

Two-Stage Programming Projects: Individual Work Followed by Peer Collaboration

Apeksha Awasthi (aawasth@ncsu.edu), Lina Battestilli (lbattestilli@ncsu.edu)

ABSTRACT

In this work, we studied 2-stage projects in an introductory computer science course. In stage-1, the students submitted their programming projects individually and then in stage-2 they were paired to work on the same project in order to create an improved solution. We used isomorphic assessments before and after each stage of the projects to gauge the students' retention of the course materials. We also studied the students' perceptions and experiences with working in 2-stages and their confidence toward computing. We found that 2-stage projects improved students' understanding of the course material. Students liked working in 2-stages because it showed them new ways to approach the same problem.

1 INTRODUCTION

In any discipline, providing constructive and detailed feedback on assignments is an essential part of the students' learning. Feedback in introductory computer science courses, however, tends to be automated, minimal and thus not very helpful. Typically, the students are provided a grade and feedback but don't have a chance to address the feedback before moving on to the next assignment. This is a problem because learning how to program is often difficult for students and leads to high attrition rates in the first-year computer science courses [Watson 2014].

One approach to this problem is to use two-stage submissions where students turn in the assignment and receive instructor's feedback instead of a grade, then the students address the feedback, and resubmit the assignment for a final grade [Szabo 2017]. Another approach is to allow students to first work alone and then in groups [Cao 2017, Smith 2009]. In this study, we apply this idea to 2-stage submission of programming projects. For the first stage, the students work alone on the projects and are given an individual grade and feedback. For the second stage, the students address the feedback with an individual reflection, then peer review each other's code and re-submit a pair-coded version of the project. The 2-stage project group submissions are graded again and potentially could increase the students' final grade for the project.

Often, when students are paired together, they just copy from each other instead of discussing the material and together coming up with a solution [Porter 2011a]. To address this issue, we paired students with other students who received a similar individual grade on the 1-stage of the project so that they had to work together to improve the code of their projects. This technique worked well because students like when the instructor creates the pairs rather than self-selection in team-based learning in computer science courses [Kirkpatrick 2017]. To assess whether the students individually made learning gains after the 2-stage pair programming, we administered pre and post assessments and developed isomorphic questions as described in [Kjolsing 2016].

In this study, we explore the effects of 2-stage programming projects in an introductory CS1 course. Our results show that the 2-stage projects improved students' understanding of the course material as it allowed them to reflect on the instructor's initial feedback and work together with a peer. We focused on the following research questions:

- RQ1.** Can 2-stage programming projects improve students' understanding of the course material?
- RQ2.** What are the students' experiences and attitudes with 2-stage projects?
- RQ3.** What is the effect of 2-stage project participation on students' attitudes and confidence towards computing?

2 BACKGROUND

Introductory programming courses can be difficult for students because many students come with the preconception that learning how to program can be very hard. Students also fear poor performance in the course due to lack of prior experience. In recent years, however, the interest in Computer Science has significantly increased the number of students enrolling in CS1 courses. Unfortunately, this leads to large size classes where the instructor's individual feedback is often limited [Szabo 2017]. *Pair programming* and *peer code review* can be used to provide students with extra feedback from their peers on programming assignments. Studies have shown that the learning outcomes are greater when students have the opportunity to review other students' assignments [Reily 2009, Li 2006]. Studies have shown that students who pair program are more likely to choose Computer Science as their major [Williams 2003]. Students who pair program show higher rates of retention and higher promise of succeeding in future courses in which they would have to work alone [Mcdowell 2006]. When working in groups, the students become more confident in their work and find it more enjoyable. Students benefit from communication with their peers and might even make new friends.

Metacognition is another way to help students benefit from grading feedback via individual reflections after each programming assignment. Metacognition is the process of analyzing whether a goal has been met or what a person could do to meet it [Livingston 1997]. Students that develop their metacognitive skills tend to have more academic success [Coutinho 2007]. Exam wrappers have been used in multiple studies to encourage metacognition by asking students to reflect on their study habits and what topics they still need to master [Craig 2016, Gezer-Templeton 2017]. Often, students get their grades for assignments but don't even look at the feedback to evaluate what they might have done wrong or what they could do to improve. Metacognition can also help students' develop *Growth Mindset*, the thinking that one's intelligence level and ability to learn is not fixed [Dweck 2008, Hochanadel 2015]. Students with a growth mindset are more likely to succeed than those with a fixed mindset.

Two-stages are also used in *Peer Instruction (PI)*, which is a student-centric pedagogy where the students move from the role of passive listeners to active participants. Recently, there has been significant research regarding the value of PI in computer science [Porter 2011b, Simon 2010, Porter 2013, Zingaro 2010]. In PI, in the first stage the students answer each question individually. In the second stage, the students discuss the question with their peers in a small group and come up with a group answer. Typically, clicker questions are used to engage students in the two-stages of PI.

