2020

What Do We Think We Think We Are Doing?: Metacognition and Self-Regulation in Programming

James Prather
Abilene Christian University

Brett A. Becker
University College Dublin

Michelle Craig
University of Toronto

Paul Denny
University of Auckland

Dastyni Loksa
University of Washington - Seattle Campus

See next page for additional authors

Recommended Citation

Prather, James; Becker, Brett A.; Craig, Michelle; Denny, Paul; Loksa, Dastyni; and Margulieux, Lauren, "What Do We Think We Think We Are Doing?: Metacognition and Self-Regulation in Programming" (2020). *Learning Sciences Faculty Publications*. 36.
doi: https://doi.org/10.1145/3372782.3406263

What Do We Think We Think We Are Doing?: Metacognition and Self-Regulation in Programming

James Prather
Abilene Christian University
Abilene, TX, USA
jrp09a@acu.edu

Brett A. Becker
University College Dublin
Dublin, Ireland
brett.becker@ucd.ie

Michelle Craig
University of Toronto
Toronto, Canada
mcraig@cs.toronto.edu

Paul Denny
University of Auckland
Auckland, New Zealand
paul@cs.auckland.ac.nz

Dastyni Loksa
University of Washington
Seattle, USA
dloksa@uw.edu

Lauren Margulieux
Georgia State University
Atlanta, USA
lmargulieux@gsu.edu

ABSTRACT

Metacognition and self-regulation are popular areas of interest in programming education, and they have been extensively researched outside of computing. While computing education researchers should draw upon this prior work, programming education is unique enough that we should explore the extent to which prior work applies to our context. The goal of this systematic review is to support research on metacognition and self-regulation in programming education by synthesizing relevant theories, measurements, and prior work on these topics. By reviewing papers that mention metacognition or self-regulation in the context of programming, we aim to provide a benchmark of our current progress towards understanding these topics and recommendations for future research. In our results, we discuss eight common theories that are widely used outside of computing education research, half of which are commonly used in computing education research. We also highlight 11 theories on related constructs (e.g., self-efficacy) that have been used successfully to understand programming education. Towards measuring metacognition and self-regulation in learners, we discuss seven instruments and protocols that have been used and highlight their strengths and weaknesses. To benchmark the current state of research, we examined papers that primarily studied metacognition and self-regulation in programming education and synthesize the reported interventions used and results from that research. While the primary intended contribution of this paper is to support research, readers will also learn about developing and supporting metacognition and self-regulation of students in programming courses.

CCS CONCEPTS

• **Social and professional topics → CS1**; • **Human-centered computing → User studies**.

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

KEYWORDS

cognition; CS1; metacognition; metacognitive awareness; programming; self-regulation; cognitive control

ACM Reference Format:
James Prather, Brett A. Becker, Michelle Craig, Paul Denny, Dastyni Loksa, and Lauren Margulieux. 2022. What Do We Think We Think We Are Doing?: Metacognition and Self-Regulation in Programming. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Computer programming is an inherently cognitive endeavor [17], making metacognition and self-regulation—collectively referred to as cognitive control—part of programming education. While computing education researchers should draw upon a wealth of work on cognitive control outside of our field, computing has unique characteristics that make learning and teaching particular topics arguably different than in other disciplines. For instance, it is normal for computing students in their first week of university to write what may be their first computer program. Students receive instantaneous and often difficult to decipher feedback in a mix of English and the likely new-to-them programming language when developing programs. This feedback can be provided anytime, often in the absence of an instructor [7], by a machine which is often believed to be infallible, yet for which many lack an accurate mental model. In addition, it has been noted that "Computer Science students struggle to develop fundamental programming skills and software development processes. Crucial to successful mastery is the development of discipline specific cognitive and metacognitive skills, including self-regulation" [23, p291]. Nonetheless, work on metacognition and self-regulation in Computer Science is limited [13].

