6-9-2022

A Qualitative Study of Emotions Experienced by First-year Engineering Students during Programming Tasks

Zahra Atiq
Ohio State University - Main Campus, atiq.2@osu.edu

Michael C. Loui
Purdue University, mloui@purdue.edu

Atiq, Zahra and Loui, Michael C., "A Qualitative Study of Emotions Experienced by First-year Engineering Students during Programming Tasks" (2022). *School of Engineering Education Faculty Publications.* Paper 75.
http://dx.doi.org/10.1145/3507696

A Qualitative Study of Emotions Experienced by First-year Engineering Students during Programming Tasks

ZAHRA ATIQ, Department of Computer Science and Engineering, The Ohio State University, Columbus, OH
MICHAEL C. LOUI, Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, Urbana, IL

In introductory computer programming courses, students experience a range of emotions. Students often experience anxiety and frustration when they encounter difficulties in writing programs. Continued frustration can discourage students from pursuing engineering and computing careers. Although prior research has shown how emotions affect students' motivation and learning, little is known about students' emotions in programming courses. In this qualitative study of first-year engineering students taking an introductory programming course, we examined the emotions that these students experienced during programming tasks and the reasons for experiencing those emotions. Our study was grounded in the control-value theory of achievement emotions. Each research participant came to two laboratory sessions: a programming session and a retrospective think-aloud interview session. In the programming session, each participant worked individually on programming problems. We collected screen capture, biometrics, and survey responses. In the interview session, each participant watched a video of their actions during the programming session. After every 2 minutes of viewing, the participants reported the emotions that they had experienced during this 2-minute period. We performed a thematic analysis of the interview data. Our results indicate that the participants experienced frustration most frequently. Sometimes they experienced multiple emotions. For example, one participant felt annoyed because she had made a mistake, but she felt joy and pride when she fixed the mistake. To promote student learning, educators should take students' emotions into account in the design of curriculum and pedagogy for introductory programming courses.

CCS Concepts: • **Social and professional topics → CS1**;

Additional Key Words and Phrases: Emotions, control-value theory of achievement emotions, computer programming, engineering

ACM Reference format:
Zahra Atiq and Michael C. Loui. 2022. A Qualitative Study of Emotions Experienced by First-year Engineering Students during Programming Tasks. *ACM Trans. Comput. Educ.* 22, 3, Article 32 (June 2022), 27 pages.
https://doi.org/10.1145/3507696

This work is supported in part by the Dale and Suzi Gallagher Professorship in Engineering Education and Bilsland Dissertation Fellowship at Purdue University.

Authors' addresses: Z. Atiq, Department of Computer Science and Engineering, The Ohio State University, Room 297, Dreese Laboratories 2015 Neil Avenue, Colubmus OH, 43210; email: atiq.2@osu.edu; M. C. Loui, Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, Coordinated Science Laboratories 1308 W Main St, Urbaba IL, 61801; email: loui@illinois.edu.

1946-6226/2022/06-ART32
https://doi.org/10.1145/3507696

1 INTRODUCTION

Since computer programming skills are essential in many careers in **STEM (Science, Technology, Engineering, and Mathematics)**, many undergraduate STEM programs require an introductory course in computer programming. In particular, undergraduate engineering students typically take an introductory programming course in their first year. In these courses, students often find that learning to write computer programs is difficult [1]. To be successful, students need patience and persistence [2]. In addition, students may encounter difficulties because they have inaccurate mental models of computer programs [1, 3]. As a consequence of these difficulties, students may experience negative emotions, and their responses to these emotions can diminish their learning and academic performance [4, 5]. Students who struggle in a programming course may leave engineering altogether [6, 7].

Prior research studies have examined how emotions affect students' learning and motivation [8–11], but little is known about students' emotions in computer programming and the specific precursors of those emotions. In this article, we address two research questions about the emotions that students experience during programming tasks:

(1) What emotions do first-year engineering students experience during a computer programming task?
(2) What reasons do these students describe for experiencing different emotions?

This article reports part of a larger study that uses the **control-value theory (CVT)** of achievement emotions and adopts a multi-modal approach to collecting data about students' emotions. By contrast, as explained in the literature review below, previous studies of emotions in computer programming have lacked a theoretical grounding or have gathered only one kind of data. CVT has been used extensively in studies of student emotions in other STEM fields [12, 13]; it has not been applied in computer programming. Although other theories have emotions as a component (e.g., Bandura's Social Cognitive Theory) [14], CVT specifically focuses on emotions in academic settings. Hence, it is an appropriate choice of theory for this study.

Because emotions are complex phenomena, previous researchers have advocated multi-modal approaches to data collection [4, 15], which we have adopted. Researchers have also called for investigations of emotions to take an interdisciplinary approach [4]. Our interdisciplinary study is grounded in evidence, theories, and methods from multiple fields, including computer science, computing education, engineering, engineering education psychology, neuroscience, and affective computing.

Our study offers implications for both research and practice in computing education. For researchers, this study demonstrates how qualitative data provides a holistic understanding of students' emotions. For computing instructors, the results of this study can inform the design of curriculum and pedagogy. By understanding what emotions students experience and the reasons for those emotions, instructors can help students develop self-regulation strategies for coping with those emotions. Instructors can account for students' emotions by structuring programming activities and choosing course practices to promote student motivation and persistence through difficulties [16].

2 THEORETICAL FRAMEWORK

2.1 What Are Emotions and Achievement Emotions?

Emotions are multi-componential psychological processes that include different affective, cognitive, motivational, expressive, and physiological components [17]. To understand each component, consider the example of a student who is anxious before performing a programming task. The

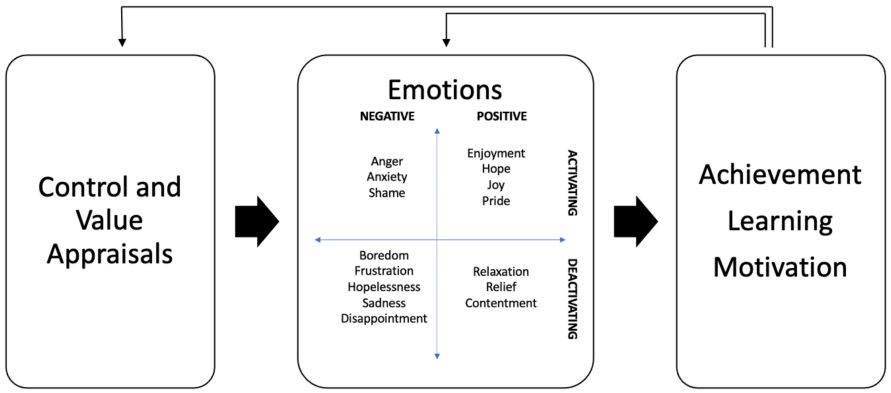


Fig. 1. Structure of the control-value theory of achievement emotions. Adapted from [19].

affective component is the subjective emotional feeling that the student experiences (e.g., feeling nervous), the cognitive component deals with the emotion-specific thoughts (e.g., worry), the motivational component deals with the students' behavioral responses (e.g., avoidance behavior), the expressive component is how the emotions are expressed (e.g., facial expressions), and the physiological component deals with physiological changes in the body caused by emotions (e.g., changes in skin conductance).

Most of the literature about emotions uses the terms "moods" and "emotions" interchangeably [18]. However, there is a difference between the two. While moods are longer lasting and may not have a particular referent, emotions consist of short and possibly intense episodes in response to a stimulus [19]. A person's situation and history play a significant role in the emotions they may feel. Two individuals may experience different emotions while facing the same situation, because of various personal histories and cultural differences [5]. Emotions in education are crucial for academic achievement, psychological well-being, motivation, beliefs in self-concept, and self-efficacy of both teachers and students. Hence, it is important to understand what emotions are and how they affect students in so many different ways [19].

2.2 Control-Value Theory of Achievement Emotions

CVT provides an integrated framework for studying emotions in an academic setting. Hence, this theory was used primarily to design this study, to interpret data, and to discuss the findings. According to CVT, achievement emotions link directly to achievement activities or achievement outcomes [19]. Pekrun describes achievement emotions in a three-dimensional taxonomy. The three dimensions of the taxonomy are object focus, activation, and valence [19]. The object focus can be either activities or the outcomes. When studying emotions, the focus should be placed not only on the activity the students engage in but also on the outcome of the activity. The valence of emotions can be either positive/pleasant or negative/unpleasant. Activation refers to the emotional arousal that a person experiences physiologically in response to a particular emotion. Both positive and negative emotions can be activating (happiness, hope, pride, anxiety, anger, shame), or deactivating (relief, hopelessness, boredom, satisfaction) [19]. Figure 1 shows the structure of CVT.

The central part of the theory revolves around the premise that students experience positive emotions when they feel in control of or value certain activities. Similarly, students experience negative emotions when they feel out of control or when they do not value certain activities. Control appraisals are the controllability of actions and their outcomes, and value appraisals are the subjective importance of activities and their outcomes [13, 19]. Different control and value

appraisals lead to different emotions. The CVT has three main components: appraisals, emotions, and outcomes. Appraisals influence emotions either directly or indirectly. Emotions in turn influence achievement. Achievement affects subsequent emotions and appraisals. Hence, achievement emotions, their appraisals, and their outcomes are linked together by reciprocal causation, resulting in a feedback loop (see Figure 1). This cyclic nature of different components in this theory also implies that emotions can be regulated by targeting any of the elements of the feedback loop [19].

According to the CVT, different control and value appraisals evoke different prospective outcome emotions, retrospective outcome emotions, and activity emotions [19]. Students experience prospective emotions like joy when there is high perceived control, but they experience hopelessness when there is lack of perceived control. Similarly, students experience hope when control is uncertain and they anticipate success. On the other hand, students experience anxiety when control is uncertain and they expect failure. Retrospective emotions such as disappointment and relief are assumed to depend on the perceived match between expectations and outcomes. The anticipated success that does not occur leads to disappointment, while anticipated failure that does not occur leads to relief. Furthermore, attributing success to oneself leads to pride, while attributing failure to oneself leads to shame. Attributing success to others leads to gratitude, while attributing failure to others leads to anger. Activity emotions are proposed to depend on an individual's competence appraisals and intrinsic values. For instance, competence and positive intrinsic qualities of action and activity lead to enjoyment of the activity, while lack of activity incentive values leads to boredom. Negative activity incentive values lead to anger and frustration.

