Exploring the Interplay of Metacognition, Affect, and Behaviors in an Introductory Computer Science Course for Non-Majors

Yinmiao Li
yinmiaoli@u.northwestern.edu
Northwestern University
Evanston, Illinois, USA

Melissa Chen
melissac@u.northwestern.edu
Northwestern University
Evanston, Illinois, USA

Ayse Hunt
aysehunt2020@u.northwestern.edu
Northwestern University
Evanston, Illinois, USA

Haoqi Zhang
hq@northwestern.edu
Northwestern University
Evanston, Illinois, USA

Eleanor O'Rourke
eorourke@northwestern.edu
Northwestern University
Evanston, Illinois, USA

ABSTRACT

Introductory computer science for non-majors, often referred to as CS0, is a course that is designed to be more accessible and less intimidating than CS1, with the goal of alleviating barriers and fears associated with learning computer science (CS). However, despite this intention, many students still struggle in CS0 and these courses do not always successfully prepare students for future CS learning experiences. In this paper, we study the experiences of CS0 students with a particular focus on the intersection of their metacognition, affect, and behaviors. To study students' daily learning experiences, we collected data from 20 participants who completed structured daily diaries and retrospective interviews over the course of a single homework assignment. Through a thematic analysis of the diaries and interviews, we identified three distinct patterns of engagement that highlight the importance of *metacognitive knowledge of strategies*, or a students' understanding of when, why, and how to effectively use regulation and disciplinary strategies while working on tasks. The three patterns of engagement include: (1) avoidance behaviors resulting from negative emotions, negative judgements, and a lack of metacognitive knowledge of strategies, (2) persistence or re-engagement behaviors despite negative emotions and judgements aided by metacognitive knowledge of strategies, and (3) persistence behaviors with evidence that metacognitive knowledge of strategies prevented students from forming negative judgements in the first place. We contribute an initial model of the interplay of metacognition, affect, and behaviors in CS learning, showing the role of metacognitive knowledge of strategies in helping students persist in the face of struggle. In our discussion, we advocate for explicit interventions that support students in developing metacognitive knowledge of strategies while also supporting their sometimes challenging emotional experiences.

CCS CONCEPTS

• **Social and professional topics** → **Computer science education**; • **Human-centered computing** → *User studies.*

KEYWORDS

CS0, Metacognition, Affect

ACM Reference Format:
Yinmiao Li, Melissa Chen, Ayse Hunt, Haoqi Zhang, and Eleanor O'Rourke. 2024. Exploring the Interplay of Metacognition, Affect, and Behaviors in an Introductory Computer Science Course for Non-Majors. In *ACM Conference on International Computing Education Research V.1 (ICER '24 Vol. 1), August 13–15, 2024, Melbourne, VIC, Australia.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3632620.3671119

1 INTRODUCTION

"*I always really disliked getting stuck. And I know I have a tendency to just give up and that's not good. But it really sucks when you have a bug. And I just don't really know if I have the patience to go through it and figure it out sometimes because those negative emotions are hitting me. Being stuck on this project was a reason some days I just didn't work on it because I don't know where to go unless I go to office hours ... I just don't know how to cope with problems in CS other than just going to a TA and trying to ask them how to figure out my problems.*"

This quote was shared by one of our participants, who was taking an introductory computer science course for non-majors (CS0), when describing her reaction to struggling while programming. CS0 courses are designed to be more accessible and less intimidating than traditional CS1 courses, with the goal of reducing barriers and fears associated with learning CS and engaging more students in the field [1, 26]. However, despite being enrolled in a CS0 course, our participant still experienced challenges that are common for many novice CS learners. Previous studies have shown that the programming experiences of novices are intricately interwoven with their emotions [30, 31] and self-assessments [23], and that negative experiences can derail the problem-solving process and decrease student persistence [37]. For example, Kinnunen and Simon found that many students feel confused when getting started on a programming assignment, feel like they are "stupid" when they encounter challenges, and ultimately experience low self-efficacy, or the belief that they are not able to learn the subject matter [30]. How can we support students in overcoming these negative emotions and judgements so that they can make progress towards their learning goals?

In this paper, we begin to explore this question by studying the intersection of metacognition, affect, and students' behaviors. Metacognition is a broad term that is comprised of *metacognitive knowledge*, or the awareness of oneself, the task, and appropriate strategies, *metacognitive regulation*, or the active regulation of one's thinking process, and *metacognitive experiences*, or subjective judgements and feelings in relation to the task, oneself, and one's learning [14, 20, 54, 57]. Many studies have established the importance of metacognition and self-regulated learning (SRL) in allowing students to identify difficulties and take strategic actions in response [11, 17, 24]. Affect, on the other hand, refers to emotions and other mental states such as mood and feelings [21], which have been shown to impact the learning process [62]. While most empirical research has studied metacognition and affect independently, a more recent line of work led by Efklides has argued for the importance of studying the two together, since affective experiences are a core part of a student's learning experience that influence and are influenced by metacognitive experiences [15, 16].

Within the field of CS education, researchers have applied models of metacognition and SRL to understand student strategies [18], improve their problem solving [40], and explore the interaction between their metacognition and self-efficacy [39]. However, very little research has considered the role of affect in student metacognition in this context. In a recent review of research on metacognition and self-regulation in programming education, Loksa et al. cited MASRL [16], a model developed by Efklides that encompasses metacognition, affect, and SRL, as an underutilised theory in CS education research [41]. Addressing this gap, we study student behaviors while working on programming assignments through a lens of both metacognitive and affective experiences to understand the relationship between these factors.

We present findings from a qualitative study that we conducted with 20 students who were enrolled in a CS0 course. Participants completed daily diaries [59] and retrospective interviews over the course of a single homework assignment to uncover their metacognitive and affective journey. Through a thematic analysis of the diary and interview data, we identified three distinct patterns of engagement that highlight the importance of metacognitive knowledge of strategies, or a student's awareness of when, why, and how to apply metacognitive and disciplinary skills effectively [20, 58], for their task engagement and metacognitive experiences. The three patterns of engagement are: (1) avoidance behaviors resulting from negative emotions, negative judgements, and a lack of metacognitive knowledge of strategies, (2) persistence or re-engagement behaviors despite negative emotions and judgements with the help of metacognitive knowledge of strategies, and (3) persistence behaviors with evidence that metacognitive knowledge of strategies prevented students from forming negative judgements in the first place.

Our work contributes an initial model of how metacognition, affect, and behaviors interact in students' learning experiences based on our empirical data. Given our findings, we advocate for increased attention to the central role of metacognitive knowledge of regulation and disciplinary strategies in student programming learning, and we propose design implications for interventions in this space.

2 THEORETICAL FRAMEWORK

In this work, we draw on theories of metacognition and self-regulated learning (SRL) from the fields of psychology and the learning sciences. Conceptualizations of metacognition have evolved and expanded over time [61], resulting in a rich and complex theoretical landscape. Metacognition can be understood as a continuous process of monitoring cognitive activity and using the information gained through monitoring to decide which action to take next [6, 20, 32, 46]. In contrast, self-regulation is the ability to manage and regulate one's thoughts, emotions, and behaviors to achieve specific goals [8, 49]. These two concepts, metacognition and self-regulation, are closely intertwined, and the terms are occasionally used interchangeably due to their overlapping definitions [24, 61]. In a more recent line of work, Efklides bridges across metacognition, affect, and self-regulated learning by introducing the metacognitive and affective model of self-regulated learning (MASRL). This model emphasizes the importance of subjective experiences, encompassing both metacognitive and affective dimensions, in shaping how individuals regulate their cognitive processes.

To ground our research, we developed a theoretical framework that focuses on the core aspects of metacognition, SRL, and MASRL most relevant to our current study, specifically metacognitive knowledge, metacognitive experiences, affective experiences, and regulation strategies. We also incorporate disciplinary strategies from CS into our framework because effective metacognition depends on knowing these strategies. We unpack each of these aspects below, and provide a visual representation of our theoretical framework in Figure 1.

Metacognitive knowledge encompasses stable beliefs about one's cognitive activities or cognition in general, including knowledge about oneself, the task, and associated strategies [5, 14, 20]. In our work, we focus primarily on metacognitive knowledge of strategies, or a student's understanding and awareness of when, why, and how to apply regulation strategies and disciplinary strategies effectively in various tasks or situations. Metacognitive knowledge is further categorized into declarative knowledge, procedural knowledge, and conditional knowledge [33, 50, 54]. Declarative knowledge involves knowing facts about oneself, the task's demands, and the strategies that are available [54, 57]. Procedural knowledge includes knowing how to use strategies [54, 57]. Finally, conditional knowledge refers to the knowing when and why to apply specific strategies in different situations [54, 57].

Metacognitive experiences encompass the momentary experiences of a learner while working on a particular cognitive task. These experiences can be further categorized into metacognitive judgements (e.g., a student's judgement of their own understanding), metacognitive feelings (e.g., a student's feeling of confidence), and the metacognitive knowledge that is in active use while monitoring the problem-solving process (e.g., a student's belief that they are not good at this) [14, 15, 20, 58]. In MASRL, metacognitive experiences are characterized as the connection between metacognition, affect, and motivation [16]. For example, emotions commonly arise

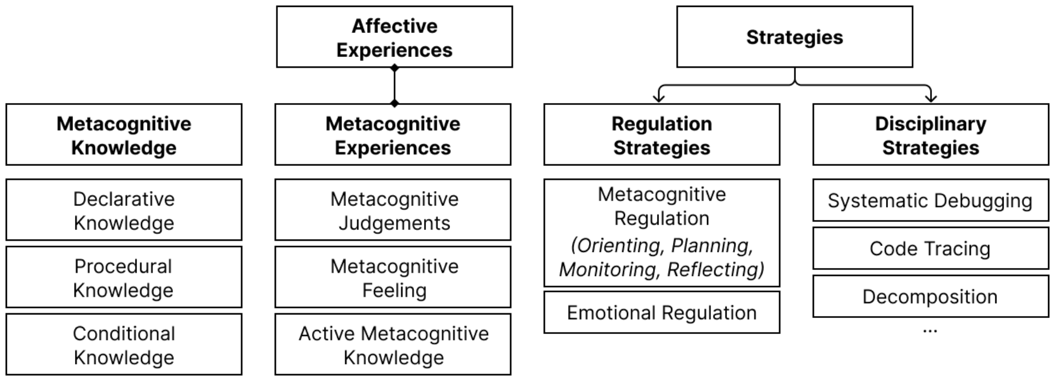

Figure 1: The theoretical framework we developed that focuses on the core aspects of metacognition, self-regulated learning, and the metacognitive and affective model of self-regulated learning (MASRL) most relevant to our current study. Metacognitive knowledge refers to stable beliefs about cognition, including knowledge about oneself, the task, and regulation and disciplinary strategies. Metacognitive experiences refers to the momentary experiences of a learner while working on a particular cognitive task, which are closely tied to their affective experiences. Regulation strategies refers to strategies related to monitoring and controlling one's learning processes. Finally, disciplinary strategies refers to the cognitive strategies employed while solving programming problems.

when students experience feelings of under-confidence or make negative judgements of their learning progress or comprehension [15]. These affective experiences and metacognitive experiences interact to impact how students employ strategies to reach their goals [16].

