>c2 = c1;</code>, what would the program print? Select ONE: ☐ A. blue , green (36.1%) → CORRECT ☐ B. blue , blue (42.9%) ☐ C. orange , orange (5.6%) ☐ D. green , orange (3.1%) ☐ E. orange , blue (12.0%)

Table A.4, Continued

#	Question
8	Which statements are true at the end of the program? Select ALL that apply: ☐ A. The variable <code>car1</code> is on the stack and stores a <code>Car</code> object (26.5%) ☐ B. The variable <code>car1</code> is on the stack and stores a reference to a <code>Car</code> object, which is on the heap (69.8%) → CORRECT ☐ C. The variable <code>car2</code> is on the stack and stores a <code>Car</code> object (25.6%) ☐ D. The variable <code>car2</code> is on the heap and stores a <code>Car</code> object (26.9%) ☐ E. The variable <code>car2</code> is on the stack and stores a reference to a <code>Car</code> object, which is also on the stack (26.9%) ☐ F. There are two <code>Car</code> objects on the heap (53.4%) → CORRECT

Table A.4, Continued

#	Question
9	Please examine the code below: <pre data-bbox="380 478 1101 800"> int result = 0; for(int j = 0; j < number.length; j++) { if(number[j] < 0) { result = result + 1; } }</pre> What is the outcome or likely purpose of the code above? Assume number is an array of integers that has already been initialized. Select ONE: ☐ A. to find the smallest number in the array ☐ B. to count the negative numbers in the array (88.0%) → CORRECT ☐ C. to sum the negative numbers in the array (3.4%) ☐ D. to add 1 to each of the negative numbers in the array (0.6%) ☐ E. to find the index of the first negative number in the array

Bibliography

- [1] Frank K. Abou-Sayf. Does the elimination of prerequisites affect enrollment and success? *Community College Review*, 36(1):47–62, 2008.
- [2] Tope Adeyemi-Bello, Amy McMillan, and Amy McMillan-Capehart. Prerequisite coursework as a predictor of performance in a graduate management course. *Journal of College Teaching and Learning*, 5:11–16, 2011.
- [3] Mikel Aickin and Helen Gensler. Adjusting for multiple testing when reporting research results: the bonferroni vs holm methods. *American Journal of Public Health*, 86(5): 726–728, 1996.
- [4] Christine Alvarado, Gustavo Umbelino, and Mia Minnes. The persistent effect of pre-college computing experience on college cs course grades. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, SIGCSE ’18, page 876–881, 2018.
- [5] Pansy W. Aycaster. Factors impacting success in community college developmental mathematics courses and subsequent courses. *Community College Journal of Research & Practice*, 25(5-6):403–416, 2001.
- [6] Maureen Biggers, Anne Brauer, and Tuba Yilmaz. Student perceptions of computer science: A retention study comparing graduating seniors with cs leavers. In *Proceedings of the 39th SIGCSE Technical Symposium on Computer Science Education*, SIGCSE ’08, page 402–406, 2008.
- [7] James H. Block. *Mastery Learning: Theory and Practice*. Holt, Rinehard and Winston, Inc., 1971.
- [8] Eric W. Burkholder, Gabriel Murillo-Gonzalez, and Carl Wieman. Importance of math prerequisites for performance in introductory physics. *Physical Review Physics Education Research*, 17:010108–1–010108–14, 2021.
- [9] Ricardo Caceffo, Steve Wolfman, Kellogg S. Booth, and Rodolfo Azevedo. Developing a computer science concept inventory for introductory programming. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*, SIGCSE ’16, page

364–369, 2016.

