WIP: An exploration into the muddiest points and self-efficacy of students in introductory computer science courses

Daniel Perez
Computer Science
Florida International University
Miami, FL 33199, USA
dpere103@fiu.edu

Leila Zahedi
Computer Science
Florida International University
Miami, FL 33199, USA
lzahe001@fiu.edu

Monique Ross
Computer Science
Florida International University
Miami, FL 33199, USA
moross@fiu.edu

Jia Zhu
Computer Science
Florida International University
Miami, FL 33199, USA
jzhu004@fiu.edu

Tiffany Vinci-Cannava
Computer Science
Southern New Hampshire University
Manchester, NH 03106, USA
tiffany.vinci-cannava@snhu.edu

Laird Kramer
STEM Transformation Institute
Florida International University
Miami, FL 33199, USA
kramerl@fiu.edu

Maria Charters
Computer Science
Florida International University
Miami, FL 33199, USA
charters@cis.fiu.edu

***Abstract*—This work-in-progress (WIP) paper in the research track explored the muddiest point and self-efficacy for students in introductory computer science courses, starting with one course at a southern public institution. With the increasing demand in computing-related careers, the increased enrollments in undergraduate computer science courses are facing challenges to ensure the average passing rate. In this work, we applied the muddiest point to capture student's perceptions of concepts of difficulty and used self-efficacy to better contextualize students' perfections of self-efficacy in relation to concepts of difficulty. In this paper we present the results of one section of one introductory computing course, that revealed the perceived difficult topics as well as changes in self-efficacy throughout the semester. Furthermore, a positive correlation was found between self-efficacy and performance. Keywords: self-efficacy, muddiest points, student performance, computer science education**

***Index Terms*—self-efficacy, muddiest points, student performance, computer science education**

I. Introduction

The demand for computing is growing in virtually all sectors of modern life. In the labor market, the growth of computing careers from 2018 to 2028 is projected at 12 percent by the United States Bureau of Labor Statistics [1]. This growth in demand coupled with the desire to find lucrative jobs deemed in demand has resulted in a surge in enrollments in undergraduate computer science courses and programs [2]. One southern public institution is no exception to this surge. Currently, computer science (CS) enrollment at this university is upwards of 2,500, up from recent years. High enrollment coupled with performance metrics that focus on four-year graduation rates exacerbates the challenges in introductory programming courses that have an average pass rate of the last eleven years of 64% [3] which is below the worldwide average pass rate of 67.7% [4]. It is worth nothing that pass rates vary considerably with lows of 23.1% and highs of 96%. While this university case has managed to increase pass rates from 41% to 64% in the past ten years with the introduction of some variation of active learning and learning assistants [3], there is still room for improvement. One proposed solution is to identify the concepts of difficulty and student misconceptions in introductory programming courses in order to develop targeted interventions for the students. In this study, the "muddiest point" and self-efficacy are used to capture student's perceptions of concepts of difficulty and to better contextualize students' perceptions of self-efficacy in relation to concepts of difficulty. The research questions guiding this work are:

1) *What topics in introductory programming course do students identify as being the hardest to understand?*
2) *How does student self-efficacy change over the semester in an introductory programming course?*
    a) *Is there a correlation between self-efficacy changes and performance in an introductory programming course?*

To answer these questions, a survey study was designed to collect data via a mobile application. With the support of one instructor in summer 2019, the survey was presented at the end of each class. The present WIP study reports data collected from this one semester's undergraduate introductory programming course.

II. Literature Review

As the first computing-related course the students encounter at a university, passing the introductory computer science course can be a difficult task. Many studies have been involved in investigating the contributing factors to pass the introductory computer science courses. Wilson's work [5] revealed the following three predictive factors, in order of importance, comfort level in the course, math background, and attribution to luck for success/failure. Learning styles have also been included as one of the factors contributing to students' successes. A study of it determined that reflective and verbal learners outperformed active and visual ones [6]. This study also pointed out that student opinions of the value of programming projects and lectures rank highest and seem to cut across learning style preference. Likewise, previous programming experience was found to have a positive effect on success in introductory courses in several studies [7] [8] [9] [10].

