Not standing at the same starting line - investigation of prior programming experience on student performance in an introductory programming course in ECE

Ms. Ziyue Li, University of Illinois - Urbana Champaign

Ziyue Li received her B.S. in Computer Engineering from the University of Illinois - Urbana Champaign in 2019. She is currently pursuing a Master of Science degree in ECE from the same institution with the Systems Networking Research Group. She has assisted with undergraduate ECE courses for six semesters and was involved with the development of numerous online courses offered through Coursera. Off campus, she has interned with various teams at Google in designing reliable systems for use in production. Her current research focuses on mobile computing and sensing systems in the acoustic domain.

Prof. Yuting W. Chen, University of Illinois Urbana - Champaign

Dr. Yuting W. Chen received the B.S. degree from the University of Illinois - Urbana Champaign in 2007, and the M.S. and Ph.D. degrees from Rensselaer Polytechnic Institute in 2009 and 2011, all in Electrical Engineering. She is currently a Teaching Assistant Professor with the Department of Electrical and Computer Engineering at University of Illinois at Urbana-Champaign. Prior to joining ECE Illinois, she worked at IBM Systems Group in Poughkeepsie, NY in z Systems Firmware Development. Her current interests include recruitment and retention of under-represented students in STEM, integrative training for graduate teaching assistants, and curriculum innovation for introductory programming courses.

Not standing at the same starting line - investigation of prior programming experience on student performance in an introductory programming course in ECE

Abstract

There have been a good number of studies on computer preparedness of incoming engineering students, but majority of them focus on simply having access to computers. As personal computers are becoming more and more prevalent, this information alone can no longer pinpoint the difference observed in programming readiness among students. A few studies have shown that students with prior programming experience have an initial advantage in introductory programming courses. This paper investigates the relevance of the abovementioned statement in a sophomore level introductory programming course in ECE at a public research university. We look at the number of years of prior programming experience students have upon entering the course and the impact it has on their performance. In particular, we are interested to understand whether the advantage from prior experience is profound only for some parts of the course or throughout the course. Four semesters of data are collected and analyzed, which include over 900 students who have completed the course. Students are categorized into four groups by number of years of prior programming experience, from less than 1 year to more than 3 years. A one-way ANOVA test is used to determine whether there is any statistically significant difference between groups in terms of performance on the following components of the course: programming assignments, computer-based quizzes, and paper exams. A Bonferroni post-hoc test is then applied to determine between which groups such difference exists. Literature has shown that women are less likely than men to enroll in high school programming classes [1] and students from lower socioeconomical status schools have limited resource to learn programming [2]. We hope the result of this study will provide insights on advising freshmen in ECE on preparation for the first programming course in the major, which in turn could help improve the retention rates of under-represented and under-privileged students.

Introduction

Previous studies on impact of prior experience

Study of prior computing experience's impact on student performance can be dated back to 1980s [3], [4], [5]. These studies either have a broad definition of prior experience [4] including usage of personal computers at home and at school or a narrow cut-off of having taken at least one high school computer science course [3], [5]. These early studies indicated that prior computing experience would give students advantage in their entry-level programming course.

Later studies have focus on prior programming experience gained through both inside and outside of school [6], [7], [8]. Results presented in these studies painted very different pictures. Bergin et al. found no significant difference between students with prior experience versus those without. They concluded that it may be due to the fact that secondary schools in their country do not offer courses similar to those at the college level. In comparison, Hagan et al. found that prior programming experience would give students initial advantages in the first two exams, but not the final, and experience with more languages also led to better performance. Last but not least, Holden et al. found that students with prior experience completed the first programming course with 1.0 point higher than those without on a 4.0 scale.

Not all of these prior studies indicated the type of assessment methods used in their introductory programming course. Depending on the type of assessment, the impact of prior programming experience on course performance could be different.

Course content and components

In our sophomore level introductory programming course, we take a bottom-up approach to introduce students to the design and programming of computing systems. Our course focuses on C programming, but starts by covering low-level concepts such as system level I/O, subroutines, and run-time stacks in LC-3 assembly language. This is a follow-up from the freshman level prerequisite course, in which students are introduced to digital circuits, basic computer organization, and machine language programming. The course then covers basic programming concepts, functions, arrays, pointers, recursion, simple data structures, and concepts in object-oriented programming.

