EXAMINATION OF VARYING PAIR PROGRAMMING STRATEGIES IN COMPUTER SCIENCE LABORATORY ENVIRONMENTS

Lily Sisouvong
MASTER OF SCIENCE THESIS

OF

LILY SISOUVONG

APPROVED:

Thesis Committee:

Major Professor  Lisa DiPippo
Natallia Katenka
Victor Fay-Wolfe
Kathleen Peno
Brenton DeBoef

DEAN OF THE GRADUATE SCHOOL

UNIVERSITY OF RHODE ISLAND

2023
Pair programming is a technique within the computer science space in which two programmers are paired on one computer to solve a related programming task. This technique is often practiced in both the industry and the academic setting, as it has a consistent history of positively impacting working professionals and students regarding personal experience and performance in programming. As a result, there has been a large focus within the computer science education field on how and why pair programming is a successful technique. More specifically, what traits of pairing students together make pair programming successful. At the University of Rhode Island (URI) we employ this technique in different formats across various computer science courses.

This study focuses on the implementation of pair programming in one of those particular courses, an introductory programming course called "Survey of Computer Science" (CSC 110). For more than five years this course has solely utilized pair programming in the laboratory environment with one specific combination of pairing traits, or pairing strategy. For each laboratory session, students are paired with new random partners on the condition that they have not paired with the same partner more than once. The implementation of this pairing strategy has proven to be successful in improving student learning, however, we recognize that no other pairing strategies have been implemented in this course. Curiosity about whether other pairing strategies exist that perform better at improving student learning had sparked the motivation to pursue this study.

We designed and experimented with two new pairing strategies in addition to the one pairing strategy that has been used in CSC 110 for over five years; we can coin this strategy as the historical strategy. The two new pairing strategies possess a few of the traits from the historical strategy as well as new ones. First, pairing
strategy one pairs students together once at the beginning of the semester with similar skill levels and they keep those same partners until the end of the course. Second, pairing strategy two pairs students together on a weekly basis similar to the historical pairing, but instead of pairing random students together, students are paired using similar skill levels.

We collected data from 98 students to measure student performance in CSC 110 and data from 80 students to measure overall change in attitude towards computer science. This data contains student grades including final course grades, final exam grades, midterm exam grades, grades from weekly assignments and assessments, and attitude scores obtained through an attitude survey conducted in the beginning and the end of the course. To assist in our analysis of the data we collected, we employed the following statistical procedures: ANOVA test, Tukey HSD test (multiple comparisons), Kruskal-Wallis test, and the Wilcoxon test (non-parametric multiple comparisons). We included non-parametric statistical tests as a result of not meeting the data assumptions necessary to perform the ANOVA and Tukey HSD tests.

Using the statistical procedures we described we found significant differences between the impact of the pairing strategies when analyzing average final course grades, final exams, midterm exams, homework assignments, overall attitude scores, and specific attitude scores that measure one dimension of attitude survey. More specifically, we found that the historical pairing strategy used in CSC 110 performed better than pairing strategy two when comparing overall average attitude scores and average scores from one dimension of the attitude survey. In all other cases, they performed similarly. Likewise, the historical pairing also performed better than strategy one when comparing average final course grades, average homework grades, and again the average scores from the one dimension
of the attitude survey. Other than those comparisons, the historical strategy and strategy one perform similarly.

As a result of our analysis, we conclude in this study that the new pairing strategies do not perform better than the historical pairing strategy. Therefore, we recommend that the historical pairing strategy is used in future occurrences of the CSC 110 course until further experimentation on pairing strategies occurs.
ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my major professor, Dr. Lisa DiPippo, it was her curiosity and insight that ignited this study. It was also with her dedicated guidance and support, as well as the invaluable support from my co-major professor Dr. Natallia Katenka, that this study was able to meet its completion. Words cannot express my gratitude, many thanks to them both.

I would also like to extend my sincere thanks to my committee including Dr. Victor Fay-Wolfe, Dr. Kathleen Peno, and Dr. Yan Sun for their involvement and feedback. My gratitude also goes to the participants of my study for their willingness to share their insight and experiences.

I would be remiss in not mentioning my family, friends, and peers for their overwhelming encouragement in pursuing this study to the very end. Lastly, I would like to share my gratitude for my parents, Phaythong Rajsombat and Sithi Sisouvong, for their consistent support in all of my endeavours.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Literature Review</td>
<td>4</td>
</tr>
<tr>
<td>2.1</td>
<td>Education</td>
<td>4</td>
</tr>
<tr>
<td>2.2</td>
<td>Implications in Pair Programming</td>
<td>6</td>
</tr>
<tr>
<td>2.2.1</td>
<td>Pairing an Expert with an Expert</td>
<td>7</td>
</tr>
<tr>
<td>2.2.2</td>
<td>Pairing a Novice with a Novice</td>
<td>8</td>
</tr>
<tr>
<td>2.2.3</td>
<td>Pairing an Expert with a Novice</td>
<td>8</td>
</tr>
<tr>
<td>2.3</td>
<td>Compatibility</td>
<td>10</td>
</tr>
<tr>
<td>2.4</td>
<td>Attitudes Towards Computer Science</td>
<td>12</td>
</tr>
<tr>
<td>3</td>
<td>Methodology</td>
<td>15</td>
</tr>
<tr>
<td>3.1</td>
<td>Pairing Strategies Design</td>
<td>15</td>
</tr>
<tr>
<td>3.2</td>
<td>Quantifying Skills</td>
<td>17</td>
</tr>
<tr>
<td>3.2.1</td>
<td>Pairing Algorithm</td>
<td>17</td>
</tr>
<tr>
<td>3.2.2</td>
<td>Pairing Strategy One (S1)</td>
<td>18</td>
</tr>
</tbody>
</table>
3.2.3 Pairing Strategy Two (S2) ..................................... 18
3.2.4 Pairing Strategy Three (S3) ................................. 21
3.3 Study Objectives .............................................. 21
3.4 Evaluation .................................................. 21
  3.4.1 Measuring Performance ................................. 22
  3.4.2 Measuring Attitudes ................................. 22
  3.4.3 Main Hypotheses .................................. 25
3.5 Data Collection ........................................... 25
3.6 Implementation of the Experiment .......................... 26

4 Experimental Results and Analysis .......................... 28
  4.1 Summary of the Data .................................... 28
    4.1.1 Performance and Attitude Data ................... 29
  4.2 Data Limitations ....................................... 35
  4.3 Performance ............................................ 38
    4.3.1 Examining Demographic Data and Final Course Grades ... 41
    4.3.2 Final Exam Comparison ............................ 41
    4.3.3 Midterm Exam Comparison ......................... 43
    4.3.4 Weekly Assignment Comparison .................... 44
    4.3.5 Looking at the Introductory Quiz and the Final Course Grades ... 46
  4.4 Attitude ................................................. 47
    4.4.1 Examining Demographic Data and Attitude Scores .... 50
    4.4.2 Examining Survey Subscales ....................... 51
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Pairing Strategy Dimensions</td>
</tr>
<tr>
<td>2</td>
<td>The formula used to calculate PVs for S1, where students are paired once at the beginning of the semester for the entire CSC 110 course using performance. $Intro_{value}$ represents the actual score after completing the Introductory Quiz and $Q_{value}$ represents the averaged score from the first quiz.</td>
</tr>
<tr>
<td>3</td>
<td>The formula used to calculate PVs for S2, where students are paired with new partners each week using performance. $Intro_{weight}$ represents the weight for the Introductory Quiz. $Intro_{value}$ represents the actual score after completing the Introductory Quiz. $Q_{weight}$ and $Q_{value}$ represent the weight and averaged score for quiz assessments, and $H_{weight}$ and $H_{weight}$ represent the weight and averaged score for homework assignments, respectively.</td>
</tr>
<tr>
<td>4</td>
<td>Sample size of varying gender identities across all pairing strategies within the performance data</td>
</tr>
<tr>
<td>5</td>
<td>Sample size of varying racial and ethnic identities across all pairing strategies within the performance data</td>
</tr>
<tr>
<td>6</td>
<td>Sample size and percentages of populations in the performance data of intersectional groups, categorized by both gender and race/ethnicity</td>
</tr>
<tr>
<td>7</td>
<td>Sample size of varying gender identities across all pairing strategies within the attitude data</td>
</tr>
<tr>
<td>8</td>
<td>Sample size of varying racial and ethnic identities across all pairing strategies within the attitude data</td>
</tr>
<tr>
<td>9</td>
<td>Sample size and percentages of populations in the attitude data of intersectional groups, categorized by both gender and race/ethnicity</td>
</tr>
<tr>
<td>10</td>
<td>Box plot visualizing the spread of the final course grades for CSC 110 among the groups using the three pairing strategies</td>
</tr>
</tbody>
</table>
11 Confidence Intervals generated from performing the Tukey HSD test on the mean final course grades for CSC 110 for each group using one of the three pairing strategies. 40

12 Confidence Intervals generated from performing the Tukey HSD test on the mean final exam grades for CSC 110 for each group using one of the three pairing strategies. 43

13 Confidence Intervals generated from performing the Tukey HSD test on the mean of the student final homework grade for CSC 110 for each group using one of the three pairing strategies. 45

14 Sample of the attitude data excluding the subscale data. Values under the pre-test and post-test scores represent student attitudes at the beginning and the end of the semester. Changed, represents changed attitude, this is the difference between the post-test and pre-test. 48

15 Confidence Intervals generated from performing the Tukey HSD test on the average changed attitude score for each group using one of the three pairing strategies. 49
# LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Performance Value weights over the course of the Fall 2022 semester</td>
</tr>
<tr>
<td>2</td>
<td>Cronbach’s Alpha Scores for the pre-test and post-test (in blue) survey sections</td>
</tr>
<tr>
<td>3</td>
<td>Distribution of demographic data including gender and race/ethnicity broken down by frequency within each pairing strategy as found in the performance data</td>
</tr>
<tr>
<td>4</td>
<td>Sample sizes for all intersectional groups identified by gender and race/ethnicity within the performance data</td>
</tr>
<tr>
<td>5</td>
<td>Distribution of demographic data including gender and race/ethnicity broken down by frequency within each pairing strategy as found in the attitude data</td>
</tr>
<tr>
<td>6</td>
<td>Sample sizes for all intersectional groups identified by gender and race/ethnicity within the attitude data</td>
</tr>
<tr>
<td>7</td>
<td>Sample sizes for all pairing strategies within both the performance data and attitude data</td>
</tr>
<tr>
<td>9</td>
<td>Significant statistical tests performed on the performance data</td>
</tr>
<tr>
<td>10</td>
<td>Significant statistical tests performed on the attitude data</td>
</tr>
</tbody>
</table>
Believe it or not, "driving" and "navigating" are terms that do not only belong on the road. They also belong in the computer science realm, more specifically, in a programming technique called "pair programming".

Conventionally, pair programming is utilized by two programmers who work together on only one computer in an effort to complete programming tasks or challenges. One programmer acts as a "driver"; their primary job is to physically type on their shared computer. The other programmer acts as a "navigator"; they are tasked with aiding the driver in various ways.

To provide a clearer picture of what the navigator does, it may be helpful to compare this technique to driving a car. In this analogy, the "driver" steers the car, and the passenger or "navigator" helps the driver by providing directions, looking ahead for any obstacles such as detours, and assisting in other tasks that the driver may need help with. Together they communicate and work to reach their shared destination in different but helpful ways.

Let’s bring our attention back to pair programming; the "navigator" will help to map out a solution, find current and potential bugs (errors) in the code, write down the solution on paper, and much more. In addition to working on just one computer, another condition of pair programming is that both programmers switch positions within intervals of some specified time.

Pair programming is widely used, most frequently studied by researchers in both the technology industry and school environments, but more so the academic setting[1]. It has proven useful for educators in increasing student participation and retention [2]. Additionally, this technique has been demonstrated to improve
student performance in course assignments and assessments, and it aids students in producing higher quality software[3]. Given these results, it is clear that a focus on pair programming should be prioritized. Especially a focus on the factors that make or will make pair programming successful.