A theoretical framework that has been used in many studies in the context of introductory computer science courses is self-efficacy. Previous studies found out that self-efficacy was the most important performance predictor of students' learning outcomes [11], [12]. Students' self-efficacy was positively associated with course grades [13], [14]. Many studies also explored whether students' perceived self-efficacy changed. One study investigated how self-efficacy among students changed during their introductory computer science course and found no significant difference [15]. Self-efficacy perception levels were found to be at the medium level by Korkmaz and Altun [16] using the Likert scale. Kinnunen and Simon's work revealed that positive or negative changes are possible for students' self-efficacy in response to both positive or negative programming assignment experiences [17]. Different results in self-efficacy change were demonstrated in different research settings. The changes in self-efficacy found to be impacted by students' performance in the introductory computer science courses [12]. Another study by Kinnunen and Simon showed that programming assignments tend to catch students off guard with difficulties and produce intense emotional reactions that have effects on their self-efficacy accordingly [18].

As one of the techniques that help to identify the difficult topic in class, the "muddiest point" was found to be an intervention that positively impacts interest, attainment, and utility value without negative cost [19]. The effectiveness and impact of using the "muddiest point" approach have been found in courses in materials science [20], [21], and chemistry [22]. A study revealed increased self-efficacy when instructors implemented interventions based on students' feedback about the "muddiest point" [23]. However, studies focused on "muddiest point" are rarely found in computer science education.

III. Theoretical Framework

There were two frameworks used to design this study. These two frameworks included self-efficacy and muddiest point. In this section of the paper we will define each framework and how it was applied in the development of this inquiry. Self-efficacy is a framework credited to Bandura that is defined as an individual's judgment of his or her abilities to accomplish specific tasks or objectives [24]. It is an effective measure of human behavior as it offers both "cognitive and motivational drive" [25]. Likewise, "muddiest point" has been identified as one of the simplest monitoring tools to help assess where students are having difficulties or confusion. The technique consists of asking students to identify the challenging content by asking questions like: "What was the most difficult or confusing part in [the lecture, discussion, homework assignment, etc.]?" [26]. It is a simple mechanism for checking students' understanding of a topic. This assessment can also illuminate any misconceptions or confusion getting in the way of a student's comprehension. Self-efficacy and the "muddiest point" frameworks were used to develop the survey instrument which we will further discuss in section IV.

This work used self-efficacy as the theoretical framework to better contextualize students' perceptions of self-efficacy in relation to concepts of difficulty and was anchored by the performance metric of final grades. We utilized grades as the performance outcomes for self-efficacy as previous research shows grades are the most predictive factor of retention and persistence of students in computer science fields [27].

IV. Method

This study aims to identify concepts of difficulty and student misconceptions in introductory programming. The survey was designed to collect data on the "muddiest point" and self-efficacy, as well as the student's identification for traceability. All identifiable information about the students were kept confidential. The survey was in correlation with the syllabus for the introductory programming class. Therefore, the "muddiest point" options were different for each survey and depended on the material being covered on the day of the class. Self-efficacy was measured by asking students to identify their perceived self-efficacy in the course. The self-efficacy question was framed, by stating " Right now, I am confident I can do well in this class" and they were presented with a 5-point likert scale from one ("Not at all true") to five ("Exactly true"). This design of surveys aided in capturing the concepts of difficulty over-time which provided insight into what concepts require additional time and/or supplemental materials. Likewise, gathering the snapshots of self-efficacy throughout the term allowed us to correlate concepts with self-belief which could aid in better course design.