Recent literature reviews suggest that metacognition and self-regulation are common topics of interest in computing education research, but there are few disciplinary-specific theories or measurements. A recent ITiCSE working group conducted a broad survey of learning theories used within the field and found that metacognition stood out as one of the top cited theories among 75 theories identified in computing education venues [67]. In addition, the co-occurrence of metacognition and self-efficacy in the literature formed one of the strongest connections between theories. Despite widespread interest in metacognition, a recent survey by Malmi et al. that reviewed theories specific to computing education found none on metacognition or self-regulation [38]. They contend that, while computing education research is informed by theories from psychology and general education literatures, discipline-specific theories are important to accurately represent what we know about how people learn programming.

Because these surveys of the literature establish metacognition and self-regulation as theories of particular interest to our community, this paper provides a focused review of their role in computing education research. We review current literature at the intersection of metacognition, self-regulation, and programming to explore theoretical bases and the instruments used to study these topics. We ground this systematic review in the wider context of research from psychology and education to examine which parts of that literature we have adopted and which parts are underutilized. Though the primary goal of the paper is to inform research, we hope that it is also useful for those who teach computer programming and design systems for students learning programming.

2 UNDERSTANDING METACOGNITION AND SELF-REGULATION

Metacognition and self-regulation are concepts that have been researched since the 1960s, but they are often still misunderstood and misused for a few fundamental reasons:

- They are internal processes, making them difficult to observe and measure reliably;
- They affect other memory, motivation, and emotion processing, making the boundary between them and other constructs difficult to determine;
- They are researched in different disciplines, such as psychology and education, meaning that definitions focus on different critical features and are translated inconsistently between disciplines.

The distinction between metacognition and self-regulation is also unclear. Some models of metacognition include self-regulation as a subcomponent and vice-versa. While definitive distinctions between the two do not exist, we will highlight the important features of, and differences between, metacognition and self-regulation.

Both metacognition and self-regulation are types of *cognitive control* [60]. As such, both involve planning strategies for learning and problem-solving, monitoring progress during a task, and evaluating the efficacy of strategies used. **Metacognition** tends to be the term used to describe **knowledge** about one's own cognitive control, including identifying strategies that have been successful or unsuccessful in the past, monitoring emotions and self-efficacy, and evaluating the validity of metacognitive knowledge based on feedback. Metacognition is most prevalent in psychology, educational psychology, and other psychology-based research. In contrast, **self-regulation** tends to be the term used to describe the process of **executing** cognitive control during a learning or problem-solving task. Thus, self-regulation focuses on selecting strategies for approaching a task, monitoring efficacy of strategy and time on task, and evaluating confidence that the task was completed accurately. Self-regulation is most prevalent in instructional design and other education-based research. Based on these different foci, it is appropriate that self-regulation would be a subcomponent of some metacognition models and vice versa. Metacognitive knowledge is important to self-regulated learning, and self-regulation of tasks is important to developing metacognition.

As an established field, metacognition, self-regulation, and self-regulated learning research has many discipline-independent reviews. Thus, we decided to limit our systematic literature review to papers published only in SIGCSE venues to best understand the state of research within this community and to inform future research. Our goal was to understand how theories of metacognition and self-regulation have guided the work of researchers investigating the learning of programming. Programming explicitly formalizes the process used to solve problems and, thus, is somewhat unique from other tasks used to study cognitive control, making the contribution of this paper different than other reviews. In addition, we sought to synthesize the measurements used to study cognitive control and the interventions used to improve or support learners' metacognitive and self-regulatory skills. As such, we hope the paper may serve as a starting point for researchers in the SIGCSE community to find relevant theories and methodologies that match their research contexts and to find relevant literature to inform their research goals. Our review was guided by two research questions:

- **RQ1**: What theories have been used by the SIGCSE community in the literature on metacognition and self-regulation in the context of programming education?
- **RQ2**: What activities and instruments that support and measure metacognition and self-regulation have been reported by the SIGCSE community in the literature on programming education?

In Section 3 we describe the methodology for our search and classification process, and we report the quantitative results of this process. In Section 4, we present theories of cognitive control and related concepts alongside the instruments for measuring various aspects of metacognition and self-regulation. Section 5 synthesizes key approaches and results from the studies which use cognitive control as a central theoretical basis.