At the University of Rhode Island, the Survey of Computer Science (CSC 110) course has practiced pair programming in one particular way for many years with anecdotal success in improving student learning. This particular implementation requires that students pair with new random partners each week. However, despite this success, there has yet to be a focus on what makes this practice successful or if other variations of pair programming can produce even more success in student learning. That is where our primary motivation for this study originates.

This thesis aims to not only address our primary motivation but we hope to add to the current literature on strategic pair programming. Our main objective is to examine factors, or combinations of factors, used to pair students together to find factors that work and do not work toward positively impacting students. We are looking at the impacts on student academic performance (course-related grades) and their attitudes toward computer science.

Given the current literature on strategic pair programming, we expect two factors to best improve student learning. These factors are dynamic pair rotation and pairing by skill level. Dynamic pair rotation requires students to pair with new partners during each laboratory session. At the same time, pairing by skill level requires students to be partnered with peers who possess similar ”skills” as themselves. Various researchers have suggested that these factors should be implemented in the classroom, as evidence shows that they positively impact student experience and performance.

Additionally, in exploring the current literature, we found one particular fac-
tor that has not been frequently studied, and information on its effects on students is still preliminary. Interestingly, that factor is static pair rotation, in which students keep the same partners for an extended period; in an academic setting, this extended period is the entire semester in which the course is active.

To put these factors to the test, we created three pairing strategies that represent three different combinations of factors. The first pairing strategy partners students once in the beginning of the course using skill level, and they keep those partners for the rest of the academic semester. The second pairing strategy pairs students with new partners each week, also using skill level. Lastly, the third pairing strategy pairs students with new random partners each week. We hope that, through this study, we can provide feedback and suggestions for future implementations of strategic student pairings in CSC 110 and other computer science courses that employ pair programming.

The rest of this thesis is organized in the following way. We begin with Chapter 2, our literature review. This chapter showcases the intersection between education and pair programming, dives into the current research on factors that affect pair programming, and introduces attitude in the computer science education space. In Chapter 3, we explain our experimental design, the three pairing strategies, how we implemented the experiment, and how we evaluated the results. In Chapter 4, we discuss our findings, what statistical tests we used, and what we can take away from these findings. Lastly, in Chapter 5, we summarize our findings and consider future work.
CHAPTER 2

Literature Review

Pair programming is a popular programming style used by many programmers in the technology industry and the classroom space [4]. One of the earliest recordings of this programming style was proposed in 1953 by graduate students Fred Brooks and Bill Wright [5]. Pair programming has since come a long way and is now an important tool used by many programming instructors.

This style or learning technique is performed by two programmers who work on one laptop to derive some programming solution(s). One programmer takes on the position of the “driver,” and the other programmer becomes the “navigator.” The pair switch positions after specific intervals of time to ensure that they have an equal opportunity to work on the solution. Informally, the ”driver” is responsible for actively typing the ideas both programmers brainstorm. At the same time, the ”navigator” guides the driver through the solution through various means, such as identifying potential bugs.

In a review of research papers spanning from 1994 to 2004, Preston indicates that this technique is an effective learning strategy to aid students in practicing programming and problem-solving tasks [6]. Students, especially first-year and second-year students, benefit in various ways, from boosting their confidence in their programming ability to increasing the production of more mature and efficient code [7].

2.1 Education

The collaboration between two programmers in the pair programming technique can be further categorized using the following educational learning strategies: peer tutoring, cooperative learning, and collaborative learning [8]. These learning
strategies model peer interactions and are rooted in Vygotsky’s Socio-Cultural Theory of Cognitive Development. This theory asserts that people psychologically develop higher mental functions such as perception, thinking, attention, and memory through social interactions[9]. Vygotsky’s theory, furthermore, postulates that with the assistance of another individual, learners can master tasks or concepts that are too difficult for them to master individually through a concept known as the “Zone of Proximal Development” (ZPD).

Typically, in the peer tutoring strategy, learners will practice ZPD by working with another individual who is knowledgeable in the area in which the learner is having difficulty mastering. In a classroom setting, the knowledgeable individual can be another student in the same grade level [10]. What sets peer tutoring apart from cooperative and collaborative learning is that it is generally done in pairs, where one student is the ”tutor” and the other is a ”tutee” [11]. In cooperative and collaborative learning, students work in small groups to complete group assignments regardless of whether one or more learners have mastered the material needed to complete the task. However, there is always a possibility in which students happen to work with another peer within the group in a ZPD like manner.

When comparing cooperative and collaborative learning, each style is distinguishable by how smaller tasks are assigned to group members. In cooperative learning, students will often assign smaller tasks to different group members to complete individually; therefore, each student will be accountable for completing their “part” of the assignment. Once each student has completed the smaller task that they are responsible for, the students will then integrate all of their work to create one cohesive solution [12]. Conversely, in collaborative learning, students are more interactive when completing more minor tasks. These smaller tasks are
not typically assigned to just one member. Many members will work across different tasks in even smaller groups or in pairs; in fact, all members may work on smaller tasks together in a joint effort.

Students practicing pair programming may exhibit behaviors relative to any of the three learning strategies presented; peer tutoring, cooperative learning, and collaborative learning. However, Williams et al. in [13] state that the ambition for pair programming is most compatible with the collaborative learning strategy, and in turn, cooperative learning should be avoided. This definition aligns with our understanding of pair programming; the goal of this technique is to give students the opportunity to work together as often as they can to program or create a solution. Therefore, the study presented in this thesis was designed to be consistent with a collaborative learning framework.

Referencing Williams et al. once more, the authors caution that pairs that are not equally matched in programming skills may begin to model the peer tutoring learning strategy because one student is more knowledgeable than the other [13]. Although the idea of a more skilled student tutoring a peer who may lack in a similar proficiency appears to be productive, it may prompt negative feelings from either or both students; these ideas will be further discussed in next section.

2.2 Implications in Pair Programming

Within the current literature focused on pair programming, there is a lack of a consensus on what makes a pair compatible or incompatible. Incompatible pairs, or as what Wood et al. describe as "dysfunctional pairs," occur when two programmers are not able to effectively collaborate; ultimately defeating the purpose of pair programming [14]. Many literature reviews and experiments have boiled down this issue into factors or strategies that negatively impact pair collaboration [3, 15, 16, 17, 14, 18, 19].
Examples of factors that contribute to partner incompatibility, according to Robe et al., include any combination of the following: “differing skill levels”, “conflicting personalities”, “imbalanced power dynamics”, inconsistent role switching (where students do not get equal time as both the ”driver” and the ”navigator”), and “inclinations to work alone” [20]. Furthermore, a literature review written by Salleh et al. concludes that “skill levels” is one of the two most commonly investigated compatibility determinants; the other is personality [15]. Skill level is further categorized by Salleh et al. as “perceived skill” and ”actual skill”. Perceived skill can be described as the perception of one’s programming skills, whereas, actual skills are the true programming skills that one possesses [15].

Before discussing the role that actual skill levels play in pairs, three scenarios should first be considered. The first scenario is when a pair is made of both high-performing or higher experienced students (expert and expert). In the second scenario, low-performing or less experienced students (novice and novice) are paired together. Lastly, when one programmer is higher performing, and one programmer is lower performing (expert and novice).

2.2.1 Pairing an Expert with an Expert

In the first scenario, pairing an expert student with another expert student has shown not always to yield positive experiences. For instance, in Chaparro et al.’s interview with post-graduate students, they found that when this type of pair is given an assignment that they both do not deem as challenging, pair programming becomes less useful to them[17]. They may complete the assigned task with the mindset that it could have been finished individually [21]. This mindset may result in some of the following situations: they may not find swapping programming roles compelling, nor would they find substantially communicating critical if one student decides to complete the task on their own. Lui and Chan’s study on novice-novice
and expert-expert pairings, through one of their established principals, have found that in these instances, solo programming may be more productive than paired programming [22].

Additionally, students have voiced a preference for having a partner that has a similar skill level as them but possesses different skill sets than themselves; they are looking for a partner that complements them [23]. This may be a suitable arrangement for some pairs, but it should be acknowledged that disengagement is likely to occur from both programmers [21]. This happens when an assigned task is split into smaller sub-tasks, then divided amongst both programmers based on programming expertise; it follows a cooperative learning framework where students complete their “assigned” tasks. Programmers are then disengaged in the tasks not assigned to them.

2.2.2 Pairing a Novice with a Novice

In the second case of pairing a novice programmer with a novice programmer, Chaparro et al. observe that in cases where students are given an assignment that they perceive as challenging, a novice and novice pairing may not promote effective pair programming [23]. As presented by the theory of ZPD, learners may require additional scaffolding from a more knowledgeable and experienced partner (or teacher) to approach difficult tasks and concepts. Conversely, compared to a novice practicing solo programming, a novice, and novice pairing is presented as the advantageous pick. Lui and Chan have found this to be a “gain” in time and code quality among novice programmers [22].

2.2.3 Pairing an Expert with a Novice

Unlike the previous cases, pairing an expert with a novice programmer is central to applying ZPD. This type of pairing promotes more opportunities for
the novice learner to understand better and apply difficult programming concepts. Additionally, this "tutor-tutee" relationship can be a scaffolding method for the more experienced programmer to solidify their knowledge of the studied concept or topic. Despite these opportunities, it has been observed that novice programmers or "less" experienced partners in these pairings may face feelings of anxiety, low self-confidence, lack of motivation, and frustration[24, 8, 21]. This may result from feeling that their partner has done more work than they have[24]. Regardless, experiencing feelings like those described may lead students to withdraw from participating in pair programming.

Jehng’s findings in a 1997 study comparing face-to-face collaboration with text-based collaboration briefly touched on the issue of having a pair with unequal programming abilities[8]. Jehng observed that the dominant partner, the partner who has higher programming skills and/or is more "aggressive", was found to exhibit more individualistic behaviors that do not support a collaborative environment. To add to that, the "dominant" or expert programmer, may feel that they are being "slowed down" by their partner and consequently they may feel frustrated [25]. It becomes more complicated for the non-dominating partner if they are not changing roles consistently. When this occurs, the non-dominating partner will often hold the navigator role[18, 21]. As the constant navigator, they may no longer feel any contingencies to participate as the dominating partner takes over the task almost wholly.

Regardless of the conflicts that may arise in the various pairings based on skill level, most researchers have suggested that students should be paired using similar skill levels [14, 15, 3, 17, 24]. In fact, as mentioned in the first scenario, some researchers have further suggested that pairs should not have the exact same skill level, instead a small gap in skill is encouraged but not too large, or students
may run into expert-novice issues [17]. Contrasting these findings, there exist records from other researchers in which a correlation between programming skills, experience, and pair interactions does not exist. For example, Müller and Padberg’s preliminary findings show that pair performance has no relationship with programming experience [26].

Briefly touching on the second most commonly investigated factor in finding what makes pair programming successful, personality-based pairings, Salleh et al. establish that there remains some ambiguity around this factor [15]. At the North Carolina State University, Katira et al. found that students in earlier computer science courses are more compatible with students with different personalities; this study leveraged the Myers Briggs Personality Test[25]. In addition, another study conducted by Sfetsos et al. concluded with a similar remark, using the Myers Briggs Personality Test, which found that pair programming was more effective when pairing students with heterogeneous personality types[27].

Contrasting those findings, Hannay et. al, found the opposite results in their study[28] Using the Big Five personality traits among other pair programming related factors such as task complexity and skill level, they found that personality had no impact on professional programmers. Salleh et. al. highlights that most personality based pair programming studies produce no significant evidence to identify personality as a predictor of pair programming performance. Even so, they found that of the few experiments that did produce significant evidence linking personality to pair programming, most suggest that students should be paired with heterogeneous personalities [15, 27].

2.3 Compatibility

Faja’s literature review on pair programming in 2011 identifies partner rotation as one of the commonly investigated strategies researchers have implemented
in their studies [16]. Partner rotation strategies can be categorized into two types of rotations. One type is implemented when partners do not change throughout the semester; this paper coins this type of rotation as "static". The other type implements a strategy in which pairs rotate periodically; this will be stamped as "dynamic" rotations. Dynamic rotations may take various forms. Examples include pairs switching multiple times a day, daily, weekly, and sometimes pairs are swapped in more specific periods.