3 LITERATURE REVIEW

3.1 Emotions in the Context of Computer Programming

Many universities offering STEM undergraduate programs require students to learn computer programming during their freshman year. In recent years, researchers have been studying the emotions of students while they learn how to program [8–11, 20–22]. In a series of studies, Kinnunen and Simon developed a framework that describes how computing majors experience emotions during each stage of tackling programming assignments in an introductory course [8, 9, 20, 21]. For instance, in the "Getting Started" stage, students who do not know how to proceed with solving a problem may feel confused or puzzled. On the other hand, in the "Succeeding" stage, students who successfully solve a problem may experience a feeling of relief and accomplishment. This work by Kinnunen and Simon studied affective states by interviewing the students four times over the course of the semester. However, they did not study the evolution of students' affective states during the semester. Building on the work of Kinnunen and Simon, Lishinski and colleagues quantitatively investigated the connection of emotions with learning outcomes and self-efficacy beliefs in an introductory computer programming course [10, 22]. According to this study, frustration was the most frequent emotion. Moreover, the strength of students' positive emotions and the absence of negative emotions positively correlated with their performance.

The original aim of Kinnunen and Simon's work was to identify the different stages that students go through to solve programming problems [8]. The authors identified student emotions as a by-product of their research. Hence, their research on emotions is not grounded in existing theories of emotions, even though many theories exist. Lishinski and colleagues used Kinnunen and Simon's work as a foundation for their work, but their research is also not grounded in any existing theory of emotion [10, 22]. Hence, these studies neglect to understand other emotions that are important for student learning and performance during programming tasks.

Mercedes and colleagues, on the other hand, investigated how affective states of novice programmers evolved over time and how emotions influenced students' performance [23–25].

Specifically, they collected data during five lab sessions over a 9-week period and analyzed the affective states of freshmen computer science students who were taking an introductory programming course. The researchers collected two types of data: student observations by trained observers and student activity logs. The affective states studied were boredom, confusion, delight, surprise, frustration, flow, and neutral. Mercedes and colleagues observed how student affective states changed over a period, concluding that confusion, flow, and neutrality varied significantly over time and variations in frustration were marginally significant [23]. The authors also found that students seemed to adapt easily to introductory topics like outputs and conditions but found constructs about object-oriented programming particularly confusing.

During the next stage of their research, Mercedes and colleagues employed student activity logs in addition to the observation data, to study the observable affective states that predict students' degree of achievement [24]. The results indicated that students who asked for help from the instructor about tasks related to the integrated development environment did not perform well on the midterm exam. In the third stage of the study, Mercedes and colleagues attempted to detect student frustration in a programming course using coarse-grained data such as student compilation logs [25]. The results indicated that students' level of frustration could be detected based on quantitative attributes of the student compilation logs (total number of errors, the number of compilations, pairs of consecutive compilations with the same error, and the average time between compilations). These research studies by Mercedes and colleagues aimed to develop intelligent tutors to help students learn programming. Hence, they used external observations and student activity logs to identify the student emotions and when students experience certain emotions. These studies did not directly interact with the students, and hence, they failed to capture students' descriptions of their own experiences. It would be beneficial to use personal accounts from students to design intelligent programming tutors, capable of identifying student emotions.

Emotions in programming courses have also been studied for non-computing majors. Bosch and D'Mello have studied the affective experience of psychology undergraduate students taking an introductory programming course [11]. Students in this study were tested individually in a 2-hour session, with each session consisting of three stages: scaffolded learning, fade-out, and retrospective affect judgment. The first two stages provided students the opportunity to work on basic programming problems with and without hints, respectively. After the completion of these two stages, student emotions were measured using a retrospective judgment protocol, which enabled students to reflect on their experience while watching the videos of their face and their on-screen activity. The learning environment kept track of student interaction events like key presses and system actions like providing feedback to students. The results of this study showed that engagement, confusion, frustration, boredom, and curiosity were the most frequently occurring affective states.

3.2 Summary and Critique of Prior Research

Most previous studies identified certain frequently occurring emotions like frustration [8, 10, 26] and patterns of emotions like confusion followed by frustration [11]. Most of these studies focused on the impact of negative emotions (e.g., frustration) on learning outcomes or self-efficacy. However, these studies do not identify the situations that trigger certain emotions in students. The current body of research also rarely explains positive emotions or a mix of multiple emotions in the context of programming. Moreover, previous studies do not describe in detail the events that trigger these emotions and how students responded to those emotions. By contrast, this study provides thick rich descriptions of how students experience both positive and negative emotions and the reasons they experienced those emotions. Finally, unlike previous studies in computing education, the findings of this study are theoretically grounded.

4 RESEARCH METHODOLOGY AND METHODS

4.1 Philosophical Underpinnings and Positionality

The purpose of this section is to promote transparency by candidly disclosing our relationship with the participants of this study, as potential sources of bias. Both authors were not involved in teaching or designing the programming course that we studied, and our only role was that of researchers. However, both authors have extensive experience teaching computer science and computer engineering, which requires extensive programming. Our motivation to design and conduct this study stems from our experiences, where we have regularly observed students' negative emotions about programming. Hence, to minimize any personal influence of our experiences in the research study, we have written memos during data collection. The memo writing helped reflect on our engagement with the data and helped analyze these data in a more neutral way [27].

4.2 Context

Purdue University is a large research university in the midwestern United States, with mostly full-time, residential, and traditional-aged undergraduate students. The **first-year engineering (FYE)** program at Purdue University serves about 2,500 new, first-year undergraduate engineering students every year. In 2018, the FYE program had an enrollment of 27% women and 9% international students. The FYE students take fundamental STEM and communications courses, including the sequence of two courses ENGR 131 and ENGR 132, which introduces students to concepts related to engineering design, data visualizations, computer programming, communications, and teamwork.

ENGR 132 focuses specifically on introducing programming skills using MATLAB to FYE students and hence is the focus of this study. ENGR 132 is offered year-round and the student population in each semester is different. Students taking ENGR 132 may take other programming courses in parallel, depending on which engineering major they want to pursue. Typically, students intending to major in computer engineering or mechanical engineering also take CS 159. CS 159 is offered by the Department of Computer Science, which is outside the College of Engineering.

ENGR 132 is a two-credit course (two 110-minute sessions per week), which is taught using active, blended, and project-based learning methodologies. ENGR 132 includes programming concepts like mathematical calculations, logical operators, if/else statements, while and for loops, nested loops, and basic data visualization techniques. Students attend lectures and work on problems in class using pair programming. Outside the class, they watch online modules and work on projects with their teams. Each section of ENGR 132 has up to 120 participants. The data collection took place immediately after the spring break. Students had learned nested loops in class just before the spring break. It is noteworthy that the second major exam for ENGR 132 took place during the data collection time period. Hence, some students may have been motivated to participate in this study to practice for the exam.

4.3 Selection Criteria

We used purposive criterion sampling to recruit participants for this study [28]. Purposive sampling ensures theoretical validity by capturing the diversity and multiple perspectives of participants [29, 30]. It is a common understanding that learning to program is a hard task for novices [1, 31–33], and because of this novices may experience a range of emotions while learning programming in an introductory programming class [30]. For this study, we considered novices who took ENGR 132 for the first time in spring 2018. Our selection criteria excluded three categories of participants. We excluded students who had had prior programming experience. In the context of this study, novices are participants who have not had any programming experience before taking

Table 1. Self-reported Demographic Information of the Study Participants

Pseudonym	Age	Gender	Race/Ethnicity	Pseudonym	Age	Gender	Race/Ethnicity
Andrew	19	Male	Black/African American	Jack	18	Male	Hispanic/Latinx
Anna	19	Female	White	John	18	Male	White
Becky	19	Female	White	Lilly	18	Female	White
Christina	18	Female	White	Mark	19	Male	White
Ella	19	Female	Asian	Martha	19	Female	White
Emily	18	Female	White	Rachel	19	Female	White
Erica	19	Female	White	Randy	18	Male	Hispanic/Latinx
George	19	Male	American Indian/Alaskan	Sarah	18	Female	Asian
Harris	19	Male	Asian				

ENGR 132, and they were not taking another programming course in parallel. We also excluded participants repeating ENGR 132 because they had had previous exposure to programming. Finally, we excluded international participants from our study. There are differences in how emotions are experienced across nations and cultures, for example, language and the ways in which emotions are expressed across cultures [34, 35]. Moreover, different cultures do not agree on what emotion is and many times there is no unified concept of emotion for certain experiences that the Western world lumps as emotional [36]. Hence, it was appropriate to exclude international students from the scope of this project.

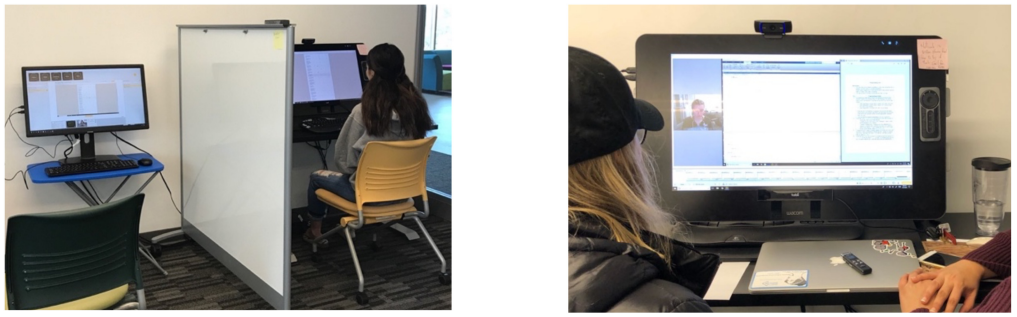
4.4 Recruitment and Participants

For selecting sample size in qualitative research, there are no concrete rules. However, a common practice is to select enough participants such that data saturation is reached [37]. Following this common practice, after approval from the human subject's protection office and conducting two pilot studies, we recruited 18 participants who were enrolled in ENGR 132 during spring 2018. According to the information from the background survey, all participants who were sent an invitation to participate in the study met the selection criteria. However, one student mentioned during data collection that she had taken CS 159 during fall 2017. Hence, she was removed from subsequent data analysis. All participants were given a $40 Amazon gift card as compensation for participating in the study. Participants' demographic information is provided in Table 1. Their demographic attributes largely reflect the diversity of the FYE population, with a slight overrepresentation of women. All participants were assigned pseudonyms for confidentiality purposes.