- [10] Angela Carbone, Linda Mannila, and Sue Fitzgerald. Computer science and it teachers’ conceptions of successful and unsuccessful teaching: A phenomenographic study. *Computer Science Education*, 17(4):275–299, 2007.
- [11] Askar Choudhury, Don Robinson, and Ramaswamy Radhakrishnan. Effect of prerequisite on introductory statistics performance. *Journal of Economics and Economic Education Research*, 8(3):19, 2007.
- [12] Eugène J.F.M. Custers. Long-term retention of basic science knowledge: A review study. *Advances in Health Sciences Education*, 15(1):109–128, 2010.
- [13] Holger Danielsiek, Wolfgang Paul, and Jan Vahrenhold. Detecting and understanding students’ misconceptions related to algorithms and data structures. In *Proceedings of the 43rd Technical Symposium on Computer Science Education*, pages 21–26, 2012.
- [14] Benoît Dompnier, Céline Darnon, Emanuele Meier, Catherine Brandner, Annique Smeding, and Fabrizio Butera. Improving low achievers’ academic performance at university by changing the social value of mastery goals. *American Educational Research Journal*, 52(4):720–749, 2015.
- [15] Hermann Ebbinghaus. *Über das Gedächtnis: Untersuchungen zur Experimentellen Psychologie (Memory: A Contribution to Experimental Psychology)*. Dunker and Humblot, 1885.
- [16] Joe Feldman. *Grading for Equity: What It Is, Why It Matters, and How It Can Transform Schools and Classrooms*. Corwin, 2018.
- [17] Joseph P. Forester, David L. McWhorter, and Maria S. Cole. The relationship between premedical coursework in gross anatomy and histology and medical school performance in gross anatomy and histology. *Clinical Anatomy: The Official Journal of the American Association of Clinical Anatomists and the British Association of Clinical Anatomists*, 15(2):160–164, 2002.
- [18] James Garner, Paul Denny, and Andrew Luxton-Reilly. Mastery learning in computer science education. In *Proceedings of the Twenty-First Australasian Computing Education Conference, ACE ’19*, page 37–46, 2019.
- [19] Tina Götschi, Ian Sanders, and Vashti Galpin. Mental models of recursion. In *Proceeding of the 34th ACM Technical Symposium on Computer Science Education, SIGCSE ’03*, pages 346–350, 2003.
- [20] Jeffrey J. Green, Courtenay C. Stone, Abera Zegeye, and Thomas A. Charles. Changes in

math prerequisites and student performance in business statistics: Do math prerequisites really matter? *Journal of Economics and Finance Education*, 6(2):27–38, 2007.

- [21] Dianne Hagan and Selby Markham. Does it help to have some programming experience before beginning a computing degree program? In *Proceedings of the 5th Annual Conference on Innovation and Technology in Computer Science Education*, ITiCSE '00, pages 25–28, 2000.
- [22] Sally Hamouda, Stephen H. Edwards, Hicham G. Elmongui, Jeremy V. Ernst, and Clifford A. Shaffer. A basic recursion concept inventory. *Computer Science Education*, 27(2): 121–148, 2017.
- [23] Edward Holden and Elissa Weeden. The impact of prior experience in an information technology programming course sequence. In *Proceedings of the 4th conference on Information technology curriculum*, CITC4 '03, pages 41–46, 2003.
- [24] Faridul Islam, Saleheen Khan, Ian Wilson, and Reed Gooch. The value of prerequisite courses for statistics. *The Journal of Business Inquiry*, 7(1):61–67, 2008.
- [25] Association for Computing Machinery (ACM) Joint Task Force on Computing Curricula and IEEE Computer Society. *Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science*. ACM and IEEE, 2013.
- [26] Colleen E. Kauffman and David A. Gilman. Are prerequisite courses necessary for success in advanced courses?. 2002.
- [27] Päivi Kinnunen and Beth Simon. Phenomenography and grounded theory as research methods in computing education research field. *Computer Science Education*, 22(2): 199–218, 2012.
- [28] Sophia Krause-Levy, Sander Valstar, Leo Porter, and William G. Griswold. Exploring the link between prerequisites and performance in advanced data structures. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, SIGCSE '20, page 386–392, 2020.
- [29] Sophia Krause-Levy, William G. Griswold, Leo Porter, and Christine Alvarado. The relationship between sense of belonging and student outcomes in cs1 and beyond. In *Proceedings of the 17th ACM Conference on International Computing Education Research*, ICER '21, page 29–41, 2021.
- [30] Sophia Krause-Levy, Adrian Salguero, Rachel S. Lim, Hayden McTavish, Jelena Trajkovic, Leo Porter, and William G. Griswold. Instructor perspectives on prerequisite courses in computing. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1*, SIGCSE' 23, page 277–283, 2023.