Watkins and Watkins’s experiment paired students together for two lab periods at a time [29]. Another dynamically rotating student implementation occurs at NCSU (North Carolina State University) [18]. Here, students work with new partners about three or four times per semester, as suggested in Williams et al.’s guidelines for implementing pair programming. They reason that students will become more social (meeting more peers), and in the event that they have problems with their partner they will only work in a conflicting environment for at most two weeks [18]. On the other hand, in a separate study run by Williams et al. at NCSU, it is pointed out that a disadvantage to periodically changing partners is that students who connected and found compatible partners will have to now adjust to a new partner who they may not be compatible with [30].

Implementation of dynamic partner change has been well-documented by Faja in a literature review on pair programming; however, only one paper is mentioned in regards to same partner rotations [16]. In that particular study, although the authors intended for students to work with the same partner for the duration of a quarter they were unable to follow this condition due to external reasons such as schedule changes [2]. Out of the papers reviewed for this study, most experiments conducted with same partner rotations focused more on student outcomes and experiences with pair programming and not from programming with the same
partner for an extended period. Furthermore, Faja notes that of the papers reviewed, none had compared the implementation of the same partners’ rotations with changing partner rotations during a semester[16]. Therefore, it is difficult to ascertain which type of partner rotation works best to improve code quality and peer interaction.

Given these varying situations, one can deduce that ultimately “compatibility” may present as a subjective term. Pair compatibility can vary depending on the assignment complexity, the personalities of the programmers, programming experience and expertise, and much more. This may explain why it is difficult for researchers to pinpoint the factors and strategies that make pair programming successful.

2.4 Attitudes Towards Computer Science

We integrate attitude towards computer science in this study as a result of being curious about the effects of pairing traits on attitudes and wanting to include some aspect of student experience. Palaigeorgiou et al. define computer science attitude as the “general evaluation or feeling of favor or antipathy toward computer technologies and specific computer-related activities” [31]. In other words, attitudes towards computer science can be perceived as the sentiment or outlook towards all computer facets (hardware, digital artifacts, activities, etc.).

It is important to discuss the impact of this type of attitude, especially from the perspective of educators in the computer science field. Facey-Shaw and Golding’s study on the effect of peer tutoring and attitude on the academic performance of introductory programming students reveals that personal confidence in learning programming, a metric to measure attitude, is the most significant contributor to the academic performance of the three metrics used to measure attitude [32]; the other metrics include teacher perception and usefulness of programming.
Many factors discourage students from studying, especially students from underrepresented populations such as women and people of color. From 2020 to 2021, the U.S. Department of Education, National Center for Education Statistics, and Integrated Postsecondary Education Data System observed that 97,047 students had Bachelor’s degrees in computer and information sciences. Of those students, about 21% identified as women. Additionally, from 2020 to 2021, of the 104,874 students who graduated with a Bachelor’s degree in computer and information sciences and support services, about 47% were identified as White students. Following that, about 8% were identified as Black students, about 11% were identified as Hispanic students, and less than 1% were identified as American Indian/Alaska Native.

Many researchers have attributed this disproportion of students to various factors. For example, Fisher and Margolis have identified factors that affect female participation in computer science. These factors include “Pre-College Computing Experience”, computing culture, and perception of what the field entails and who makes up students in computer science[33]. More importantly, they cite that confidence and interest, often used to measure attitude, are another factor in low female participation in computer science.

To measure computer science attitudes, various surveys have been developed and tested [34, 35, 36, 37, 31]. One survey in particular, “Computer Science Attitude Survey,” authored by Miller et al., has become one of the central methods for the study presented in this paper [38]. It was modeled after a mathematics-oriented survey created in 1976 by Fennema and Sherman [39]. Fennema and Sherman’s survey was designed to measure attitudes toward mathematics, focusing on gender. The Computer Science Attitude Survey was then modified and published by Miller et al. in 2003 to measure computer science attitudes.
This survey uses five subscale categories, including “Confidence in learning computer science and programming”, “Attitude towards success in computer science,” “Computer Science as a male domain,” “Usefulness of computer science and programming,” and “Effective motivation in computer science and programming.”
CHAPTER 3

Methodology

In pursuit of adding to the current literature on pairing programming, we devised an experiment that implemented three different pairing strategies in an effort to find which strategy (and factors) promotes a more positive attitude in computer science and improved academic success.

This experiment took place at the University of Rhode Island in an introductory computer science course called "Survey of Computer Science" (CSC110) in the fall of 2022. Generally in this class, students are required to partake in weekly laboratory sessions in which they are paired with another student to complete an assignment that involves programming and problem-solving. Historically students have been paired with new partners each week, randomly, with the condition that they do not pair with the same partner again for the rest of the semester. We apply this condition in our experiment as well, for all three strategies. Additionally, there are generally three laboratory sections each semester as there are typically around 120 students that enroll in this course, and lab sections are kept to around 30 or 40 students each. This fact led us to create the three pairing strategies, with each pairing strategy applied to a specific laboratory section.

3.1 Pairing Strategies Design

The pairing strategies comprise two dimensions as shown in Figure 2. The first dimension determines whether or not students are paired using performance data (or paired randomly). The second dimension determines pair rotation, if pairs keep the same partner all semester (static) or if the pairs change partners each week (dynamic). Based on this structure we have implemented the following three pairing strategies. The first strategy (S1) paired students with similar skill levels together
and they kept the same partners for the entire semester (performance/static). The second strategy (S2) also paired students together using similar skill levels but paired them with new partners each week (performance/dynamic). The last strategy (S3) mimics the historical assignment, where pairs will be formed at random and they will have a new partner each week (random/dynamic). We chose not to implement a fourth strategy as shown in Figure 2 (random/static) due to our limited sample size. Additionally, because most of the current literature on pair programming suggests that students and educators would find greater success in pairing students together using skill level with a dynamic rotation, we assume that the static and random assignment may not be a successful implementation [14, 15, 18].

![Figure 1. Pairing Strategy Dimensions](image)

Figure 1. Pairing Strategy Dimensions
3.2 Quantifying Skills

To pair students who are assigned the S1 and the S2 strategies, in which pairs are required to have similar skills, we needed to be able to measure student skills. To do that we have developed a formula to generate a quantitative value called “Performance Value” (PV); the value for PV ranges anywhere from zero to one hundred. PV is different for both S1 and S2, because S1 generates pairs once at the beginning of semester, using the PVs only once. Meanwhile, S2 generates pairs weekly, creating new PVs each week as well.

PVs for S1 uses grades from an “Introductory Quiz” (assessment) and grades from the first in course quiz. The “Introductory Quiz” is required to be completed by all students in the first laboratory of the semester. The purpose of this assessment is to evaluate students on their programming knowledge and skills before engaging in the CSC110 course. PVs for S2 use homework grades, quiz grades, grades from the “Introductory Quiz”, and grades from the course midterm exam. After students complete their midterm exams, we no longer calculate PVs for S2 using the “Introductory Quiz” instead we use the midterm exam grades. In S2, the midterm exam grades are incorporated into the process that creates PVs.

When pairs were first created for the S1 and the S2 strategies, we discerned that the first homework assignment is the least complex of all the assignments the students were given. The course professor has indicated that students in the past generally do well on this assignment. As a result, we have decided to omit the homework grade for the first generation of the PV for both strategies.

3.2.1 Pairing Algorithm

After computing the PV for all students, we then ranked the students from highest performing to lowest performing in the laboratory sections that implemented the S1 and S2 strategies. Once we had determined these rankings, we then


\[ PV_1 = (\text{Intro}_{0.8weight} \times \text{Intro}_{value}) + (Q_{0.2weight} \times Q_{value}) \]

Figure 2. The formula used to calculate PVs for S1, where students are paired once at the beginning of the semester for the entire CSC 110 course using performance. \( \text{Intro}_{value} \) represents the actual score after completing the Introductory Quiz and \( Q_{value} \) represents the averaged score from the first quiz.

traversed down the ordered list and consecutively assigned six students into one pairing group. We assumed that students who were in the same pairing group had similar skills. Once these groups were formed, we shuffled the students together and paired two random students.

### 3.2.2 Pairing Strategy One (S1)

We created pairs using the S1 strategy after their first laboratory session; at this time students are required to complete their Introductory Quiz assessment. Essentially, pairs under the S1 strategy are created only once at the beginning of the semester after the first laboratory session. To create these pairs, we calculated the PVs for all (S1) students using the equation in Figure 2. In this equation, the "Introductory Quiz" weight was set to (80%) and the quiz weight for the first CSC 110 quiz was set to (20%). We chose these weights because we felt that the "Introductory Quiz" represented students’ programming skill and knowledge more extensively than the first CSC 110 quiz. We then ranked the students using their PV values, created pairing groups, shuffled them within those groups, and paired two students. Those pairs then continued working together until the end of the semester on laboratory assignments.

### 3.2.3 Pairing Strategy Two (S2)

Under the S2 strategy, new pairing groups were computed each week using the formula in Figure 3. This formula generates new PVs for all (S2) students every
Figure 3. The formula used to calculate PVs for S2, where students are paired with new partners each week using performance. $Intro_{weight}$ represents the weight for the Introductory Quiz. $Intro_{value}$ represents the actual score after completing the Introductory Quiz. $Q_{weight}$ and $Q_{value}$ represent the weight and averaged score for quiz assessments, and $H_{weight}$ and $H_{value}$ represent the weight and averaged score for homework assignments, respectively.

week using weights that changed over time as we observed student performances each week. These weights are listed in Table 1. As the weeks progress from week two to week seven, the Introductory Quiz weight moves from the largest to the smallest value. At the seventh week mark, students completed their midterm exam. Therefore, we removed the Introductory Quiz weight from the PV formula and introduced the Midterm Exam weight in its place. From week seven to week eleven, the midterm exam weight also decreased over time and was assigned at a higher weight than the Introductory Quiz weight.

<table>
<thead>
<tr>
<th>Week</th>
<th>Intro</th>
<th>ME</th>
<th>Q</th>
<th>HW</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.8</td>
<td>–</td>
<td>0.2</td>
<td>–</td>
</tr>
<tr>
<td>3</td>
<td>0.6</td>
<td>–</td>
<td>0.25</td>
<td>0.15</td>
</tr>
<tr>
<td>4</td>
<td>0.4</td>
<td>–</td>
<td>0.35</td>
<td>0.25</td>
</tr>
<tr>
<td>5</td>
<td>0.3</td>
<td>–</td>
<td>0.4</td>
<td>0.3</td>
</tr>
<tr>
<td>6</td>
<td>0.2</td>
<td>–</td>
<td>0.45</td>
<td>0.35</td>
</tr>
<tr>
<td>7</td>
<td>–</td>
<td>0.4</td>
<td>0.25</td>
<td>0.35</td>
</tr>
<tr>
<td>8</td>
<td>–</td>
<td>0.45</td>
<td>0.2</td>
<td>0.35</td>
</tr>
<tr>
<td>9</td>
<td>–</td>
<td>0.45</td>
<td>0.2</td>
<td>0.35</td>
</tr>
<tr>
<td>10</td>
<td>–</td>
<td>0.35</td>
<td>0.25</td>
<td>0.4</td>
</tr>
<tr>
<td>11</td>
<td>–</td>
<td>0.35</td>
<td>0.25</td>
<td>0.4</td>
</tr>
</tbody>
</table>

Table 1. Performance Value weights over the course of the Fall 2022 semester
In Table 1, the weights for week two to week seven comprise the Introductory Quiz, course quizzes, and homework assignments. The Introductory Quiz is weighted at higher weights at the beginning of the semester and declines over time. This is because the Introductory Quiz is a measurement of knowledge and programming skills before taking CSC 110. Therefore, as time passes, we assume that students will become more knowledgeable in programming concepts/topics and will develop more/sharpen their programming skills. Moreover, we believe that a more accurate measurement of these metrics (knowledge and skills) comes from the homework assignments and course quizzes that are assigned on a week-to-week basis; this is why these performance weight variables "Q" and "HW" are assigned at higher rates as the semester progresses. We treat the Introductory Quiz weight and these weights (quiz and homework) as a trade-off. Additionally, week one is excluded from the table, as we do not begin pair programming activities until week two and because students complete the Introductory Quiz at this time (week one).