4.5 Research Design

This article is part of a larger study that collected different forms of multi-modal data. However, for the purpose of answering the research question addressed in this article, we used one primary source of data (retrospective think-aloud interviews). We also collected secondary data (video screen capture and eye gaze) to help participants recall their experience during the retrospective think-aloud interview.

4.5.1 Experimental Setup. For data collection, we used iMotions [38], a software platform that integrates and synchronizes different data sources (e.g., biomarkers, surveys, screen capture). There are two sides of the iMotions experimental setup (see Figures 2(a) and 2(b)). The right side



(a) (b)

Fig. 2. (a) Setup of the experiment. (b) Retrospective think-aloud interview with one participant.

is for the participant being tested, which includes the workstation consisting of a computer with a frontal camera and eye tracker. The left side is a live view that enables the researcher to view participants' activity in real time and make necessary changes, if needed (e.g., to prompt the participant if their face was out of the range of the camera).

4.5.2 Data Collection. For research on emotions, simultaneously collecting different kinds of data increases the validity and accuracy of the study [15]. We collected both quantitative and qualitative data. However, the data analysis methods are primarily qualitative. We collected data in two rounds: (1) programming task and (2) retrospective think-aloud interview.

4.5.2.1 First Round (Programming Task). During the first round of data collection, the first author welcomed the participant, introduced herself, and brought them to the room where the experiment was set up. She provided a detailed overview of the study to the participant and then handed them the consent form. Once they had read and signed the consent form, she connected a noninvasive device on one foot, which collected the participant's electrodermal activity.

Before the task started, baseline data for biometrics was collected. The participant filled out the pre-survey, in which they self-reported the prospective emotions during the programming task. Once they completed the pre-survey, the programming task started, and the participant was given 30 minutes to work on the problems. After completing the task, the participant filled out the post-survey, in which they self-reported the retrospective emotions experienced during the programming task. The findings from the pre- and post-surveys and biometric data will be discussed in other publications about the overarching study.

Programming Task: The programming task consisted of four programming problems in MATLAB. We carefully selected these problems by considering the difficulty level of the problems so that most participants could solve these problems during a 30-minute period. Moreover, these problems were selected because they cover a range of programming constructs that participants had already studied in their ENGR 132 course. Appendix A shows the four programming problems on the programming task. Participants had the option to work on the problems in any order and could also switch back and forth between problems. Unless specified, participants could use a predefined MATLAB function to solve a problem or they could write code. For example, participants could use the built-in cumsum function to compute the running sum or they could write code from scratch. Participants saved their code scripts (m files) for every problem in a folder on the desktop. In ENGR 132, participants normally work on assignments in groups and seek help from their peers and from the teaching assistants. However, for this programming task, they worked alone, but they could use the available resources (MATLAB or online help).

4.5.2.2 Second Round (Retrospective Think-Aloud Interview). During the second round of data collection, which took place 3 to 7 days after the programming task, the participants returned for a retrospective think-aloud interview. The first author conducted the retrospective think-aloud interviews. She replayed the video of participants' programming task for 2-minute intervals, during which participants verbalized their thoughts. After each interval, she paused the video and asked these questions: (1) What emotion do you think you were experiencing here? (2) What do you think are some reasons for experiencing these emotions? Figure 2(b) shows a retrospective think-aloud interview with one student.

We provided participants with a sheet of paper with a list of emotions derived from Pekrun's taxonomy of academic emotions [19]; the list had a range of positive and negative emotions that students frequently experience in academic settings: hope, joy, pride, relief, gratitude, anger, anxiety, shame, boredom, frustration, hopelessness, disappointment, contentment, and relaxation. The list of emotions was provided to help participants assign appropriate labels to the emotions that they thought they had experienced during the programming task [11]. Participants were also given the "other" option to articulate an emotion not mentioned on the list and to provide a definition for that emotion. Often participants reported feeling neutral, that they were not feeling any type of positive or negative emotions [11].

Retrospective Think-aloud Interview: A retrospective think-aloud interview is a verbalization technique in which the participants verbalize their thoughts while watching a recording of their performance on a task that they had previously performed [39]. We chose a retrospective think-aloud interview instead of a concurrent think-aloud interview for two reasons. The programming tasks required full concentration of the participant; hence, concurrently verbalizing their thoughts may have distracted them. Moreover, this research aims to assess emotions; a distraction from simultaneously working on programming problems and verbalizing thoughts may have influenced their emotions. One drawback of a retrospective think-aloud interview is that the participants may not retain a strong memory of their actions during the programming task. To overcome this drawback and to aid participants' memory, we collected three types of secondary data (video of screen capture, eye-gaze data, and facial expressions) to help participants recall their experience on the programming task during the retrospective think-aloud interview [40].

4.5.3 Data Analysis. We analyzed the retrospective think-aloud interviews using a mix of deductive and inductive thematic analysis [28, 41, 42]. The deductive thematic analysis was driven by the theoretical framework and literature that guided the study [42], whereas in inductive thematic analysis, themes emerged from these data [28]. The steps for thematic analysis are explained in the following sections.

Getting Familiar with the Data: Audio files for all participants were transcribed and then transferred to a qualitative data analysis tool (NVivo). The first author read all transcripts to check the accuracy of the transcripts and to become familiar with the depth and breadth of these data [42].

Development and Reliability of the Codebook: We developed the initial codebook by using the CVT [13] and Kinnunen and Simon's process model [8]. The codebook kept evolving as the analysis progressed. To improve the reliability of the coding process, we requested a fellow researcher to code four transcripts with the first author. Both researchers coded the transcripts independently, after which they came together to consolidate their respective codes.

Assigning or Generating Initial Codes: Once the codebook was refined using codes from the first four transcripts, the first author assigned codes to the remaining 13 transcripts independently. During this process, if the code was available in the codebook, it was assigned to the relevant excerpt; otherwise, a new code was generated and then assigned to an excerpt. Following this process, some new codes were generated and added to the codebook.

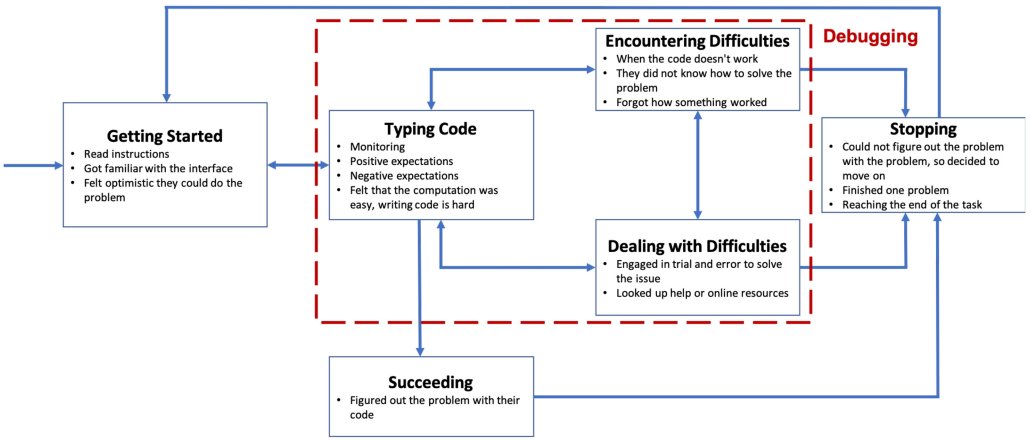


Fig. 3. Process model that participants follow while working on the programming task.

Categorization: For the first round of categorization, we used CVT [13] and Kinnunen and Simon's framework [8, 21]. Kinnunen and Simon's process model has six stages: (1) Getting Started, (2) Encountering Difficulties, (3) Dealing with Difficulties, (4) Succeeding, (5) Submitting, and (6) Stopping. However, participants in the current study were not required to submit their MATLAB scripts, so there is no Submitting stage in the adapted process model for this study. Moreover, data from the current research captures student behaviors and emotions while they typed code. Hence, the adapted process model has an additional stage called Typing Code. The six stages of the adapted model are (1) Getting Started, (2) Typing Code, (3) Encountering Difficulties, (4) Dealing with Difficulties, (5) Succeeding, and (6) Stopping. Figure 3 shows the adapted process model, which will be discussed in depth in the Findings section.

Generating Themes and Writing Memos: Once we had categorized codes, we further categorized codes within each category into sub-categories (or themes), based on similar characteristics. This process was inductive in nature, where the themes emerged from the codes [42]. Throughout the duration of the study, the first author wrote memos, which were used during data analysis and to write a compelling argument to answer the research questions.

4.6 Quality of Research

We used the framework proposed by Walther et al. to inform the different methodological decisions made to ensure the quality of our research study [29]. This framework provides guidelines for "making the data" during the collection phase and "handling the data" during the analysis phase. We used purposive criterion sampling to recruit participants for this study [28]. For handling data, we did not conduct the analysis in isolation. We kept on discussing findings with our peers. We wrote memos during each step of data collection and analysis. Communicative validation works to establish interpretation with the internal and external customers of the research [29]. While analyzing these data, we checked the reliability of the code book with another researcher for communicative validation. Process reliability is the means by which the research process is made independent from random influences [29]. To collect data in a dependable way, we conducted two pilot studies to check the appropriateness of the task difficulty and for the phrasing of the interview questions. During the pilot studies, we also refined the protocol for the interviews. Moreover, we embedded secondary data collection (biometrics and screen capture) to aid student recall during the retrospective think-aloud interview. We also provided participants with a list of emotions so that they could accurately label the emotions that they thought they had experienced.

5 FINDINGS

Figure 3 shows the process model that participants followed while working on the programming task. This process model is a result of multiple rounds of deductive and inductive analysis of our qualitative data (as described in Section 4.5.3). The double-sided arrows between two stages in Figure 3 indicate that participants could move back and forth between those stages. For instance, while encountering errors, participants could go back and type something to fix the error.

In all, 515 instances of different emotions were coded in the qualitative data. The predominant emotions were frustration (123 instances—24%), anxiety (52 instances—10%), confusion (49 instances—9.5%), neutral (46 instances—9%), and relief (45 instances—9%). All 17 participants reported feeling frustrated, while 15 participants reported feeling anxious at various times during the programming task. Additionally, participants also reported some emotions that did not appear on the list provided to the participants. These emotions were annoyance, nervousness, overwhelmed, and sadness. However, these emotions did not occur too frequently in these data. The following sections discuss in detail the emotions participants experienced during each stage of the process model and the reasons they provided for experiencing those emotions.