3 METHODOLOGY

To address our research questions, we collected articles published in SIGCSE-sponsored or in-cooperation venues that referenced both programming and metacognition or self-regulation. We searched the ACM Full-Text Collection using the following search string: "programming" AND ("metacognition" OR "self-regulation"), using a full-text search of the content of each article and automatically matching variations of each term (e.g., metacognitive or self-regulated). We conducted this search on November 26, 2019 and found the following number of results:

- 2531 results (all venues across ACM Full-Text Collection)
- 357 results (when refined by sponsor: SIGCSE)

These numbers exclude duplicate articles that appeared in both a conference proceedings and in a reprint (e.g., SIGCSE Bulletin or ACM Inroads). We applied the following set of exclusion criteria to refine the entries under consideration:

- removed 5 items which were not papers (e.g. full conference proceedings, or symposium reports)

- removed 27 articles with two or fewer pages (to exclude posters and short papers)

We omitted articles with two or fewer pages because they do not provide enough space to thoroughly describe theory, methodology, and results and, therefore, would not provide adequate information to address our research questions. Applying these criteria resulted in 325 articles. We then conducted a manual search of these articles to exclude those for which the search terms occurred only in the References section of the paper. This ensured that the search terms appeared within the body of the article, resulting in the removal of a further 111 papers. Following these exclusion steps, we were left with a total of 214 papers to review from the following range of venues: ACE, ACSE, CompEd, FDG, ICER, ITiCSE (including working group reports), Koli Calling, SIGCSE, WCCCE, and WiPSCE.

In the final step of the review process, we categorized the extent to which the paper used theories and measurements of metacognition and self-regulation to guide the research and interpret results. A draft set of descriptors were defined for these categories, and all authors initially read and classified the same subset of eight papers, selected at random. The descriptors were then refined following a round of discussion involving all authors. The final categories used to classify the papers are shown in Table 1.

Table 1: Categories and descriptors for paper classification

Category	Descriptor
passing	the terms are mentioned in passing only, often in only one sentence
peripheral	the terms are discussed more thoroughly, but are not used to support the methodology of the paper or interpret the results
depth	the paper uses metacognition or self-regulation as a central theoretical basis of the study and measures and/or interprets the results using these theories

The 214 papers were divided into three groups as there were three pairs of authors, and each pair took one group for classification. Within pairs, after both authors had independently classified each paper, the pair met to reconcile differences. At this stage, seven papers were excluded because they used inclusion keywords in a completely different context to the one we are investigating, bringing the total down to 207 papers. For instance, one was a column on ethical responsibilities of computer professionals which called for a "self-regulated certification process". Table 2 provides statistics for our reconciliations including those that were simple oversights on the part of one of the authors and those that required non-trivial reconciliation. In Table 2 we also report the pairwise inter-rater reliability (IRR) using the unweighted Cohen's Kappa statistic based on these non-trivial reconciliations. Cohen's Kappa was > 0.81 (very good agreement) for all pairs. Calculating an all-author IRR would have required all authors to classify all papers (or a statistically representative subset). Given that our pair-wise IRR figures are considered very good, we deemed this unnecessary. We would expect an all-author IRR to be lower, but not enough to warrant concern. In addition, IRR values are not absolutely central to this study. Our research questions could still be answered had these been low or not calculated, but the results might have been less complete.

We further categorized the 31 papers identified as *depth*. Two researchers independently coded these by identifying the types of interventions depicted within the papers. These post-hoc categories and the papers within them are discussed in Section 5.