Regulation strategies is a term we define to encompass a set of strategies discussed in the metacognition and SRL literatures that relate to the ability to monitor and control one's learning processes. Metacognitive regulation strategies include orienting, planning, monitoring and adjusting, and reflecting [3, 60, 66]. Orienting involves assessing and understanding the task and evaluating what one knows [3, 66]. Planning entails setting goals and subgoals, as well as thinking about implementation [3, 66]. Monitoring and adjusting refers to continuously checking one's progress, knowing the current state, comparing with goal states, and making modifications as necessary [3, 66]. Reflecting involves evaluating one's strategies or learning process, and thinking about alternative strategies after completing a task [3, 66]. Emotional regulation, taken from SRL, includes strategies for managing negative affect such as using reassuring self-talk, redirecting, or taking a break and coming back to the task later [8].

Disciplinary strategies, which is a term we use to refer to the cognitive strategies used in the discipline of computer science, encompass approaches for solving programming problems. For example, strategies such as systematic debugging [45], code tracing [27], and decomposing problems [56] are all categorized as disciplinary strategies. We include these strategies in our theoretical framework because they are important for understanding student problem-solving behavior. Disciplinary strategies are used to make progress on the task at hand, while regulation strategies are used to monitor whether the disciplinary strategies are effective and control decisions about which strategies to use next. When a student does not have a strong understanding of disciplinary strategies, they are not able to select and apply the most appropriate strategies [8].

3 RELATED WORK

3.1 Introductory Computer Science Courses for Non-Majors

While increasing numbers of students are enrolling in computer science courses [67], many struggle to learn programming and retention rates in computing programs are low [51]. Low retention is especially noticeable among women and students of color, which negatively impacts diversity in computing [35]. Researchers have explored a variety of approaches for helping novice students learn programming more effectively, including developing new curricula [28], programming languages [19], and collaborative programming approaches [43].

More recently, researchers have established the importance CS0 courses in helping increase diversity and address the growing relevance of computing to other disciplines [53]. In traditional CS1 courses, students often struggle with a lack of understanding of problem-solving approaches, and demotivation from low self-efficacy [44]. In an effort to make CS0 more accessible and less intimidating, researchers have explored several ways of restructuring course content, including a focus on computational thinking instead of programming [9], using media-based creative tasks [25], and reducing workload to make learning more manageable [10]. While studies show that CS0 courses can alleviate fears associated with learning CS and help to improve confidence [2, 26], there is also evidence that they do not necessarily prepare students for future computing learning experiences [52]. Our research contributes new understanding of the challenges that CS0 students encounter while learning to program, which can foster the design of new curricula and interventions to better support their learning.

3.2 Metacognitive and Affective Experiences in CS Education

CS education researchers have argued that introductory courses must go beyond teaching syntax and semantics, as metacognitive skills are a key component of student success in the field [4, 40, 55]. Loksa et al. conducted a systematic review of the growing body of research on metacognition and self-regulation in computing education, highlighting the usage of different theoretical frameworks in recent articles [41]. Some of this research aims to understand student strategies [18] and the interaction between self-efficacy and metacognitive strategies [39]. Other research explores the design of interventions that cultivate metacognitive regulation while programming. For example, Loksa et al. introduced a novel intervention that teaches problem-solving and metacognition explicitly by providing lessons on problem-solving techniques, visualizing student progress across problem-solving stages, supporting self-reflection guided by prompts, and providing contextual assistance through the code [40]. *Metacodenition* [48] provides metacognitive scaffolding for problem-solving based on the six stages of problem-solving framework presented in [40]. *Pensieve* surfaces students' programming process through visualizations that support one-on-one conversations between teaching staff and students around strategies and practices [65]. *Code replays* offer opportunities for students to reflect on their programming process, which had the potential to support self-regulation in novice programmers [64].

Affective experiences, which often arise alongside metacognitive experiences when students make judgements of their learning progress or comprehension [15], represent another significant area of research within CS education. Lishinski and Rosenberg found that students' long-term interest and learning outcomes are significantly influenced by factors including momentary self-efficacy assessments and affective experiences [38]. Kinnunen and Simon showed that the programming experiences of CS1 students are interwoven with their emotions [30]. For instance, CS1 students often feel confused when getting started on a programming task, feel like they have been "struck by lightning" and that they are "stupid" when encountering challenges, feel fed up and frustrated after several debugging attempts, and ultimately experience low self-efficacy in CS [30].

Despite the clear importance of both metacognitive and affective experiences, few studies within CS education have studied the intersection. In their literature review, Loksa et al. emphasized the need for further research grounded in MASRL model that includes both metacognition and affect. By exploring the interplay between metacognition, affect, and behaviors in the context of CS education, we aim to fill this gap.

4 METHOD

To better understand the relationship between students' metacognition, affect, and their programming behaviors, we conducted a qualitative study of students' experiences as they navigated a CS0 course. We collected data throughout the 12-week course, with a few students participating each week. Participation was structured around the due dates of programming homework assignments, as we were interested in learning about students' programming process. Students have two lectures and one tutorial session each week, where the tutorial is a small group session with about 10 people led by a teaching assistant. Most students participated for one week, but some students participated over a three-week period during the course's final project. While participating in the study, students were asked to complete daily journals to reflect on their experiences leading up to, during, and after working on their homework assignment. The diaries also asked them to reflect on supplementary course activities such as attending office hours, lectures, and preparing for quizzes. Students were not required to complete diaries on days when they did not work CS0 course-related tasks. After the homework due date, a researcher conducted an interview with each participant to learn more about their experience.

4.1 Participants

We recruited 20 students from a large private university in the Midwestern United States. All participants were enrolled in the same CS0 course. 11 of the 20 participants reported that they had prior programming experience, such as in Scratch or taking a Statistics course in R. Of the participants with prior experience, three reported taking AP Computer Science Principles or AP Computer Science A in high school. These are Advanced Placement (AP) courses offered in the United States to introduce students to CS and prepare them for college-level coursework.

Students were invited to participate in the study through an announcement made in class and via email. We chose to work with students in CS0 because we hypothesized that non-majors would be more likely to hold negative beliefs about their CS abilities, a phenomenon we were particularly interested in studying.

In Table 1, we present a brief summary of the homework assignment each participant worked on while taking part in our study. This table also shows how our data collection was distributed across the 12-week course.

4.2 Procedure

4.2.1 Daily reflective journals. To surface participants' affective and metacognitive experiences, we asked them to complete daily journal entries [59]. Participants were directed to an online form that prompted them to answer a few short answer questions about any course activities they took part in that day (see Table 2).

Our goal was to gain insight into participants' experiences without a long period of time between the experience itself and when they documented and reflected on it. Additionally, we were interested in capturing any differences in participants' affective and metacognitive experiences across the week, such as at the beginning of a homework assignment vs. when the deadline was approaching.

4.2.2 Retrospective interview. We asked participants to schedule one-on-one interviews with the first author immediately after they submitted their homework. All interviews occurred within five days of the associated homework deadline. During the interviews, the researcher asked follow-up questions based on their daily journal entries. Additionally, the researcher asked participants to reflect on their overall experiences in the class. The interviews allowed participants to expand on the journal entries they wrote the previous week. These responses were recorded using Zoom's built-in functionality, and then automatically transcribed using Rev. These

Table 1: Brief Summary of the Homework Assignment Each Participant Worked On

Week	Homework	Homework Description	Participants
W1-W3	Various	Various	None
W4	HW3	Create an original musical piece with a length of 16 to 32 beats from scratch. Use list or tuple, variables, and functions to create four tracks.	P1, P2, P3, P4, P6, P7, P8, P9
W5	None	None	None
W6	HW4	Change the example code to draw basic shapes Practice writing functions that enable you to create more complex shapes.	P11, P12
W7	HW5	Write a program to draw a creature or your own design using Tkinter.	P15, P16, P17
W8-W9	HW6	Write a text-based version of the game Wordle.	P14, P18
W10-W12	Final Project	Create a visualization of a real-world data set.	P20, P21, P23, P25, P27

Table 2: Daily journal questions

Section	Questions
Task completion	What course-related tasks did you do today? How long you spent on each task, what you did and why you decided to work on it? For the task that took you the longest time, could you walk me through your thought process and how you approached the task?
Challenges	What did you struggle with today, if anything, and why did you struggle? What emotions, if any, did you feel while struggling and why did you think you felt that way? How did your struggles and your emotions affect your behaviors, if any? How did your struggles and your emotions affect your beliefs about yourself or [cs0], if any? Were you able to overcome your struggles? If so, what actions did you take to overcome them? Or what did you plan to do next?
Overall reflection	Which activities you engaged in today contributed most to your learning? Reflecting on your learning process today, what have you done well today? What, if any, are things you wish you had done differently? Anything else you want to share?

automated transcripts were edited by the first author during analysis to correct any transcription errors.