Cao et al. extends the PI concept to **2-Stage Exams** where the students take some parts of a test individually and other parts in small groups [Cao 2017]. They found that the group portion of the test improved students' individual knowledge on the topic. The benefits of two-stage exams have been explored across other fields such as physiology [Cortright 2003], physics [Singh 2005], and speech and language pathology [Dahlström 2012].

Our study builds upon PI, 2-stage exams, pair programming, code review and reflections. We extend the 2-stage concept of working individually and then collaboratively to programming projects in an introductory computer science course.

3 METHOD

Three sections of a CS1 Java course at a large public university participated in this study during spring 2017. At our institution, this course is the first programming course that students take in the Computer Science major. The course learning objectives are basic programming such as variables, data types, loops, conditionals, methods and introductory object-oriented concepts. There was an experimental section and two control sections with a total

enrollment of 87 students, 29 per section. There were two different instructors, one instructor taught the experimental section and is the author of this paper and another instructor taught the two control sections.

The course has six large programming projects, two tests, a final exam, and other smaller assignments. In this study, we focus on the programming projects which were the same across all three sections and were written by a third instructor who is the overall course coordinator. The grading rubric and guidelines for the projects were identical across all sections of the course.

The demographic data for the experimental section is shown in Figure 1. All students were undergraduates and the section consisted of mostly males, STEM majors, who were freshmen and sophomores. The demographic data for the control section was not collected but students are assigned to sections of the course randomly based on their schedule. The size and demographics of the experimental section are typical for our institution.

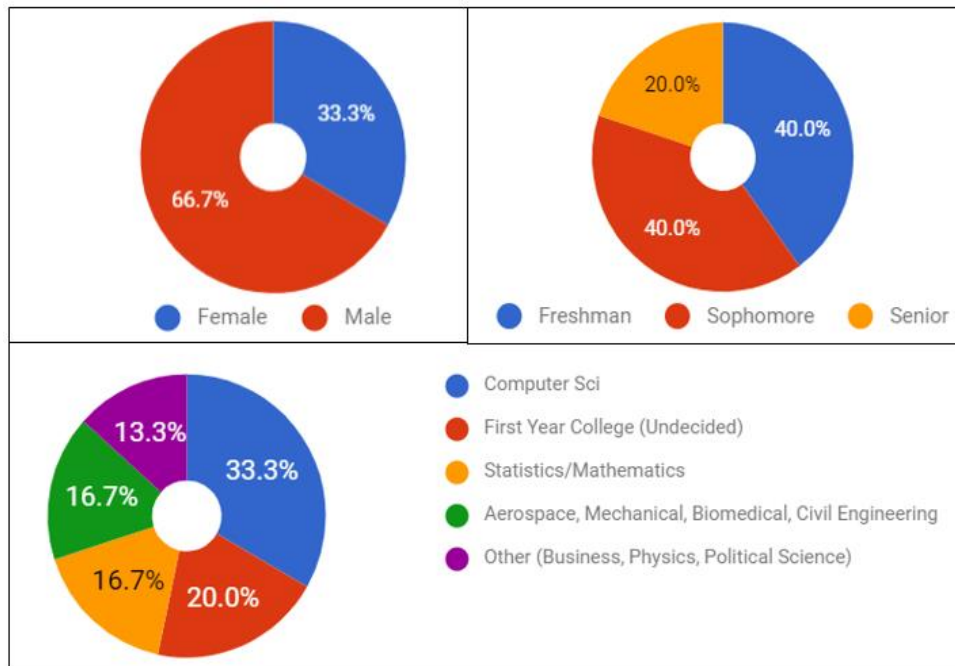


Figure 1: Demographics of the students in the experimental section.

3.1 Experimental Design

Typically for this course, the students work on the programming projects and submit them individually. To implement 2-stage projects, we add the components from Table 1 to the experimental section. The students in control sections only did the pre-assessment component PRE_A in order to evaluate their learning after they had submitted their individually-coded projects, i.e. stage 1. This was needed to compare how the students did with and without the 2-stage.