Table 2: Statistics for classification of *n* papers by three groups (G) of coders showing inter-rater reliability (IRR). Paper types are PAssing, PEripheral, DEpth. Reconciliation types are oversight (Rec OS) or non-trivial (Rec NT)

G	*n*	PA	PE	DE	Rec OS	Rec NT	IRR
A	67	37	20	10	13	5	0.899
B	71	39	15	17	12	4	0.907
C	69	47	18	4	16	2	0.937
Total	207	123	53	31	41	11	

3.1 Threats to validity

The intended scope and audience of this study is the SIGCSE community. Therefore when deciding which literature to review, we selected papers published in the ACM Digital Library at venues sponsored by SIGCSE. While we supplemented this paper with background and theory-focused papers from other communities, our systematic review includes only SIGCSE publications. Members of the SIGCSE community may have published papers in other venues not represented here. In addition, authors may have published work about cognitive control without using our search terms. We know of one paper that, had it made it into the initial dataset, would have been considered a depth paper because its focus was on in-process scaffolding very similar to Prather et al. [53], building on the work of Loksa et al. [37]. However, it was excluded early in the process because it did not use the terms explicitly. This oversight suggests that our methods may have excluded other papers that could have been included. We do not expect that the number of overlooked papers would be substantial; therefore, we do not expect that potential oversights would substantially diminish the contributions of the paper. Our primary goals were to survey the literature, identify themes, and create a resource that can support and guide future work. The synthesis presented in the following sections would likely not be materially different by including papers that report on cognitive control without using the terms metacognition or self-regulation.

4 RESULTS

Interest in metacognition and self-regulation has accelerated sharply in recent years. The earliest paper in our dataset dates back to 1978, yet only 20% of articles appeared prior to 2010. Half of the papers (106) were published in the last 5 years (2015 or later) and the two years with greatest number of papers were 2018 (with 24) and 2019 (with 35). This trend illustrates growing recognition of the importance of cognitive control within computing education research and suggests a review of the literature thus far is timely.

We now discuss the types of papers covered in our review and the theories and instruments for metacognition and self-regulation found in papers we identified as *peripheral* and *depth*.

4.1 Papers tagged as *passing*, *peripheral*, and *depth*

Nearly two-thirds (123 of 207) of the papers we reviewed made only *passing* reference to cognitive control. One interpretation of this is that many authors recognize the relevance of metacognition and self-regulation to programming education, but the connections made were often superficial. We frequently observed terms used in isolation when claiming techniques would "develop metacognition" or help students "self-regulate", often without further elaboration, evidence, or references. For example, in one paper the only mention of metacognition or self-regulation was the sentence "This helps to develop students' metacognition with regard to virtual classroom use, which can enhance their capacity to utilize the environment effectively." A claim like this needs to be supported by evidence and buttressed by theory. Other papers classified as *passing* did not make explicit connections to self-regulation or metacognition, but were motivated by related concepts such as self-efficacy. One such paper included metacognition in a list of skills that could be developed by social learning activities, but did not connect this to other parts of the research on self-efficacy.

We classified 53 papers as *peripheral*. These papers discussed metacognition and self-regulation literature, usually in a related work section, but did not use this literature as a theoretical basis for the study. About half of them discussed only studies that focused on metacognition or self-regulation published in computing education conferences. The other half discussed metacognition or self-regulation in light of theories from education, psychology, and cognitive science. Twelve of these papers employed some kind of measurement from outside the discipline: MSLQ (5), SRQ-L (1), SPOCK (1), Bloom's Taxonomy (3), EBQ (1), and Growth Mindset (1). We review these measurements and others in Section 4.3. However, even though these papers used these measurements, the results were ancillary to the study and sometimes not discussed. These papers also mentioned multiple theories related to cognitive control, such as Temporal Motivation Theory, Cognitive Load Theory, and Self-Efficacy. These theories are included in Section 4.2.

In Section 5, we synthesize the 31 papers tagged as *depth* into categories of similar research approaches and highlight pedagogical implications. Below we discuss in detail the theories and instruments used in the papers tagged as *peripheral* and *depth*.

4.2 Theories of metacognition and self-regulation

We center our discussion on four theories of cognitive control that are commonly used outside of our field and that we found to be commonly used in computing education research. Then we discuss 1) other less common theories used in our field, 2) theories related to cognitive control used in our field, and 3) other common theories of cognitive control that are underutilized in our field.