The weights consist of the midterm exam, course quizzes, and homework assignments from week seven to week eleven. One interesting thing to note here is the repetition of weights. For instance, week nine uses the same weights as week eight. The reason for this contrast, in comparison with the steady weight changes in the first half of the table, are as follows.

Unfortunately, in the second half of the fall semester, we encountered pair assignment issues that we did not anticipate. This problem stems from our sample size; by the end of the semester, 98 students were enrolled in CSC 110. We found that performance groups remained relatively static; some groups consisted of the same students for over a week. We had no choice but to duplicate pairs because of this. Additionally, the script created to make the pairs did not account for static
groups, leading to many technical issues. As a result, most of the effort towards the end of the semester was focused on fixing these problems, and in turn, the weights were accidentally repeated twice.

3.2.4 Pairing Strategy Three (S3)

The S3 strategy follows the same pairing procedure that has been replicated over the last five years. Similar to the S2 strategy, students were grouped with new partners each week with the condition that students were not paired with the same partner more than once. However, unlike the S1 and S2 strategies, we did not calculate performance values at all. Instead, students will be paired regardless of their skills.

3.3 Study Objectives

With the aim of this research in mind, we formed the following list of study objectives.

- Determine if there is an effect from any or all of the pairing strategies on student course scores in assignments and assessments
- Determine if there is an effect from any or all of the pairing strategies on student attitudes towards computer science
- Compare the difference in score outcomes and attitudes across the pairing strategies if a difference or many differences exist

3.4 Evaluation

To measure the effectiveness of the various pairing strategies we employed, we used two types of metrics - quantitative and qualitative. The quantitative metrics we analyzed include final course grades, final exam grades, midterm exam grades, homework assignment grades, and quiz grades. The laboratory assignments
have been analyzed in addition to these metrics; however, it is not included as a primary metric. This is because most students earn full points (100%) on these assignments. Explanations for this event include the fact that students are placed in a laboratory environment where multiple teaching assistants are available to help, and these assignments were designed to be finished in the allotted time; this time is one hour and forty-five minutes.

3.4.1 Measuring Performance

Our quantitative metrics include graded course components such as the weekly homework assignments and quizzes. These metrics are already measured upon grading. Once the professor and teaching assistants grade the metrics, there are no additional weights or modifications to the metrics. The range for this metric is typically 0 to 100, but some components/assignments allow students to receive bonus points. To access this information a consent form was distributed to students, which permitted us to download their grades from a platform called "BrightSpace". The University of Rhode Island (URI) uses this platform to support learning for most classes.

The qualitative metrics we analyzed are the responses from the surveys we conducted twice in the semester. This includes determining whether or not student attitudes toward computer science have changed and if that change is positive or negative. As stated, one of our objectives for this study is to discover if any of the three strategies impact student attitudes. The following subsection ("Measuring Attitudes") provides more details on the qualitative metrics for this study.

3.4.2 Measuring Attitudes

The method we used to measure student attitudes toward computer science is through a survey authored by Miller et al. described in the literature review
under the section (Attitude Towards Computer Science) [38]. This survey acts as a “Pre-test” and a “Post-test.” The difference in survey scores reflects a change in attitude towards computer science.

Both the pre-test and the post-test surveys have been modified from the original survey state by omitting one of the five subscales that measure attitude towards computer programming and computer science as a whole. This is the third subscale of “Computer Science as a male domain”. The remaining subscales are confidence in learning computer science and programming (confidence), attitude toward success in computer science (attitude), usefulness of computer science and programming (usefulness), and effective motivation in computer science and programming (effective). These subscales are further described as follows:

- The confidence subscale measures respondents’ ability to learn and to perform well on computational tasks
- The attitude subscale measures respondents’ anticipation of consequences (positive and negative) as a result of success in computer science
- The usefulness subscale measures respondents’ beliefs about the usefulness of computer science in the present and the future in relation to education, work, and other instances of life in which computer science is relevant
- The effective subscale measures respondents’ motivation as a result of their desire to be effective, or effectance when engaging in computer science

One of our more specific objectives was to determine if any (or all) of the three strategies results in more positive attitudes toward computer science. This specific subscale will not provide any insight to meet this goal. In addition to that, we decided to omit the first question. This asks participants if they will “…plan to major in computer science” because the CSC 110 class used to experiment in is
Table 2. Cronbach’s Alpha Scores for the pre-test and post-test (in blue) survey sections

<table>
<thead>
<tr>
<th>Survey subscale</th>
<th>Cronbach’s Alpha</th>
</tr>
</thead>
<tbody>
<tr>
<td>Confidence in Learning Computer Science and Programming</td>
<td>0.91</td>
</tr>
<tr>
<td>Attitude Toward Success in Computer Science</td>
<td>0.82</td>
</tr>
<tr>
<td>The usefulness of Computer Science and Programming</td>
<td>0.91</td>
</tr>
<tr>
<td>Effective Motivation in Computer Science and Programming</td>
<td>0.88</td>
</tr>
<tr>
<td>Confidence in Learning Computer Science and Programming</td>
<td>0.96</td>
</tr>
<tr>
<td>Attitude Toward Success in Computer Science</td>
<td>0.85</td>
</tr>
<tr>
<td>Usefulness of Computer Science and Programming</td>
<td>0.91</td>
</tr>
<tr>
<td>Effective Motivation in Computer Science and Programming</td>
<td>0.89</td>
</tr>
</tbody>
</table>

taken by students who are already majoring in computer science. The pre-test was further modified by adding additional questions separate from the survey sections. This was done to gather demographic data from survey participants. We included two questions related to gender identification and ethnicity/race identification.

After modifying the survey, we recalculated the Cronbach’s Alpha for each survey subscale to ensure that each section or subscale measures a specific characteristic of ”attitude” in a reliable manner. We derive the following results in Table 2. These results show that we could move forward and use the surveys. In addition to that, a codebook was developed to label and provide in-depth information about each question from both surveys. The codebook, Figure A, can be found in the first chapter of the Appendix titled ”Codebook”.

Moreover, the original design of this study included a psychological pairing condition using the Felder and Silverman’s Index of Learning Styles [40]. We hoped to categorize students as active, neutral, or reflective learners. We used survey questions from the Felder and Silverman’s Index of Learning Styles Questionnaire to do that. In order to strictly determine if students were active, neutral, or
reflective learners, we isolated eleven questions [41]. These eleven questions were added to the pre-test. Unfortunately, after collecting this data, we concluded that we should not pair students using learning styles due to the limitation in the sample size for this study. Unlike the pre-test, the post-test did not include demographic questions and the learning style questions.

3.4.3 Main Hypotheses

Based on the list of study objectives, we developed the following two primary hypotheses:

**Hypothesis 1 (H1):** The mean of the final grades for CSC 110 for students using the three pairing strategies will not be equal.

**Hypothesis 2 (H2):** The average difference between the pre-test and post-test attitude survey scores completed by students using the three pairing strategies will not be equal.

When we initially created these hypotheses, we had the following in mind. Upon testing the hypotheses, if we find any significant differences between groups within our hypotheses, we will continue to explore our data and create more hypotheses. Otherwise, we will look at our data to find other significant findings.

3.5 Data Collection

Due to the nature of this study (human participants), a human subject research certificate certified by "CITI Program" has been attained by both the principal and student investigator. This certificate certifies that both investigators have the proper training and background in researching with human participants and, accompanied by the required IRB approval, certifies that data will be collected in a valid and reliable procedure. Physical consent forms that were approved by an IRB
representative were given to students to volunteer their data prior to completing the pre-survey.

The consent forms emphasize that identifying data such as name and student ID will be kept confidential. To meet this expectation, we replaced personal data such as student names with unique IDs that were generated in the following anonymizing algorithm.

The data was organized into a data frame, including student names and IDs. The data frame was then stripped of the student ID, as it was not necessary for the analysis portion of this experiment. The data frame was then shuffled to keep students anonymous. Student names were then replaced with an identifier that begins with the word "student" and ends in a unique number between one and the total amount of students. It is important to note that not all students completed both the pre-test and post-test surveys; out of 98 students, only 80 students completed both surveys; therefore, only 80 surveys were eligible to be analyzed. A look-up table was created to ensure that, if necessary, student identification would be possible for the study investigators. These tasks were completed using the R Studio software.

3.6 Implementation of the Experiment

We began this experiment by emailing the experiment to students enrolled in CSC 110. Additionally, the student investigator visited the first laboratory to introduce the experiment and answer any questions students had. A physical consent form was passed out to all students during this visit; if they chose to participate, they would complete it. Otherwise, they left it blank. Only two students in the section using strategy one opted out of participating. The student investigator was able to pass out and collect the consent forms as they did not partake in grading any assignments in the course.
The three laboratory sections for CSC 110 were then assigned a pairing strategy. The first laboratory (section 1) was given the S1 pairing strategy, the second laboratory (section 2) was assigned the S2 strategy, and the third laboratory (section 3) was assigned the S3 strategy.
CHAPTER 4
Experimental Results and Analysis

Our experiment is set out to measure the effect of the various pairing strategies on student academic performance and attitude toward computer science. As a result, we have decided to organize this chapter into three sections. Our first section, "Summary of the Data", introduces the data we collected and analyzed and the limitations of this study. The second section addresses our findings on student academic performance, and the third section discusses our findings on student attitudes toward computer science. We begin each section by addressing our two primary hypotheses. As a reminder, those are: (H1) The mean of the final grades for CSC 110 for students using the three pairing strategies will not be equal, and (H2) The average difference between the pre-test and post-test attitude survey scores completed by students using the three pairing strategies will not be equal.

4.1 Summary of the Data

In the Fall of 2022, we collected performance and attitude data from students in CSC 110. We title data relating to performance, "performance data". These values correspond to course grades, such as final exams, midterm exams, introductory quizzes, weekly quizzes, daily in-class activities, and weekly laboratory assignments. We use the term, "attitude data" to refer to the data outputted from the "Computer Science Attitude" survey that students completed at the beginning and end of the semester.

As discussed in Chapter 3, Section (3.4.2) "Measuring Attitudes", the Computer Science Attitude survey was authored by William et al. and published in 2003[38]. To better assist our investigation into strategic pair strategies, the modi-
fication of the survey in this study limited the survey to four out of its five original subscales. These subscales shown in Table 2, are made of twelve questions each and represent different dimensions of "attitude" towards computer science. Each question is answered using a Likert Scale, where individuals can correspond to each prompt with values ranging from one through five; one represents that a respondent "Strongly disagrees" with the prompt, and five represents respondents "Strongly agrees." Therefore, a response can score as high as 240 and as low as 48 on the overall survey. These scores derived from each subscale are then summed together to make up one numerical score to measure their attitude toward computer science when they partake in the survey. To measure their change in attitude after practicing their assigned pairing strategy, we conducted both pre-test and post-test surveys. We subtracted their pre-test scores from each respondent’s post-test scores.

By the end of the Fall 2022 semester, 100 students were enrolled in the course. Of those 100 students, 98 were eligible and interested in participating in this study. In the following subsection and our "Data Limitations" section, we highlight the distribution of gender and race/ethnicity demographic data within the performance and attitude data and why there are two types (performance and attitude) of data instead of one.

4.1.1 Performance and Attitude Data

In the following Figures (4, 5, 7, 8) and Tables (3 and 5), we visualize the distribution of gender identities and race/ethnicity identities across all pairing strategies. Two identities within these figures are substantially higher than others.

First, as illustrated in Figure 4 and Figure 7, male students represent a large number within the course and study population. Of the 98 students participating in this study, male participants make up about 70% of the participant population.
in comparison to other gender identities such as female (9%) and non-binary (1%) participants.