Table 1: Components of the experiment

Component	Description
Stage 1 (S1) Individually-coded Project	Students were assigned a project based on the recent course material. In this first stage, the students were to complete and submit the project individually.
Stage 2 (S2) Pair-coded Project	The second stage of the project was to be completed by students in pairs. This stage was optional and the students chose to do it or not after they had completed Stage 1 of each project. Students who signed up for the second stage were assigned a partner based on their project grades of Stage 1. Students with similar grades were paired together. This was done so they can work together to

	improve their grades. However, if a student got a failing grade then they were paired with someone who got a high grade. The purpose for this is that the student with the low grade can see a good solution for the project. The two partners were to work on the project together and submit an improved pair-coded version of the project together.
Pre Assessment (PRE_A), Post Assessments ($POST_A$)	These assessments measured the learning of the students after each stage. The pre-assessment and post-assessment were short isomorphic quizzes that each student had to take individually. The pre-assessment was given before the second-stage and measured learning after Stage 1, i.e., individually working on the project. The post-assessment was given after the second-stage and measured learning after pair-coding and discussion with a partner. The assessments had one to two questions based on the learning objectives of the project. The pre-assessment was generally given the day after the first stage was submitted and the post-assessment was generally given the day after the second stage was submitted.
Reflection (RE):	Students who had signed up for the second stage were also encouraged to complete an individual reflection. The reflection consisted of a series of question about why the students lost points in the first stage, how they could improve for future projects, and how the second stage experience was for them.

As noted previously, stage-2 of the projects was not mandatory for the students in the experimental section. The students were given the option to sign up for stage-2. For the students who participated in stage-2, a new grade was calculated which replaced their stage-1 project grade only if it increased their grade. The final project grade was calculated using the following formula:

$$Final\ Grade = S1 + (S2 - S1) * 0.3 + RE * 0.01 + \frac{(PRE_A + POST_A)}{2} * 0.02$$

Figure 2 shows an example timeline of how the projects and assessments were assigned. There were six projects in over the course of the semester and a project was due every two weeks. Because of the tight timeline, the students were generally given about three days to work on Stage 2 before they had to start working individually on the next project.

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
1	2 Project Stage 1 Due	3 In-class pre-assessment Participation request	4	5	6 Receive grade & pair assignment	7
8	9 Project Stage 2 Due	10 In-class post-assessment	11	12	13	14

Figure 2: Project Submission Timeline

3.2 Projects

Six programming projects are typically assigned in this course. Each project has a set of learning objectives which are based on what was recently taught in the course, see Table 2. For each project, the students submit source files, structure diagrams, black box and white box tests. The project grading rubric takes into account implementation, Javadoc documentation, correctness of structure diagram and teaching staff tests. The projects are graded and feedback is provided to the students as to why they lost points if they did.

Table 2: Descriptions of the six projects in the course.

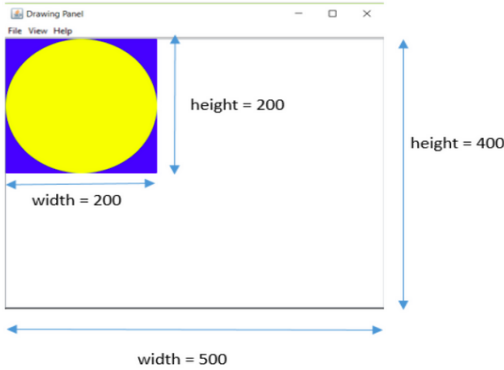
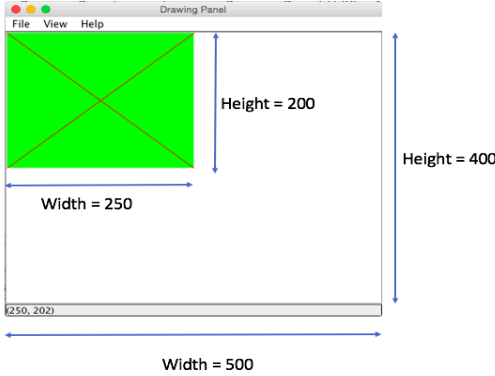
Project	Learning Objectives
Project 1	For loops and Java classes for text-based output to the Console Window
Project 2	Loops and Java classes for text-based inputs/outputs, selection logic with if/else statements, static methods, Graphics/Basic Animation
Project 3	While Loops, Using Java classes for text based input/output, methods with multiple parameters and return, White-Box Testing.
Project 4	Reading/Writing to files, Loops, writing methods and using Java classes
Project 5	Construct and use one and two dimensional arrays, Arrays of Objects
Project 6	Object-Oriented Design, Writing Classes that drive a GUI

3.3 Isomorphic Questions for Pre/Post Assessments

To test whether the students' individual learning had improved due to the stage-2 of the projects, the pre-assessment and post-assessment quizzes had to be isomorphic [Kjolsing 2016, Porter 2011a]. The questions were created based on the learning outcomes for a project. Instructor experienced in teaching a CS1 course were brought in to review the assessments for isomorphic equality.

Sample Assessment Questions

For example, Project 2's main learning objectives were using Java Graphics objects and reading user input from the Console Window. The first question in both the Pre/Post Assessments had to do with creating Graphics objects and the second question pertained to reading from user input.