Flavell's model [24] is the foundation of most contemporary theories and research about metacognition. Flavell described metacognition as "knowledge or cognitive activity that takes as its object, or regulates, any aspect of any cognitive enterprise" [24, p104]. This definition broadly includes learning, problem-solving, and other types of cognitive tasks. Typically in educational settings, Flavell's definition is applied to predicting performance and monitoring current levels of mastery [12]. The seminal book "How People Learn" [12] expands upon Flavell's work with more recent and education-focused literature, adding that metacognition includes self-regulation but stipulates that self-regulation is difficult in disciplines about which the learner has little content knowledge.

A foundational self-regulation theory differs from Flavell's theory of metacognition by focusing on cognitive control's influence over social and environmental factors of behavior. Bandura [5] incorporated self-regulation into his social cognitive theory of behavior. He defined self-regulation as self-observation, self-judgment, and self-reflection, which served as an additional influence on behavior beyond social and environmental influences. As such, cognitive control may override other predictors of behavior.

The last two major theories we will discuss build upon Bandura's work but focus on tasks related to learning to explain self-regulated learning. Zimmerman [74] defines self-regulated learning as a three-phase process in which learners consciously plan, practice (i.e., monitor), and evaluate their progress towards achieving learning goals. Pintrich and de Groot [51] differ by focusing on the relationship between self-regulated learning and motivation. Their model has four phases, 1) forethought, planning, and activation, 2) monitoring, 3) control, and 4) reaction and reflection. Each phase is defined by four components: cognition, motivation/affect, behavior, and context. It is also unique compared to previous theories in that it explicitly includes prior knowledge activation.

4.2.1 Other Theoretical Bases in CEdR. Research in computing education builds upon a broad range of work about cognitive control other than these main theories. One of the most common frameworks that we identified from our review was the Revised Bloom's Taxonomy [10]. This taxonomy is a framework for categorizing learning objectives rather than a theory about cognitive control. The original Bloom's Taxonomy did not include metacognition or self-regulation in its six levels of learning objectives, but the revised version includes metacognitive knowledge as one of the added knowledge dimensions that learners can develop. Content knowledge was the only knowledge dimension that learners could develop in the original taxonomy [10].

Focusing on the learner rather than on learning objectives, Ertmer & Newby's model of an expert learner heavily includes cognitive control [22]. Their model states that expert learners take a systems-view including themselves, task requirements, and strategies to select, control, and monitor progress towards learning goals, i.e., use metacognitive knowledge to self-regulate learning. Furthermore, they define the two components of expert learning as metacognitive knowledge (cognitive, motivational, and environmental) of the task requirements and of personal resources and metacognitive control (or self-regulation) of strategy selection, task monitoring, and strategy evaluation. All of these features are common in previously discussed theories of cognitive control.

4.2.2 Theories and Concepts Related to Cognitive Control. Cognitive control is related to many other motivational, cognitive, and social constructs that can affect learning experiences and outcomes. The most commonly associated construct is self-efficacy [67], which stems from Bandura's Self-Efficacy Theory [6]. This theory is a precursor to Bandura's social cognitive theory of behavior, which would later include self-regulation in 1986 as discussed earlier in this section. Self-efficacy is an individual's belief that they can achieve a goal or accomplish a task. It determines how learners react to obstacles and failure, particularly whether they engage in coping behavior rather than defensive behavior and expend additional effort or change tasks, and for how long. Self-efficacy matters for cognitive control because metacognition and self-regulation are often directly related to self-efficacy but often only indirectly related to performance. Self-efficacy is the stronger, direct predictor of performance [15] [35] [43]. In addition to self-efficacy, many other theories and concepts are related to cognitive control, albeit less commonly. In the papers included in our review, researchers studied cognitive control in tandem with the theories listed in Table 3.

4.2.3 Cognitive Control Theories Underutilized in CEdR. In this section we discuss some common theories of cognitive control from literature in psychology and education that appeared only once or were absent from the papers in our systematic literature review. The first is Boekaerts' Adaptable Learning Model [11], which is unique because it emphasizes the role of self-regulation of motivation. The model has two interconnected sides: cognitive self-regulation and motivational self-regulation. Each side has the same three components: planning strategies for achieving a goal, strategy use, and knowledge (i.e., content knowledge on the cognitive side and metacognitive knowledge on the motivational side). These two sides operate in parallel with the cognitive side focused on learning and the motivational side focused on coping or maintaining well-being.