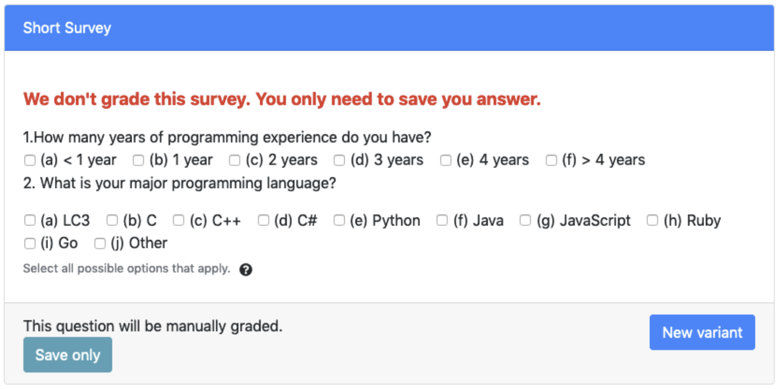
Student grades in our course are calculated from the following components: 12 programming assignments, 6 computer-based quizzes, 2 midterm exams and a final exam. For each programming assignment, students have at least one week from assignment release date to complete it. Computer-based quizzes are 50-minute each taken on a terminal inside a campus testing facility. Quiz questions generally require no more than 25 lines of code. Midterm exams are 90-minute long each and final exam is 180-minute long, all in paper format. The first midterm focuses on low-level concepts and programming in LC-3 assembly language. The second midterm focuses on concepts related to C programming. The final is cumulative but weights towards data structures and object-oriented programming. These exams have a mix of short programming questions and concept questions. In this paper, we examine how prior programming experience impacts student's grade in each course component.

Historical background of changes in assessment

Assessments in our introductory programming course has gone through several phases of changes since its inception. When the course was first created, all exams were paper-based. A curriculum redesign in 2013 compressed a three-course sequence into a two-course sequence and assessments in the introductory programming course changed to computer-based exams. In 2017, assessments changed yet again to a hybrid model with computer-based quizzes and paper exams.

Data Collection and Categorization

In our course, prior programming experience information is collected in a pre-course survey. Students are asked to indicate their level of experience ranging from less than 1 year to 4 or more years. They are also asked to indicate their major programming languages. In order to have groups of comparable size for our study, we define 4 groups based on students' prior experience - Group 1: less than one year, Group 2: one to two years, Group 3: two to three years, and Group 4: three or more years. The pre-course survey is shown in Figure 1.


Short Survey

We don't grade this survey. You only need to save you answer.

1.How many years of programming experience do you have?

☐ (a) < 1 year ☐ (b) 1 year ☐ (c) 2 years ☐ (d) 3 years ☐ (e) 4 years ☐ (f) > 4 years

2. What is your major programming language?

☐ (a) LC3 ☐ (b) C ☐ (c) C++ ☐ (d) C# ☐ (e) Python ☐ (f) Java ☐ (g) JavaScript ☐ (h) Ruby
☐ (i) Go ☐ (j) Other

Select all possible options that apply.

This question will be manually graded.

Figure 1: Pre-course survey questions.

Students' grades in course assignments and assessments are also collected from Fall 2017 to Fall 2019. Entries with no pre-course survey response or a zero grade in any course component were removed. Based on students' prior programming experience, they are then divided into 4 groups defined previously. Table 1 reports the total number of valid student data used in our analysis and their distribution among 4 groups across 4 semesters from Fall 2017 to Fall 2019. Data from Fall 2018 is not presented in this paper because pre-course survey was not conducted that semester. In all tables, Fall 2017 is represented as FA17, Spring 2018 as SP18 and so on.