5.1 Getting Started

Participants entered the "Getting Started" stage when they started working on a new problem. The following sections describe participants' emotions and the reasons for those emotions during the "Getting Started" stage.

5.1.1 Reading the Instructions. In this stage, while reading the problems on the programming task, participants experienced positive deactivating emotions like relief, as evidenced from an excerpt by Rachel:

Alright my first thought, I was just kind of going over the task. It was a lot shorter than I thought it was going to be, which I felt pretty relieved about. I thought it was going to be a lot more like what our homework assignments are. ***(Rachel, Problem 1)***

According to the CVT, a deactivating emotion like relief is induced when failure has been averted [13]. Rachel was anticipating failure and because of this she was expecting to encounter challenges that would hinder her success on the programming task. However, after reading the instructions, she realized that the problems were familiar or were shorter than her expectations, hence averting failure and experiencing relief.

5.1.2 Getting Familiar with the Interface. Some participants started working on the programming task by familiarizing themselves with the interface or by saving files in the right location, which introduced incidental challenges. These incidental challenges are not related to the programming task itself because they take up additional time from the task. During this stage, most participants experienced negative emotions, as evidenced by one example from Rachel:

I normally work on a Mac so just, I don't even know what this interface is so just working on it [. . .] it was a lot more blank than what I am used to [. . .], this was kind of empty [referring to the blank MATLAB script], kind of overwhelming, especially with the scripts we were supposed to make. They [referring to ENGR 132] have a whole header for us already so when you started the assignment, you already had words down, kind of helps you a little bit [. . .]. ***(Rachel, Problem 1)***

Rachel encountered two unfamiliar situations (Windows interface and lack of code templates) and because of this she was uncertain about the outcome of the programming task. Hence, she was overwhelmed because she was feeling anxious. Rachel's experience corroborates with the

CVT, which suggests that anxiety is induced when participants are uncertain about the outcome of a certain event [13].

5.1.3 Optimistic They Could Do the Problem. Once the participants had selected a problem, they evaluated if they would be successful at writing code for that problem. If participants felt confident about their ability to write code for the chosen problem, they experienced positive emotions like joy, as shown in the following example by Ella:

So the first one was pretty straightforward. I realized that I could just write, um, if statements form, which should be pretty easy and it was something that I had just done, so I was pretty happy about [. . .] I would say joy because it's um, yeah. And I feel like I could probably enjoy writing the code too. ***(Ella, Problem 1)***

Ella's experience conforms with the CVT, which suggests that positive emotions like joy are experienced when there is high probability of success and non-occurrence of failure [13]. In addition to experiencing positive emotions, data from Ella exhibits confidence about their abilities to successfully solve some problems on the programming task. Confidence in their abilities could be explained by the CVT, which states that participants exhibit action-control expectancies when they are confident about their ability to successfully perform an action [19].

5.2 Typing Code

The following sub-sections explain participants' behaviors and expectations while "Typing Code" and the emotions they experienced during this stage or anticipated experiencing in the subsequent stages.

5.2.1 Monitoring. Some participants expected to encounter errors in their code and hence decided to double-check everything as they typed. These participants adopted preemptive behavior, where they knew they had the propensity to make mistakes. To avoid experiencing negative emotions, they made sure they typed correct code the first time, instead of making errors and then fixing them. Examples from Randy and Rachel explained how they double-checked everything while writing code:

I feel like I'll get to the end of it and something will be wrong and I can't find out what it is, it just gives me a lot of anxiety. I mean the only thing I can do about it is power through and double check everything which is why I go back and I take a really long time to type it because I want to make sure it's right the first time and I don't have to go back and figure out what's wrong. ***(Randy, Problem 2)***

I just, with coding it's very temperamental, like it's very picky and has to be exact and so if you don't know exactly how it's supposed to be done and the computer can't understand what you are trying to do, it's frustrating because you can't get too much feedback from it. You just kind of have to keep working through it, it might be something simple like an extra period when there isn't supposed to be or the order of something is wrong or might be a spelling error you skipped over so that frustration comes out a lot when I am coding. ***(Rachel, Problem 1)***

Although both Randy and Rachel were cautious and double-checked everything as they wrote code, both of them had different reasons for adopting these behaviors. While Randy attributed his cautious coding style to his own propensity to make mistakes, Rachel attributed her caution to the fact that the syntax of the code had to be precise before it is executed. Both participants did not experience frustration and anxiety while typing their code, but they were expecting to experience negative emotions in case their code did not work. Hence, to avoid feeling those emotions, they took necessary precautions.

5.2.2 Positive Expectations. Some participants had positive expectations; hence, they felt confident their work would yield the correct outcomes. Thus, they felt proud of their abilities and in some cases relief, as shown by John's example:

I think pride and relief because I correctly translated the math language to the MATLAB language. Okay. Um, so that felt good to have a working equation. And right now I'm plotting which is something I'm fairly comfortable with. ***(John, Problem 4)***

John felt proud because he had successfully converted a mathematical equation into MATLAB code, and he felt confident that his code would execute successfully. According to the CVT, pride is experienced when participants achieve something they had perceived as hard or unattainable [13, 43]. For these participants, writing code was a hard task, but once they had written the code, they felt proud of their accomplishment. These participants had not yet executed the code and hence, they had not yet seen the output of the code. However, they had positive expectations and felt confident that their code would execute correctly.

5.2.3 Negative Expectations. Some participants had negative expectations from what they had typed, and hence they thought their code would be faulty. In such a case, participants felt anxious, as shown in an example by Jack:

At this point I was nervous and anxious, so I had to test my code to see if it worked [. . .] I kinda was expecting my code was not going to work but I didn't know how begin to fix the issue. So, I decided to go back to the browser. ***(Jack, Problem 2)***

According to the CVT, anxiety is induced if participants focus on experiencing failure [13]. Even though participants had not yet executed the code, some participants felt anxious because they were anticipating their code to be faulty.

5.2.4 Computation Is Easy, Writing Code Is Hard. Sometimes participants perceived they could find a solution of a problem in their minds but faced difficulty converting their solution into syntax. In such a case, they experienced frustration because they thought they knew what to do but were unable to implement their thoughts correctly. One example is of Rachel, who explained her feeling of frustration when she could not convert her thoughts into MATLAB code:

Oh, but this last problem on problem 2, I knew what it was saying, I would know how to do it on a piece of paper, it's just that doing it on a computer, putting in these commands so that the computer understands you, I couldn't do it. [. . .]. It's frustrating knowing how to do something without coding but then not knowing how to do it with coding because it seems so easy but it's just figuring it out, it's not actually that the computations are difficult, it's finding the commands to do that or the ways to do it on MATLAB. I don't ever remember doing like that in class which again is why I felt frustrated. ***(Rachel, Problem 2)***

Here Rachel used the phrase "piece of paper" metaphorically to refer to the computation she did in her mind. However, she was unable to write code corresponding to the solution in her mind, and consequently, she felt frustrated. To be able to convert a solution into code, participants construct mental models, called "notional machines" in the context of computer science [3]. However, novice participants lack the detailed mental models needed to convert problem statement into code [1]. This difficulty of converting problem statement into code leads participants to feel frustrated. According to the CVT, failure induces frustration [13]. Since Rachel failed to translate the solution in her mind to executable code, she felt frustrated. Even though Rachel thought she could do the problem in her mind, she moved on to problem 3 as soon as she realized she could not write code

for running sum. Rachel's example also shows that participants did not have to struggle on one problem for too long to feel frustrated.

5.3 Encountering Difficulties

In the "Encountering Difficulties" stage, participants encountered numerous challenges, which are discussed in the following sections.

5.3.1 When the Code Does Not Work. Most participants experienced negative emotions when their code did not execute successfully. Participants predominantly experienced frustration, but some also experienced anxiety. The following example by Becky shows her feeling of frustration and anxiety when her code did not execute successfully:

So here I was a little angry and a little frustrated 'cause of the same reasons as why is this not working, I do not know what's wrong with the program at this point, I don't know what I am doing wrong. I was starting to blame myself for getting it wrong, blaming MATLAB for getting it wrong [...] Also getting slightly anxious with the time limit, 'cause I knew there was a time limit and I had wasted a lot of time on this. ***(Becky, Problem 2)***

Becky experienced two emotions at the same time, due to different but related events. She encountered syntax errors in her code, which required time to fix. However, she did not have enough time left; hence, she felt anxious. These findings conform with the CVT, which suggests that anxiety is induced if participants focus on failure [13]. Becky's feeling of anxiety is not related to programming per se, but because she was running out of time on the programming task. She also felt frustrated because her code did not execute successfully. Her experience conforms with the CVT, which suggests that failure induces frustration [13].

5.3.2 They Did Not Know How to Solve the Problem. Some participants experienced negative emotions when they realized they did not know how to solve a certain problem. The negative emotions included frustration and shame. It must be noted that the same situation (not knowing how to solve a problem) may induce different emotions in different participants [43], as shown in the following examples from Rachel and Randy:

I was kind of hoping that I would have had that "aha" moment, I would remember something, I would think of something, some way how to do it or I look back at it, I'd read it again and interpret it in a different way, which is why I moved to the next problem. Didn't let it get to me too much then, but I did feel frustrated that I didn't initially know how to do it. ***(Rachel, Problem 3)***

Kind of shameful that I didn't know how to do that. In general, like after this whole programming task, I just kind of felt a lot of shame because I felt like a lot of these problems were really easy compared to like what I'm normally given and I flat out didn't know how to do half of them so that was kind of frustrating and a little bit of shame. ***(Randy, Problem 3)***

Rachel experienced frustration when she realized she could not solve the problem. However, she hoped to figure out the solution later and hence moved to the next problem. Rachel's experience aligns with the CVT, which suggests that participants experience frustration when they encounter activities that are not sufficiently controllable [13, 44]. Randy felt ashamed after the programming task because he did not know how to do most problems on the task. According to the CVT, shame is experienced when failure is judged to be caused by oneself [13]. Randy attributed his failure on the task to himself. He explained that most of the problems on the task were easy, and even then he was unable to do them successfully, so he experienced shame.