The second is Efklides' Metacognitive and Affective Model of Self-Regulated Learning [20], which focuses on metacognition. This model builds upon Bandura's socio-cognitive theory [5] by representing the person, the task, and the interaction of person-and-task as separate entities. The person-, or macro-, level is considered to be top-down because it is driven by the learner's goals for the task, which affect the person's metacognitive knowledge, metacognitive skills, motivation, self-concept, and affect. The person-and-task level, or micro-level, is considered to be bottom-up because the task dictates use of knowledge and skills, while performance and progress on the task provides feedback to the learner. Similarly, this affects the learner's metacognitive knowledge and skills, motivation, self-concepts, and affect.

The third is Winne & Hadwin's model of self-regulated learning [72], which focuses on metacognition. It conceptualizes self-regulated learning as an information processing task in which information, primarily feedback, comes from both internal and external sources. Their model defined four phases that are connected and iterative based on a feedback loop. The phases are 1) task definition, 2) goal setting and planning, 3) enacting tactics and strategies, and 4) metacognitive adaptation depending on the difference between the current state and goal state. Across these four phases, there are five components of the information processing task of self-regulation represented by the COPES acronym: conditions (i.e., personal and environmental resources), operations (i.e., cognitive processes), products (i.e., information created through operations), evaluations (i.e., feedback about differences between products and standards), and standards (i.e., criteria for goal completion).

The last is Hadwin et al.'s Socially Shared Regulated Learning (SSRL) model [25], which builds upon Winne & Hadwin's model [72] to include collaborative learning contexts. Collaboration impacts many of the common features in theories of cognitive control, including cognition, motivation, social interaction, and learning environment. In addition, groups collectively undertake skills associated with cognitive control—planning, strategy use, and evaluation. The SSRL model, thus, include three types of regulation: self-regulation, co-regulation (supporting others' task regulation), and shared regulation of task strategies and progress towards achieving goals. The phases of SSRL are the same as in Winne & Hadwin's model, but they are completed by the group rather than an individual. The COPES components apply to SSRL as well.

4.3 Measures of metacognition and self-regulation

Measuring metacognition or self-regulation is complicated by the fact that the associated models have evolved over the decades and given rise to numerous assessment instruments, each reflecting novel definitions of the terms [11]. In this section, we report the approaches used to measure self-regulation and metacognition in the literature under review. A summary, which includes a reference to a source article for each measure, is provided in Table 4.

4.3.1 Motivated Strategies for Learning Questionnaire. The most common instrument used in the papers we reviewed was the **Motivated Strategies for Learning Questionnaire** (MSLQ). The MSLQ is also one of the most widely used instruments for measuring self-regulated learning in the wider educational literature [18]. The MSLQ is a self-report questionnaire, created by Pintrich et al. [52], and designed to assess the motivational orientations of students as well as their use of different learning strategies. The first version of this questionnaire, published in 1990 [51], includes 44 items organized into five scales: self-efficacy, intrinsic value, test anxiety, cognitive strategy use, and self-regulation. A longer version, published the following year [50], is composed of 15 scales, and divided into a motivation section (31 items) that assesses students' goals, anxiety about being tested and beliefs about their skill, and a learning strategies section (50 items) which assesses students' use of cognitive and metacognitive strategies and use of resources. The scales are modular and can be administered independently, and students' responses to items are on a 7-point Likert-type scale (from "not at all true of me" to "very true of me").