	FA17	SP18	SP19	FA19
Group 1 (<1 year)	94	80	64	54
Group 2 (1-2 years)	66	67	62	47
Group 3 (2-3 years)	52	55	56	50
Group 4 (>3 years)	35	58	57	32
Total	247	260	239	183

Table 1: Number of students in each group from FA17 to FA19.

Discussion

Impact of prior programming experience
The mean score and standard deviation for each group in each course component is shown in Table 2. In general, the more years of prior programming experience a student has, the higher their score is in most course components. One-way ANOVA tests are performed using IBM SPSS Statistics [9] on each semester's data to determine whether this trend in performance between groups is statistically significant. Table 3 summarizes the p values from the one-way ANOVA test in each course component from Fall 2017 to Fall 2019. A p value less than 0.05 indicates there is a statistically significant difference between groups. To generalize the results over multiple semesters, we focus on course components that show statistically significant difference in student performance between groups for three or more semesters. Therefore, prior programming experience has a statistically significant impact on a student's performance in quizzes, the second midterm (MT2) and the final, but no statistically significant impact is found in performance in programming assignments nor the first midterm (MT1).

	Mean (Std. Dev.)	Group 1	Group 2	Group 3	Group 4
Prog. Asgmt.	FA17	91.45(16.5)	95.21(11.0)	93.32(13.5)	94.90(9.6)
	SP18	91.36(13.2)	93.46(11.9)	96.47(5.6)	96.46(9.2)
	SP19	92.88(11.1)	94.55(12.0)	95.55(8.9)	95.37(13.8)
	FA19	92.02(16.2)	95.65(7.5)	92.14(16.1)	92.41(14.9)
Quiz	FA17	83.83(19.9)	89.09(13.8)	92.69(12.5)	95.14(10.2)
	SP18	86.51(18.8)	92.25(13.1)	94.75(8.6)	97.17(7.7)
	SP19	87.41(20.5)	92.14(15.9)	95.38(10.5)	97.16(7.8)
	FA19	85.89(18.7)	90.81(15.1)	88.74(17.5)	96.81(8.0)
MT1	FA17	83.37(15.7)	83.18(14.2)	80.87(19.5)	89.24(8.4)
	SP18	82.11(14.7)	86.86(10.9)	82.63(15.3)	89.57(9.2)
	SP19	86.10(11.5)	85.96(10.3)	87.17(11.2)	88.50(9.9)
	FA19	89.94(8.1)	90.41(8.0)	90.35(7.7)	92.44(5.3)
MT2	FA17	69.61(20.5)	73.38(20.9)	78.38(15.5)	83.00(14.9)
	SP18	66.10(19.5)	74.38(13.8)	77.33(15.0)	83.14(12.4)
	SP19	68.22(20.6)	71.86(15.3)	72.61(17.1)	76.47(17.0)
	FA19	76.64(14.7)	80.57(14.9)	80.60(12.5)	85.86(10.6)
Final	FA17	61.61(19.9)	70.11(17.8)	70.31(17.5)	80.36(11.7)
	SP18	70.34(21.4)	80.28(16.9)	75.71(18.8)	84.52(10.9)
	SP19	70.67(18.5)	71.48(15.6)	75.73(13.4)	78.55(14.6)
	FA19	74.41(18.8)	78.40(18.1)	79.82(14.7)	85.31(9.0)

Table 2: Mean and standard deviation of each group by course components from FA17 to FA19.

p value	FA17	SP18	SP19	FA19
Prog. Asgmt.	0.327	0.014	0.565	0.548
Quiz	0.001	0.000	0.002	0.023
MT1	0.098	0.003	0.547	0.496
MT2	0.002	0.000	0.088	0.027
Final	0.000	0.000	0.021	0.027

Table 3: One-way ANOVA test results in all course components among groups from FA17 to FA19.