5.3.3 Forgot How Something Worked. Some participants experienced negative emotions when they realized they had previously worked on something but forgot how to apply their previous learning during the programming task. Randy's example encapsulates his experience of feeling frustration and anger because he forgot the MATLAB function for exponent:

I couldn't remember that e was exp and then I tried e in the command window, saw it didn't work and still put it in anyway so yeah, a lot of frustration and kind of anger towards myself because I was angry at myself for not remembering what it was. Have I used e before? Yeah, absolutely. I'm realizing now that on the exam, I had forgot it too but it's fine so yeah, kind of anger and frustration in this moment. ***(Randy, Problem 4)***

Randy's experience could be explained by the CVT, which suggests that participants experience negative emotions like anger and frustration when they fail on the task they undertake [13]. Since he had forgotten how to use the exponential function in MATLAB, he hardcoded the constant value of exponential (e = 2.7). After hardcoding the value for exponential, Randy worked on fixing syntax errors in the code. When he executed the code, his code ran without any errors, but as soon as he executed the code, the time for the programming task finished.

5.4 Dealing with Difficulties

In the "Dealing with Difficulties" stage, most participants adopted some strategies to overcome the challenges they were facing. Interestingly, these data show that most participants persevered through the programming task and did not give up completely despite the challenges they faced. However, some participants abandoned individual problems, with the intention of returning to them later.

5.4.1 Engaged in Trial and Error to Overcome Errors. All participants who participated in the study encountered syntax errors while working on the programming task. However, different participants dealt with the errors differently. The most common strategy that the participants adopted was trial and error. While engaging in trial and error, participants went back and forth over their work and tried to identify errors in their code. This process induced negative emotions like frustration in participants, as shown in the following example by Becky:

So, at this point I made up a random vector to test [whether] it was MATLAB screwing up or if I was doing something wrong. So, I purposefully used negatives and positives so I could check what it was doing so yeah. So, then I got the same error in Y and then that's when I realized that I was doing something wrong. At this point, it was pretty neutral, slightly frustrated just 'cause it was not working. [. . .] Just using trial an error and then figuring out what was happening, so it was pretty neutral at this point. ***(Becky, Problem 2)***

To resolve the syntax errors in her code, Becky engaged in the process of trial and error. When she was unable to fix errors, Becky felt frustrated. According to the CVT, participants experience frustration when they experience failure [13]. Previous research suggests that most novices normally do not have the developed mental models to correct errors in their code [1]. Like novices, Becky probably lacked the mental models required to correct errors in her code, and hence she felt frustrated.

5.4.2 Looked at Help or Online Resources. Participants looked at help when they encountered syntax errors or if they did not know the syntax of a function. According to the instructions provided, participants could raise their hand and ask the researcher questions, but none of the participants took that option.

5.4.2.1 Successfully Overcame Errors Using Help. When participants encountered syntax errors, they looked at either local MATLAB or online help resources. Many times, participants were able to resolve errors by looking at help. In this case, participants experienced joy and relief, as explained in the following example excerpts by Ella:

Once I googled it and like the first thing that came up, like it said, like cumulative sum, I'm like, oh, that makes sense. And then I'm sure I felt relief and joy. I was like, oh good. Then I'm. So, I clicked on it and then I read through it briefly and then it was exactly what I wanted it to. So, I put in my code and it worked. ***(Ella, Problem 2)***

According to the CVT, joy is experienced when participants are in control of the activity [13], and relief is experienced when failure is averted [13]. Ella averted failure by searching for a correct solution online. Interestingly, Ella experienced two emotions at the same time: joy and relief. According to the CVT, joy and relief are experienced when there is a high probability of success and low probability of failure [13, 44].

5.4.2.2 Could Not Resolve Errors Even after Looking at Help. Some participants were unable to resolve syntax errors in their code, even after consulting MATLAB or online help. These participants experienced shame, as shown by an excerpt from Christina:

I actually tried looking up help in MATLAB and I found results but I couldn't really figure it out from that so that was when I experienced a little shame because you would think, if I do help and you get a sample code, you can figure it out but I just, for whatever reason, didn't get it. And then jumped to hard coding it which was really bad programming practice was but I guess it made me feel a little better because they give you the vector so you know which values you should get. ***(Christina, Problem 2)***

According to the CVT, shame is induced when failure is judged to be caused by oneself [13]. Christina felt that she should have been able to resolve errors in her code, but she was unable to. She attributed the failure to herself and hence experienced shame. However, Christina did not give up even though she could not find a solution online. She resorted to hardcoding the function in her code, expecting to come back if she found a better solution.

5.5 Succeeding

Participants reached the "Succeeding" stage when they had successfully finished a problem on the programming task. After encountering and dealing with difficulties, many participants successfully resolved the errors. Once they realized their code executed successfully, they experienced positive emotions like relief and pride. The following example except by Anna explained the emotions they experienced when they successfully fixed the problem with their code:

I was proud of myself because I had successfully figured out how to do something that I didn't know how to do, that always gives into pride and almost, something like relief, faith in my abilities to do the coding. ***(Anna, Problem 2)***

Anna's experience corroborates with the CVT, which explains that participants experience positive emotions when they have high control of the activity [13, 44]. Moreover, the CVT explains that participants experience relief when they are able to prevent failure [13]. Anna experienced relief because she was able to successfully do something she did not know; that is, she figured out how to calculate the running sum.

5.6 Stopping

During the "Stopping" stage, participants encountered different types of events resulting in different types of emotions, which are discussed in the following sections.

5.6.1 Could Not Understand the Problem, so Decided to Move On. When participants were unable to successfully complete one problem, they decided to move on to the next problem, instead of spending time trying to figure out a solution for the problem. In this case, participants experienced frustration and hopelessness, as described by an example excerpt from Jack:

I remember thinking that problem 3 is just hopeless. I am going to attempt it last if I ever get to it. After reading problem 3 thoroughly I decided that I don't think I can do this, so I won't attempt it until I solve the other three. ***(Jack, Problem 3)***

Jack experienced hopelessness because he realized he could not solve problem 3. Jack's experience aligns with the CVT, which suggests that hopelessness is experienced when participants have low expectancy of success [13]. Since Jack was not expecting to succeed on a problem, he felt hopeless and decided to move on to another problem, expecting to come back to it later.

5.6.2 Finished a Problem. When participants successfully finished one problem, they felt relieved. They also felt proud of their capabilities. However, participants felt disappointed when they had spent too much time on one problem. These experiences are explained by the excerpts from Sarah:

Here I felt a little relief because I finished the first problem. I didn't really know how the time had gone by, but I was just glad I was past the first problem. I was a little disappointed that I took so long but I kind of pushed that aside because moving on the next problem was like Okay, well I finished one, which means I can finish another one, I can do this now. ***(Sarah, Problem 1)***

According to the CVT, relief is experienced when failure is averted [7]. Sarah was relieved because she finished the first problem on the task, but she also added that she was disappointed because she had taken longer than expected on one problem. According to the CVT, the non-occurrence of expected success instigates disappointment [13]. Sarah expected to complete one problem in less time, but she spent almost the entire duration of the programming task on problem 1.

5.6.3 Reaching the End of the Task. Most participants experienced a mix of positive and negative emotions when the task ended after 30 minutes. Almost all participants were relieved that the task was over. However, participants experienced a variety of emotions at the end of the task, primarily frustration, shame, and relief. The following example excerpt by John show mixed emotions at the end of the task:

When the screen goes black. I suppose it's frustration because I never got to figure out that last problem. I'm a little bit of shame because I wasn't able to do it and I feel like I could have done it easily. Okay. Um, but also relief because the session was over. ***(John, Problem 3)***

Most participants felt relieved after the task finished. According to the CVT, when participants work on a relatively unimportant task, that is, they do not value the task, they experience relief when the task ends [13]. Since this task was low stakes for the participants, with no grade attached to their performance, participants did not value the task. Hence, most participants felt relieved when the task ended. At the same time, these participants experienced negative emotions because they were not satisfied with their performance on the task. John was frustrated at the end of the task because he was unable to successfully complete the last problem he was working on. John's experience corroborates with the CVT, which explains that participants experience frustration when they deal with difficult tasks or when they experience failure [13]. John also experienced shame because he was unable to successfully complete the problem. His experience corroborates

with the CVT, which suggests that participants experience shame when they think failure was caused by themselves [13].

6 DISCUSSION

6.1 Predominant Emotions Experienced by Participants

6.1.1 Frustration and Confusion. All participants reported feeling frustrated at various points during the programming task. The main events that triggered frustration were when participants got stuck somewhere (e.g., unable to overcome an error) or if they failed at something (e.g., they were unable to complete a problem successfully). These events mostly occurred during the "Encountering Difficulties" and "Dealing with Difficulties" stages, instigating frustration in participants. Participants experienced frustration in all stages except "Getting Started" and "Succeeding."

The current study shows that many times, participants had negative expectations about the outcome while they were writing code in the "Typing Code" stage. When their negative expectations became a reality, participants experienced frustration. The process model in [8] does not have a "Typing Code" stage because the authors could not observe participants as they typed code. The current study adds to the process model a new stage that explains student behaviors and emotions as they typed code.

During the "Encountering Difficulties" stage, participants experienced frustration when their code did not execute successfully, when they forgot something, or when they did not know how to solve a problem (e.g., problem 3 and running sum from problem 2). The literature suggests that learning to program-solve involves an interplay of complex cognitive activities, mental models, program design, understanding, modification, and debugging [1, 3]. Furthermore, novice programmers do not have the accurate mental models required to learn programming, and they use surface knowledge for problem-solving [1]. Hence, when the difficulty of a task exceeds the competence of the novice programmers, they may experience negative emotions, like frustration [44].

Most participants found problem 3 difficult, as it involved the use of nested loops. Moreover, nested loops were introduced to participants just before the spring break, while the data were collected immediately after the spring break. Participants may not have had enough practice with nested loops to be able to perform adequately on problem 3; hence, they experienced frustration when trying to write code or while resolving errors. Findings from this study agree with the research literature, which suggests that participants experience frustration when they encounter difficulties with their code [10, 11, 20, 21]. For instance, according to [21], participants experience negative emotions like frustration when they encounter errors suddenly when they least expected them (as if struck by lightning).