4.3 Data Analysis

We conducted a thematic analysis [29] of the daily journal responses and interview transcripts to identify themes related to students' affective and metacognitive experiences and their employment of strategies. The first and second authors conducted inductive, open coding, discussed initial themes, and reached an agreement on initial codes and definitions.

Following the initial round of coding, we compared our categories to existing theories in the literature and adopted terminology that matched the theory. The two authors then clustered the initial codes into six categories: disciplinary strategies, regulation strategies, metacognitive experiences, metacognitive knowledge, affective experiences, and environmental factors. Environmental factors refers to additional aspects of student learning experiences that can influence their behaviors, beliefs, and emotions, such as class structures, the topic of assignments, and social interactions. We aimed to identify patterns and tensions within each category, specifically focusing on affective and metacognitive experiences during struggles, task engagement in response to emotions and metacognitive judgements, and participants' metacognitive knowledge of strategies. The themes that resulted from this second round of coding were presented to all authors, who discussed the data

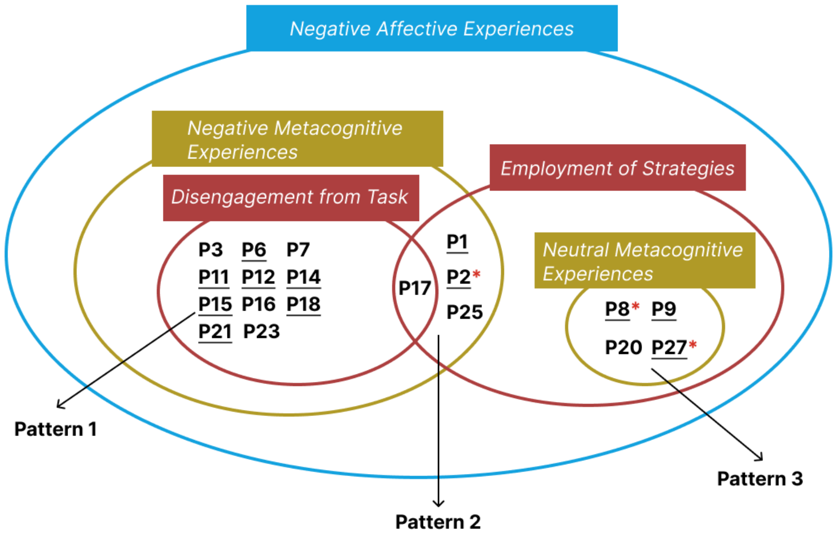

Figure 2: A Characterization of our participants according to their affective experiences, metacognitive experiences, and behaviors. The underlined participants have prior experiences with programming, including Scratch, code.org, or AP CS Principles/A (which is further denoted with an asterisk.)

supporting each theme and confirmed them. Together, we refined and finalized the findings presented below.

5 RESULTS

In our data analysis, we found that all students (except for one who encountered no significant challenges) reported that they experienced negative emotions when they struggled with their programming assignments. Of these, 15 participants described having negative metacognitive experiences related to their task progress and confidence in response to the struggle. The most commonly reported negative metacognitive experiences were feeling incompetent about coding ability, feeling under-confident, and feeling unable to complete tasks. These negative affective and metacognitive experiences stemmed from programming experiences like being unable to complete homework independently, struggling to understand and fix errors, failing to start coding or fixing errors quickly, not understanding the example code, and feeling uncertain about how to apply concepts, which align with previous research [22].

These intertwined negative affective and metacognitive experiences often influenced participants' decision-making regarding task engagement. We observed three distinct patterns in response to these experiences among our participants. 12 out of the 19 participants demonstrated avoidance behaviors, such as putting off homework until they received help from friends or teaching assistants or giving up on assignments for partial credits. In contrast, eight participants demonstrated resilience in overcoming their struggles, including one participant who shifted from initially avoiding to re-engaging. We divide these eight into two separate patterns: four participants initially experienced negative metacognitive judgements and feelings related to their coding competence and task completion but ultimately gained confidence in their abilities, and four participants never experienced negative metacognitive experiences in the first place. Figure 2 presents a characterization of 19 participants who reported facing struggles.

In this section, we describe three broad patterns of student engagement: avoidance of struggle, persistence through struggle despite negative emotions and metacognitive experiences, and persistence through struggle with an absence of negative metacognitive experiences. We draw connections between students' engagement patterns and their emotions, their metacognitive experiences, and their metacognitive knowledge of strategies.

5.1 Negative Affective and Metacognitive Experiences Lead Many Students to Disengage

In this section, we present the avoidance behaviors we saw in participants who disengaged, along with the reasons they gave for resorting to giving up and waiting for help. We highlight that participants who fell into this pattern expressed not knowing alternate strategies for moving forward. Then, we discuss why this pattern of behavior could harm students' learning.

5.1.1 A pattern of disengagement due to not knowing alternative strategies. When participants face struggles and experience negative emotions such as annoyance, frustration, and tiredness, make momentary negative metacognitive judgements regarding their coding competence, or express uncertainty about their ability to complete tasks, a common reaction is to avoid the task and rely on others for help.

Instead of actively engaging in strategies to overcome challenges, participants were often overwhelmed by negative feelings, opting instead to put off the task until the deadline, wait for lectures and office hours to give them guidance for the next steps, or even give up to earn partial credit. For example, P7 talked about feeling frustrated and irritated because she did not understand how to implement a task she perceived as simple, and how this made her feel uncertain of her ability to fulfill the assignment requirements. This made her "put off the homework for the weekend", which was close to the assignment deadline on Monday. Some students also described overpowering negative emotions that led to their decision to disengage with learning, leaving them feeling a lack of coding competence. For instance, P14 expressed her tendency to give up "because I was tired of feeling bad." P11 mentioned, "I very quickly gave up doing the homework because I was annoyed with being upset." P21 also expressed her feeling of discouragement and wanting to give up: "I just don't really know if I have the patience to go through it and figure it out sometimes because those negative emotions are hitting me." Additionally, P6 described that the struggle made him feel "stressed and discouraged", leading to doubts about his level of understanding, which demotivated him from incorporating concepts required in practice. All of these examples illustrate the pattern of participants resorting to avoidance due to overwhelming negative emotions and metacognitive experiences, ultimately derailing their learning process.

We found that participants who gave up and avoided their homework ultimately moved forward by asking for help in person. Eight of the 11 participants who disengaged from their homework reported that they waited for lectures, tutorial sessions, or office hours to make progress. For instance, P6 decided to wait until the tutorial to figure out the problem with his homework. P14 planned to seek clarification from others in person so "flagged it to ask in office hours." Additionally, P21 expressed that, when overwhelmed by emotions and losing their patience to persist at task, "I don't know where to go unless I go to office hours." Five participants reported overcoming their struggles after getting help from TAs.

Our analysis revealed that the participants who avoided tasks and sought help from others did not report trying many effective strategies to address challenges by themselves. The strategies they did report using included: orienting through identifying the homework requirements and what they learned; planning through thinking about desired outcomes and identifying analogous examples; tinkering with code; referencing course materials, notes, and similar examples; debugging by trial-and-error; and memorizing syntax to become familiar with how to implement certain concepts. The predominant strategies reported by all participants for approaching homework were referencing course resources to find the exact code or working from similar examples. P15 mentioned that if she needed to create a visualization that uses shapes like circles and ovals, she would prefer that the course provides similar examples showing how to implement them, as she felt the need to "work with a foundation instead of starting from scratch." P23 also expressed similar ideas that "without the example provided by the professor, I don't think I would've been able to figure out [how to complete the homework.]" Other than referencing course resources, several students use trial-and-error strategies to debug, where they "changed all elements, one at a time, and tried everything until it produced what was expected." P3, P7, and P14 said they memorized syntax to learn concepts and apply them to homework. As P14 elaborated, "I have to remember what the functions are, what the exact formatting is, if there's a dot something, it has to be in your vocabulary."

From these participants' experiences, there is a noticeable absence of a variety of effective strategies, particularly when compared to the participants who engaged with struggle, who will be discussed in Section 5.2. Apart from referencing course resources, most students reported no use of effective regulation and disciplinary strategies, such as planning through thinking about code implementation, monitoring their progress, switching strategies to achieve goals, decomposition, systematic debugging, and reading documentation. Students who tried to debug suggested that they changed elements randomly until the desired outcome was achieved rather than employing structured methods to identify errors, such as tracing, unit testing, etc. Some students used memorization as a way to learn programming. While memorization can be beneficial for recalling syntax, professional programming emphasizes understanding of principles, the use of problem-solving techniques, systematic debugging, and adaptive planning in varied contexts [36]. Memorization will not help with these core skills.

In fact, some participants explicitly mentioned that they didn't have access to strategies to help them move forward, other than asking for help from course staff. P21 elaborated that she wouldn't re-engage in the task because she did not have enough strategies to fix errors when got stuck:

> I always really disliked getting stuck. And I know I have a tendency to just give up and that's not good at all ... some days I just didn't work on it because I don't know where to go unless I go to office hours or something ... I just don't know how to cope with problems in CS or in math other than just going to a TA and trying to ask them how to figure out my problems.

Even though she recognized that giving up did not benefit her learning, the absence of metacognitive knowledge regarding strategies, coupled with overpowering emotions and negative self-judgements, led to the tendency to stop working. P15 also mentioned that she found debugging hard and would give up due to lacking effective strategies and knowledge for debugging, as well as experiencing frustration.

Participants' lack of metacognitive knowledge of strategies was also evident in their ineffective use of strategies, such as trial-and-error for debugging, which often resulted in disengagement from the task. When they tried to debug through trial-and-error, they tried all concepts from the course resources to see which would work. P3 elaborated she used trial-and-error by "changing the parentheses to a bracket or trying a different line of code method that I happen to know, trying to fit the puzzle piece until it works." P11 said: "you're just going to have to type random things and hope it works ... if it's helping then I'll just keep doing that, or you kind of just stare at your computer." While trial-and-error involves experimentation, in reality, it is a systematic testing approach that relies on the programmer's understanding of the problem and programming concepts. It appears that many participants don't know how to use this strategy effectively. It's also possible that participants don't have the content knowledge needed to identify the problem and form a hypothesis, a requirement for applying this strategy effectively. P15 explained: "I try to go to the ones that most obviously seem like they could be the problem, but if that doesn't work, I do just try everything." P15 resorted to ineffective trial-and-error when he lacked the necessary content knowledge to be more systematic. Ineffective strategy use is important because it ultimately contributes to disengagement from tasks.