Content of the different course components is examined to understand why prior programming experience only impacts student performance in certain components of the course but not others. For the programming assignments, students are given one to two weeks to complete. They are allowed to consult lecture materials, seek help from course staff, and work in groups. One-way ANOVA tests indicate that prior programming experience has no statistically significant impact on student performance in programming assignments. We believe that when students are given ample time and resources, they are able to perform at similar levels regardless of their prior exposure to programming. On the other hand, prior programming experience has a statistically significant impact on a student's performance in the quizzes. Contrary to the programming assignments, students are only given 50 minutes to complete each computer-based quiz, and they are not allowed to consult outside material. During each quiz, students are expected to read the problem statement, write no more than 25 lines of code with hints from provided algorithm, compile and test their solution locally, and finally submit their solution to a server where our auto-grader will run test cases and provide feedback. We think that prior programming experience familiarizes students with the development process that involves compiling, testing and debugging. As shown in Figure 2, students with more years of prior programming experience tend to have experience in more languages. In addition, they are also likely to be more familiar with language syntax. Majority of the quizzes are in C, which has similar syntax as C++. As shown in Figure 3, only 9% of students in Group 1 reported that one of their major programming languages is C++, compared to 19.3% in Group 2, 31.9% in Group 3, and 24.3% in Group 4.

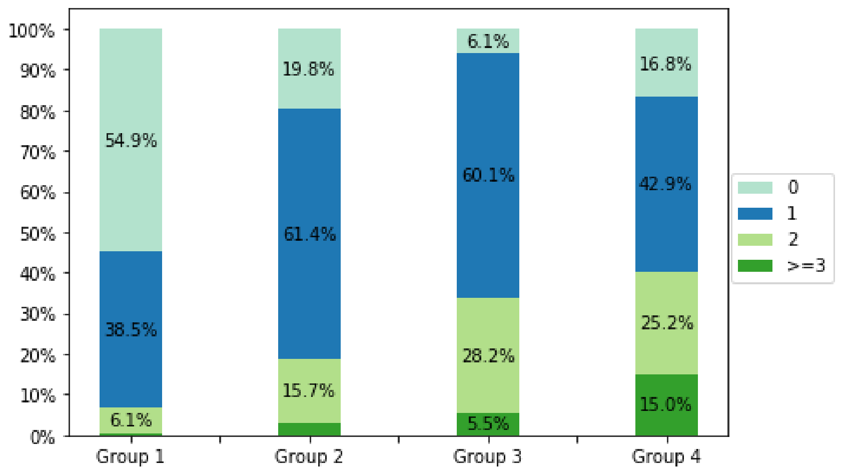


Figure 2: Number of major programming languages known by each group, excluding LC-3 and C which are introduced in the prerequisite course.