- [31] Soohyun Nam Liao, Sander Valstar, Kevin Thai, Christine Alvarado, Daniel Zingaro, William G. Griswold, and Leo Porter. Behaviors of higher and lower performing students in CS1. In *Proceedings of the 25th Conference on Innovation and Technology in Computer Science Education*, ITiCSE '19, pages 196–202, 2019.
- [32] Alex Lishinski, Aman Yadav, Jon Good, and Richard Enbody. Learning to program: Gender differences and interactive effects of students' motivation, goals, and self-efficacy on performance. In *Proceedings of the 12th Conference on International Computing Education Research*, ICER '16, pages 211–220, 2016.
- [33] Raymond Lister. Ten years after the mccracken working group. *ACM Inroads*, 2(4):18–19, 2011.
- [34] Jane Margolis and Allan Fisher. *Unlocking the Clubhouse: Women in Computing*. MIT press, 2003.
- [35] Jane Margolis, Rachel Estrella, Joanna Goode, Jennifer Jellison Holme, and Kim Nao. *Stuck in the Shallow End: Education, Race, and Computing*. MIT Press, 2010.
- [36] Marshall A. Martin. Course prerequisites and undergraduate student performance. *NACTA Journal*, 33(1):38–42, 1989.
- [37] Karen B. McCarron and Alan N. Burstein. The importance of mathematics as a prerequisite to introductory financial accounting. *Community College Journal of Research and Practice*, 41(9):543–550, 2017.
- [38] Mark McCourt. *Teaching for Mastery*. John Catt Educational, Limited, 2019.
- [39] Michael McCracken, Vicki Almstrum, Danny Diaz, Mark Guzdial, Dianne Hagan, Yifat Ben-David Kolikant, Cary Laxer, Lynda Thomas, Ian Utting, and Tadeusz Wilusz. A multi-national, multi-institutional study of assessment of programming skills of first-year cs students. *ACM SIGCSE Bulletin*, 33(4), 2001.
- [40] Amy McMillan-Capehart and Tope Adeyemi-Bello. Prerequisite coursework as a predictor of performance in a graduate management course. *Journal of College Teaching & Learning (TLC)*, 5(7), 2008.
- [41] Karthik Muralidharan, Abhijeet Singh, and Alejandro J. Ganimian. Disrupting education? experimental evidence on technology-aided instruction in india. Technical report, National Bureau of Economic Research, 2016.
- [42] Laurie Murphy and Josh Tenenbergs. Do computer scientists know what they know? a calibration study of data structure knowledge. In *Proceedings of the 10th Annual ACM Conference on Innovation and Technology in Computer Science Education*, ITiCSE '05,

pages 148–152, 2005.

- [43] Johnathon Nash, Laura E. Boucheron, and Steven J. Stochaj. A correlative analysis of course grades as related to curricular prerequisite structure and inter-class topic dependencies. In *2021 IEEE Frontiers in Education Conference, FIE '21*, pages 1–5, 2021.
- [44] University of California San Diego Computer Science and Engineering (CSE) Department. Computer Science and Engineering (CSE) Course Catalog. <https://catalog.ucsd.edu/courses/CSE.html>, 2023.
- [45] Claudia Ott, Brendan McCane, and Nick Meek. Mastery learning in cs1 - an invitation to procrastinate?: Reflecting on six years of mastery learning. In *Proceedings of the 26th Annual ACM Conference on Innovation and Technology in Computer Science Education, ITiCSE '21*, page 18–24, 2021.
- [46] Miranda C. Parker, Mark Guzdial, and Shelly Engleman. Replication, validation, and use of a language independent cs1 knowledge assessment. In *Proceedings of the 2016 ACM Conference on International Computing Education Research, ICER '16*, page 93–101, 2016.
- [47] Jack Parkinson and Quintin Cutts. Investigating the relationship between spatial skills and computer science. In *Proceedings of the 14th ACM Conference on International Computing Education Research, ICER '18*, pages 106–114, 2018.
- [48] Joe Gibbs Politz. Computer Organization Course Syllabus Politz (2017). <https://github.com/ucsd-cse30-f17/syllabus>, 2017. Accessed: 2018-11-05.
- [49] Joe Gibbs Politz. Basic Data Structures and Object-Oriented Design Politz (2019). <https://ucsd-cse12-w19.github.io/>, 2019.
- [50] Leo Porter and Beth Simon. Retaining nearly one-third more majors with a trio of instructional best practices in cs1. In *Proceeding of the 44th ACM Technical Symposium on Computer Science Education, SIGCSE '13*, page 165–170, 2013.
- [51] Leo Porter, Cynthia Bailey Lee, and Beth Simon. Halving fail rates using peer instruction: A study of four computer science courses. In *Proceeding of the 44th ACM Technical Symposium on Computer Science Education, SIGCSE '13*, page 177–182, 2013.
- [52] Leo Porter, Daniel Zingaro, Soohyun Nam Liao, Cynthia Taylor, Kevin C. Webb, Cynthia Lee, and Michael Clancy. Bdsi: A validated concept inventory for basic data structures. In *Proceedings of the 15th ACM Conference on International Computing Education Research, ICER '19*, page 111–119, 2019.
- [53] Vennila Ramalingam, Deborah LaBelle, and Susan Wiedenbeck. Self-efficacy and mental