Both versions of the MSLQ were used in the literature we reviewed. Bergin et al. used the 1991 version of the questionnaire and analyzed data collected on 11 of the 15 scales [8]. They found that greater use of metacognitive strategies, such as planning and monitoring, correlated positively with programming performance data. This differs from the results of Campbell et al., who used the 1990 version of the questionnaire after finding its items more suitable for use at the beginning of a course [13]. They used the self-regulation scale to assess metacognitive strategies and management of effort, but found no evidence that self-regulation was correlated with course scores. In comparing their work to Bergin et al., they suggest the conflicting results may be due to the different versions of the MSLQ that were used. A further critique of the MSLQ was offered by Leppänen et al., who explored how students managed their time while working on programming tasks in a CS1 course [34]. They found no correlation between the number of days

Table 3: Theories related to cognitive control featured in reviewed papers

	Theory (Author)	Connection to Cognitive Control
Motivational Theories	Self-efficacy theory (Bandura)	Self-efficacy affects strategy use and coping mechanisms
	Mindset theory (Dweck)	Metacognitive knowledge affects strategy selection and progress monitoring
	Goal oriented theory (Ford et al.)	Self-regulation is related to monitoring progress towards goals
	Temporal motivation theory (Steel)	Self-regulation affects goal setting behaviors and procrastination
	Self-determination theory (Ryan & Deci)	Regulation of behaviors comes from both intrinsic (i.e., self-regulation) and extrinsic motivators
	Control-value theory of achievement emotions (Pekrun et al.)	Achievement emotions can affect learning strategy planning and regulation
Cognitive Theories	Cognitive development (Piaget)	Strategy and planning are not possible until later (i.e., formal operational) stages of development
	Cognitive load theory (Sweller)	Learners need cognitive resources to use metacognitive knowledge for strategy selection, monitoring, and evaluation
Social Theories	Model of school learning (Carroll)	Generalized skills, such as self-regulation, affect time needed to complete a task
	Social learning theory (Bandura & Walters)	Cognitive control are often not visible tasks, making them difficult to learn by observing others
	Cognitive apprenticeship (Collins & Brown)	Cognitive control is learned as part of skill development through apprenticeship

Table 4: Instruments for measuring metacognition and self-regulation constructs, as reported in the reviewed literature

Instrument	Instrument source	Studies using instrument
Motivated Strategies for Learning Questionnaire	Pintrich & De Groot [51]	[8], [13], [34], [35], [71], [2], [55], [27], [32], [1], [26]
Classroom Assessment Techniques	Angelo & Cross [3]	[61], [31]
Self-Efficacy Scales	Ramalingam & Wiedenbeck [56]	[40], [76]
Self-Regulation Questionnaires	Ryan & Connell [58]	[57], [68]
Student Perceptions of Classroom Knowledge Building	Shell et al. [64]	[19]
Epistemological Beliefs Questionnaire	Schommer [59]	[42]
Achievement Goal Framework Questionnaire	Elliot & McGregor [21]	[28]

students spent programming and the observed MSLQ variables, also conflicting with the correlations reported previously by Bergin et al. [8]. Leppänen et al. suggest this may be due to a construct validity threat, in that the MSLQ may measure what students think they do, rather than what they actually do.

Other research involving the MSLQ has used various subsets of the scales. Lishinski et al. chose four of the subscales: self-efficacy, metacognitive self-regulation, intrinsic goal orientation, and extrinsic goal orientation, and they studied how these constructs interacted to influence students when learning to program [35]. Similarly, Watson et al. selected 12 of the MSLQ subscales in their study comparing traditional and dynamic predictors of success in an introductory programming course [71].

4.3.2 Classroom Assessment Techniques. Classroom Assessment Techniques (CATs) refer to a collection of strategies compiled by Angelo and Cross [3] for assessing student comprehension of important concepts during a lesson. One such strategy, **Recall, Summarize, Question, Comment and Connect** (RSQC^2), provides a structure within which students rehearse information from a lecture, identify questions on the material they would like answered, and connect the new material to content they have previously learned. Another strategy, called **muddiest point**, has students identify the parts of a lesson they find least clear.

In the literature we reviewed, CATs were used much less frequently than the MSLQ. Kirkpatrick & Prins used a combination of both the RSQC^2 and muddiest point strategies in an effort to "bring metacognition into the course" [31]. Although they did find some evidence that use of these techniques contributed to positive student outcomes in the course, they abandoned some of the questions that specifically targeted metacognition after finding student responses quite varied and unhelpful. Schwarm & VanDeGrift [61] report more positive results from the use of several CAT techniques in an information technology course. Their analysis of student responses uncovered evidence of metacognitive thinking and revealed the processes by which students constructed knowledge.