In summary, we observed a pattern of participants' avoidance behavior influenced by negative affective and metacognitive experiences. Further analysis showed that the participants who fell into this pattern used fewer and less effective strategies than those who fell into the engagement pattern (see Section 5.2). Some participants mentioned that they relied on help from TAs because they lacked strategies to move forward independently. This suggests that students in this group may have insufficient metacognitive knowledge regarding strategies to address their issues on their own, ultimately derailing participants' engagement with practice.

5.1.2 Why is this pattern of avoidance and reliance on external help a problem? When participants don't have access to effective strategies, it is natural for them to depend on TAs and other help to move forward. While help-seeking is an important part of learning, prior studies have also pointed out that not all help-seeking is effective [34, 42, 47]. Students and TAs prioritize fixing immediate problems in homework assignments over teaching students effective strategies [34, 42]. Students often expect TAs to provide explanations of code implementation or even direct answers, and as a result TAs feel pressure to offer students detailed guidance even if it means providing explicit answers to problems [42]. In fact, some of our participants reported that they asked for or received answers from TAs. For instance, P27 mentioned that she showed her code to a TA who suggested that she change the list to a dictionary, a direct answer to her issue. P12 shared that one time she and her classmates didn't know how to get started on the homework. When asked, the TA went over the beginning of the solution. When P15 talked about her help-seeking experience with the TA, the interviewer asked whether P15 understood how the TA had solved the problem, and she answered "I'm not sure, she just suggested it after changing some values." This reflects a concerning reliance on external support as a way to receive answers, rather than to learn disciplinary strategies that would help students overcome struggles, which potentially impedes their self-confidence.

Another potential harm of avoiding independent work and relying on TAs to resolve issues is that, when students do not receive the assistance they need, it can exacerbate negative emotions and reduce interest in class. P6 went to office hours but the TA wasn't able to fix his issue, which added to his stress and disappointment. He felt that that office hours were "unproductive," and that he was incapable of completing the task. Consequently, P6 decided to not continue working on the homework because he was uncertain about what to do next, and decided to "try winging the assignment and hope for partial credit." He further reported, "I'm starting to like this course way less now." This case highlights that when students fail to receive help from a TA, especially when they heavily rely on assistance, it can lead to heightened stress and disappointment, further disengagement from learning, even a loss of interest in class.

The students who fell into this pattern of avoiding tasks and over-relying on help often led to negative metacognitive experiences that extended beyond the task. Their negative emotions and feelings of incompetence during moments of struggle led to concerns about their ability to complete coursework, get good grades, pursue CS, and whether they possessed the innate ability needed to succeed in CS. P18 talked about how not being able to understand the example code triggered feelings of frustration stemming from "not being able to start sooner", which ultimately led her to "quit studying sooner". This frustration further made her "question my ability to successfully complete this course [...] with an A." This demonstrates that her negative emotions were so overpowering that she could not spend more time on the task. Her emotions escalated her feelings of incompetence, undermining her confidence that she could get a good grade in the course. P17 reported having trouble applying concepts and feeling frustrated, stating that "these emotions led me to stop the work for today in hopes of getting answers at a tutorial session or office hours this week." This experience left her with a feeling of "believ[ing] that I couldn't adequately complete the work or any other work that may be given to me. This also mitigated my beliefs about completing a minor in CS." The negative metacognitive judgement of her ability in coding extends beyond the current task to questioning her capability to complete any future task, leaving her uncertain about pursuing a CS minor.

The combination of struggle, negative affective and metacognitive experiences, and social comparison led some students to believe that they don't have the necessary innate ability to succeed in CS. P7 said her friends had told her that CS0 would be easy if she had the right way of thinking. When she struggled with tasks she perceived as simple, she said it made her feel like "I'm bad at computer science and don't have the way of thinking". P21 also expressed her belief that the way of thinking in CS is innate. She felt disappointed in herself and reported that "I do wish I were clever enough to solve these problems on my own." She further elaborated in the interview that:

> I can't intuitively tell where my problem is. It seems like some of the TAs just look at your code and they can intuitively tell, that's so impressive … some of my friends could even look at some of my code and be like, I can tell what's wrong with this ... So clearly they have some intuitive sense of what is wrong and what is right.

P16 also reported feeling incompetent because she could not start coding right away. She believed that some people could start right away because they have the CS way of thinking, which made her feel discouraged and not gifted in CS. This reveals a fixed mindset [12, 13] regarding the innate nature of CS, which could discourage students who do not perceive themselves as naturally gifted from learning disciplinary knowledge and further persisting in CS.

Overall, our data reveals a troubling pattern among students who have insufficient metacognitive knowledge of the strategies needed to move forward when they face struggles. Figure 3 is a diagram demonstrating the synthesis of all participants' cases falling under this disengagement pattern. Participants experienced negative emotions and negative metacognitive judgements of their coding ability and task progress. They used limited regulation and

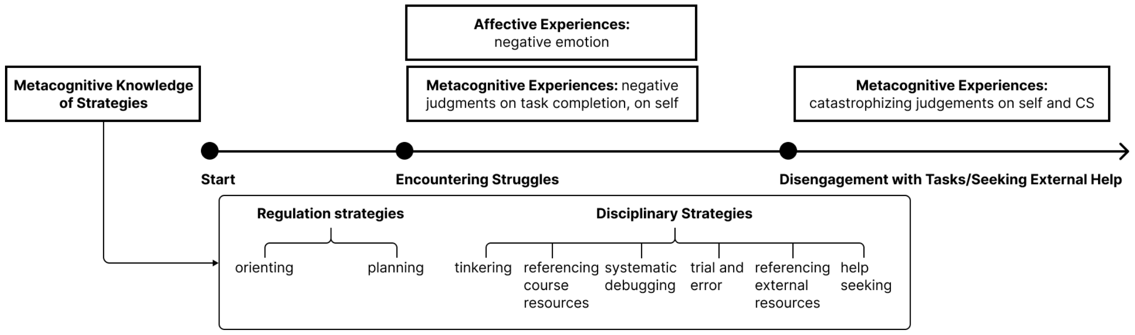
Affective Experiences: negative emotion
Metacognitive Knowledge of Strategies
Metacognitive Experiences: negative judgments on task completion, on self
Metacognitive Experiences: catastrophizing judgements on self and CS
Start
Encountering Struggles
Disengagement with Tasks/Seeking External Help
Regulation strategies
Disciplinary Strategies
orienting
planning
tinkering
referencing course resources
systematic debugging
trial and error
referencing external resources
help seeking

Figure 3: A synthesized diagram that demonstrates all participants' cases falling under the disengagement pattern

disciplinary strategies compared to the participants who fell into the engagement pattern (see Section 5.2). Some students experienced a spiral effect in the end. We demonstrated that this pattern of behavior does not typically support student learning, and can lead to diminished confidence levels and the formation of problematic beliefs about themselves as CS learners.

5.2 Metacognitive Knowledge of Strategies Can Help Students Engage with Tasks Despite Struggles

In this section, we show how students who possess metacognitive knowledge of both regulation strategies (e.g., orienting, planning, monitoring, reflecting) and disciplinary strategies (e.g., systematic debugging) are able to successfully engage in tasks even when they struggle. First, we show that when students experience self-doubt, this metacognitive knowledge can help them regulate their emotions and re-engage in tasks, ultimately increasing their confidence. Then, we show how having access to this metacognitive knowledge may prevent students from having negative metacognitive experiences.

5.2.1 Metacognitive knowledge of strategies helps students persist despite negative emotions and judgements. Four participants in our sample experienced negative metacognitive judgements and self-doubt in reaction to negative emotions and struggle, but regulated their emotions and re-engaged in tasks successfully. One of these participants, P17, experienced a shift in their behavior from initially intending to avoid challenges and seek help to actively employing strategies and re-engaging with the task. The other three participants did not exhibit any avoidance behaviors. We found that these four students had an awareness of their task progress and the strategies they could use to regulate their emotions and make progress on tasks. We noticed that when these participants had negative emotions and metacognitive experiences, they employed strategies such as taking a break (P17, P25), giving themselves encouragement (P1), and re-engaging in tasks using diverse strategies (P2). Overall, these students exhibited much more effective metacognitive strategies than those in the avoidance pattern, including planning, assessing their understanding, monitoring their process, and switching approaches when needed. They also used a much more diverse and mature set of disciplinary strategies, including debugging techniques like decomposing the task and tracing line-by-line execution, and referencing a broader set of external resources in addition to the course materials. In this section, we will provide a more detailed description of these four participants to characterize how they overcame struggles and ultimately grew confident in their capability.

P17 experienced emotional fluctuation and a loss of confidence in pursuing CS while completing the visualization project. Initially, she approached the task by planning what to create, considering which example code could assist her, and determining what errors she might encounter that would require a TA's help. However, by the fourth day, she felt overwhelmed by the assignment's complexity and questioned her ability to complete the homework or any other future homework. She decided to "take a break and stop the work for today in hopes of getting answers during office hours." Similar to the pattern in Section 5.1.1, P17 experienced negative emotions, made negative metacognitive judgements of herself and the task completion, and decided to ask TAs for help.

Though P17 had decided to wait for the next day and ask for help, in her retrospective interview, she shared her belief that TAs typically offer general guidance rather than specific solutions to student problems and expressed a desire to engage in trial-and-error independently before seeking help to effectively support her learning. She also mentioned that "usually if I just know I'm not doing anything productive, I'll wait for the next day." So the next day, after taking a break that helped her regulate emotions, she re-engaged in the task. She reported that she "tried to dissect the different shapes that were needed to create an animal." She explained in the interview that she changed her approach by using shapes that were already familiar. She said, "figuring out how to do the task gave me more self-confidence. This could easily be done for the rest of the shape." In the following days, she continued to utilize external resources and engage in trial-and-error to debug. With more work done, her confidence increased and her interest in pursuing a CS minor persisted. The case of P17 demonstrates the importance of emotional regulation to navigate through frustration, as well as the disciplinary strategies that allowed her to try alternate implementations for solving the problem, helping her overcome obstacles and build confidence in her programming abilities.