Pre-Assessment	Post Assessment
<p>Part1: Write the code to create the following figure. The square is blue and its top left corner is at (0,0). The enclosed circle is yellow.</p>  <p>SOLUTION</p> <pre>public static void main (String[] args){ Scanner console = new Scanner (System.in); DrawingPanel panel = new DrawingPanel(500, 400); Graphics g = panel.getGraphics(); g.setColor(Color.BLUE); g.fillRect (0,0,200,200); g.setColor(Color.YELLOW); g.fillOval (0,0,200,200); }</pre>	<p>Part1: Write the code to create the following figure. The rectangle is green and its top left corner is at (0,0). The diagonal lines are red.</p>  <p>SOLUTION</p> <pre>public static void main (String[] args){ DrawingPanel panel = new DrawingPanel(500, 400); Graphics g = panel.getGraphics(); g.setColor(Color.GREEN); g.fillRect (0,0,250,200); g.setColor(Color.RED); g.drawLine(0, 0, 250, 200); g.drawLine(250, 0, 0, 200); }</pre>

Part 2: Write a program called UserInfo.java that prompts the user for their full name, height and age and then prints the information entered to the Console Window. Here is example output:

```
Full Name:Jane Java
Height:5.37
Age:102
Hi Jane Java [height:5.4, age:102]
```

SOLUTION

```
public static void main (String[] args){
    Scanner console = new Scanner (System.in);
    System.out.print("Full Name:" );
    String name = console.nextLine();
    System.out.print("Height:" );
    double h = console.nextDouble();
    System.out.print("Age:" );
    int age = console.nextInt();
    System.out.printf("Hi %s [height:%.1f, age:%d]", name, h, age);
}
```

Part 2: Write a program called Restaurant.java that prompts the user the main course they would like to order, the serving size they want, and their table number and then prints the information entered to the Console Window. Here is example output:

```
Hello, What would you like for your main course? Pizza
How many servings would you like?
5.7
What is your table number? 8
You have ordered a(n) Pizza with 5.7 servings at table number 8
```

SOLUTION

```
public static void main (String[] args){
    Scanner console = new Scanner (System.in);
    System.out.print("Hello, What would you like for your main course? " );
    String entree = console.nextLine();
    System.out.println("How many servings would you like? " );
    Double serving = console.nextDouble();
    System.out.print("What is your table number? " );
    int tableNumber = console.nextInt();
    System.out.print("You have ordered a(n) " + entree + " with "
        + serving + " servings at table number " + tableNumber);
}
```

To gauge the learning gains, the questions were designed so that a student would have to have knowledge on the subject and couldn't answer the post-assessment simply based on the pre-assessment. These assessment questions were given towards the beginning of the semester.

3.4 Reflections

Students completed individual reflections after stage-1 of the projects. They were asked to reflect on where they had lost points, how they had managed the individual completion of stage-1 and about their experience of working with a partner for stage-2. Most of the reflection consisted of open ended questions, except of the project management part which consisted of 5-point Likert scale statements, shown in Figure 3. The reason for these questions was to have the students reflect on their strategy of completing stage-1 of the project and to provide ideas on how they can approach better the completion of the next project.

	Strongly Agree	Agree	Neutral	Disagree	Strongly disagree
I managed my time well on this project	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I met all of the requirements for this project	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I attended office hours when I had questions about the project	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I posted questions on Piazza when I had questions about this project	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I read the whole project description prior to starting the project	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
It took me longer to complete the project that I initially expected	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I will start earlier on the next project	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Figure 3: Management Questions in the Reflection

3.5 End-of-Semester Survey and Focus Group

At the end of the semester the students were given a survey inquiring about their opinions on the course overall, as well as their experiences with 2-stage projects. The survey also asked questions about confidence and attitude toward computing which were pulled from existing surveying instruments [Wiebe 2003, Dorn 2015]. The students who had participated in the stage-2 of the projects were asked what they did or didn't like about the second stage. The students who had not participated were asked why they had not chosen to participate in any of the second stages of the projects.

After the final course grades were submitted, a small focus group was held with students from the experimental section. The students were asked about how the 2-stage projects might have affected their attitudes and confidence towards the course material and what they did and didn't like.

4 RESULTS

The collected data, described in the previous sections, were analyzed in order to answer the three main research questions.

4.1 RQ1 Understanding the Material

The pre/post assessments were used to determine if the 2-stage projects improved the students' understanding of the course material. Figure 4 shows an improvement from the pre-assessment to the post-assessment on all the projects except for Project 4. The reason for this might be that the concepts of reading/writing to/from files from Project are often too complex for novice Java programmers.