The second identity to highlight relates to race/ethnicity distributions. Figures 5 and 8 depict significant differences in the distribution size of Caucasian students compared to the other race/ethnicity identities across all three pairing strategies. Caucasian students make up about 61% of the participant population in this study. These results are also out of the 98 students. Again, in comparison to this, Asian students represent about 14% of the population, Black/African students make up about 11%, Middle Eastern students about 4%, and lastly, mixed race/ethnicity students about 1%.

Figures 6 and 9 display the distributions of all intersectional groups including both gender and racial/ethnicity within both the performance and attitude data. From these figures, we identify that a significant makeup of both of our data are made of male Caucasian students. They represent close to half of our study population, while other intersectional groups such as male Middle Eastern and female Caucasian make up less than a quarter of the study population.
### Gender

<table>
<thead>
<tr>
<th></th>
<th>Strategy 1</th>
<th>Strategy 2</th>
<th>Strategy 3</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Female</td>
<td>6</td>
<td>3</td>
<td>3</td>
<td>12</td>
</tr>
<tr>
<td>Male</td>
<td>29</td>
<td>31</td>
<td>24</td>
<td>84</td>
</tr>
<tr>
<td>Non-binary</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Prefer not to say</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

### Race/Ethnicity

<table>
<thead>
<tr>
<th></th>
<th>Strategy 1</th>
<th>Strategy 2</th>
<th>Strategy 3</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asian</td>
<td>7</td>
<td>4</td>
<td>3</td>
<td>14</td>
</tr>
<tr>
<td>Black/African</td>
<td>6</td>
<td>4</td>
<td>1</td>
<td>11</td>
</tr>
<tr>
<td>Caucasian</td>
<td>18</td>
<td>23</td>
<td>19</td>
<td>60</td>
</tr>
<tr>
<td>Hispanic/Latinx</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>Middle Eastern</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>Mixed</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Prefer not to say</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 3. Distribution of demographic data including gender and race/ethnicity broken down by frequency within each pairing strategy as found in the performance data

![Gender Distribution within the Performance Data](image)

Figure 4. Sample size of varying gender identities across all pairing strategies within the performance data
Figure 5. Sample size of varying racial and ethnic identities across all pairing strategies within the performance data.

Figure 6. Sample size and percentages of populations in the performance data of intersectional groups, categorized by both gender and race/ethnicity.
<table>
<thead>
<tr>
<th>Intersectional groups</th>
<th>Sample Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Female Asian</td>
<td>1</td>
</tr>
<tr>
<td>Female Black/African</td>
<td>2</td>
</tr>
<tr>
<td>Female Caucasian</td>
<td>8</td>
</tr>
<tr>
<td>Female Hispanic/LatinX</td>
<td>1</td>
</tr>
<tr>
<td>Male Asian</td>
<td>13</td>
</tr>
<tr>
<td>Male Black/African</td>
<td>9</td>
</tr>
<tr>
<td>Male Caucasian</td>
<td>50</td>
</tr>
<tr>
<td>Male Hispanic/LatinX</td>
<td>5</td>
</tr>
<tr>
<td>Male Middle Eastern</td>
<td>4</td>
</tr>
<tr>
<td>Male Not Specified</td>
<td>2</td>
</tr>
<tr>
<td>Male Mixed</td>
<td>1</td>
</tr>
<tr>
<td>Prefer not to say Caucasian</td>
<td>1</td>
</tr>
<tr>
<td>Non-binary Caucasian</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 4. Sample sizes for all intersectional groups identified by gender and race/ethnicity within the performance data

<table>
<thead>
<tr>
<th></th>
<th>Strategy 1</th>
<th>Strategy 2</th>
<th>Strategy 3</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gender</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Female</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>9</td>
</tr>
<tr>
<td>Male</td>
<td>25</td>
<td>27</td>
<td>17</td>
<td>69</td>
</tr>
<tr>
<td>Non-binary</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Prefer not to say</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Race/Ethnicity</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Asian</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>11</td>
</tr>
<tr>
<td>Black/African</td>
<td>2</td>
<td>5</td>
<td>2</td>
<td>9</td>
</tr>
<tr>
<td>Caucasian</td>
<td>20</td>
<td>16</td>
<td>13</td>
<td>49</td>
</tr>
<tr>
<td>Hispanic/Latinx</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>Middle Eastern</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>Mixed</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Prefer not to say</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 5. Distribution of demographic data including gender and race/ethnicity broken down by frequency within each pairing strategy as found in the attitude data
Figure 7. Sample size of varying gender identities across all pairing strategies within the attitude data.

Figure 8. Sample size of varying racial and ethnic identities across all pairing strategies within the attitude data.
Figure 9. Sample size and percentages of populations in the attitude data of intersectional groups, categorized by both gender and race/ethnicity

<table>
<thead>
<tr>
<th>Intersectional groups</th>
<th>Sample Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Female Black/African</td>
<td>2</td>
</tr>
<tr>
<td>Female Caucasian</td>
<td>6</td>
</tr>
<tr>
<td>Female Hispanic/LatinX</td>
<td>1</td>
</tr>
<tr>
<td>Male Asian</td>
<td>11</td>
</tr>
<tr>
<td>Male Black/African</td>
<td>7</td>
</tr>
<tr>
<td>Male Caucasian</td>
<td>41</td>
</tr>
<tr>
<td>Male Hispanic/LatinX</td>
<td>4</td>
</tr>
<tr>
<td>Male Middle Eastern</td>
<td>4</td>
</tr>
<tr>
<td>Male Mixed</td>
<td>1</td>
</tr>
<tr>
<td>Male Prefer not to say</td>
<td>1</td>
</tr>
<tr>
<td>Prefer not to say Caucasian</td>
<td>1</td>
</tr>
<tr>
<td>Non-binary Caucasian</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 6. Sample sizes for all intersectional groups identified by gender and race/ethnicity within the attitude data

4.2 Data Limitations

We begin by addressing that we organized two separate data sets for performance data and one for attitude data. When we initially explored the values
from the attitude surveys, we found that not all students completed both surveys, one at the beginning and one at the end of the semester. Therefore, unlike our performance data, we do not have a pair of survey results for all 98 students. In fact, at the beginning of the semester, 97 students completed the pre-test survey, and by the end of the semester, 89 students completed the post-test survey. Of those completed surveys, only 80 survey results were eligible from both the pre-test and post-test surveys to be analyzed. Consequently, our attitude data had a sample size of 80, comprised of the 80 paired survey results. As a result, there were 18 missing survey responses. Surveys that are not eligible for analysis include surveys completed by students who are not eligible to participate in the survey, students who enrolled late into the course, preventing them from completing the introductory quiz and the pre-test survey, and surveys with non-response bias in which students completed the pre-test survey but not the post-test survey for other reasons not listed here. In contrast to this, we have access to student grades from CSC 110 for all 98 students, making the sample size for our performance data 98.

We would also like to highlight the role that missing data played in this study. We recognize that not all students completed all homework assignments, laboratory assignments, quizzes, and in-class activities. Consequently, if students missed any of those assignments and assessments, they were given a score of zero. Furthermore, we found extreme outliers when exploring the performance and attitude data used in the analysis. In the performance data, all extreme outliers were below the average spread of the data. Box plots of all performance data indicates there existed individual assignment and assessment scores that were much lower than the average distribution of scores.
Figure 10. Box plot visualizing the spread of the final course grades for CSC 110 among the groups using the three pairing strategies.

For example, a box plot of the final course grades for the CSC 110 course and the pairing strategies in Figure 10 show outliers in all three groups (S1, S2, and S3). The outliers for S1 have a final course grades below 40, whereas the outliers for S2 and S2 have final course grades below 60 and 40 respectively.

This could have occurred due to students not performing well, especially in performance metrics such as final and midterm exams, amongst many other reasons. Additionally, the extreme outliers in the attitude survey scores were found to be both higher and lower than the average spread of attitude scores. This could be attributed to students consistently marking down extreme Likert scale answers, either all one’s or all five’s.

In addition to the limitations to our data collection, there are limitations to
the statistical methods we employed in the analysis portion of this study. The primary statistical method we use is the Analysis of Variance (ANOVA). This statistical method or test uses mean values to find significant differences between three or more populations. To properly perform an ANOVA test, the sample data used in this test has to align with the following assumptions: (1) all groups in the sample are independent, (2) the examined samples are random, (3) the sample is either large (size 30 or more according to CLT) or the sample comes from normally distributed populations, and (4) equal variance is present [42]. When analyzing our data, we found that not all data conditions for the ANOVA test were satisfied. As a result, we performed an alternative non-parametric statistical test called the "Kruskal Wallis" test, which is comparable to the ANOVA test. Generally, non-parametric tests allow individuals to analyze data without meeting data distribution assumptions. On that note, after reflecting on the sample sizes of the pairing strategies in both the performance and attitude data, as shown in Table 5, we have decided to set our alpha level to 10%. Alpha level, or significance level, is used in statistical testing to represent the probability of rejecting some null hypothesis when it is actually true. In other words, how likely will an individual incorrectly reject their null hypothesis? In our analysis, we set this probability to 10%.

<table>
<thead>
<tr>
<th></th>
<th>Strategy 1</th>
<th>Strategy 2</th>
<th>Strategy 3</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance Data</td>
<td>35</td>
<td>35</td>
<td>28</td>
<td>98</td>
</tr>
<tr>
<td>Attitude Data</td>
<td>29</td>
<td>29</td>
<td>22</td>
<td>80</td>
</tr>
</tbody>
</table>

Table 7. Sample sizes for all pairing strategies within both the performance data and attitude data

### 4.3 Performance

We start this section by discussing our findings on the first hypothesis. Here, we compared the impact of the three pairing strategies on the final course grades.
Table 9. Significant statistical tests performed on the performance data for the CSC 110 course. We employed the ANOVA test to test the differences between the three strategies. Our test revealed that there was a significant difference in mean final course grades between at least two sections using their corresponding strategies ($p = 0.0216$). As a result, this led us to perform a multiple comparison test, Tukey’s HSD test, to find which exact groups are significantly different. Tukey’s HSD test found that the mean final course grade was significantly different between S2 and S1 ($p = 0.0570$, $\text{diff} = 8.704$, C.I. = [0.929, 16.478]) and S3 and S1 ($p = 0.0361$, $\text{diff} = 9.969$, C.I. = [1.723, 18.215]). We reject our null hypothesis and determine that at a 10% significance level, that a significant difference exists between the impacts of the pairing strategies on the final course grade. Moreover, the results from Figure 11 suggest that the true average difference in points regarding the final course grade could be between 1.723 and 18.215 for students using S3 instead of S1. Additionally, the actual average difference in points between students using S2 and S1 could be between 0.929 and 16.478. Possibly, if the true average differences in final grades for CSC 110 were at the higher end
of the confidence interval, that can indicate that students using S3 and S2 scored close to two letter grades higher than students using S1.

Figure 11. Confidence Intervals generated from performing the Tukey HSD test on the mean final course grades for CSC 110 for each group using one of the three pairing strategies.

Generally, students who were paired dynamically (new partners each week) using random assignment (S3) and students who were paired dynamically using similar performance skills (S2) performed better than students who were paired statically (same partners for the entire semester) using similar performance skills at the beginning of the semester (S1). Pairing students using similar skill levels, measured at the beginning of the course, and having those students continue to
4.3.1 Examining Demographic Data and Final Course Grades

As discussed, our demographic data is not large enough to make inferences about populations that do not identify as "Caucasian" and as "Male." We recognize that it is not likely that any relationship will exist between demographic data, pairing strategy assignment, and final course grade due to the differences in sample sizes. Regardless, we continued to check for this relationship about our primary hypotheses. To do this, we leverage a two-way ANOVA test instead of a one-way. The difference between these two tests is that the two-way test allows individuals to analyze the effects of two categorical variables on one numerical variable instead of just one categorical variable on one numerical variable. The results from this test indicate that no significant interactions exist between gender (Female, Male, Non-binary, and Prefer not to say) and the pairing strategies (S1, S2, and S3) on final course grades (p = 0.4970). Similarly, there is no evidence to show that there is a significant interaction between race/ethnicity (Asian, Black/African, Caucasian, Hispanic/Latinx, Middle Eastern, Mixed, and Not Specified) and the pairing strategies on final course grades (p = 0.9980). As anticipated, no relationship exists within our data. As a result, we assume that other relationships between demographic data, pairing strategies, and other performance metrics will not be significant.