- models in learning to program. In *Proceedings of the 9th Conference on Innovation and Technology in Computer Science Education*, ITiCSE '04, pages 171–175, 2004.
- [54] Christine F. Reilly and Emmett Tomai. Should college algebra be a coerequisite for computer science 1? FECS: Position paper. In *Proceedings of the International Conference on Frontiers in Education: Computer Science and Computer Engineering (FECS)*, FECS '13, page 1, 2013.
 - [55] Anthony Robins. Learning edge momentum: a new account of outcomes in cs1. *Computer Science Education*, 20:37–71, 2010.
 - [56] Adam Rosenstein, Aishma Raghu, and Leo Porter. Identifying the prevalence of the impostor phenomenon among computer science students. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, SIGCSE '20, page 30–36, 2020.
 - [57] Allen A. Rovick, Joel A. Michael, Harold I. Modell, David S. Bruce, Barbara Horwitz, Thomas Adamson, Daniel R. Richardson, Dee U. Silverthorn, and Shirley A. Whitescarver. How accurate are our assumptions about our students' background knowledge? *Advances in Physiology Education*, 276(6):S93, 1999.
 - [58] Adrian Salguero, Julian McAuley, Beth Simon, and Leo Porter. A longitudinal evaluation of a best practices cs1. In *Proceedings of the 16th ACM Conference on International Computing Education Research*, ICER '20, page 182–193, 2020.
 - [59] Adrian Salguero, William G. Griswold, Christine Alvarado, and Leo Porter. Understanding sources of student struggle in early computer science courses. In *Proceedings of the 17th ACM Conference on International Computing Education Research*, ICER '21, page 319–333, 2021.
 - [60] Brian K. Sato, Amanda K. Lee, Usman Alam, Jennifer V. Dang, Samantha J. Dacanay, Pedro Morgado, Giorgia Pirino, Jo Ellen Brunner, Leanne A. Castillo, Valerie W. Chan, and Judith H. Sandholtz. What's in a prerequisite? a mixed-methods approach to identifying the impact of a prerequisite course. *CBE—Life Sciences Education*, 16(1):ar16, 2017.
 - [61] Burton F. Schaffer and D. Ordell Calkins. An appraisal of prerequisites to business finance. *Journal of Financial Education*, Fall, 1980(9):51–55, 1980.
 - [62] Justin F. Shaffer, Jennifer V. Dang, Amanda K. Lee, Samantha J. Dacanay, Usman Alam, Hollie Y. Wong, George J. Richards, Pavan Kadandale, and Brian K. sato. A familiar(ity) problem: Assessing the impact of prerequisites and content familiarity on student learning. *PloS one*, 11(1):e0148051, 2016.
 - [63] Judy Sheard, Angela Carbone, Raymond Lister, Beth Simon, Errol Thompson, and Jacqueline L. Whalley. Going solo to assess novice programmers. In *Proceedings of the 13th*

ACM Conference on Innovation and Technology in Computer Science Education, pages 209–213, 2008.