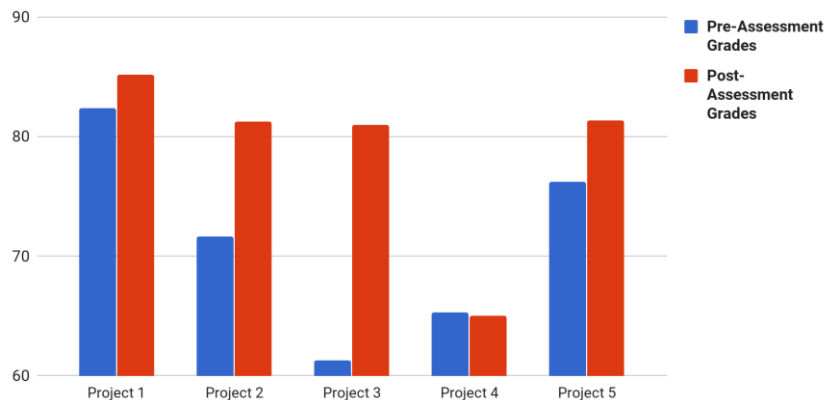


Figure 4: Average pre/post-assessment grades of the stage-2 participants from the experimental section

We also analyzed the same post-assessment grades for the two control sections. The control sections scored lower on every topic in comparison to the students who participated in the 2-stage projects in the control section, see Figure 5. Data was not collected for Project 3 for the control section due to scheduling issues.

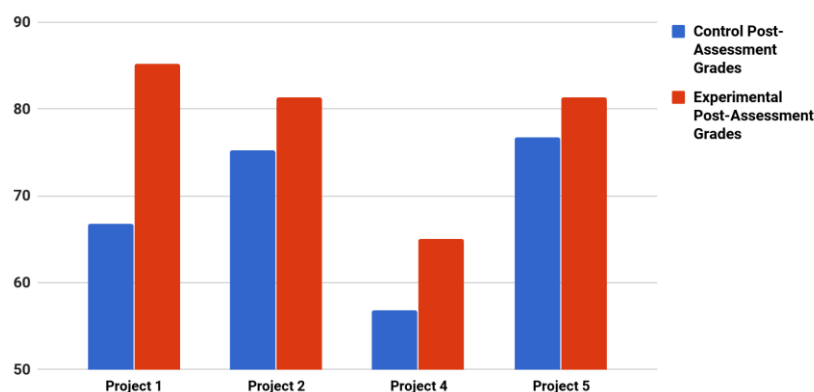


Figure 5: Average post-assessment grades of the Control vs. the Experimental Section.

We also examined which students chose to participate in the 2- stages of the projects based on their average test grades. Tests were taken individually by students, thus those grades weren't affected by external factors like extra credit and were a good indicator of the students' understanding of the course material. Figure 6 shows that mostly A, B, and C students participated in the 2-stage. Therefore, it might be better to make the 2-stage of the projects mandatory rather than optional and thus also help the learning of the lower grade students. The reason we made stage-2 optional during our experiment is because we wanted the assigned partners to be both willing and eager to work together and create a better pair-coded project solution.

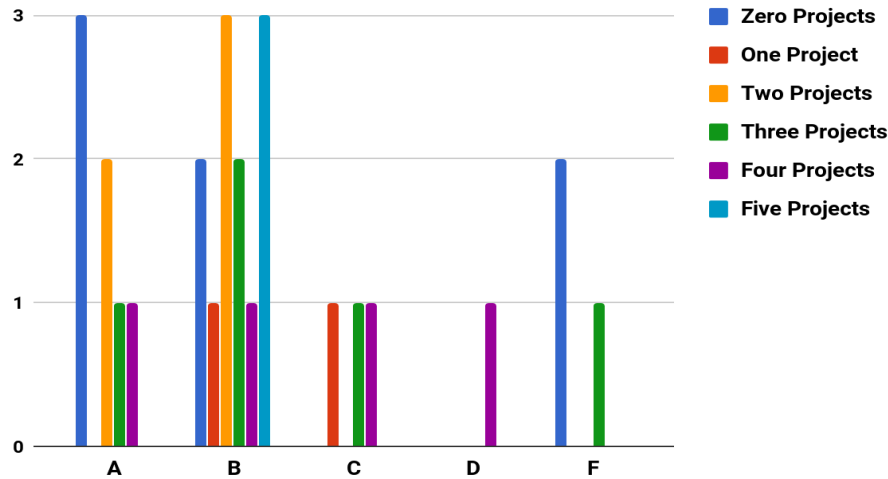


Figure 6: Average Test Grades vs. the number of 2-stages that they chose to participate in

4.2 RQ2 Students Experiences

The End-of-Semester (EoS) survey asked students how they felt about the 2-stage projects. 77% of the students had participated in at least one out of the five project second stages. The experience for these students is coded in Table 3. Many of the students showed growth mindset and felt that working with a partner in stage-2 helped them learn. However, there were also students who were simply focused on grades rather than learning. Many of the students that chose to participate in stage-2 already had good grades on stage-1 and thus chose not to submit new code for stage-2.