This research study was IRB approved and 81 students consented to participate. All participants were taking the same introductory programming class at a public research university

TABLE I: Muddiest points with high frequencies per survey

Survey Number	Muddiest Point with Highest Frequency	Survey Number	Muddiest Point with Highest Frequency
Survey 1	Netbeans	Survey 13	N/A
Survey 2	Variables	Survey 14	N/A
Survey 3	Data types for variables	Survey 15	Traversing an array from beginning to end
Survey 4	Integer division vs. regular division vs. modulus division	Survey 16	Creating an array of domain objects from reading a file
Survey 5	Defining constructors, getters, setters, and toString	Survey 17	Defining an arrayList of primitives using Wrapper classes
Survey 6	Creating a Method that receives input and produces output	Survey 18	Finding the max, min, sum, and average of a value in an ArrayList
Survey 7	When to use a specific type of if-statement	Survey 19	How String methods work
Survey 8	When to use a switch statement	Survey 20	Creating an arrayList of domain objects from reading a file
Survey 9	The 3 types of loops and when to use each	Survey 21	Pseudo code to encrypt or decrypt a String variable using Caesar cypher
Survey 10	Variations of a for-loop	Survey 22	Files, reading, writing (Final review)
Survey 11	Loops (Midterm review)	Survey 23	N/A
Survey 12	Loops (Midterm review)	Survey 24	N/A

which is a Hispanic Serving Institution (HSI) with a large percentage of transfer students. The participants completed the survey online that was disseminated through Qualtrics Survey Software (Qualtrics, Provo, UT, USA), a web-based system for surveys. The online survey was made available to be completed on any electronic device with internet access at any time and location convenient for the student on a weekly basis. Each student's participation only lasted as long as it took her or him to fill out the survey, which probably occupied about 5 minutes of their time. The number of students who responded to the survey each week ranged from 11 to 61 and the total number of surveys conducted was 24 throughout the semester. At the conclusion of every class, the students that consented to participate were pushed a text message with a link to the survey to complete. Descriptive and inferential statistics were used to report the results of this study. In regards to identifying the muddiest points in the introductory programming course, descriptive analysis was used and results are represented in frequency and percentage. Also, linear regression was used to test the correlation between the level of self-efficacy and performance of the students at the end of the semester. Statistical significance was tested using a p-value of less than 0.05. Statistical analyses were managed using R version 3.6.1 in RStudio, version 1.1.456.

V. Results

A total of 81 students were surveyed in this research study. However, 11 students who did not answer enough surveys relevant to this study (less than 3 surveys) were removed from the analysis. Four of the aforementioned students dropped the course. The "muddiest point" with the highest frequency per survey, with the exception for exam weeks (marked as N/A), are reflected in Table I.

In the lectures where exam reviews were given, the surveys asked for the "Muddiest Point" for the topics being covered on that exam. In the midterm review, the muddiest point topics ranged from those being identified as the muddiest point from survey 1 through survey 10. For the final review, the same approach was adopted for muddiest point topics ranged from survey 15 through survey 22. In the midterm review, the "muddiest point" with the highest frequency was "Loops." In the final review, the "muddiest point" with the highest frequency was "Files, reading, writing."

To evaluate the trend in students' self-efficacy throughout the semester, we applied a regression model. First, we applied a step analysis to estimate a regression equation for each individual, which contained parameters for an intercept and the slope. The slope parameter was also used to assess self-efficacy over the course of the semester, leading to a trendline that illustrated if a student's level of self-efficacy increased, decreased or remained neutral. Additionally, this quantified the change in the level of self-efficacy. The descriptive analysis of the self-efficacy classification is reflected in Table II.

TABLE II: Students' self-efficacy trendline

Classification	Percentage
Increased	24.29%
Decreased	48.57%
Neutral	27.14%

Furthermore, the intercept parameter was used to measure each student's initial self-efficacy, at the beginning of the semester. In the next step, we applied a regression model to evaluate the correlation between these two parameters - slope (self-efficacy trend) and intercept (initial self-efficacy)- with the students' performance metric of final grades. In Table III, correlations among independent variables and the dependent variable are presented. As can be seen, there were statistically significant positive relationships between students' self-perception of self-efficacy and performance.

TABLE III: Regression model

Predictor	SE	Significance
Intercept	0.8097	0.2128
Initial self-efficacy (x1)	0.1877	0.0106*
Self-efficacy trend (x2)	1.1636	0.0018**

N = 70 and Adjusted R2 = 0.12

As mentioned earlier, a regression analysis was used to determine predictors of performance, and the self-efficacy

variables contributed to a significant difference in the final grade. The final model was where y is the dependent variable (final grades) and x1 and x2 are the independent variables (self-efficacy changes and initial self-efficacy). As we can see in Table III, these variables have a positive correlation with the final grade with $p < 0.01$. Our results showed the higher the students' self-confidence, the higher the final grades. Also, there was a significant correlation with initial self-confidence, meaning if a student started the semester with high self-efficacy, then it is likely for them to receive higher grades in comparison to those who started the semester with lower self-efficacy levels.