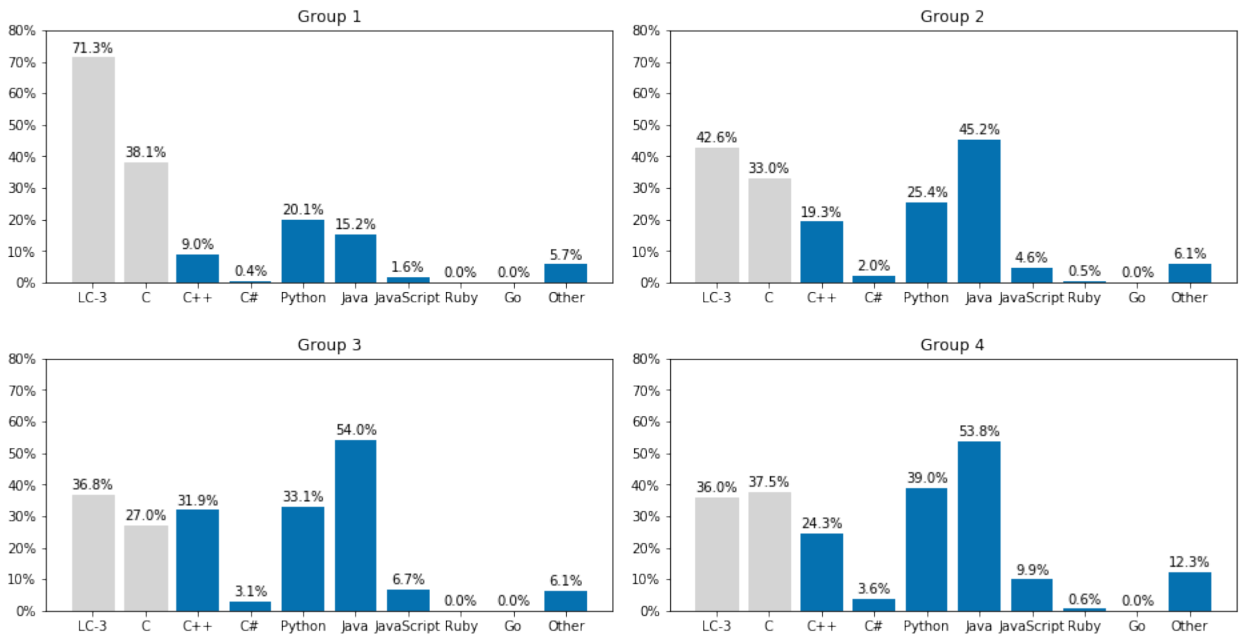


Figure 3: Distribution of students' major programming languages by group. Students are expected to have basic knowledge of LC-3 and C through the prerequisite course.

Out of the three paper exams in this course, prior programming experience has a statistically significant impact on student performance on the second midterm and the final. It does not have a statistically significant impact on the first midterm which covers concepts on LC-3 Assembly. Assembly language is vastly different from the languages students have prior exposure to as shown in Figure 3. Furthermore, the prerequisite for our course, which most students take during their freshman year, introduces them to LC-3 Assembly. Regardless of which group a student is in, they have similar amount of exposure to this topic. We believe the lack of prior programming experience does not set students back in the situation where they are introduced to the topic at the same time in a prerequisite course. On the contrary, the second midterm and the final exam covers topics on basic C programming, data structures, algorithms, and object oriented programming. Lack of prior programming experience puts students at a disadvantage in these topics [6], [7] compared to those with prior programming experience. Additionally, around 50% of students in Group 2, Group 3 and Group 4 reported that Java is one of their major programming languages, while this number is only 15.2% for students in Group 1. Java is the language used in the AP Computer Science exam [10] and according to its curriculum, students learn object oriented programming, basic control structures, arrays, recursion, etc. which are all topics assessed in our second midterm and final. If these students learned Java through the AP Computer Science curriculum, they would have already had exposure to these topics. Therefore, students with limited prior programming experience would be at a disadvantage.

Observing that prior programming experience impacts student performance in course components differently, we gain insight into how assessments could be modified or curriculum could be adjusted so that performance in an introductory programming course correlates less to prior programming experience a student has. For example, courses could put more weight on programming assignments, instead of timed programming quizzes. Exams could focus more on conceptual questions that bridge prerequisite material with existing course content, instead of programming questions that depend on familiarity with language syntax.

Extent of prior programming experience

Since the one-way ANOVA test has shown that prior programming experience's impact on quizzes, the second midterm and the final is statistically significant, a Bonferroni post-hoc test is then carried out using IBM SPSS Statistics [9] to determine whether the performance difference is between any two groups or only some groups. Table 4 summarizes the results. Statistically significant difference in performance is observed between Group 4 and Group 1 in quizzes, the second midterm and the final. There is also a statistically significant difference in performance between Group 3 and Group 1 in quizzes only. However, there are no statistically significant difference in performance between all other groups. This result shows that having less than one year of prior programming experience puts students at a disadvantage compared to those with more years of experience. However, the difference in performance is statistically insignificant once students have at least one year of prior programming experience. In other words, once students have one year of prior programming experience, they are no longer at an disadvantage. This discovery provides insight into how to best prepare students with almost no prior programming experience to have an equal opportunity to perform as well as those with extensive programming experience. For example, curriculum designers could offer programming studio courses concurrently with the prerequisite course to those with no programming experience. Freshman advisors could suggest students to join student organizations or study online material that would expose them to programming prior to the start of the course.