Table 3: 2-stage participants' on why they did or didn't like the 2-stage projects

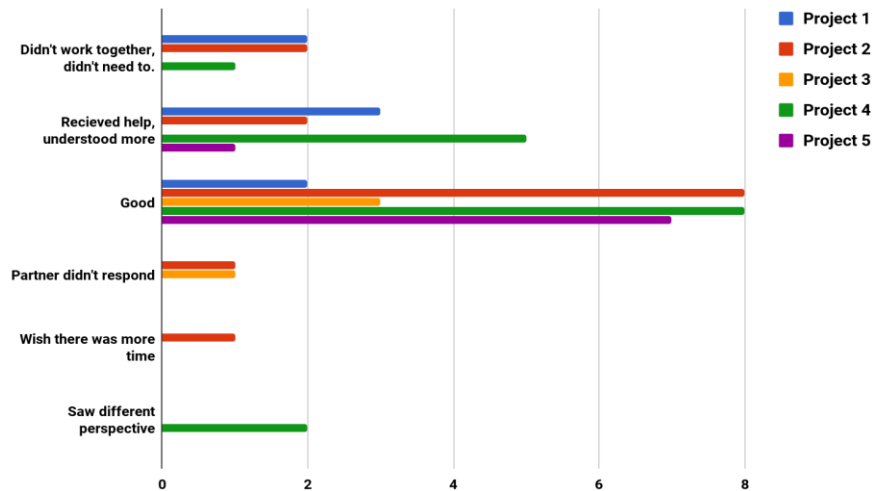
Answer	Number of students
Showed me a different perspective	6
Helped me understand what I missed	3
Helped me understand difficult material	3
Helped me improve my grade	6
Didn't help me	2
Helped the other person, but not me	1
My code didn't need improvement	2
Time was very limited	3
Didn't like being randomly paired	1
Partner wanted extra credit, but didn't want to talk about concepts	2
Didn't answer	3

Table 4 shows that students mainly chose not to participate in stage-2 because of the limited time allotted to work with a partner. In our experiment, this tight schedule was necessary because there were 6 projects and the due dates were approximately every two weeks.

Table 4: Students on why they chose not to participate in stage-2.

Answer	Number of students
Not enough time allotted	3
Didn't need to	1
Didn't know the grade for stage-1	1
Did not provide an answer	1

The individual reflections revealed that as the projects got harder, the time the students spent on stage-1 increased from less than 3 hours to 8 or more hours, while the time that students spent on stage-2 stayed around 1-2 hours. Also the reflections showed that the majority of the students felt that working with a partner in stage-2 was a good experience, they received help, saw a different perspective and reviewed someone else's code, see Figure 7. A minority of the students felt that stage-2 didn't help them because their grade was already good enough or their partner didn't respond.

**Figure 7: Students' responses on their overall experience of working with a partner for stage-2**

In the focus group we asked questions pertaining to students experiences and attitudes towards two-stage projects/ The students' answers reinforced some of the findings from the EoS Survey. Students expressed that the 2-stage projects helped them see new ways to approach a problem. One student said the 2-stages helped them see more efficient ways to solve the same problem. Students also enjoyed the 2-stage because it opened communication amongst their peers in a class and sometimes it is difficult to connect and talk to peers in. The time-constraint was their biggest challenge with the stage-2. Also, the students did not like the automated and limited feedback from stage-1. If the paired students for stage-2 had failed the same exact teaching staff test case in stage-1, then they did not know what the problem was and how to fix it.

4.3 RQ3 Attitudes and confidence towards computing

The EoS survey started with questions pertaining to how the students felt about computer science and programming. Figure 8 shows that the 2-stage participants answered neutral to more of the questions. The students who did not participate in stage-2 tend to have more confidence in their own abilities than students who participated in stage-2. The reason behind this could be that students' computing confidence and attitude drove their decisions to participate and not to participate in stage-2. Students with less confidence towards the material were more likely to participate in at least one 2-stage. Students who didn't participate in stage-2 disagreed with the statements concerning enjoying programming and being interested in computer science.