P1 expressed feeling stressed, frustrated, and under-confident in her coding ability at the beginning of the assignment. Notably, this was the first project-based assignment, rather than a problem set. P1 also explained in the retrospective interview that: "that jump from the first assignment to the second assignment, there was where I stopped feeling so confident in my programming skills." In her diaries, she said her code is not working as expected. Although she felt stressed and under-confident in her ability, she tried to regulate her emotions first: "I needed to ground myself. Give myself a pep talk in a sense since I was alone and in a panicked state." She also said that: "I knew I was going to struggle in this assignment, it does regard a subject I am not familiar with at all ... I knew I needed to try my very best with the information that was provided to me and what I could gather from other sources." Other than regulating emotions, she expressed the metacognitive feeling of unfamiliarity with the task topic and the need to learn from different resources. Therefore, she thought about what other strategies she could use, including referencing resources and decomposing the task into smaller parts and testing each part out to build the entire melody: "I started playing notes individually, to later started playing around with them and seeing what note combinations sounded best with my chords. I was able to define three functions and use them to build a melody I like." As she made progress, she felt proud of her outcome, and gained confidence with her coding abilities. She also expressed her increased interest in computing: "now that I'm seeing what code is and because I still have that curiosity, I would see myself taking other computer science courses." P1's experiences highlight the importance of emotional regulation and a strategic approach to overcoming challenges and fostering increasing confidence in programming.

P17 and P1 showed how they regulated their emotions first, and moved forward through applying disciplinary strategies. We also see other participants exhibiting their use of metacognitive regulation strategies such as monitoring their progress, evaluating their understanding, and planning. During the life cycle of P25's final homework, which required creating a data visualization with a selected dataset, he experienced several days of frustration when the code didn't execute as expected. At times, he would have self-doubt, questioning if computer science was the right path for him: "I don't know if computer science is for me"; "I felt like I was stuttering in computer science and not really getting it well". P25 shared that when he faced negative emotions, he would take a break to cope with the emotions first. When working on the task, he intentionally checked his understanding and monitored his progress. For example, at the start of the homework, he said that he asked himself "what I didn't know how to do when it came to data cleaning, and then asked myself how I could learn how to do it." Each day, he would ask himself how to approach the task, and what needed to be implemented. He also monitored progress by asking himself what was missing and why, as well as whether he needed to look up any information to complete the task. During the interview, when asked about his approach to completing the tasks, P25 said he transferred the learning strategies from a previous math class:

> It started because of my calculus class and I just realized that everyone was really good at a lot of small things that built up the algebra. And I felt like it was the same thing here. So it was like, what don't I know and how can I learn it? And I just applied the same thing to my computer science work. When I'm working on something very hard, [I ask myself] why can't I do this? And usually, it comes down to a lot of very simple things that add up. So if I know what I'm missing, then I can ask myself how do I add it in? ... Then I'll just look it up.

The way P25 worked through tasks demonstrated his awareness of the task progress and a systematic strategy for monitoring tasks and navigating challenges by checking his understanding. Though P25 has no prior experience in programming, he transferred the metacognitive strategies (evaluating understanding and monitoring) learned from another class and applied them to programming. He also reported frequently using external resources such as online tutorial videos or blog posts to address his uncertainty in understanding. Through his diaries, he documented a positive trajectory in his emotions, a growing confidence in his programming capability, and an increased interest in CS. The use of regulation strategies helps him re-engage in practice and overcome the struggles, negative emotions, and negative metacognitive experiences regarding his competence, highlighting the importance of self-regulated strategies in facilitating personal growth.

P2 also reported using metacognitive strategies such as planning how to approach tasks in advance, and placed clear value on the importance of planning in programming:

> From my personal perspective, it makes it easier when you know what specific tasks you're going to do instead of just trying to take on all of it at once. And I think it's a little bit like when you're cooking something, I think if that's a good analogy and you read through the recipe first, you're not scrambling in the middle of it to look for maybe you're missing an ingredient or [tool]. It saves time in the middle.

In his diary entries, P2 described planning in the first several days of working on his assignment, such as familiarizing himself with the task and looking for resources to gain a better understanding. However, on the last day of his diary, P2 reported feeling anxiety and frustration, especially since his assignment was due later that night. He made negative metacognitive judgements of his task completion, said "I initially felt a bit discouraged and did not think I could solve it at first." The time crunch made him "code in a rushed manner, I had more errors and had to make more corrections than I had on previous assignments." When he was reflecting in the interview, he said, "I think I was just for the sake of completing the assignment, I had just gone too quickly and focused much on just trying to get the code completed instead of like, is there a more efficient way that I can do this?" From P2's reflection, the get-it-done mindset while under time pressure superseded his belief that planning is important.

However, despite encountering errors and experiencing negative metacognitive feelings around his ability to fix errors, he remained engaged in the task. He reported that, based on the output, he identified the errors and proposed two alternative approaches to try. He also reported that "I ran through each line and worked out the logic step by step to understand where my logic and the computer's logic were not in agreement. I then made adjustments and ran the code again to ensure it was giving the result I wanted." By adopting a systematic debugging process, P2 reported that "with more time and corrections I felt better and ... more optimistic about my ability

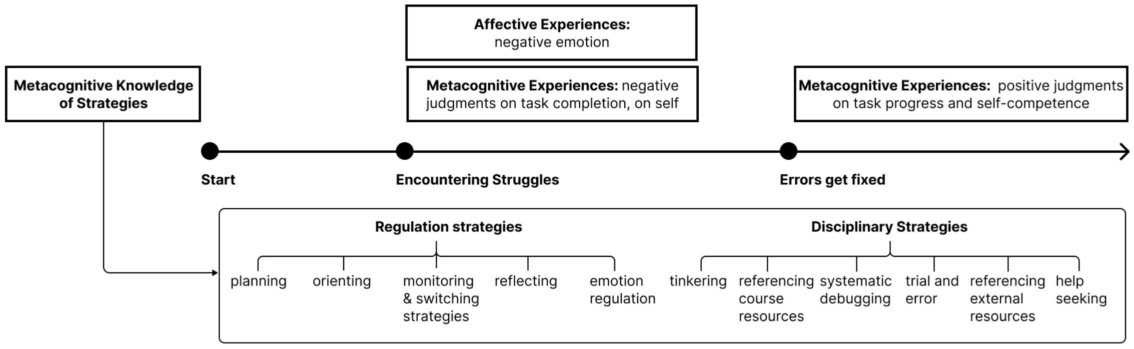
Affective Experiences: negative emotion
Metacognitive Knowledge of Strategies
Metacognitive Experiences: negative judgments on task completion, on self
Metacognitive Experiences: positive judgments on task progress and self-competence
Start
Encountering Struggles
Errors get fixed
Regulation strategies
Disciplinary Strategies
planning
orienting
monitoring & switching strategies
reflecting
emotion regulation
tinkering
referencing course resources
systematic debugging
trial and error
referencing external resources
help seeking

Figure 4: A synthesized diagram that demonstrates all participants' cases falling under the persistence or re-engagement despite negative emotions and judgements pattern

to troubleshoot code in the future." From the experience of P2, we see the importance of having metacognitive knowledge of disciplinary strategies such as identifying errors, considering alternative approaches, and employing debugging techniques. These strategies helped P2 resolve his errors and build confidence in his ability to debug code effectively in the future, despite initially struggling and having negative affective and metacognitive experiences.

In this section, we presented a different pattern in response to struggles, negative emotions, and negative metacognitive experiences. Figure 4 is a diagram showing the persistence pattern with the help of metacognitive knowledge of strategies that combined all of our participants' cases and strategies mentioned. Instead of avoiding their work and relying on TAs, these participants demonstrated an awareness of their emotions, progress, and the variety of strategies available to help them make progress. Through employing regulation strategies and disciplinary strategies, they persisted, overcame struggles, and built confidence in their programming capability.

5.2.2 Metacognitive knowledge of strategies helps prevent negative metacognitive experiences. Four of our participants did not experience negative metacognitive feelings and judgements about themselves or their task progress, even though they encountered struggles and felt frustrated. They all started the task early and progressed daily with a goal in mind, demonstrating a good awareness of themselves, their task progress, and potential strategies for approaching the task.

Throughout P20's diaries, she documented facing struggles almost daily, which often led to feelings of frustration or stress. However, she never mentioned questioning her ability to complete the tasks or in general. Instead, she adopted different strategies to engage with the tasks, such as applying what she learned from instructors' demonstrations during the lecture to understand error messages and readjust her code , employing trial-and-error, referencing course materials, and looking up documentation. During the interview, P20 explained why she didn't question herself or let her emotions influence her behaviors:

> I found that if I am frustrated or if I get angry, I'm never productive. I try to move past that as quickly as possible. And I think what also helped was that I knew I still had a week left to do this. Even when I had frustrating times, it didn't really affect how I believed about myself or how I actually behaved. I knew I had time, I knew I was on a good track to get there, and I knew what I wanted to do at the end. So I think that helped me get through everything.

She acknowledged that feeling frustrated or upset would impede her progress. Her metacognitive judgements of the time remaining and her task progress prevented her from negative metacognitive experiences and helped her stay engaged in the task. Her reflection emphasized that having metacognitive judgements of task progress and metacognitive knowledge of strategies can help with managing negative emotions and persisting through struggle.

P8 also showed how planning, monitoring the task, and adjusting her approach helped her with both completing the task and regulating her emotions. P8 reported feeling frustrated the first day because "I didn't plan anything and I was just trying to go from the beginning. It just seemed like a lot, and everything was in disarray. I felt like I spent most of my time trying to figure things out." Consequently, at the end of the first day, she reflected and decided to "try a more organized approach". She explained in her retrospective interview that:

> I started planning things out because I figured out it wasn't working. It doesn't really mean anything when you say you start working on it. It probably stresses you out more … if you have a clear plan, it breaks things down into something that's more palatable for me to do … even if you didn't finish the thing, you could be like, oh, I finished what I was supposed to do … not the whole, but that is something.