	Groups	4-1	4-2	4-3	3-1	3-2	2-1
Prog. Asgmt.	FA17	1.000	1.000	1.000	1.000	1.000	0.534
	SP18	0.039	0.731	1.000	0.043	0.757	1.000
	SP19	1.000	1.000	1.000	1.000	1.000	1.000
	FA19	1.000	1.000	1.000	1.000	1.000	1.000
Quiz	FA17	0.002	0.412	1.000	0.008	1.000	0.237
	SP18	0.000	0.259	1.000	0.003	1.000	0.064
	SP19	0.002	0.394	1.000	0.022	1.000	0.448
	FA19	0.016	1.000	0.016	1.000	0.754	0.754
MT1	FA17	0.336	0.370	0.083	1.000	1.000	1.000
	SP18	0.005	1.000	0.027	1.000	0.428	0.158
	SP19	1.000	1.000	1.000	1.000	1.000	1.000
	FA19	0.839	1.000	1.000	1.000	1.000	1.000
MT2	FA17	0.003	0.096	1.000	0.047	0.935	1.000
	SP18	0.000	0.013	0.308	0.000	1.000	0.010
	SP19	0.066	0.940	1.000	1.000	1.000	1.000
	FA19	0.016	0.539	0.525	0.824	1.000	0.878
Final	FA17	0.000	0.039	0.064	0.031	1.000	0.020
	SP18	0.000	1.000	0.054	0.513	0.949	0.005
	SP19	0.038	0.090	1.000	0.480	0.866	1.000
	FA19	0.017	0.385	0.815	0.539	1.000	1.000

Table 4: Bonferroni post-hoc test results.

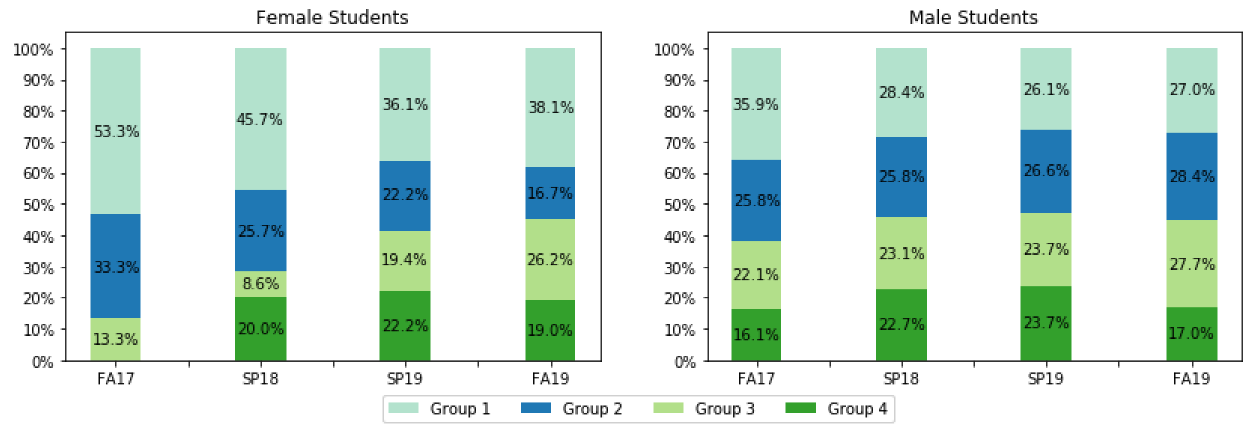


Figure 4: Group distribution by gender.

Access to pre-college Computer Science education
As indicated by literature [1], it is less likely for women than men to take programming courses in high school although their access to computer is equivalent. To check whether that is the case in our course, we compare how students are distributed among the four groups across semesters by gender. Figure 4 shows that for every semester, there is a lower percentage of female students than male students coming in with more than one year of prior programming experience. We

believe performance difference between male and female students, if any, can be attributed to the difference in prior programming experience at the start of the course. However, the situation is promising if we look at the trend over 2 years. In Fall 2017, only 46.7% of female students were coming in with at least one year of prior programming experience, compared to the 64.1% for male students. However, this number grew to 61.9% for female students in Fall 2019. Moreover, the number of female students who have two or more years of programming experience grew from 13.3% to 45.2%, while the distributions have stayed about the same for male students.