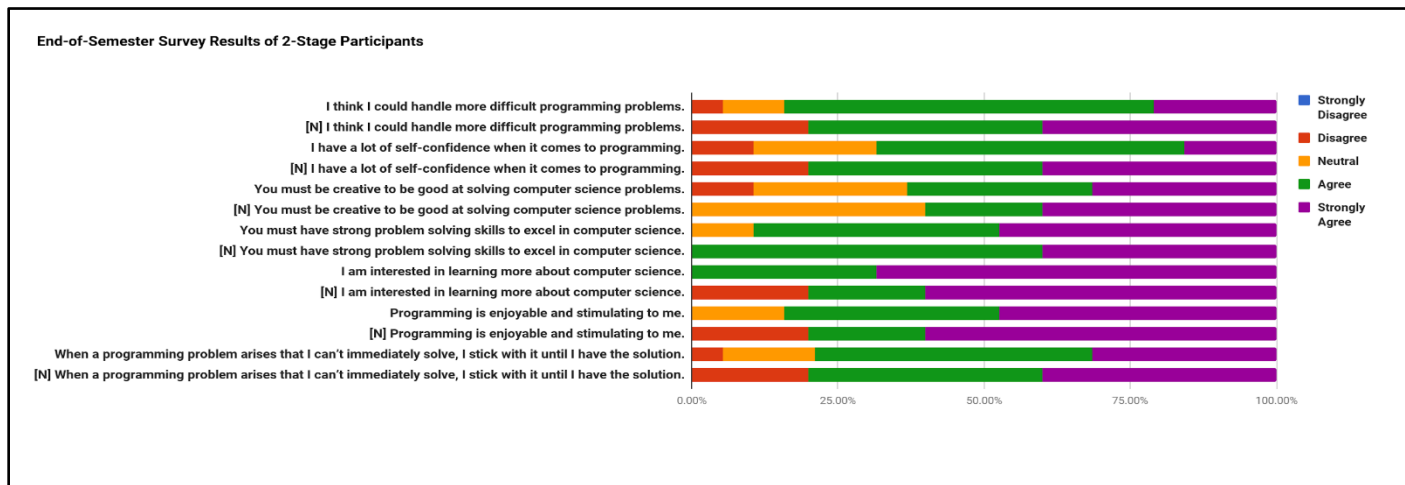


Figure 8: The results of the questions that asked students about computer science and programming from the EoS survey. The questions answered by non-participating in stage-2 students begin with [N].

In the focus group, we found that the students felt that their attitudes didn't change very much throughout the semester. They had started the semester with an open and conscientious attitude towards programming and the 2-stage projects played no role in their attitudes. The students expressed that their confidence in the course material had significantly improved because they either had no experience before the course or felt that the course improved their understanding of the material. The students agreed that the 2-stage projects overall helped them have more confidence in explaining their own code.

4.4 THREATS TO VALIDITY

The experimental group and control group were taught by two different instructors, which could affect the results of the assessments. However, the learning gains before/after stage-2 for the control section alone indicate the advantage of this intervention. Also, the demographic info of the control sections was unknown.

5 CONCLUSIONS AND FUTURE WORK

This study examined the effects of 2-stage programming projects on students' understanding of the course material and their attitude and confidence towards programming and computer science. Students showed improvement in their learning and enjoyed working with a partner on the second stage. The two-stage projects, however, had little effect on students' attitudes or confidences toward computing. Further studies are needed to determine the effects of longer timeline for the students to complete stage-2 and figure out how to provide more detailed feedback to the students for stage-1. We recommend the stage-2 projects be made mandatory rather than optional to encourage students to adapt to a growth mindset, review other's code, and practice working with partners.

ACKNOWLEDGMENTS

We would like to thank Dr. Paul Chao for discussions on 2-stage and for detailed review of the polymorphic assessment questions. Also, we want to thank Dr. Jessica Schmidt who taught the two control sections.

REFERENCES

- [Dorn 2015] B. Dorn and A. E. Tew. Empirical Validation and Application of the Computing Attitudes Survey. *Computer Science Education*, 25(1):1-36, 2015
- [Cao 2017] Cao, Yingjun, and Leo Porter. "Evaluating Student Learning from Collaborative Group Tests in Introductory Computing." *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education - SIGCSE '17* (2017): 99-104. *ACM Digital Library*. Web.

[Szabo 2017] Claudia Szabo, and Nick Falkner (2017) Silence, Words, or Grades: The Effects of Lecturer Feedback in Multi-Revision Assignments.

[Cortright 2003] Cortright, R. N., H. L. Collins, D. W. Rodenbaugh, and S. E. Dicarlo. "Student Retention Of Course Content Is Improved By Collaborative-Group Testing." *AJP: Advances in Physiology Education* 27.3 (2003): 102-08. Web.

[Coutinho 2007] Coutinho, Savia A. "The Relationship between Goals, Metacognition, and Academic Success." *Educate - The Journal of Doctoral Research in Education* 7.1 (2007): n. pag. Print.

[Craig 2016] Craig, Michelle, Diane Horton, Daniel Zingaro, and Danny Heap. "Introducing and Evaluating Exam Wrappers in CS2." *SIGCSE '16 Proceedings of the 47th ACM Technical Symposium on Computing Science Education* (2016)

[Dahlström 2012] Dahlström, Örjan. "Learning during a Collaborative Final Exam." *Educational Research and Evaluation* 18.4 (2012): 321-32. Web.