4.3.2 Final Exam Comparison

A comparison of the effects of the pairing strategies on the final exam grades, again using the ANOVA statistical test with a 10% significance level, reveals that there is a difference between at least two pairing strategies (p = 0.0947). Con-
sequently, we performed Tukey’s HSD test and found contradictory results. The p-values, which represent whether a significant difference exists in the average final grade exams between any two pairing strategies were found to be at or above 0.1152. These p-values and the confidence intervals in Figure 12 suggest that no significant differences exist between any of the pairing strategies with regard to the average final exam scores. As a result of the value zero appearing in the confidence intervals in Figure 12, there is a possibly that a true difference can be zero. Further indicating no significant differences. It is possible that this contradiction occurred because the p-value from the ANOVA test is very close to the significance level we set; 10%.
4.3.3 Midterm Exam Comparison

When preparing the midterm exam data for the ANOVA test, we found that our data violates one of the ANOVA assumptions. That is the assumption of equal variance, we checked for this using Levene’s test. As a result we performed both the ANOVA test and the non-parametric test called Kruskal-Wallis test.

At a 10% significance level, when comparing the influence of the pairing strategies on the midterm exam grades using the Kruskal-Wallis test, we find that a significant difference exists between at least two of the sections ($p = 0.0487$). As
a result, we performed a non-parametric multiple comparisons test to determine which sections are significantly different. This test is called the Wilcoxon test. We included a p-value adjustment method called Bonferonni when running the Wilcoxon test and the results from performing this test indicate that there is a significant difference between S2 and S1 (p = 0.6800). We then performed similar testing but with the ANOVA (p = 0.0135) and Tukey Test (p = 0.0175, diff = 12.164, C.I. = [1.772, 22.556]). Those findings support similar results as the non-parametric tests, however Tukey’s test indicates that S3 and S1 are significantly different in addition to S2 and S1. Due to the violation of the assumption of equal variance, it is more likely that only S2 and S1 are significantly different. We comprehend these results as the following, students who were paired dynamically and randomly performed better on the midterm exam than students who were paired statically and using similar performance values. This serves as additional evidence to support that S1 does not perform better than S2 and S3.

4.3.4 Weekly Assignment Comparison

The last performance metrics that we compare across the pairing strategies are the assignments and assessments that students complete weekly. This include the homework assignments, laboratory assignments, and quizzes. The one-way ANOVA test was applied to all of the mentioned metrics, and of the three tests only one presented significant results. This test compares the influence of the pairing strategies on homework assignments. Where our homework assignment data represents students’ final homework grade, this is an average of all CSC 110 homework assignments assigned to students in the fall 2022 semester. Our results from the ANOVA test (p = 0.0678) led us to complete the Tukey test (p = 0.0679, diff = 11.383, C.I. = [0.878, 21.889]), this revealed that there exists a significant difference in S3 and S1 in relation to the weekly homework assignments. We find in
Figure 13 that the true average point difference between the pairing strategies with regard to homework assignments could be between 0.878 and 21.889. Potentially, the true average difference could present as a letter grade difference. On average, students using S3 perform higher than students using S1 on final homework grades; this is the overall homework grade after averaging all of the homework assigned.

![Graph showing confidence intervals](image)

Figure 13. Confidence Intervals generated from performing the Tukey HSD test on the mean of the student final homework grade for CSC 110 for each group using one of the three pairing strategies.

The other metrics investigated including weekly quizzes, in-class activities, and laboratory assignments, show no significant differences between the pairing strategies.
4.3.5 Looking at the Introductory Quiz and the Final Course Grades

In addition to looking at each performance metric, we also examined the relationship between the Introductory Quiz, Final Course Grades, and the pairing strategies. We had expected to see evidence to indicate that some type of relationship existed, such as the following: Students using a specific pairing strategy who scored low on the introductory quiz, wound up scoring higher on in their course grade.

To aid our examination we utilized the multiple linear regression statistical test. This test is used to evaluate the effect of two or more variables on some variable of interest. Here we are looking to see if there is an effect from the Introductory Quiz and all of the pairing strategies on the final course grades. A summary of model indicates there is no significant evidence to indicate that any relationship exists between the final course grades for CSC 110, the pairing strategies, and the Introductory Quiz. The only significant p-values from the summary are for the following variables: (intercept) S1 (p-value: 1.61e-15), S3 (p-value: 0.0602), and the Introductory Quiz (p-value: 0.953e-09). There is a much stronger relationship between the Introductory Quiz and the final course grades.
4.4 Attitude

<table>
<thead>
<tr>
<th>Attitude Metrics</th>
<th>Statistical Tests</th>
<th>P-values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comparing the average changed attitude between all three pairing strategies</td>
<td>ANOVA</td>
<td>0.0526</td>
</tr>
<tr>
<td></td>
<td>Tukey HSD</td>
<td>S3 - S2: 0.0427</td>
</tr>
<tr>
<td>Comparing the average score derived from the &quot;Attitude Towards Success in Computer Science&quot; subscale between all three pairing strategies</td>
<td>ANOVA</td>
<td>0.0337</td>
</tr>
<tr>
<td></td>
<td>Tukey HSD</td>
<td>S3-S1: 0.0692</td>
</tr>
<tr>
<td></td>
<td></td>
<td>S3-S2: 0.0431</td>
</tr>
</tbody>
</table>

Table 10. Significant statistical tests performed on the attitude data

Here we address our second hypothesis where we set out to find if there were any significant differences between the impact of the pairing strategies on student attitudes towards computer science. In Section 4.1, we briefly described the subscales within the Computer Science Attitude survey and what scores are expected as an output of the survey. The score range for this survey is at least 48 and at most 240. We use the scores from both the pre-test and post-test to measure students’ change in attitude towards computer science from the beginning of the semester to the end of the semester. We subtract the pre-test scores from the post-test scores for each student, and we use this difference to make inferences about student attitudes towards computer science; we refer to the difference as ”changed attitude” as shown in Figure 14. Scores such as 20 and -23 represent an increase and decrease in positive attitudes towards computer science, respectively. These are the values we test in this study.
Figure 14. Sample of the attitude data excluding the subscale data. Values under the pre-test and post-test scores represent student attitudes at the beginning and the end of the semester. Changed, represents changed attitude, this is the difference between the post-test and pre-test.

An ANOVA test, at a 10% significance level, reveals that a significant difference between the changed attitude for at least two strategies exists ($p = 0.0526$). Tukey’s HSD test then reveals that the mean changed attitude significantly differed between S3 and S2 ($p = 0.0427$, diff = 12.148, C.I. = [1.837, 22.460]). We reject our null hypothesis and determine that at a 10% significance level, a significant difference exists between the impacts of the pairing strategies on student attitudes toward computer science. Moreover, we highlight that our true average difference in changed attitude scores, as shown in Figure 15, could be between 1.837 and 22.460. We can further interpret these results within the classroom space. Students who utilize S3, on average, score as having more positive attitudes toward computer science than students who employ the S2 strategy.

<table>
<thead>
<tr>
<th>ID</th>
<th>Section</th>
<th>Gender</th>
<th>Ethnicity</th>
<th>Pre-test</th>
<th>Post-test</th>
<th>Changed</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>student11</td>
<td>Male</td>
<td>Middle Eastern</td>
<td>214</td>
<td>191</td>
<td>-23</td>
</tr>
<tr>
<td>2</td>
<td>student37</td>
<td>Female</td>
<td>Caucasian</td>
<td>207</td>
<td>210</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>student68</td>
<td>Female</td>
<td>Black/African</td>
<td>177</td>
<td>197</td>
<td>20</td>
</tr>
<tr>
<td>4</td>
<td>student06</td>
<td>Male</td>
<td>Black/African</td>
<td>145</td>
<td>146</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>student55</td>
<td>Male</td>
<td>Caucasian</td>
<td>227</td>
<td>238</td>
<td>11</td>
</tr>
<tr>
<td>6</td>
<td>student40</td>
<td>Prefer not to say</td>
<td>Caucasian</td>
<td>211</td>
<td>226</td>
<td>15</td>
</tr>
<tr>
<td>7</td>
<td>student07</td>
<td>Male</td>
<td>Middle Eastern</td>
<td>208</td>
<td>185</td>
<td>-23</td>
</tr>
</tbody>
</table>
Figure 15. Confidence Intervals generated from performing the Tukey HSD test on the average changed attitude score for each group using one of the three pairing strategies.

Unlike the results that compare the impact of the pairing strategies on student final course grades, there is no significant evidence in this test related to S1, S1 students on average are just as positive as students in S2 and S3. Additionally, we find that although S2 and S3 perform similarly on final course grade outcomes, S3 students have a more positive outlook on computer science. The main takeaway is that we advise computer science educators to implement S3 over S2 and S1. First, evidence shows that students in S3 perform just as well as students in S2 and better than students in S1 in reference to their final course grades. Moreover,
S3 students acquired attitudes toward computer science that are, on average, just as positive as the students using S1 and more positive than students using S2.

4.4.1 Examining Demographic Data and Attitude Scores

Similar to our test examining the interaction between demographic data, the pairing strategies, and final course grades, we cannot make inferences about student groups, pairing strategies, and attitudes toward computer science. The reasoning stands that there still exists a substantial difference in size across student demographic groups. Again, we continue our test regardless.

We will use the two-way ANOVA test again to examine the role of demographic data in the pairing strategies and the attitude scores. We complete two tests in this section, one that incorporates gender into the ANOVA test and one that incorporates race and ethnicity into the test. The results from exploring the effects of gender (Female, Male, Non-binary, and Prefer not to say) and pairing strategies on attitude scores show that there is no significant interaction between gender and pairing strategies ($p = 0.6922$). Moving forward, when examining the effects of Race/Ethnicity (Asian, Black/African, Caucasian, Hispanic/Latinx, Middle Eastern, Mixed, and Prefer not to say) and pairing strategies on attitude scores, we are met with the same results. No significant interactions exist between race/ethnicity and pairing strategies ($p = 0.7982$).

Given these results, we find that within our attitude data, pairing strategies and demographic data (gender and race/ethnicity) are not impactful on attitude scores. However, our sample size on student demographic groups that do not identify as male and Caucasian do not truly reflect their true populations. Therefore this interpretation should not be examined as a general inference but specific to our attitude data.
4.4.2 Examining Survey Subscales

In Chapter 3, we provided short descriptions for each of the four subscales within the "Attitude Towards Computer Science" survey. These subscales include confidence in learning computer science and programming, attitude toward success in computer science, usefulness of computer science and programming, and effective motivation in computer science and programming. Each of these subscales represents a dimension of overall attitudes toward computer science, alone they can more about student ability, motivation, beliefs, and expectations toward computer science. Hence, we examined the subscale scores within the "Attitude Towards Computer Science" survey. The primary aim in exploring these subscales is to find whether there is or is not a difference in the impact of the three pairing strategies on all subscales.

To investigate this, we began by employing the MANOVA statistical test. This is another extension of the ANOVA test. The exception with this test is that instead of measuring the impact of the strategies on one variable such as attitude scores, it is done with multiple variables such as the four subscale values. Unfortunately, the data we used in this test violates the assumption of multivariate normality. As a result, we moved to examine each of the subscales individually.