We observed from P8's experiences that she actively monitored her task progress, reflected, and adjusted her strategy for approaching the task. According to P8, planning not only helped her make progress on the problem, but also helped her regulate her emotions. As she said, having and following a plan helped her monitor her progress and feel accomplished despite only completing part of the assignment.

Both P27 and P9 demonstrated their metacognitive knowledge of regulation strategies and various disciplinary strategies for approaching their assignments. P9 shared her approach of writing

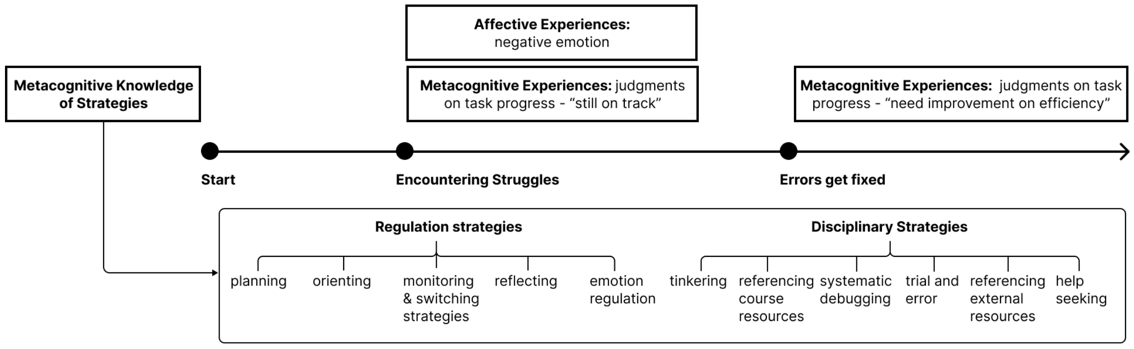
Affective Experiences: negative emotion
Metacognitive Knowledge of Strategies
Metacognitive Experiences: judgments on task progress - "still on track"
Metacognitive Experiences: judgments on task progress - "need improvement on efficiency"
Start
Encountering Struggles
Errors get fixed
Regulation strategies
Disciplinary Strategies
planning
orienting
monitoring & switching strategies
reflecting
emotion regulation
tinkering
referencing course resources
systematic debugging
trial and error
referencing external resources
help seeking

Figure 5: A synthesized diagram that demonstrates all participants' cases falling under the non-negative metacognitive experiences pattern

down the structure of the code after understanding the homework requirements: "I write the bones and then I go back to step one and fill them in." And while working through the problem, despite feeling overwhelmed with error messages, she shared that "I overcame my struggles by taking a deep breath, using my resources, and patiently debugging my code until the functions with parameters worked properly." P27 also exhibited behaviors of monitoring and adjusting her strategies when she felt confused or frustrated while encountering struggles. She "searched up and tried to note the confusion down" and came back to it the next day to help regulate her emotions. She also felt annoyed when the code output differed from her expectations. However, she switched approaches and eventually felt accomplished.

Figure 5 is a synthesis of these four participants who experienced negative emotions when struggling, but who demonstrated metacognitive knowledge of both regulation strategies and disciplinary strategies. Overall, these four participants demonstrated that their monitoring of themselves and their progress, and their employment of strategies, were instrumental in helping them recover from negative emotions, avoid negative metacognitive judgements, and ultimately complete the task and foster self-confidence.

6 DISCUSSION

Through our analysis of students' daily diaries and retrospective interviews, we identified three behavioral patterns that arose when students encountered struggles and had negative affective and metacognitive experiences. Figure 6 presents an initial model depicting the interplay between metacognition, affect, and task engagement based on our data.

When students encounter struggles, they often have an affective response and engage in metacognitive judgements of themselves, their task progress, and their learning. The first pattern we presented (Section 5.1.1) describes cases where these affective and metacognitive experiences were so overpowering that students could not persist in the task. These learners described negative emotions such as discouragement, fatigue, and frustration, coupled with judgements that they could not resolve errors or complete the task. These factors impacted students' task engagement, leading them to avoid work and seek external help. Negative affective and metacognitive experiences may even exacerbate emotions and contribute to the formation of negative, stable self-beliefs associated with CS. Participants who fall under this pattern also employed a limited set of strategies and often used them ineffectively, suggesting that they had insufficient metacognitive knowledge of strategies, which also influenced task engagement. Finally, we notice a circular effect, where the decision to disengage results in students experiencing catastrophizing emotions and judgements about themselves (Section 5.1.2).

The second pattern (Section 5.2.1) describes cases where participants experienced similar affective and metacognitive experiences as those in the first pattern, but where they ultimately persisted rather than disengaging from the task. One notable difference between students in this pattern and the first one is that they effectively employed regulation strategies with the goal of gaining metacognitive awareness of their progress and emotional status, as well as a variety of disciplinary strategies, which helped them persist through struggles, positively influencing task engagement. This suggests that students in this pattern had better metacognitive knowledge of strategies. Successfully persisting through struggles also had a positive impact on these students' affective and metacognitive experiences.

The third pattern (Section 5.2.2) describes cases where participants experienced negative affective experiences but did not engage in negative metacognitive judgements of themselves. These students also exhibited strong metacognitive knowledge of strategies through their effective employment of both regulation and disciplinary strategies. They consistently monitored their progress and evaluated their understanding, thereby influencing their affect and metacognitive experiences. It is possible that because these students were able to make judgements of their task progress and confirm that they were still on track, they did not experience the emotional spiral and negative judgements we saw in other participants. In whatever case, the students who employed regulation and disciplinary strategies effectively were also observed to persist on the task despite negative emotions, and have more positive metacognitive experiences.

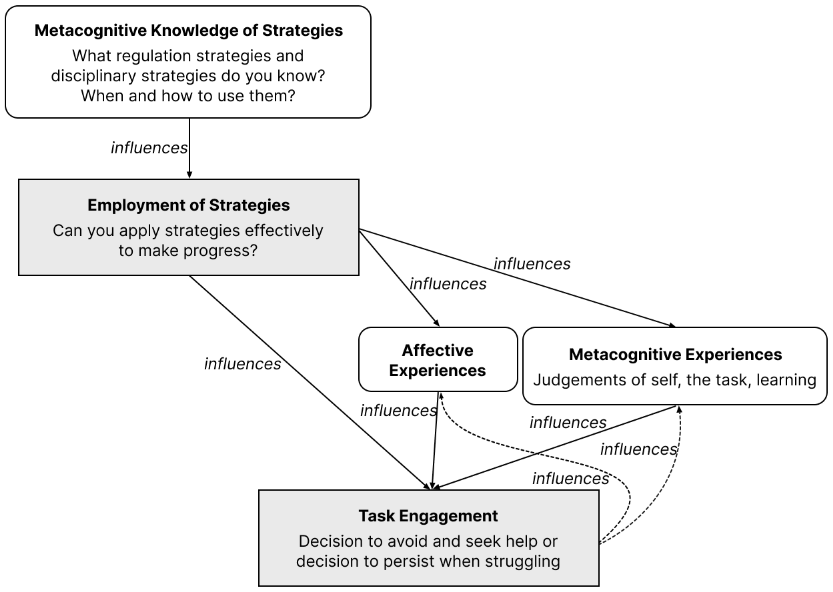
Metacognitive Knowledge of Strategies
What regulation strategies and disciplinary strategies do you know? When and how to use them?
influences
Employment of Strategies
Can you apply strategies effectively to make progress?
influences
influences
influences
Affective Experiences
Metacognitive Experiences
Judgements of self, the task, learning
influences
influences
influences
influences
Task Engagement
Decision to avoid and seek help or decision to persist when struggling

Figure 6: An initial model demonstrating the interplay of metacognition, affect and task engagement. Negative affective and metacognitive experiences impacted students' task engagement, leading them to avoid work and seek external help. However, having access to metacognitive knowledge of regulation and disciplinary strategies helped students persist through negative affective and metacognitive experiences, leading to sustained task engagement. The dotted line suggests that the decision to avoid or persist could further influence students' affective and metacognitive experiences.

6.1 Design Implications

The model we present in Figure 6 positions metacognitive knowledge of strategies as the root cause underlying the differences we observed in student task engagement. Based on these findings, we believe that students could overcome struggles and negative affective and metacognitive experiences through more effective employment of strategies. To use strategies effectively, students must possess metacognitive knowledge of strategies, including knowing what strategies exist and when and how to use them [20, 58]. Below we outline design opportunities grounded in these findings.

First, we believe that learners would benefit from explicit interventions around metacognitive knowledge of strategies, which could be delivered through lectures, TA office hours, or computer-based systems. Such interventions should go beyond simply introducing strategies; they should also explain why these strategies are beneficial, when they should be employed, and how to utilize them effectively. For instance, we found that some students did not know how to progress; in such cases, an intervention could suggest potential strategies to use. We also found that some students were aware of strategies such as trial-and-error but struggled to use them effectively, which could stem from a lack of metacognitive knowledge of how to employ them well (Section 5.1.1). In such cases, providing support around how to apply strategies could improve students' ability to navigate challenges. This aligns with existing literature, which suggests that instructors could model when and how to use strategies and provide opportunities for students to practice them [57].

Second, we recommend teaching both regulation strategies and disciplinary strategies. Our findings highlight that students equipped with regulation strategies for task completion and emotional management were better able to stay engaged with the task, and had more positive affective and metacognitive experiences. This aligns with studies indicating that students achieve better academic performance with robust metacognitive regulation [7, 57, 63]. One key difference between participants who fell into the disengagement pattern and those who remained engaged is their use of monitoring strategies. Participants who actively checked their progress, evaluated what knowledge they needed to acquire, switched implementation strategies, and updated their plans were able to prevent emotional spirals and overcome struggles. In contrast, those who did not use monitoring strategies to regulate their learning progress experienced negative emotions and catastrophizing metacognitive judgements. This suggests that designing interventions to prompt or scaffold students to check their progress and consider alternative plans before their emotions spiral could help them navigate challenges more effectively.