In the "Dealing with Difficulties" stage, some participants felt frustrated when dealing with errors in their code. To resolve these errors, participants engaged in trial and error or looked at help, like participants in a previous study [45]. Engaging in trial and error and looking up help are not mutually exclusive behaviors, and participants can switch between these two activities. Participants who switch between trial and error and help resources can become confused when they are unable to incorporate new information from help into existing code, thus reaching an impasse [46]. In short, participants' confusion leads to frustration when they fail to overcome the impasse [8, 11, 21].

In the "Stopping" stage, participants felt frustrated when they could not understand a problem or when the programming task ended and they had failed to solve the problem. For instance, John felt frustrated because he was unable to complete the last problem he was working on. John's experience aligns with the peak and end rule [41], which states that a person judges their overall experience based on how they felt at the peak of the experience and at the end of the experience.

Jack was working on a problem when the programming task ended, and hence he experienced frustration.

6.1.2 Anxiety. Some participants experienced anxiety in the "Getting Started," "Typing Code," and "Encountering Difficulties" stages mainly because of the negative expectations and uncertainty they had about problems on the programming task. However, in the subsequent stages, participants' anxiety dissipated and changed into either frustration and shame if their negative expectations were fulfilled or joy and relief if the negative expectations were averted. These findings align with the literature, which suggests that prospective anxiety is experienced when there is uncertainty and participants focus on anticipated failure [44].

6.1.3 Relief. Participants experienced relief primarily when they overcame a challenging situation or when their negative expectations were averted. Participants experienced relief in the "Getting Started," "Dealing with Difficulties," "Succeeding," and "Stopping" stages. However, participants did not experience relief in the "Typing Code" and "Encountering Difficulties" stages, because they were cognitively engaged in completing the task. Participants experienced relief in the "Getting Started" stage when they estimated that the problems were easier than they had expected a priori, based on their experiences from ENGR 132. However, for many participants, the feeling of relief was short lived because they struggled on various problems on the task. All participants experienced relief when the programming task ended (in the "Stopping" stage). The literature suggests that participants may have experienced relief because they had completed a cognitively engaging task and they could now focus on other things [21].

6.1.4 Pride and Shame. Findings from this study suggest that participants experienced pride when they attributed their success to their own efforts, and they experienced shame when they attributed the cause of failure to themselves. These findings align with the literature, which suggests that shame and pride are self-conscious emotions; when participants experience self-conscious emotions, they engage in self-evaluation to understand the reasons for their failure and success [47].

In this study, participants experienced pride in the "Typing Code," "Dealing with Difficulties," "Succeeding," and "Stopping" stages when they overcame certain challenges and attributed the cause of their success to their own efforts. For instance, some participants experienced pride when they successfully executed code on the first attempt. These findings align with the literature from computing education, which suggests that participants feel proud when they find a solution to a difficulty based on their own efforts [13]. It is noteworthy that participants experienced pride only after they achieved success in an endeavor. Hence, none of the participants experienced pride in the "Getting Started" stage because they had not achieved anything yet or in the "Encountering Difficulties" stage because they were faced with difficulties.

Participants in this study experienced shame in the "Encountering Difficulties," "Dealing with Difficulties," and "Stopping" stages. The literature suggests that shame is mostly experienced in evaluation settings where judgment by others is likely to happen [48]. Participants experienced shame most commonly when they were unable to overcome challenges on a seemingly easy problem or after looking up resources. In such cases participants attributed the cause of failure to themselves; hence, they experienced shame. For instance, Christina was unable to solve the running sum problem even after looking at help. She explained that she should have been able to solve the problem after looking at help, and hence she experienced shame. Our findings mostly align with the literature. For instance, Randy experienced shame in an evaluative setting because he was unable to solve problems he perceived as simple. However, unlike the literature, Randy is

not being judged by others, but he is being critical of himself. Moreover, shame is a context-specific emotion [48]. Randy felt ashamed of his performance after a timed task. He also thought the problems were easier than what he was used to, but because he was working on a timed task, he was unable to successfully complete those problems, hence experiencing shame. He may not experience shame in another context where he may not be dealing with a timed task. Although shame is a seldom-reported academic emotion in the context of programming, experiencing shame may seriously hamper student learning and motivation, particularly for performance-oriented participants because participants tend to blame themselves for their failure [47].

6.1.5 Neutral. Participants reported feeling neutral in the "Getting Started" stage, when participants were not engaged in tasks that involved high cognitive load. The CVT does not include neutral as an emotion. However, findings from this study conform with previous literature, which has identified neutral as an affective state where participants experienced no apparent emotion [11].

6.1.6 Multiple Emotions. Participants experienced multiple emotions simultaneously or in different patterns during the programming task. In some instances, participants reported a range of negative emotions (e.g., frustration, anxiety, and hopelessness), while in other instances, participants experienced multiple positive emotions like joy and relief. These fluctuations in emotions are important dynamic patterns of positive and negative emotions at a given point during the task and may be good or bad for learning [49].

Some participants also experienced both negative and positive emotions because of the same event. For instance, both Jack and Rachel moved to the next problem when they could not solve a problem, intending to return to it later. However, both participants experienced different emotions. While Rachel was hopeful she could solve the problem when she returned to it, Jack felt hopeless about the problem. These findings align with the literature, which suggests that similar experiences could be interpreted differently by different people, yielding different emotional reactions [43].

6.2 Credibility of the Study

In qualitative research, to justify the trustworthiness of the methods and results, studies should demonstrate credibility [28]. There were multiple instances in our data where participants connected experiences during the programming task with their actual experiences in their ENGR 132 course, providing evidence that students would have had similar experiences if they had done these problems in the classroom or in a test. The following excerpt by Anna indicates that participants who participated in the study connected their experiences in the study with their experiences in the ENGR 132 course:

I know I'm going to have to go and sit down and do it all over again. Just because I have one programming task over, I know that like right now, I have another one to go and sit and work on so it's just like a constant stressor, that you don't really get away from. ***(Anna)***

Anna did not feel relieved after the programming task ended, because she knew that programming was not over for her and that she would have to go and do more programming for her class. Although this task was part of a research study and was not related to the ENGR 132 course, she connected the experience of the programming task for this study with the programming she does for the course.

Moreover, Emily alluded to the fact that she treated the lab study as a test. Drawing a parallel, Emily mentioned using the same methods for solving problems that she used in an actual test for her ENGR 132 class, as shown in Emily's excerpt:

I mean, I still wanted to do well, and I still wanted to figure it out. It was an extremely low stress, low pressure thing but I treated it as a test in the way like, my methods for solving everything were similar. ***(Emily)***

Although Emily used the same problem-solving strategies as she used in ENGR 132, the lab study differed from an actual test because there was no grade assigned for the lab study. Hence, findings of this study are transferrable only to a certain extent to an actual test-like setting.

6.3 Transferability of Findings and Future Directions

Transferability of findings is an aspect of trustworthiness of a qualitative study [29]. In the following sections, we will discuss two types of transferability.

Transferability across Different Programming Environments. MATLAB was used in the current study because MATLAB was used in ENGR 132, the course from which we recruited participants. Since MATLAB differs significantly from programming languages like C or Python [50], findings from this study might not be transferable to other programming languages. For instance, MATLAB is interpreted, which means that it executes code line by line. In contrast, programming languages like C and Java are compiled, which means that the code executes only after the compiler has ensured that there are no syntax errors in the code. More research needs to be conducted in different contexts (e.g., lecture, homework assignments, and different languages like C or Python) to understand student emotions in those contexts.

Transferability across Majors and Institutions. Findings from this study may be transferable to a certain extent to participants from other majors and institutions. The following excerpt by George explains potential differences between engineering and computing participants and reasons their emotional experiences may vary:

Yeah, I mean I knew that you are not looking for people who are CS majors and have done coding all through high school and you know, [. . .] So, I didn't have this real high expectations going in and you know, I was able to finish quite a bit. Didn't feel bad about any of that. ***(George)***

It is interesting that George perceives that CS majors are expected to do well in programming, while engineering majors are not. It is probably a widespread belief among engineering students that programming is meant for CS majors, and because of this they are supposed to be very good at it. It is established that students from different majors and institutions experience negative emotions while working on programming [8, 10, 21, 51]. However, it is important to note that students in different majors may have different motivations to take programming course(s). Additionally, different introductory courses might be taught with different instructors, different examples, and different pedagogies. Hence, different students may experience emotions differently. It is thus worth investigating the different emotions that students from different majors and institutions experience.

Although the data collection was multi-modal, this article only provides findings from the qualitative data, that is, the retrospective think-aloud interview. Currently, the authors are in the process of preparing two other manuscripts that employ multi-modal data collected for this study. The first manuscript investigates students' self-regulation behaviors and emotions as they work on the programming task. The second focuses on discussing the methodology of the study focusing specifically on triangulating multiple sources of data (self-report and biometric) to see where these data align with qualitative data and where these data contradict.

The team is conducting a follow-up study on the interplay of emotions and self-efficacy beliefs during programming problems. This study aims to further disaggregate based on factors such as gender and ethnicity to understand how students experience emotions and how their self-efficacy

beliefs change as a result. Pride and shame are rarely studied in the context of learning programming. However, our findings point to the fact that these emotions are important for students' motivation and performance. Furthermore, shame may also be short-lived and may be replaced by pride once the participant persists and converts their failure to a success. Hence, it may be worthwhile to conduct an in-depth study to understand students' pride and shame while learning programming.

7 LIMITATIONS OF THE STUDY

Limitations in the Methodology and Methods. It is generally understood that the controlled laboratory setting is less authentic than higher-stakes activities student engage with in a course, e.g., tests, assignments, and homework [13]. Hence, the emotions participants experienced in this study may have been less intense than emotions that they experience in real academic settings [52]. This limitation cannot be eliminated entirely. However, participants regularly connected their experiences in the lab setting with their experiences in the ENGR 132 class (see Section 6.2). The intensity of participants' emotions may increase in a high-stakes task like a programming quiz. However, findings from this study provide valuable insights on how participants might behave emotionally when faced with similar tasks in class.

Limitations in the Data and Findings. Participants in this study encountered syntax errors only, and no logical errors. Hence, this study does not provide any understanding about the emotions that participants experience when they encounter logical errors. A more detailed investigation is needed to understand participants' emotions when they encounter and deal with logical errors. In the same vein, there is limited data about participants' behaviors while they execute test cases to test their code. One of the four problems on the programming task provided test cases in the problem description. Participants executed the test cases provided to them and did not test code for the other three problems on the task. It may be because they had limited time to finish four problems, and their focus was to finish as many problems as possible, instead of testing the code for multiple test cases.