Interestingly, we only found significant results for the "attitude toward success in computer science" subscale. In this examination, we use the ANOVA test and Tukey’s HSD test to determine if a difference exists between the effects of the pairing strategies on the subscales. At a 10% significance level, a significant difference exists between at least two strategies ($p = 0.0337$). Additionally, we find that those strategies are (S3) and (S1) ($p = 0.0692$, diff = 3.857, C.I. = [0.285, 7.429]), as well as the strategies (S3) and (S2) ($p = 0.0431$, diff = 4.202, C.I. = [0.630, 7.773]). As described in Chapter 3, the "attitude towards success in computer
science” subscale measures students’ expectations of consequences as a result of succeeding in computer science. Our results from the Tukey test show that there is a difference between S3 and S1 as well as S3 and S2. Additionally, the true average score differences between students who use S3 instead of S1 are among the following range -0.239 and 7.954. For students who use S3 instead of S2, that range is 0.104 and 8.299. In other words, students who have new random partners each week, when succeeding in computer science are more positively reinforced than students using S2 and S1; they expect more positive outcomes from succeeding.
CHAPTER 5

Conclusion

The original motivation behind this study stemmed from curiosity about whether other pairing strategies exist aside from CSC 110’s historical pairing strategy that leads to better student experience and performance. After creating the pairing strategies, our initial intuitive guess was that students paired dynamically using similar performance skills would prove to be the best pairing strategy. To investigate this, we created two main hypotheses that address the average final course grades and the average changed attitudes toward computer science and compared them across our three pairing strategies. Our intuition and literature review made us believe that pairing strategy 2 would increase student academic performance in CSC 110 and changed attitude scores higher than the other two pairing strategies. Our results demonstrate that this is not the case.

We found that in both hypotheses, there are significant results to indicate differences between the pairing strategies. In relation to the hypotheses, students paired with new random partners each week (S3) performed just as well as other pairing strategies and, in some cases, better. For instance, when comparing the final course grades, S3, on average, outperformed students who were paired statically using similar performance skills while performing similarly to students who were paired with new partners each week using similar performance skills. In addition, S3 performed better on average than S2 when comparing attitude scores while performing similarly to S1. Neither of the other pairing strategies performed better than the historical pairing strategy (S3).

Perhaps these results emerged as a result of our experiment design. Students using S1 were paired together for the duration of the semester using preliminary
grades (the introductory quiz and the first-in-class quiz), maybe other data prior
to the course should have been used instead. Additionally, the weights for the
performance values could have potentially affected our results. Other variations
of weights might have led to results that aligned with our guess regarding the
"best" pairing strategy. Moreover, we could have chosen measurements that did
not measure the true impact of these pairing strategies. For instance, we recognize
that using metrics such as final course grades might not showcase the true influence
of these strategies as many factors outside of this study could have attributed to
students’ grades.

Additionally, to shed light on the pairing strategies from an instructor’s point
of view, although S3 and S2 perform similarly, pairing students randomly is much
quicker to prepare on a week-to-week basis. While conducting the study, our
experience with preparing pairs for strategy 2 every week was more strenuous.
Creating pairs for S2 had taken more time, leading to more pairing issues compared
to the two other pairing strategies. Each week we needed to collect, clean, merge,
and process new data to enter into the performance value equation for strategy 2
to group students into similar skill groups. Moreover, to clarify the pairing issues,
one example is not having enough students within the classroom to ensure that we
would not have to pair students up with the same partner more than once due to
the limited class size.

We suggest that for CSC 110 that the historical assignment (S3) should be
implemented in future iterations of the course until further experimentation with
the pairing strategies takes place. Additionally, if S3 is not used in the future and
perhaps a new pairing strategy is tested, we advise that static pairing rotations
should be avoided. In most of the tests run in this analysis, S1 performed worse (fi-
nal course grades, midterm exam grades, homework grades, and "Attitude Towards
Success in Computer Science” subscale score) than one or both remaining pairing strategies. Moreover, if future class sizes for CSC 110 stay at 100 or less and if the instructor plans to ensure that students do not work with the same partner more than once, we suggest that they avoid pairing students together by programming skill level or trying new methods to calculate programming skill level. Doing this will help avoid the pairing issues we faced and possibly provide interesting results that give more clarity to the ”best” pairing strategy. Although we cannot make general specifications for all computer science courses, these results might provide insight into how other educators can use pairing strategies in their classrooms.

5.1 Future Work

When conducting the experiment in this study, we collected weekly survey results with numerical and anecdotal data attesting to students’ experience with the pair programming and the pairing strategies they were assigned. We also collected data related to student learning styles with the hope of implementing learning styles into the pairing strategy designs, but we chose not to include that in this study due to our sample size. Upon additional reflection on our work, we consider that maybe the metrics we used to measure the influence of the pairing strategies were not appropriate and effective. Choosing metrics more closely related to the laboratory sessions, such as the weekly surveys, may provide a better measurement of the true impact of the pairing strategies on student learning.

On the other hand, another possible alternative to gaining more accurate insight on this impact may be to complete more in depth analyses on the metrics we are currently using. For example, it may be advisable to investigate if a relationship between student academic performance in CSC 110, their attitudes, and the pairing strategies exists. In addition to that, it may also be interesting to look into the effects of the pairing strategies on students who start the course with high In-
introducary Quiz results and low results. Moreover, although a closer examination of the weekly assignments and assessments (quizzes, homework, and laboratory assignments) was out of the scope of this analysis, it may be beneficial to take a closer look at this data using more complex statistical tests such as the Mixed Effect Model.

On that note, we also reflected on what would have been our fourth pairing strategy, had our sample size been larger. That strategy requires static pair rotations with random assignments of students. Our study shows that pairing strategy 3 (S3) impacts student academic performance as well or better than the other two strategies. However, we cannot pinpoint whether these results came from students rotating partners frequently, students working with random partners each week, or if it is indeed a combination of those pairing factors. Experimenting with a fourth strategy or other pairing strategy variations may help answer that question.

Lastly, we recognize that this study targets student performance and attitudes, but not necessarily their learning experiences. An anecdote from one student who has previously used pairing strategy three, in which students are paired with new random partners each week, sheds light on how this pairing strategy positively affected their learning experience. They shared that as a student who commuted to campus, they looked forward to participating in laboratory sessions such as the one in CSC 110 because it gave them an opportunity to socialize with their peers. Using pairing strategy three in these sessions allowed them to meet new people, make friends, and work with students of varying programming skills/knowledge. Possibly, what we share in this study is not limited to just our results and interpretations, but also a suggestion to investigate student personal experience.
LIST OF REFERENCES