Further, our study underscores the importance of emotional regulation. We found that intense negative emotions often lead to task disengagement. Previous designs for developing disciplinary strategies and supporting metacognition have not considered student's affective experiences while learning [40, 64, 65]. While these interventions teach students strategies and facilitate monitoring and reflection, it remains challenging for students to apply these strategies when they experience negative emotions and self-judgements. In our study, the participants who persisted often intentionally regulated their emotions first. Therefore, there is a need to design interventions that increase students' emotional awareness and help them employ emotional regulation strategies and re-engage when they experience negative emotions.

6.2 Limitations and Future Work

We recognize a few limitations to our work that can be addressed in the future. First, participants self-reported how they approach programming practices through diaries. Many students are beginners who might not report their strategy usage accurately and comprehensively. As a result, future work could use programming environment log data to triangulate the diaries and interview data. Second, our participants are from a single CS0 course and university, so we do not know whether these findings will generalize to other courses and contexts. Future work should study other populations of students. Finally, the initial model we present describes the relationships we saw between metacognitive knowledge of strategies, metacognitive experiences, affective experiences, and behaviors. However, the learning process is influenced by many other factors including motivation, self-concepts, and social comparison. Future work should consider these factors to expand our model.

7 CONCLUSION

This paper explores an initial model of how metacognition, affect, and students' behaviors interact in students' learning experiences. Our results demonstrate three different behavioral patterns in response to struggles. We highlight the importance of metacognitive knowledge of both regulation and disciplinary strategies in overcoming struggles, mitigating negative emotions and judgements of self and task progress, and fostering confidence in programming ability. We further discuss opportunities for designing interventions to foster the development of metacognitive knowledge of strategies.

ACKNOWLEDGMENTS

This project is supported by the National Science Foundation under Grant IIS-2016900, and a Design Cluster Research Fellowship from the Center for Human-Computer Interaction and Design at Northwestern University. We thank Michael Horn and the Delta Lab for their feedback, and the participants and instructors of this course for their time.

REFERENCES

[1] Noura Albarakati, Lisa DiPippo, and Victor Fay-Wolfe. 2021. Rethinking CS0 to improve performance and Retention. *Proceedings of the 23rd Australasian Computing Education Conference* (Feb 2021). https://doi.org/10.1145/3441636.3442314

[2] Noura Albarakati, Lisa DiPippo, and Victor Fay-Wolfe. 2021. Rethinking CS0 to improve performance and Retention. *Proceedings of the 23rd Australasian Computing Education Conference* (Feb 2021). https://doi.org/10.1145/3441636.3442314

[3] Susan A. Ambrose. 2010. *How learning works: Seven research-based principles for smart teaching.*

[4] Susan Bergin, Ronan Reilly, and Desmond Traynor. 2005. Examining the role of self-regulated learning on Introductory programming performance. *Proceedings of the 2005 international workshop on Computing education research - ICER '05* (2005). https://doi.org/10.1145/1089786.1089794

[5] Monique Boekaerts. 1997. Self-Regulated Learning: A new concept embraced by researchers, policy makers, educators, teachers, and students. *Learning and Instruction* 7, 2 (Jun 1997), 161–186. https://doi.org/10.1016/s0959-4752(96)00015-1

[6] Ann L. Brown. 1978. *Knowing when, where, and how to remember: A problem of metacognition.* Eric Reports.

[7] Michelene T.H. Chi, Miriam Bassok, Matthew W. Lewis, Peter Reimann, and Robert Glaser. 1989. Self-explanations: How students study and use examples in learning to solve problems. *Cognitive Science* 13, 2 (Apr 1989), 145–182. https://doi.org/10.1207/s15516709cog1302_1

[8] Lyn Corno. 1986. The metacognitive control components of self-regulated learning. *Contemporary Educational Psychology* 11, 4 (Oct 1986), 333–346. https://doi.org/10.1016/0361-476x(86)90029-9

[9] Thomas J. Cortina. 2007. An introduction to computer science for non-majors using principles of computation. *Proceedings of the 38th SIGCSE technical symposium on Computer science education* (Mar 2007). https://doi.org/10.1145/1227310.1227387

[10] Jessica Q. Dawson, Meghan Allen, Alice Campbell, and Anasazi Valair. 2018. Designing an introductory programming course to improve Non-Majors' experiences. *Proceedings of the 49th ACM Technical Symposium on Computer Science Education* (Feb 2018). https://doi.org/10.1145/3159450.3159548

[11] Hester de Boer, Anouk S. Donker, Danny D.N.M. Kostons, and Greetje P.C. van der Werf. 2018. Long-term effects of metacognitive strategy instruction on student academic performance: A meta-analysis. *Educational Research Review* 24 (Jun 2018), 98–115. https://doi.org/10.1016/j.edurev.2018.03.002

[12] Carol S. Dweck. 1999. *Self-theories: Their role in motivation, personality, and development.* Psychology Press.

[13] Carol S. Dweck. 2006. *Mindset: the new psychology of Success.* Random House.

[14] Anastasia Efklides. 2001. Metacognitive experiences in problem solving. *Trends and Prospects in Motivation Research* (2001), 297–323. https://doi.org/10.1007/0-306-47676-2_16

[15] Anastasia Efklides. 2006. Metacognition and affect: What can metacognitive experiences tell us about the learning process? *Educational Research Review* 1, 1 (Jan 2006), 3–14. https://doi.org/10.1016/j.edurev.2005.11.001

[16] Anastasia Efklides. 2011. Interactions of metacognition with motivation and affect in self-regulated learning: The MASRL model. *Educational Psychologist* 46, 1 (Jan 2011), 6–25. https://doi.org/10.1080/00461520.2011.538645

[17] Anneli Eteläpelto. 1993. Metacognition and the expertise of computer program comprehension. *Scandinavian Journal of Educational Research* 37, 3 (Jan 1993), 243–254. https://doi.org/10.1080/0031383930370305

[18] Katrina Falkner, Rebecca Vivian, and Nickolas J.G. Falkner. 2014. Identifying computer science self-regulated learning strategies. *Proceedings of the 2014 conference on Innovation & technology in computer science education - ITiCSE '14* (2014). https://doi.org/10.1145/2591708.2591715

[19] Matthias Felleisen, Robert Bruce Findler, Matthew Flatt, and Shriram Krishnamurthi. 2004. The teachscheme! project: Computing and programming for every student. *Computer Science Education* 14, 1 (Jan 2004), 55–77. https://doi.org/10.1076/csed.14.1.55.23499

[20] John H. Flavell. 1979. Metacognition and cognitive monitoring: A new area of cognitive-developmental inquiry. *American Psychologist* 34, 10 (1979), 906–911. https://doi.org/10.1037//0003-066x.34.10.906

[21] Joseph P Forgas. 1994. The role of Emotion in Social Judgments: An introductory review and an affect infusion model (AIM). *European Journal of Social Psychology* 24, 1 (Jan 1994), 1–24. https://doi.org/10.1002/ejsp.2420240102

[22] Jamie Gorson and Eleanor O'Rourke. 2019. How Do Students Talk About Intelligence? An Investigation of Motivation, Self-efficacy, and Mindsets in Computer Science. In *Proceedings of the 2019 ACM Conference on International Computing Education Research* (Toronto ON, Canada) *(ICER '19)*. Association for Computing Machinery, New York, NY, USA, 21–29. https://doi.org/10.1145/3291279.3339413

[23] Jamie Gorson and Eleanor O'Rourke. 2020. Why do CS1 students think they're bad at programming? *Proceedings of the 2020 ACM Conference on International Computing Education Research* (Aug 2020). https://doi.org/10.1145/3372782.3406273

[24] Lin Guo. 2022. Using metacognitive prompts to enhance self-regulated learning and learning outcomes: A meta-analysis of experimental studies in computer-based Learning Environments. *Journal of Computer Assisted Learning* 38, 3 (Feb 2022), 811–832. https://doi.org/10.1111/jcal.12650

[25] Mark Guzdial and Andrea Forte. 2005. Design process for a non-majors computing course. *Proceedings of the 36th SIGCSE technical symposium on Computer science education* (Feb 2005). https://doi.org/10.1145/1047344.1047468

[26] Emma Hogan, Ruoxuan Li, and Adalbert Gerald Soosai Raj. 2023. CS0 vs. CS1: Understanding Fears and Confidence amongst Non-majors in Introductory CS Courses. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1* (Toronto ON, Canada) *(SIGCSE 2023)*. Association for Computing Machinery, New York, NY, USA, 25–31. https://doi.org/10.1145/3545945.3569865

[27] Sorva Juha. 2013. Notional machines and introductory programming education. *ACM Trans. Comput. Educ* 13, 2 (2013), 1–31.