VI. Discussion and limitation

With one semester's worth of data collected for this current study, our findings underscore that self-efficacy is likely to play an important role in students' learning processes in the introductory computer science course, which is in alignment with previous studies in a similar context. Our results also reported increased, decreased and neutral changes in students' self-efficacy, which reinforced Kinnunen and Simon's work [17]. In the meantime, we observed that many of the students in the class had decreased self-efficacy. This could in part contribute to the high levels of students' attrition seen in computing fields. Through the current analysis, "loops" and "files, reading, writing" were identified as the hardest to understand which suggests additional efforts in those topics to help improve the passing rate for the course. We hope with the intervention implemented based on those student identified topics (including but not limited to extra lectures, lab assignments, in-class activities, etc), the students' perception of self-efficacy will increase along with the improved passing rate. Overall, these results provide a foundation for both the computing community and education researchers to look for more targeted teaching strategies to reduce barriers encountered by students in computing fields. Consideration of such strategies will be essential to our national efforts to improve self-efficacy and/or diversity in computing fields while providing benefits to all students.

This WIP study had some limitations. First, the participants did not have enough time to digest the new concepts they just covered in class when filling out the surveys. Therefore, every new concept they learned ended up being a "muddiest point." In future work for this research study, we plan on having only one survey per week. The surveys would consist of the concepts that the participants learned the week prior to when the survey is distributed. This would allow the participants to have more time to deliberate on which concept is truly their "muddiest point." Currently, students were provided a list of topics for them to choose from as their identified "muddiest point." The future work should also consider allowing students to supply their own suggestions for topics that are most challenging to them. Second, the analyzed results did not include the students who 1) dropped the course, 2) did the survey less than 3 times, or 3) opted out for not participating in the survey. The students who fell into either the first or third options stated above could be the result of perceived low or high self-efficacy. For example, a student who dropped the introductory programming course could be an indication for low self-efficacy. Students who opted out the survey could be an indicator for low or high self-efficacy as well. Furthermore, the relationships between students' identified "muddiest points" and their perceived self-efficacy through the course period would be worth exploring. In addition, the current analysis only focused on the performance outcome of final grade. It would be interesting to explore non-performance related aspects of self-efficacy in future work. Lastly, the current analysis was based on limited sample size. With a larger sample size, we will be able to test the generalizability of the results and illuminate the effects of many other effective measures.

References

[1] U.S. Bureau of Labor Statistics, "Computer and information technology occupations : Occupational outlook handbook," Sep 2019. [Online]. Available: https://www.bls.gov/ooh/computer-and-information-technology/home.htm

[2] Computing research Association, "Generation cs: Report on cs enrollment," Apr 2017. [Online]. Available: http://cra.org/data/Generation-CS/

[3] Florida International University, "Pass and fail report by instructors for cop courses," 2019.

[4] C. Watson and F. W. Li, "Failure rates in introductory programming revisited," in *Proceedings of the 2014 conference on Innovation & technology in computer science education*, 2014, pp. 39–44.

[5] B. C. Wilson and S. Shrock, "Contributing to success in an introductory computer science course: a study of twelve factors," *ACM SIGCSE Bulletin*, vol. 33, no. 1, pp. 184–188, 2001.

[6] J. Allert, "Learning style and factors contributing to success in an introductory computer science course," in *IEEE International Conference on Advanced Learning Technologies, 2004. Proceedings.* IEEE, 2004, pp. 385–389.

[7] H. G. Taylor and L. C. Mounfield, "Exploration of the relationship between prior computing experience and gender on success in college computer science," *Journal of educational computing research*, vol. 11, no. 4, pp. 291–306, 1994.

[8] E. D. Bunderson and M. E. Christensen, "An analysis of retention problems for female students in university computer science programs," *Journal of Research on Computing in Education*, vol. 28, no. 1, pp. 1–18, 1995.