The situation is less optimistic for under-privileged students because their access to high school computer science courses is limited. As early as 1994, it has reported that ownership of a computer is the most significant factor on success in college level computer science [5]. Unlike today, owning a computer more than 25 years ago can be considered a symbol of high socioeconomic status. Recent studies have shown that socioeconomic status remains a problem for access to high school computer science resource. In South Carolina, more than 70% of computer science classes are offered in the three largest cities; 46.2% of public schools in the state receive federal Title I funding yet only 25.3% of public schools offer computer science education fall under such category, resulting in a 20.9% gap [11], [12]. While in California, only 24% of schools with the highest percentage of low income students offer any kind of computer science course while that number is 61% for those with the lowest percentage of low income students [2]. Although we do not have demographic information on students in our course, we hope that these insights could guide us in better preparing students with limited exposure to computer science courses in high school to have equal opportunities to succeed.

Conclusion

We surveyed four semesters of students in an introductory programming course in ECE about their prior programming experience and analyzed the responses and their performance in five different course components. We found that students with more years of programming experience generally perform better in certain course components. This difference is only statistically significant in quizzes, the second midterm and the final. It showed that students with limited prior programming experience are at a disadvantage when the assessment format is timed programming at a terminal, and when topics assessed are covered in high school Computer Science curriculum in which they had limited exposure to. However, prior programming experience has limited impact on student performance when the assessment is not timed and allows students to seek help, or when the topics assessed are only covered in a college level prerequisite course in which all students took around the same time.

Furthermore, we found that students with less than one year of prior programming experience are at a disadvantage in the course components mentioned above compared to those with more years of experience. However, the difference in performance among students with at least one year of experience is statistically insignificant. We conclude that by having at least one year of prior programming experience places students at a similar starting line in a college level introductory programming course compared to those with extensive programming experience.

The extent of prior programming experience of female students and male students is analyzed and it shows that, as indicated by literature, a lower percentage of women took computer science courses in high school. However, the percentage of female students coming in with at least one year of prior programming experience has grown tremendously over the course of two years. It is encouraging to see that a much higher percentage of female students are starting the course already having extensive programming experience.

References

[1] J. Margolis and A. Fisher. *Unlocking the Clubhouse: Women in Computing*. The MIT Press. MIT Press, 2002. ISBN 9780262632690. URL `https://books.google.com/books?id=StwGQw45YoEC`.

[2] Alexis Martin, Frieda McAlear, and Allison Scott. Path not found: Disparities in Access to Computer Science Courses in California High Schools. Technical report, Level Playing Field Institute, 2015.

[3] Roger E Franklin Jr. What academic impact are high school computing courses having on the entry-level college computer science curriculum? *ACM SIGCSE Bulletin*, 19(1):253–256, 1987.

[4] Zoe A Kersteen, Marcia C Linn, Michael Clancy, and Curtis Hardyck. Previous experience and the learning of computer programming: The computer helps those who help themselves. *Journal of Educational Computing Research*, 4(3):321–333, 1988.

[5] Harriet G Taylor and Luegina C Mounfield. Exploration of the relationship between prior computing experience and gender on success in college computer science. *Journal of educational computing research*, 11 (4):291–306, 1994.

[6] 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*, pages 41–46, 2003.

[7] 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 SIGCSE/SIGCUE ITiCSEconference on Innovation and technology in computer science education*, pages 25–28, 2000.

[8] Susan Bergin and Ronan Reilly. Programming: factors that influence success. In *Proceedings of the 36th SIGCSE technical symposium on Computer science education*, pages 411–415, 2005.

[9] IBM Corp. IBM SPSS statistics for macintosh. URL `https://www.ibm.com/products/spss-statistics`.

[10] College Board. AP computer science A course at a glance, effective fall 2019. 2019.

[11] Barbara Ericson, W Richards Adrion, Renee Fall, and Mark Guzdial. State-based progress towards computer science for all. *ACM Inroads*, 7(4):57–60, 2016.

[12] Quinn Burke, Madeleine Schep, and Travis Dalton. Cs for sc: A landscape report of k-12 computer science in south carolina. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*, pages 705–705, 2017.