8 IMPLICATIONS AND SUGGESTIONS FOR INSTRUCTORS

In recent years, there has been an increased awareness about the role of emotions in education [16, 53, 54]. Positive emotions generally promote learning and creativity, and negative emotions may help during tasks that require close attention [4]. To capture students' attention, promote their learning, and foster their motivation, instructors should plan for potential emotional impacts in their teaching and curriculum. In particular, to align with the CVT [44], programming instructors should foster students' perceptions of control and value of programming. The following section provides some suggestions for instructors.

Findings from this study suggest that most participants adopted a trial-and-error strategy to debug their code (see Section 5.4.1), during the loop between the "Typing Code," "Encountering Difficulties," and "Dealing with Difficulties" stages in Figure 3. Sometimes participants even engaged in repeated trial and error, called "hamster wheel" by Kinnunen and Simon [9]. These debugging behaviors are typical of novices because they tend to process and plan in their minds, and they do not use "paper and pencil" or print statements to debug and trace their code [45], as also evidenced in the current study. Prior research suggests that debugging instruction should incorporate metacognition or self-reflection questions where novices ask themselves questions like "What should I try?" and "What are the possible sources of bugs?" [45, 55]. These metacognitive questions can help novices cope with frustration and other negative emotions while debugging, thus breaking the "hamster wheel."

Some participants in our study were frustrated when they realized that they could perform the computation in their mind but were unable to convert their thinking into code (see Section 5.2.4). To help students with the process of converting a descriptive problem to code, one effective technique is to use visual icons for each programming concept by constructing flowcharts. Flowcharts have been found to enhance students' learning and also lessen their frustration [56]. Hence, it is recommended that flowcharts be introduced to students early in an introductory programming course and used throughout the course to conceptualize problems and convert them to code.

Our findings indicate that students experience shame when they fail at a certain task (see Section 5.3.2). Shame is a context-dependent emotion and participants fear other people's judgment when they experience failure [47, 48]. Hence, shame may hamper students' learning during a programming course. To reduce shame, instructors may adopt pedagogy of respectful discourse [57] to minimize judgment when students fail at a programming task. For instance, instructors must create a "safe space" where both instructors and students refrain from calling out and judging students who have been unsuccessful at performing a task. Furthermore, instructors can develop low-stakes assessments and provide timely and specific feedback to the students. This strategy can help students overcome momentary shame due to failing at low-stakes assignments. Moreover, they can use timely feedback to work on future assessments, thus enhancing learning and possibly succeeding at the task, leading to pride. To enable students to respond constructively to anxiety, instructors can create an environment of "felt safety" by providing scaffolding for difficult problems that connects students' prior knowledge with their present challenges [58]. Instructors can reframe difficulties in debugging as opportunities for students to learn troubleshooting skills, thereby promoting a growth mindset [59], rather than attributing difficulties to a lack of ability [16].

The CVT implies that students who have positive values about an academic task also experience positive emotions [44, 60]. Students' value for programming can be affected by the messages conveyed by instructors, either directly or indirectly. For instance, when instructors imply that CS1 is a "weed out" course, they create a stressful learning environment for the students [61], leading students to believe that they are "not cut out" for programming. Instructors must also realize that their own emotional disposition may transfer to their students, an idea called the "instructor-to-student contagion" [16]. For instance, the instructor's positive emotions may lead to greater enthusiasm in delivery. Instructors' enthusiasm creates a pleasant atmosphere in the class, leading to increased motivation in students, subsequently impacting learning.

A shift to a team-oriented approach to instruction and community building may also foster positive emotions in students [16]. For this purpose, some instructors have implemented pair programming in their CS1 classes [62]. Although an adapted version of pair programming is implemented in ENGR 132, more instructors need to adopt this technique in their classes to promote teamwork. During pair programming, students provide feedback to each other about their work. Feedback plays an important role in participants' learning and growth. Feedback that emphasizes punishment promotes anxiety [6]. Hence, instructors must provide constructive feedback to students that would foster their learning while promoting their emotional well-being.

9 CONCLUSION

This study provides a theory-backed qualitative understanding of the emotions that novices experience when they work on programming tasks. As explained in the discussion section, most findings of this study conform with the control-value theory of achievement emotions. However, we have explained some slight deviations. These deviations help extend the body of knowledge. One major contribution of this study is that it provides fine-grained understanding of the emotions that students experience as they work on a programming task and the reasons they provide

for experiencing those emotions, which most literature does not explain. Similarly, most literature on emotions in the context of programming focuses on frustration [26]. This study provides an understanding of other emotions (e.g., shame, pride) during programming tasks and the nuanced reasons they experience those emotions. Understanding these nuances will help educators design appropriate interventions to help students cope with negative emotions and foster positive emotions.

APPENDIX

A PROGRAMMING PROBLEMS ON THE PROGRAMMING TASK

No.	Programming Problem
Problem 1	An engineering student and his family are planning activities for their summer vacation to Lake X (blinded for review). The student needs a MATLAB function to help them decide what to do. Here are the criteria: 1. If the temperature is greater than or equal to 90°F, they will swim. 2. If the temperature is greater than or equal to 80°F and less than 90°F, they will go boating. 3. If the temperature is less than 80°F, they will go fishing. Write the MATLAB function to determine which activity the family will do based on the outside temperature, and then display the decision to the MATLAB command window. Include appropriate comments with your code. Test your function using the temperatures 70°F, 85°F, and 90°F.
Problem 2	Given the vector x = [1 8 3 9 0 1], write the MATLAB code necessary to: 1. Add up the values of all of the elements. 2. Compute the sine of the given vector containing x-values (values may be assumed to be in radians). 3. Compute a running sum. A running sum is the summation of a sequence of numbers that is updated every time a new number is added to the sequence. So the running sum of this x vector would be new vector with the values [1 9 12 21 21 22].
Problem 3	Consider an M-by-N array of random numbers. Write the MATLAB code necessary to move systematically through the array, element by element, and set any value that is less than 0.2 to 0 and any value that is greater than (or equal to) 0.2 to 1.
Problem 4	Write the MATLAB code necessary to evaluate and plot the function $f(t) = 4 - t^2 e^{-3t}$ for the time range $0 \leq t \leq 3$ second. Use an appropriate step size for t to create a smooth curve. Label the axes on the plot appropriately.

ACKNOWLEDGMENTS

The authors thank Edward Berger for access to the experimental setup; Brent Jesiek, Jennifer DeBoer, and Idalis Villanueva for advice on the design of the study; and Saira Anwar for assistance with data analysis.

REFERENCES

[1] A. Robins, J. Rountree, and N. Rountree. 2003. Learning and teaching programming: A review and discussion. *Comput. Sci. Educ.* 13 (2003), 137–172.

[2] C. Rogerson and E. Scott. 2010. The fear factor: How it affects students learning to program in a tertiary environment. *J. Inf. Technol. Educ. Res.* 9, 1 (2010), 147–171.

[3] M. Guzdial. 2015. *Learner-Centered Design of Computing Education: Research on Computing for Everyone* 8. Retrieved March 6, 2018, from http://www.morganclaypool.com/doi/abs/10.2200/S00684ED1V01Y201511HCI033.

[4] R. Pekrun and L. Linnenbrink-Garcia. 2014. Conclusions and future directions. In *International Handbook of Emotions in Education*. Routledge.

[5] M. Zeidner. 2014. Anxiety in education. In *International Handbook of Emotions in Education*. Routledge, 265–288.

[6] S. Secules, A. Gupta, A. Elby, and C. Turpen. 2018. Zooming out from the struggling individual student: An account of the cultural construction of engineering ability in an undergraduate programming class. *J. Eng. Educ.* 107, 1 (2018), 56–86.

[7] M. Meyer and S. Marx. 2014. Engineering dropouts: A qualitative examination of why undergraduates leave engineering. *J. Eng. Educ.* 103, 4 (2014), 525–548. DOI: 10.1002/jee.20054

[8] P. Kinnunen and B. Simon. 2012. My program is ok – Am I? Computing freshmen's experiences of doing programming assignments. *Comput. Sci. Educ.* 22, 1 (2012), 1–28. DOI: 10.1080/08993408.2012.655091

[9] P. Kinnunen and B. Simon. 2011. 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* 19–26.

[10] A. Lishinski, A. Yadav, and R. Enbody. 2017. Students' emotional reactions to programming projects in introduction to programming: Measurement approach and influence on learning outcomes. In *Proceedings of the 2017 ACM Conference on International Computing Education Research*. 30–38.

[11] N. Bosch and S. D'Mello. 2015. The affective experience of novice computer programmers. *Int. J. Artif. Intell. Educ.* 27, 1 (2015), 181–206. DOI: 10.1007/s40593-015-0069-5

[12] A. C. Frenzel, R. Pekrun, and T. Goetz. 2007. Girls and mathematics—A "hopeless" issue? A control-value approach to gender differences in emotions towards mathematics. *Eur. J. Psychol. Educ.* 22, 4 (2007), 497. DOI: 10.1007/BF03173468

[13] R. Pekrun. 2006. The control-value theory of achievement emotions: Assumptions, corollaries, and implications for educational research and practice. *Educ. Psychol. Rev.* 18, 4 (2006), 315–341. DOI: 10.1007/s10648-006-9029-9

[14] A. Bandura. 1991. Social cognitive theory of self-regulation. *Organ. Behav. Hum. Decis. Process.* 50, 2 (1991), 248–287. DOI: 10.1016/0749-5978(91)90022-L

[15] S. Afzal and P. Robinson. 2015. Emotion data collection and its implications for affective computing. In *The Oxford Handbook of Affective Computing*, R. A. Calvo, S. K. D'Mello, J. Gratch, and A. Kappas (Eds.). Retrieved August 29, 2017, from http://www.oxfordhandbooks.com/view/10.1093/oxfordhb/9780199942237.001.0001/oxfordhb-9780199942237-e-002.

[16] S. R. Cavanagh. 2016. *The Spark of Learning: Energizing the College Classroom with the Science of Emotion*. West Virginia University Press. Retrieved April 12, 2019, from https://muse.jhu.edu/book/47958.