[Gezer-Templeton 2017] Gezer-Templeton, P. Gizem, Emily J. Mayhew, Debra S. Korte, and Shelly J. Schmidt. "Use of Exam Wrappers to Enhance Students' Metacognitive Skills in a Large Introductory Food Science and Human Nutrition Course." *Journal of Food Science Education* 16.1 (2017): 28-36. Web.

[Dweck 2008] Dweck, Carol S.. (2008) *Mindset :the new psychology of success* New York : Ballantine Books

[Hochanadel 2015] Hochanadel, Aaron, and Dora Finamore. "Fixed And Growth Mindset In Education And How Grit Helps Students Persist In The Face Of Adversity." *Journal of International Education Research (JIER)* 11.1 (2015): 47. Web.

[Reily 2009] Ken Reily, Pam Ludford Finnerty, and Loren Terveen. "Two Peers Are Better Than One: Aggregating Peer Reviews for Computing Assignments Is Surprisingly Accurate." *Proceedings of the ACM 2009 International Conference on Supporting Group Work* (2009): 115-24. *ACM Digital Library*. Web.

[Kirkpatrick 2016] Kirkpatrick, Michael S. "Student Perspectives of Team-Based Learning in a CS Course: Summary of Qualitative Findings." *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education - SIGCSE '17* (2017): 327-33. *ACM Digital Library*. Web. 03 Aug. 2017.

[Kjolsing 2016] Kjolsing, Eric, and Lelli Van Den Einde. "Peer Instruction: Using Isomorphic Questions to Document Learning Gains in a Small Statics Class." *Journal of Professional Issues in Engineering Education and Practice* 142.4 (2016): 04016005. Web.

[Porter 2011a] Porter, Leo, Cynthia Bailey Lee, Beth Simon, and Daniel Zingaro. "Peer Instruction." *Proceedings of the Seventh International Workshop on Computing Education Research - ICER '11* (2011): 45-52. Web.

[Porter 2013] Porter, Leo, Cynthia Bailey Lee, and Beth Simon. "Halving Fail Rates Using Peer Instruction." *Proceeding of the 44th ACM Technical Symposium on Computer Science Education - SIGCSE '13* (2013): 177-82. Web.

[Porter 2011b] Porter, Leo, Cynthia Bailey Lee, Beth Simon, Quintin Cutts, and Daniel Zingaro. "Experience report: a multi-classroom report on the value of peer instruction." *Proceedings of the 16th Annual Joint Conference on Innovation and Technology in Computer Science Education - ITiCSE '11* (2011): 138-42. Web.

[Li 2006] Li, Xiaosong. "Using Peer Review to Assess Coding Standards - A Case Study." *Frontiers in Education Conference, 36th Annual* (2006): n. pag. Web.

[Livingston 2006] Livingston, Jennifer A. "Metacognition: An Overview." (1997): n. pag. Web.

[Mcdowell 2006] Mcdowell, Charlie, Linda Werner, Heather E. Bullock, and Julian Fernald. "Pair Programming Improves Student Retention, Confidence, and Program Quality." *Communications of the ACM* 49.8 (2006): 90-95. Web.

[Simon 2010] Simon, Beth, Michael Kohanfars, Jeff Lee, Karen Tamayo, and Quintin Cutts. "Experience Report: Peer Instruction in Introductory Computing." *Proceedings of the 41st ACM Technical Symposium on Computer Science Education - SIGCSE '10* (2010): 341-45. Web.

[Singh 2005] Singh, Chandralekha. "Impact of Peer Interaction on Conceptual Test Performance." *American Journal of Physics* (2005): 446-51. Web.

[Smith 2009] Smith, M. K., W. B. Wood, W. K. Adams, C. Wieman, J. K. Knight, N. Guild, and T. T. Su. "Why Peer Discussion Improves Student Performance on In-Class Concept Questions." *Science* 323.5910 (2009): 122-24. Web.

[Watson 2014] Watson, Christopher, and Frederick W.b. Li. "Failure Rates in Introductory Programming Revisited." *Proceedings of the 2014 Conference on Innovation & Technology in Computer Science Education - ITiCSE '14* (2014): n. pag. Web.

[Wiebe 2003] Wiebe, E.N., Williams, L., Yang, K. and Miller, C. 2003. Computer Science Attitude Survey. North Carolina State University Technical Report TR-2003-1. (2003).

[Williams 2003] Williams, L., C. Mcdowell, N. Nagappan, J. Fernald, and L. Werner. "Building Pair Programming Knowledge through a Family of Experiments." *2003 International Symposium on Empirical Software Engineering, 2003. ISESE 2003. Proceedings.* (2003): 143-53. Web.

[Zingaro 2010] Zingaro, Daniel. "Experience Report: Peer Instruction in Remedial Computer Science." *Proceedings of the 22nd World Conference on Educational Multimedia, Hypermedia & Telecommunications* (2010): n. pag. Web.