- [64] Simon, Judy Sheard, Daryl D’Souza, Peter Klemperer, Leo Porter, Juha Sorva, Martijn Stegeman, and Daniel Zingaro. Benchmarking introductory programming exams: Some preliminary results. In *Proceedings of the 12th Conference on International Computing Education Research*, ICER ’16, page 103–111, 2016.
- [65] Elliot Soloway. Learning to program = learning to construct mechanisms and explanations. *Communications of the ACM*, 29(9):850–858, 1986.
- [66] Krista M. Soria and Lori Mumpower. Critical building blocks: Mandatory prerequisite registration systems and student success. *NACADA Journal*, 32(1):30–42, 2012.
- [67] James H. Steiger. Tests for comparing elements of a correlation matrix. *Psychological Bulletin*, 87(2):245–251, 1980.
- [68] Matthew Tennyson and Marc Beck. A study of knowledge retention in introductory programming courses. *Journal of Computing Sciences in Colleges*, 33(4):13–20, 2018.
- [69] Allison Elliott Tew, W. Michael McCracken, and Mark Guzdial. Impact of alternative introductory courses on programming concept understanding. In *Proceedings of the First International Workshop on Computing Education Research*, page 25–35, 2005.
- [70] Sander Valstar, William G. Griswold, and Leo Porter. The relationship between prerequisite proficiency and student performance in an upper-division computing course. In *Proceedings of the 50th Technical Symposium on Computer Science Education*, SIGCSE ’19, pages 794–800, 2019.
- [71] Sander Valstar, Sophia Krause-Levy, Alexandra Macedo, William G. Griswold, and Leo Porter. Faculty views on the goals of an undergraduate cs education and the academia-industry gap. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, SIGCSE ’20, page 577–583, 2020.
- [72] Sander Valstar, Sophia Krause-Levy, Adrian Salguero, Leo Porter, and William G. Griswold. Proficiency in basic data structures among various subpopulations of students at different stages in a cs program. In *Proceedings of the 26th ACM Conference on Innovation and Technology in Computer Science Education*, ITiCSE ’21, page 429–435, 2021.
- [73] J.D. Vitale, S.P. Wanger, and D.C. Adams. Explaining student performance in an undergraduate agricultural economics classroom. *NACTA Journal*, 54(1):2, 2010.
- [74] Henry M. Walker. Prerequisites: Shaping the computing curriculum. *ACM Inroads*, 1(4): 14–16, 2010.

- [75] Jennifer Wang, Hai Hong, Jason Ravitz, and Sepehr Hejazi Moghadam. Landscape of k-12 computer science education in the u.s.: Perceptions, access, and barriers. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*, SIGCSE '16, page 645–650, 2016.
- [76] Christopher Watson and Frederick W.B. Li. Failure rates in introductory programming revisited. In *Proceedings of the 2014 Conference on Innovation and Technology in Computer Science Education*, ITiCSE '14, page 39–44, 2014.
- [77] Eric J. Williams. The comparison of regression variables. *Journal of the Royal Statistical Society. Series B (Methodological)*, 21(2):396–399, 1959.
- [78] Brenda Cantwell Wilson and Sharon Shrock. Contributing to success in an introductory computer science course: A study of twelve factors. In *Proceedings of the 32nd Technical Symposium on Computer Science Education*, SIGCSE '01, pages 184–188, 2001.
- [79] Robin Wright, Sehoya Cotner, and Amy Winkel. Minimal impact of organic chemistry prerequisite on student performance in introductory biochemistry. *CBE—Life Sciences Education*, 8(1):44–54, 2009.
- [80] Lisa Zhang and Sonya Allin. Just-in-time prerequisite review for a machine learning course. In *Proceedings of the 25th Western Canadian Conference on Computing Education*, WCCCE '23, 2023.
- [81] Daniel Zingaro and Leo Porter. Impact of student achievement goals on cs1 outcomes. In *Proceedings of the 47th Technical Symposium on Computing Science Education*, SIGCSE '16, pages 279–296, 2016.
- [82] Daniel Zingaro, Michelle Craig, Leo Porter, Brett A. Becker, Yingjun Cao, Phill Conrad, Diana Cukierman, Arto Hellas, Dastyni Loksa, and Neena Thota. Achievement goals in CS1: Replication and extension. In *Proceedings of the 49th Technical Symposium on Computer Science Education*, SIGCSE '18, pages 687–692, 2018.
- [83] Stuart Zweben. Enrollment and retention in u.s. computer science bachelor's programs in 2016-17. *ACM Inroads*, 10(4):47–59, 2019.
- [84] Stuart Zweben and Betsy Bizot. 2019 taulbee survey: Total undergrad cs enrollment rises again, but with fewer new majors; doctoral degree production recovers from last year's dip. 2(5), 2019.