[28] Judy Kay, Michael Barg, Alan Fekete, Tony Greening, Owen Hollands, Jeffrey H. Kingston, and Kate Crawford. 2000. Problem-based learning for foundation computer science courses. *Computer Science Education* 10, 2 (Aug 2000), 109–128. https://doi.org/10.1076/0899-3408(200008)10:2;1-c;ft109

[29] Michelle E. Kiger and Lara Varpio. 2020. Thematic analysis of qualitative data: Amee Guide no. 131. *Medical Teacher* 42, 8 (May 2020), 846–854. https://doi.org/10.1080/0142159x.2020.1755030

[30] Paivi Kinnunen and Beth Simon. 2010. Experiencing programming assignments in CS1. *Proceedings of the Sixth international workshop on Computing education research* (Aug 2010). https://doi.org/10.1145/1839594.1839609

[31] Päivi Kinnunen and Beth Simon. 2012. My program is ok – am I? computing freshmen's experiences of doing programming assignments. *Computer Science Education* 22, 1 (Mar 2012), 1–28. https://doi.org/10.1080/08993408.2012.655091

[32] Asher Koriat and Morris Goldsmith. 1996. Monitoring and control processes in the strategic regulation of memory accuracy. *Psychological Review* 103, 3 (1996), 490–517. https://doi.org/10.1037//0033-295x.103.3.490

[33] Deanna Kuhn. 2000. Metacognitive development. *Current Directions in Psychological Science* 9, 5 (Oct 2000), 178–181. https://doi.org/10.1111/1467-8721.00088

[34] Harrison Kwik, Haoqi Zhang, and Eleanor O'Rourke. 2022. How Do Students Seek Help and How Do TAs Respond?: Investigating Help-Seeking Strategies in CS1 Office Hours. In *SIGCSE 2022: The 53rd ACM Technical Symposium on Computer Science Education, Providence, RI, USA, March 3-5, 2022, Volume 2*, Larry Merkle, Maureen Doyle, Judithe Sheard, Leen-Kiat Soh, and Brian Dorn (Eds.). ACM, 1130. https://doi.org/10.1145/3478432.3499099

[35] Kathleen J. Lehman, Julia Rose Karpicz, Veronika Rozhenkova, Jamelia Harris, and Tomoko M. Nakajima. 2021. Growing Enrollments Require Us to Do More: Perspectives on Broadening Participation During an Undergraduate Computing Enrollment Boom. In *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education* (Virtual Event, USA) *(SIGCSE '21)*. Association for Computing Machinery, New York, NY, USA, 809–815. https://doi.org/10.1145/3408877.3432370

[36] Paul Luo Li, Amy J. Ko, and Andrew Begel. 2019. What distinguishes Great Software Engineers? *Empirical Software Engineering* 25, 1 (Dec 2019), 322–352. https://doi.org/10.1007/s10664-019-09773-y

[37] Yinmiao Li, Haoqi Zhang, and Eleanor O'Rourke. 2024. The Undervalued Disciplinary and Emotional Support Provided By Teaching Assistants in Introductory Computer Science Courses. *International Society of the Learning Sciences* (2024), 1498–1501.

[38] Alex Lishinski and Joshua Rosenberg. 2021. All the pieces matter: The relationship of momentary self-efficacy and affective experiences with CS1 achievement and interest in computing. *Proceedings of the 17th ACM Conference on International Computing Education Research* (Aug 2021). https://doi.org/10.1145/3446871.3469740

[39] Alex Lishinski, Aman Yadav, Jon Good, and Richard Enbody. 2016. Learning to program: Gender differences and interactive effects of students' motivation, goals, and self-efficacy on performance. *Proceedings of the 2016 ACM Conference on International Computing Education Research* (Aug 2016). https://doi.org/10.1145/2960310.2960329

[40] Dastyni Loksa, Amy J. Ko, Will Jernigan, Alannah Oleson, Christopher J. Mendez, and Margaret M. Burnett. 2016. Programming, problem solving, and self-awareness. *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (May 2016). https://doi.org/10.1145/2858036.2858252

[41] Dastyni Loksa, Lauren Margulieux, Brett A. Becker, Michelle Craig, Paul Denny, Raymond Pettit, and James Prather. 2022. Metacognition and self-regulation in programming education: Theories and exemplars of use. *ACM Transactions on Computing Education* 22, 4 (Dec 2022), 1–31. https://doi.org/10.1145/3487050

[42] Yana Malysheva, John Allen, and Caitlin Kelleher. 2022. How do teaching assistants teach? characterizing the interactions between students and TAS in a computer science course. *2022 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)* (Sep 2022). https://doi.org/10.1109/vl/hcc53370.2022.9832962

[43] C. Mcdowell, L. Werner, H.E. Bullock, and J. Fernald. 2003. The impact of pair programming on student performance, perception and persistence. *25th International Conference on Software Engineering, 2003. Proceedings.* (2003). https://doi.org/10.1109/icse.2003.1201243

[44] Rodrigo Pessoa Medeiros, Geber Lisboa Ramalho, and Taciana Pontual Falcao. 2019. A systematic literature review on teaching and learning introductory programming in Higher Education. *IEEE Transactions on Education* 62, 2 (May 2019), 77–90. https://doi.org/10.1109/te.2018.2864133

[45] Tilman Michaeli and Ralf Romeike. 2019. Improving Debugging Skills in the Classroom: The Effects of Teaching a Systematic Debugging Process. In *Proceedings of the 14th Workshop in Primary and Secondary Computing Education* (Glasgow, Scotland, Uk) *(WiPSCE '19)*. Association for Computing Machinery, New York, NY, USA, Article 15, 7 pages. https://doi.org/10.1145/3361721.3361724

[46] Thomas O. Nelson. 1990. Metamemory: A theoretical framework and new findings. *Psychology of Learning and Motivation* (1990), 125–173. https://doi.org/10.1016/s0079-7421(08)60053-5

[47] Sharon Nelson-Le Gall. 1981. Help-seeking: An understudied problem-solving skill in children. *Developmental Review* 1, 3 (Sep 1981), 224–246. https://doi.org/10.1016/0273-2297(81)90019-8

[48] Yulia Pechorina, Keith Anderson, and Paul Denny. 2023. Metacodenition: Scaffolding the problem-solving process for novice programmers. *Proceedings of the 25th Australasian Computing Education Conference* (Jan 2023). https://doi.org/10.1145/3576123.3576130

[49] Paul R. Pintrich. 2000. The role of goal orientation in self-regulated learning. *Handbook of Self-Regulation* (2000), 451–502. https://doi.org/10.1016/b978-012109890-2/50043-3

[50] Paul R. Pintrich, Christopher A. Wolters, and Gail P. Baxter. 2000. *Assessing Metacognition and Self-Regulated Learning.* Lincoln, NE: Buros Institute of Mental Measurements, 43–97.

[51] Leo Porter and Beth Simon. 2013. Retaining nearly one-third more majors with a trio of instructional best practices in CS1. In *Proceeding of the 44th ACM Technical Symposium on Computer Science Education* (Denver, Colorado, USA) *(SIGCSE '13)*. Association for Computing Machinery, New York, NY, USA, 165–170. https://doi.org/10.1145/2445196.2445248

[52] Christine Reilly, Emmett Tomai, and Laura M. Grabowski. 2015. An evaluation of how changes to the introductory computer science course sequence impact student success. *2015 IEEE Frontiers in Education Conference (FIE)* (Oct 2015). https://doi.org/10.1109/fie.2015.7344029

[53] Linda J. Sax, Kathleen J. Lehman, and Christina Zavala. 2017. Examining the Enrollment Growth: Non-CS Majors in CS1 Courses. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education* (Seattle, Washington, USA) *(SIGCSE '17)*. Association for Computing Machinery, New York, NY, USA, 513–518. https://doi.org/10.1145/3017680.3017781

[54] Gregory Schraw and David Moshman. 1995. Metacognitive theories. *Educational Psychology Review* 7, 4 (Dec 1995), 351–371. https://doi.org/10.1007/bf02212307

[55] Duane F. Shell and Leen-Kiat Soh. 2013. Profiles of motivated self-regulation in college computer science courses: Differences in major versus required non-major courses. *Journal of Science Education and Technology* 22, 6 (Feb 2013), 899–913. https://doi.org/10.1007/s10956-013-9437-9

[56] Raja Sooriamurthi. 2009. Introducing abstraction and decomposition to novice programmers. *SIGCSE Bull.* 41, 3 (jul 2009), 196–200. https://doi.org/10.1145/1595496.1562939

[57] Julie Dangremond Stanton, Amanda J. Sebesta, and John Dunlosky. 2021. Fostering metacognition to support student learning and performance. *CBE—Life Sciences Education* 20, 2 (Jun 2021). https://doi.org/10.1187/cbe.20-12-0289

[58] Pina Tarricone. 2011. *The taxonomy of Metacognition.* Psychology Press.

[59] Christine Unterhitzenberger and Kate Lawrence. 2022. Diary method in Project Studies. *Project Leadership and Society* 3 (Dec 2022), 100054. https://doi.org/10.1016/j.plas.2022.100054

[60] Marcel Veenman and Jan J. Elshout. 1999. Changes in the relation between cognitive and metacognitive skills during the acquisition of expertise. *European Journal of Psychology of Education* 14, 4 (Dec 1999), 509–523. https://doi.org/10.1007/bf03172976

[61] Marcel V. Veenman, Bernadette H. Van Hout-Wolters, and Peter Afflerbach. 2006. Metacognition and learning: Conceptual and methodological considerations. *Metacognition and Learning* 1, 1 (Mar 2006), 3–14. https://doi.org/10.1007/s11409-006-6893-0

[62] J. D. Vermunt. 1996. Metacognitive, cognitive and affective aspects of learning styles and strategies: A phenomenographic analysis. *Higher Education* 31, 1 (Jan 1996), 25–50. https://doi.org/10.1007/bf00129106

[63] Margaret C. Wang, Geneva D. Haertel, and Herbert J. Walberg. 1990. What influences learning? A content analysis of review literature. *The Journal of Educational Research* 84, 1 (Sep 1990), 30–43. https://doi.org/10.1080/00220671.1990.10885988

[64] Benjamin Xie, Jared Ordona Lim, Paul K.D. Pham, Min Li, and Amy J. Ko. 2023. Developing novice programmers' self-regulation skills with code replays. *Proceedings of the 2023 ACM Conference on International Computing Education Research V.1* (Aug 2023). https://doi.org/10.1145/3568813.3600127

[65] Lisa Yan, Annie Hu, and Chris Piech. 2019. Pensieve: Feedback on Coding Process for Novices. *Proceedings of the 50th ACM Technical Symposium on Computer Science Education* (Feb 2019). https://doi.org/10.1145/3287324.3287483

[66] Qing Zhang and Barbara B. Lockee. 2022. Designing a framework to facilitate metacognitive strategy development in computer-mediated problem-solving instruction. *Journal of Formative Design in Learning* 6, 2 (Aug 2022), 127–143. https://doi.org/10.1007/s41686-022-00068-y

[67] Stuart Zweben. 2019. Enrollment and retention in U.S. computer science bachelor's programs in 2016-17. *ACM Inroads* 10, 4 (Nov 2019), 47–59. https://doi.org/10.1145/3366690