[17] K. R. Scherer. 2009. The dynamic architecture of emotion: Evidence for the component process model. *Cogn. Emot.* 23, 7 (2009), 1307–1351. DOI: 10.1080/02699930902928969

[18] E. A. Linnenbrink. 2006. Emotion research in education: Theoretical and methodological perspectives on the integration of affect, motivation, and cognition. *Educ. Psychol. Rev.* 18, 4 (2006), 307–314. DOI: 10.1007/s10648-006-9028-x

[19] R. Pekrun and L. Linnenbrink-Garcia. 2014. *International Handbook of Emotions in Education*. Routledge, 2014.

[20] P. Kinnunen and B. Simon. 2010. Building theory about computing education phenomena. In *Proc. 10th Koli Call. Int. Conf. Comput. Educ. Res. (Koli Call'10)*. 37–42. DOI: 10.1145/1930464.1930469

[21] P. Kinnunen and B. Simon. 2010. Experiencing programming assignments in CS1: The emotional Toll. In *Icer'10*. 77–85. DOI: 10.1145/1839594.1839609

[22] A. Lishinski. 2016. Cognitive, affective, and dispositional components of learning programming. In *Proceedings of the 2016 ACM Conference on International Computing Education Research*. 261–262.

[23] M. Mercedes et al. 2009. Affective and behavioral predictors of novice programmer achievement. In *Proceedings of the 14th Annual ACM SIGCSE Conference on Innovation and Technology in Computer Science Education*. 156–160.

[24] M. Mercedes, T. Rodrigo, R. Shaun, and J. Baker. 2009. Coarse-grained detection of student frustration in an introductory programming course. *Presented at the International Workshop on Computing Education Research*.

[25] M. Mercedes, T. Rodrigo, J. O. Sugay, R. S. Baker, and E. Tabanao. 2009. Monitoring novice programmer affect and behaviors to identify learning bottlenecks. In *Philippine Computing Society Congress*. 1–7.

[26] M. Coto, S. Mora, B. Grass, and J. Murillo-Morera. 2021. Emotions and programming learning: Systematic mapping. *Comput. Sci. Educ.* 0, 0 (2021), 1–36. DOI: 10.1080/08993408.2021.1920816

[27] L. Tufford and P. Newman. 2012. Bracketing in qualitative research. *Qual. Soc. Work* 11, 1 (2012), 80–96. DOI: 10.1177/1473325010368316

[28] M. Q. Patton, *Qualitative Research & Evaluation Methods: Integrating Theory and Practice*, 4th ed. SAGE Publications, Inc.

[29] J. Walther et al. 2017. Qualitative research quality: A collaborative inquiry across multiple methodological perspectives. *J. Eng. Educ.* 106, 3 (2017), 398–430.

[30] Y. S. Lincoln and E. G. Guba. 1985. *Naturalistic Inquiry*. Sage.

[31] B. D. Boulay. 1986. Some difficulties of learning to program. *J. Educ. Comput. Res.* 2, 1 (1986), 57–73. DOI: 10.2190/3LFX-9RRF-67T8-UVK9

[32] E. Soloway, J. Bonar, and K. Ehrlich. 1983. Cognitive strategies and looping constructs: An empirical study. *Commun. ACM* 26, 11 (1983), 853–860.
[33] S. Wiedenbeck, D. Labelle, and V. N. Kain. 2004. Factors affecting course outcomes in introductory programming. In *Proceedings of 16th Workshop of the Psychology of Programming Interest Group*. Carlow, Ireland, P. 11.
[34] H. A. Elfenbein and N. Ambady. 2002. On the universality and cultural specificity of emotion recognition: A meta-analysis. *Psychol. Bull.* 128, 2 (2002), 203–235. DOI: 10.1037/0033-2909.128.2.203
[35] R. Jack, O. Garrod, H. Yu, R. Caldara, and P. Schyns. 2012. Facial expressions of emotion are not culturally universal. *Proc. Natl. Acad. Sci.* 109, 19 (2012), 7241–7244. DOI: 10.1073/pnas.1200155109
[36] L. Feldman Barrett. 2016. How emotions are made: The secret life of the brain. Nov. 28, 2016. Retrieved June 13, 2021, from https://lisafeldmanbarrett.com/books/how-emotions-are-made/.
[37] B. Marshall, P. Cardon, A. Poddar, and R. Fontenot. 2013. Does sample size matter in qualitative research?: A review of qualitative interviews in is research. *J. Comput. Inf. Syst.* 54, 1 (2013), 11–22. DOI: 10.1080/08874417.2013.11645667
[38] iMotions. 2019. Retrieved August 29, 2019, from 2022 https://imotions.com/.
[39] S. Elling, L. Lentz, and M. de Jong. 2011. Retrospective think-aloud method: Using eye movements as an extra cue for participants' verbalizations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 1161–1170.
[40] D. L. Paulhus and S. Vazire. 2009. The self-report method. In *Handbook of Research Methods in Personality Psychology*, R. W. Robins, R. C. Fraley, and R. F. Krueger, Eds. Guilford Press, 224–239.
[41] J. Aronson. 1995. A pragmatic view of thematic analysis. *Qual. Rep.* 2, 1 (1995), 1–3.
[42] V. Braun and V. Clarke. 2006. Using thematic analysis in psychology. *Qual. Res. Psychol.* 3, 2 (2006), 77–101. DOI: 10.1191/1478088706qp063oa
[43] L. F. Barrett. 2006. Solving the emotion paradox: Categorization and the experience of emotion. *Personal. Soc. Psychol. Rev.* 10, 1 (2006), 20–46. DOI: 10.1207/s15327957pspr1001_2
[44] R. Pekrun and R. P. Perry. 2014. Control-value theory of achievement emotions. In *International Handbook of Emotions in Education*, R. Pekrun and L. Linnenbrink-Garcia, Eds. 120–141.
[45] S. Fitzgerald et al. 2008. Debugging: Finding, fixing and flailing, a multi-institutional study of novice debuggers. *Comput. Sci. Educ.* 18, 2 (2008), 93–116. DOI: 10.1080/08993400802114508
[46] A. C. Graesser and S. D'Mello. 2014. Confusion. In *International Handbook of Emotions in Education*, R. Pekrun and L. Linnenbrink-Garcia, Eds. Routledge Handbooks. DOI: 10.4324/9780203148211-21
[47] G. V. Oades-Sese, T. A. Matthews, and M. Lewis. 2014. Shame and pride and their effects on student achievement. In *International Handbook of Emotions in Education*, R. Pekrun and L. Linnenbrink-Garcia, Eds. Routledge Handbooks Online. DOI: 10.4324/9780203148211.ch13
[48] D. Raccanello, M. Brondino, and M. Pasini. 2015. Two neglected moral emotions in university settings: Some data on pride and shame. *J. Beliefs Values* 36, 2 (2015), 231–238. DOI: 10.1080/13617672.2015.1031535
[49] C. Sansone and D. B. Thoman. 2005. Does what we feel affect what we learn? Some answers and new questions. *Learn. Instr.* 15, 5 (2005), 507–515. DOI: 10.1016/j.learninstruc.2005.07.015
[50] H. Fangohr. 2004. A comparison of C, MATLAB, and python as teaching languages in engineering. In *Computational Science (ICCS'04)*, 1210–1217.
[51] N. Bosch and S. D'Mello. 2013. Sequential patterns of affective states of novice programmers. In *the First Workshop on AI-supported Education for Computer Science (AIEDCS'13)*. 1–10.
[52] R. W. Picard. 2016. Automating the recognition of stress and emotion: From lab to real-world impact. *IEEE Multimed.* 23, 3 (2016), 3–7. DOI: 10.1109/MMUL.2016.38
[53] D. Brooks. 2019. Opinion | Students learn from people they love. *The New York Times*, Apr. 2, 2019. Retrieved April 12, 2019, from https://www.nytimes.com/2019/01/17/opinion/learning-emotion-education.html.
[54] J. R. Swallow. 2018. Students today need colleges to value emotions as well as the intellect. *Inside Higher Ed.* Retrieved April 12, 2019, from https://www.insidehighered.com/views/2018/07/10/students-today-need-colleges-value-emotions-well-intellect-opinion.
[55] D. Loksa and A. J. Ko. 2016. The role of self-regulation in programming problem solving process and success. In *Proceedings of the 2016 ACM Conference on International Computing Education Research*. 83–91.
[56] D. Gudmundsen, L. Olivieri, and N. Sarawagi. 2011. Using visual logic: Three different approaches in different courses - general education, CS0, and CS1. *J. Comput. Sci. Coll.* 26, 6 (2011), 23–29.
[57] K. Cushman. 2016. 3 Ways to keep shame from blocking deeper learning. *Education Week*, Jan. 19, 2016. Retrieved June 11, 2021, from https://www.edweek.org/leadership/opinion-3-ways-to-keep-shame-from-blocking-deeper-learning/2016/01.
[58] J. R. Eyler. 2018. *How Humans Learn: The Science and Stories behind Effective College Teaching*. West Virginia University Press. Retrieved May 26, 2020, from http://muse.jhu.edu/book/62882/.

[59] D. S. Yeager and C. S. Dweck. 2012. Mindsets that promote resilience: When students believe that personal characteristics can be developed. *Educ. Psychol.* 47, 4 (2012), 302–314. DOI: 10.1080/00461520.2012.722805

[60] R. Pekrun, T. Goetz, A. C. Frenzel, P. Barchfeld, and R. P. Perry. 2011. Measuring emotions in students' learning and performance: The Achievement Emotions Questionnaire (AEQ). *Contemp. Educ. Psychol.* 36, 1 (2011), 36–48. DOI: 10.1016/j.cedpsych.2010.10.002

[61] L. F. Denton and D. McKinney. 2004. Affective factors and student achievement: A quantitative and qualitative study. In *34th Annual Frontiers in Education, 2004 (FIE'04)*. T1G-6-11 1. DOI: 10.1109/FIE.2004.1408474

[62] J. Brougham, S. Freeman, and B. Jaeger. 2003. Pair programming: More learning and less anxiety in a first programming course. June 2003, 8.912.1–8.912.9. Retrieved May 1, 2016, from https://peer.asee.org/pair-programming-more-learning-and-less-anxiety-in-a-first-programming-course.

Received June 2020; revised June 2021; accepted December 2021