[9] P. Byrne and G. Lyons, "The effect of student attributes on success in programming," in *Proceedings of the 6th annual conference on Innovation and technology in computer science education*, 2001, pp. 49–52.

[10] D. Hagan and S. Markham, "Does it help to have some programming experience before beginning a computing degree program?" in *Proceedings of the 5th annual SIGCSE/SIGCUE ITiCSEconference on Innovation and technology in computer science education*, 2000, pp. 25–28.

[11] C. Watson, F. W. Li, and J. L. Godwin, "No tests required: comparing traditional and dynamic predictors of programming success," in *Proceedings of the 45th ACM technical symposium on Computer science education*, 2014, pp. 469–474.

[12] A. Lishinski, A. Yadav, J. Good, and R. Enbody, "Learning to program: Gender differences and interactive effects of students' motivation, goals, and self-efficacy on performance," in *Proceedings of the 2016 ACM Conference on International Computing Education Research*, 2016, pp. 211–220.

[13] V. Ramalingam, D. LaBelle, and S. Wiedenbeck, "Self-efficacy and mental models in learning to program," in *Proceedings of the 9th annual SIGCSE conference on Innovation and technology in computer science education*, 2004, pp. 171–175.

[14] S. Wiedenbeck, "Factors affecting the success of non-majors in learning to program," in *Proceedings of the first international workshop on Computing education research*, 2005, pp. 13–24.

[15] K. Davidson, L. Larzon, and K. Ljunggren, "Self-efficacy in programming among sts students," *Retrieved August*, vol. 12, p. 2013, 2010.

[16] Ö. Korkmaz and H. Altun, "Adapting computer programming self-efficacy scale and engineering students' self-efficacy perceptions," *Participatory Educational Research*, vol. 1, no. 1, pp. 20–31, 2014.
[17] P. Kinnunen and B. Simon, "Cs majors' self-efficacy perceptions in cs1: results in light of social cognitive theory," in *Proceedings of the seventh international workshop on Computing education research*, 2011, pp. 19–26.
[18] Kinnunen, Päivi and Simon, Beth, "Experiencing programming assignments in cs1: the emotional toll," in *Proceedings of the Sixth international workshop on Computing education research*, 2010, pp. 77–86.
[19] A. Carberry, S. Krause, C. Ankeny, and C. Waters, ""unmuddying" course content using muddiest point reflections," in *2013 IEEE Frontiers in Education Conference (FIE)*. IEEE, 2013, pp. 937–942.
[20] S. J. Krause, D. Baker, A. Carberry, T. Alford, C. Ankeny, M. Koretsky, B. Brooks, C. Waters, B. Gibbons, S. Maass *et al.*, "Characterizing and addressing student learning issues and misconceptions (slim) with muddiest point reflections and fast formative feedback," in *Proceedings of the 121st American Society for Engineering Education Conference*.
[21] C. Waters, S. J. Krause, J. Callahan, B. Dupen, M. B. Vollaro, and P. Weeks, "Revealing student misconceptions and instructor blind spots with muddiest point formative feedback," 2016.
[22] D. B. King, "Using clickers to identify the muddiest points in large chemistry classes," *Journal of chemical education*, vol. 88, no. 11, pp. 1485–1488, 2011.
[23] S. J. Krause, "Muddiest point formative feedback in core materials classes with youtube, blackboard, class warm-ups and word clouds," *age*, vol. 23, p. 1, 2013.
[24] A. Bandura, "Social foundations of thought and action," *Englewood Cliffs, NJ*, vol. 1986, 1986.
[25] S. Yasar, *Discourse in freshman engineering teams: The relationship between verbal persuasions, self-efficacy, and achievement*. Arizona State University, 2008.
[26] T. A. Angelo and K. P. Cross, *Classroom assessment techniques*. Jossey Bass Wiley, 2012.
[27] L. Zahedi, S. Lunn, S. Pouyanfar, M. Ross, and M. Ohland, "Leveraging machine learning techniques to analyze persistence in undergraduate computing programs," 06 2020.