## APPENDIX

### Codebook

<table>
<thead>
<tr>
<th>Variable</th>
<th>Survey/Question Subscale</th>
<th>Variable Description</th>
<th>Value Label</th>
<th>Variable Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>N/A</td>
<td>Unique identifier for each participant</td>
<td>String</td>
<td>Nominal</td>
</tr>
<tr>
<td>Lab</td>
<td>N/A</td>
<td>Lab section/pairing strategy (lab section is associated with a particular pairing strategy)</td>
<td>Range: 1-3 (1 - S1, 2 - S2, 3 - S3)</td>
<td>Nominal</td>
</tr>
<tr>
<td>Gender</td>
<td>N/A</td>
<td>To which gender identity do you most identify with?</td>
<td>1 = Female, 2 = Male, 3 = Prefer not to say, 4 = Other... (Transgender, Non-binary)</td>
<td>Nominal</td>
</tr>
<tr>
<td>Ethnicity</td>
<td>N/A</td>
<td>To which ethnicity do you most identify with?</td>
<td>1 = Asian, 2 = Black/African, 3 = Caucasian, 4 = Hispanic/LatinX, 5 = Native American, 6 = Pacific Islander, 7 = Other... (Bosnian, mixed, middle eastern...)</td>
<td>Nominal</td>
</tr>
<tr>
<td>L1</td>
<td>Learning Style: <strong>Active/Reflective</strong></td>
<td>I understand something better after I</td>
<td>1 = try it out, 2 = think it through.</td>
<td>Nominal</td>
</tr>
<tr>
<td>L2</td>
<td>Learning Style: <strong>Active/Reflective</strong></td>
<td>When I am learning something new, it helps me to</td>
<td>1 = talk about it, 2 = think about it.</td>
<td>Nominal</td>
</tr>
<tr>
<td>L3</td>
<td>Learning Style: <strong>Active/Reflective</strong></td>
<td>In a study group working on difficult material, I am more likely to</td>
<td>1 = jump in and contribute ideas, 2 = sit back and listen.</td>
<td>Nominal</td>
</tr>
<tr>
<td>L4</td>
<td>Learning Style: <strong>Active/Reflective</strong></td>
<td>In classes I have taken</td>
<td>1 = I have usually got to know many of the students, 2 = I have rarely got to know many of the students.</td>
<td>Nominal</td>
</tr>
<tr>
<td>L5</td>
<td>Learning Style: <strong>Active/Reflective</strong></td>
<td>When I start a homework problem, I am more likely to</td>
<td>1 = start working on the solution immediately, 2 = try to fully understand the problem first.</td>
<td>Nominal</td>
</tr>
<tr>
<td>L6</td>
<td>Learning Style: <strong>Active/Reflective</strong></td>
<td>I prefer to study</td>
<td>1 = in a group, 2 = alone.</td>
<td>Nominal</td>
</tr>
<tr>
<td>L7</td>
<td>Learning Style: <strong>Active/Reflective</strong></td>
<td>I would rather first</td>
<td>1 = try things out, 2 = think about how I'm going to do it.</td>
<td>Nominal</td>
</tr>
<tr>
<td>L8</td>
<td>Learning Style: <strong>Active/Reflective</strong></td>
<td>I more easily remember</td>
<td>1 = something I have done, 2 = something I have thought about a lot about.</td>
<td>Nominal</td>
</tr>
<tr>
<td>Question</td>
<td>Description</td>
<td>Response Options</td>
<td>Scale</td>
<td></td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
<td>-----------------</td>
<td>-------</td>
<td></td>
</tr>
<tr>
<td>L9</td>
<td>Learning Style: <strong>Active/Reflective</strong>&lt;br&gt;When I have to work on a group project, I first want to</td>
<td>1 = have a &quot;group brainstorming&quot; where everyone contributes ideas. 2 = brainstorm individually and then come together as a group to compare ideas.</td>
<td>Nominal</td>
<td></td>
</tr>
<tr>
<td>L10</td>
<td>Learning Style: <strong>Active/Reflective</strong>&lt;br&gt;I am more likely to be considered</td>
<td>1 = outgoing. 2 = reserved.</td>
<td>Nominal</td>
<td></td>
</tr>
<tr>
<td>L11</td>
<td>Learning Style: <strong>Active/Reflective</strong>&lt;br&gt;The idea of doing homework in groups, with one grade for the entire group.</td>
<td>1 = appeals to me. 2 = does not appeal to me</td>
<td>Nominal</td>
<td></td>
</tr>
<tr>
<td>Q1</td>
<td>Computer Science Attitude Survey: <strong>Confidence in learning computer science and programming</strong>&lt;br&gt;Generally I have felt secure about attempting computer programming problems.</td>
<td>1 = strongly disagree 2 = disagree 3 = neutral 4 = agree 5 = strongly agree</td>
<td>Ordinal</td>
<td></td>
</tr>
<tr>
<td>Q2</td>
<td>Computer Science Attitude Survey: <strong>Confidence in learning computer science and programming</strong>&lt;br&gt;I am sure I could do advanced work in computer science.</td>
<td>1 = strongly disagree 2 = disagree 3 = neutral 4 = agree 5 = strongly agree</td>
<td>Ordinal</td>
<td></td>
</tr>
<tr>
<td>Q3</td>
<td>Computer Science Attitude Survey: <strong>Confidence in learning computer science and programming</strong>&lt;br&gt;I am sure that I can learn programming.</td>
<td>1 = strongly disagree 2 = disagree 3 = neutral 4 = agree 5 = strongly agree</td>
<td>Ordinal</td>
<td></td>
</tr>
<tr>
<td>Q4</td>
<td>Computer Science Attitude Survey: <strong>Confidence in learning computer science and programming</strong>&lt;br&gt;I think I could handle more difficult programming problems.</td>
<td>1 = strongly disagree 2 = disagree 3 = neutral 4 = agree 5 = strongly agree</td>
<td>Ordinal</td>
<td></td>
</tr>
<tr>
<td>Q5</td>
<td>Computer Science Attitude Survey: <strong>Confidence in learning computer science and programming</strong>&lt;br&gt;I can get good grades in computer science.</td>
<td>1 = strongly disagree 2 = disagree 3 = neutral 4 = agree 5 = strongly agree</td>
<td>Ordinal</td>
<td></td>
</tr>
<tr>
<td>Q6</td>
<td>Computer Science Attitude Survey: <strong>Confidence in learning computer science and programming</strong>&lt;br&gt;I have a lot of self-confidence when it comes to programming.</td>
<td>1 = strongly disagree 2 = disagree 3 = neutral 4 = agree 5 = strongly agree</td>
<td>Ordinal</td>
<td></td>
</tr>
<tr>
<td>Q7</td>
<td>Computer Science Attitude Survey: <strong>Confidence in learning computer science and programming</strong>&lt;br&gt;I'm no good at programming.</td>
<td>1 = strongly agree 2 = agree 3 = neutral 4 = disagree 5 = strongly disagree</td>
<td>Ordinal</td>
<td></td>
</tr>
</tbody>
</table>
| Q8   | Computer Science Attitude Survey: **Confidence in learning computer science and programming** | I don't think I could do advanced computer science.  
1 = strongly agree  
2 = agree  
3 = neutral  
4 = disagree  
5 = strongly disagree | Ordinal |
| Q9   | Computer Science Attitude Survey: **Confidence in learning computer science and programming** | I'm not the type to do well in computer programming.  
1 = strongly agree  
2 = agree  
3 = neutral  
4 = disagree  
5 = strongly disagree | Ordinal |
| Q10  | Computer Science Attitude Survey: **Confidence in learning computer science and programming** | For some reason even though I work hard at it, programming seems unusually hard for me.  
1 = strongly agree  
2 = agree  
3 = neutral  
4 = disagree  
5 = strongly disagree | Ordinal |
| Q11  | Computer Science Attitude Survey: **Confidence in learning computer science and programming** | Most subjects I can handle O.K., but I have a knack for flubbing up programming problems.  
1 = strongly agree  
2 = agree  
3 = neutral  
4 = disagree  
5 = strongly disagree | Ordinal |
| Q12  | Computer Science Attitude Survey: **Confidence in learning computer science and programming** | Computer science has been my worst subject.  
1 = strongly agree  
2 = agree  
3 = neutral  
4 = disagree  
5 = strongly disagree | Ordinal |
| Q13  | Computer Science Attitude Survey: **Attitude toward success in computer science** | It would make me happy to be recognized as an excellent student in computer science.  
1 = strongly disagree  
2 = disagree  
3 = neutral  
4 = agree  
5 = strongly agree | Ordinal |
| Q14  | Computer Science Attitude Survey: **Attitude toward success in computer science** | I'd be proud to be the outstanding student in computer science.  
1 = strongly disagree  
2 = disagree  
3 = neutral  
4 = agree  
5 = strongly agree | Ordinal |
| Q15  | Computer Science Attitude Survey: **Attitude toward success in computer science** | I'd be happy to get top grades in computer science.  
1 = strongly disagree  
2 = disagree  
3 = neutral  
4 = agree  
5 = strongly agree | Ordinal |
| Q16  | Computer Science Attitude Survey: **Attitude toward success in computer science** | It would be really great to win a prize in computer science.  
1 = strongly disagree  
2 = disagree  
3 = neutral  
4 = agree  
5 = strongly agree | Ordinal |
<p>| Q17 | Computer Science Attitude Survey: Attitude toward success in computer science | Being first in a programming competition would make me pleased. | 1 = strongly disagree 2 = disagree 3 = neutral 4 = agree 5 = strongly agree | Ordinal |
| Q18 | Computer Science Attitude Survey: Attitude toward success in computer science | Being regarded as smart in computer science would be a great thing. | 1 = strongly disagree 2 = disagree 3 = neutral 4 = agree 5 = strongly agree | Ordinal |
| Q19 | Computer Science Attitude Survey: Attitude toward success in computer science | Winning a prize in computer science would make me feel unpleasantly conspicuous. | 1 = strongly agree 2 = agree 3 = neutral 4 = disagree 5 = strongly disagree | Ordinal |
| Q20 | Computer Science Attitude Survey: Attitude toward success in computer science | People would think I was some kind of a nerd if I got A's in computer science. | 1 = strongly agree 2 = agree 3 = neutral 4 = disagree 5 = strongly disagree | Ordinal |
| Q21 | Computer Science Attitude Survey: Attitude toward success in computer science | If I had good grades in computer science, I would try to hide it. | 1 = strongly agree 2 = agree 3 = neutral 4 = disagree 5 = strongly disagree | Ordinal |
| Q22 | Computer Science Attitude Survey: Attitude toward success in computer science | If I got the highest grade in computer science I'd prefer no one knew. | 1 = strongly agree 2 = agree 3 = neutral 4 = disagree 5 = strongly disagree | Ordinal |
| Q23 | Computer Science Attitude Survey: Attitude toward success in computer science | It would make people like me less if I were a really good computer science student. | 1 = strongly agree 2 = agree 3 = neutral 4 = disagree 5 = strongly disagree | Ordinal |
| Q24 | Computer Science Attitude Survey: Attitude toward success in computer science | I don't like people to think I'm smart in computer science. | 1 = strongly disagree 2 = disagree 3 = neutral 4 = agree 5 = strongly agree | Ordinal |
| Q25 | Computer Science Attitude Survey: Usefulness of computer science and programming | I'll need programming for my future work. | 1 = strongly disagree 2 = disagree 3 = neutral 4 = agree 5 = strongly agree | Ordinal |</p>
<table>
<thead>
<tr>
<th>Question</th>
<th>Description</th>
<th>Likert Scale</th>
<th>Scale Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q26</td>
<td>Computer Science Attitude Survey: <em>Usefulness of computer science and programming</em></td>
<td>I study programming because I know how useful it is.</td>
<td>1 = strongly disagree 2 = disagree 3 = neutral 4 = agree 5 = strongly agree</td>
</tr>
<tr>
<td>Q27</td>
<td>Computer Science Attitude Survey: <em>Usefulness of computer science and programming</em></td>
<td>Knowing programming will help me earn a living.</td>
<td>1 = strongly disagree 2 = disagree 3 = neutral 4 = agree 5 = strongly agree</td>
</tr>
<tr>
<td>Q28</td>
<td>Computer Science Attitude Survey: <em>Usefulness of computer science and programming</em></td>
<td>Computer science is a worthwhile and necessary subject.</td>
<td>1 = strongly disagree 2 = disagree 3 = neutral 4 = agree 5 = strongly agree</td>
</tr>
<tr>
<td>Q29</td>
<td>Computer Science Attitude Survey: <em>Usefulness of computer science and programming</em></td>
<td>I'll need a firm mastery of programming for my future work.</td>
<td>1 = strongly disagree 2 = disagree 3 = neutral 4 = agree 5 = strongly agree</td>
</tr>
<tr>
<td>Q30</td>
<td>Computer Science Attitude Survey: <em>Usefulness of computer science and programming</em></td>
<td>I will use programming in many ways throughout my life.</td>
<td>1 = strongly disagree 2 = disagree 3 = neutral 4 = agree 5 = strongly agree</td>
</tr>
<tr>
<td>Q31</td>
<td>Computer Science Attitude Survey: <em>Usefulness of computer science and programming</em></td>
<td>Programming is of no relevance to my life.</td>
<td>1 = strongly agree 2 = agree 3 = neutral 4 = disagree 5 = strongly disagree</td>
</tr>
<tr>
<td>Q32</td>
<td>Computer Science Attitude Survey: <em>Usefulness of computer science and programming</em></td>
<td>Programming will not be important to me in my life's work.</td>
<td>1 = strongly agree 2 = agree 3 = neutral 4 = disagree 5 = strongly disagree</td>
</tr>
<tr>
<td>Q33</td>
<td>Computer Science Attitude Survey: <em>Usefulness of computer science and programming</em></td>
<td>I see computer science as a subject I will rarely use in my daily life.</td>
<td>1 = strongly agree 2 = agree 3 = neutral 4 = disagree 5 = strongly disagree</td>
</tr>
<tr>
<td>Q34</td>
<td>Computer Science Attitude Survey: <em>Usefulness of computer science and programming</em></td>
<td>Taking computer science courses is a waste of time.</td>
<td>1 = strongly agree 2 = agree 3 = neutral 4 = disagree 5 = strongly disagree</td>
</tr>
<tr>
<td>Q</td>
<td>Question</td>
<td>Description</td>
<td>Scale</td>
</tr>
<tr>
<td>---</td>
<td>--------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Q35 | Computer Science Attitude Survey: *Usefulness of computer science and programming* | In terms of my adult life it is not important for me to do well in computer science in college. | 1 = strongly disagree  
2 = disagree  
3 = neutral  
4 = agree  
5 = strongly agree | Ordinal |
| Q36 | Computer Science Attitude Survey: *Usefulness of computer science and programming* | I expect to have little use for programming when I get out of school. | 1 = strongly agree  
2 = agree  
3 = neutral  
4 = disagree  
5 = strongly disagree | Ordinal |
| Q37 | Computer Science Attitude Survey: *Effective motivation in computer science and programming* | I like writing computer programs. | 1 = strongly disagree  
2 = disagree  
3 = neutral  
4 = agree  
5 = strongly agree | Ordinal |
| Q38 | Computer Science Attitude Survey: *Effective motivation in computer science and programming* | Programming is enjoyable and stimulating to me. | 1 = strongly disagree  
2 = disagree  
3 = neutral  
4 = agree  
5 = strongly agree | Ordinal |
| Q39 | Computer Science Attitude Survey: *Effective motivation in computer science and programming* | When a programming problem arises that I can't immediately solve, I stick with it until I have the solution. | 1 = strongly disagree  
2 = disagree  
3 = neutral  
4 = agree  
5 = strongly agree | Ordinal |
| Q40 | Computer Science Attitude Survey: *Effective motivation in computer science and programming* | Once I start trying to work on a program, I find it hard to stop. | 1 = strongly disagree  
2 = disagree  
3 = neutral  
4 = agree  
5 = strongly agree | Ordinal |
| Q41 | Computer Science Attitude Survey: *Effective motivation in computer science and programming* | When a question is left unanswered in computer science class, I continue to think about it afterward. | 1 = strongly disagree  
2 = disagree  
3 = neutral  
4 = agree  
5 = strongly agree | Ordinal |
| Q42 | Computer Science Attitude Survey: *Effective motivation in computer science and programming* | I am challenged by programming problems I can't understand immediately. | 1 = strongly disagree  
2 = disagree  
3 = neutral  
4 = agree  
5 = strongly agree | Ordinal |
| Q43 | Computer Science Attitude Survey: *Effective motivation in computer science and programming* | Figuring out programming problems does not appeal to me. | 1 = strongly agree  
2 = agree  
3 = neutral  
4 = disagree  
5 = strongly disagree | Ordinal |
| Q44 | Computer Science Attitude Survey: Effective motivation in computer science and programming | The challenge of programming problems does not appeal to me. | 1 = strongly agree 2 = agree 3 = neutral 4 = disagree 5 = strongly disagree | Ordinal |
| Q45 | Computer Science Attitude Survey: Effective motivation in computer science and programming | Programming is boring. | 1 = strongly agree 2 = agree 3 = neutral 4 = disagree 5 = strongly disagree | Ordinal |
| Q46 | Computer Science Attitude Survey: Effective motivation in computer science and programming | I don’t understand how some people can spend so much time on writing programs and seem to enjoy it. | 1 = strongly agree 2 = agree 3 = neutral 4 = disagree 5 = strongly disagree | Ordinal |
| Q47 | Computer Science Attitude Survey: Effective motivation in computer science and programming | I would rather have someone give me the solution to a difficult programming problem than to have to work it out for myself. | 1 = strongly agree 2 = agree 3 = neutral 4 = disagree 5 = strongly disagree | Ordinal |
| Q48 | Computer Science Attitude Survey: Effective motivation in computer science and programming | I do as little work in computer science courses as possible. | 1 = strongly agree 2 = agree 3 = neutral 4 = disagree 5 = strongly disagree | Ordinal |
BIBLIOGRAPHY


68