4.3.3 Self-efficacy scales. Self-efficacy has long been considered an important factor for academic success, and plays a crucial role in some self-regulated learning models such as those of Pintrich [49] and Zimmerman [75]. Numerous scales have been defined for measuring self-efficacy, such as the general self-efficacy scale of Schwarzer & Jerusalem which consists of 10-items and is used to measure a general sense of perceived self-efficacy [62]. Self-efficacy for learning and performance also appears as one of the 15 subscales in the 1991 version of the MSLQ [50]. Instruments for measuring self-efficacy that are directly relevant to computing have also been created, such as the widely used **Computer Programming Self-Efficacy Scale** developed by Ramalingam & Wiedenbeck [56].

The literature we reviewed included both the use of existing scales and the development of new scales. For example, Zingaro used a trivial adaptation of Ramalingam & Wiedenbeck's scale to measure an increase in self-efficacy for CS1 students in a course taught using peer instruction [76]. Mannila et al. present a self-efficacy scale designed for teachers, consisting of 27 items derived from competency statements from the EU framework for digital competence [40]. They present data from use of this scale with more than 500 teachers, identifying areas for which there is a need for teacher professional development.

4.3.4 Self-Regulation Questionnaires. Ryan & Connell developed the **Self-Regulation Questionnaire** (SRQ) for assessing the extent to which individuals act autonomously when performing particular tasks [58]. The questionnaire format has since been customized for various domains and demographics. The original SRQ, now known as the Academic Self-Regulation Questionnaire (SRQ-A), was designed for elementary school students whereas a later version designed by Black & Deci, known as the Learning Self-Regulation Questionnaire (SRQ-L) was intended for adult students. The SRQ-L consists of a series of statements which probe the reasons why students engage in learning-related behaviors [9].

Both the SRQ-A and SRQ-L have been applied in the literature we reviewed. For example, Ruf et al. used the 17-item SRQ-A with 7th grade students using two block-based environments, Scratch and Karel [57]. They found that students using Karel gave higher scores on the "identified regulation" sub-scale, which the authors interpret as a sign of greater perceived relevance. At the university level, Toma & Vahrenhold use the "autonomous motivation" and "controlled motivation" sub-scales of the SRQ-L in their exploration of psychological factors that affect self-efficacy and emotions in a collaborative algorithms lab [68].

4.3.5 Student Perceptions of Classroom Knowledge Building. As part of a larger project to help teachers incorporate collaborative learning tools in their classrooms, Shell et al. developed the **Student Perceptions of Classroom Knowledge Building** (SPOCK) instrument [63]. This instrument assesses student perceptions of their intentional learning behaviors and includes explicit scales to measure both self-regulation and lack of regulation. Items relating to the former scale include "In this class, I try to monitor my progress when I study", with items such as "In this class, I have trouble figuring out how to approach studying" targeting the latter scale [64]. In our review of the literature, Eck et al. used four scales from the SPOCK instrument, including the two just mentioned, to assess students' metacognitive strategies [19]. These assessments were used, along with other survey data, to successfully predict patterns of engagement in online collaborative activities.

4.3.6 Epistemological Beliefs Questionnaire. The **Epistemological Beliefs Questionnaire** (EBQ) was introduced by Schommer as an instrument for measuring the beliefs of teachers and students about the nature of knowledge and its acquisition [59]. Such beliefs play an important role in how students apply cognitive control. Schommer's questionnaire consists of 63 items and measures five epistemological dimensions, three of which (structure, certainty, and source) relate to knowledge itself and two (control and speed) to its acquisition. From our review, the EBQ was used by McDermott et al. to explore the personal epistemological beliefs of computer science students [42]. Their exploratory factor analysis yielded only partial agreement with Schommer's original work. Although they suggest this could be due to cohort differences, independent studies have called into question the reliability of Schommer's EBQ [14].

4.3.7 Achievement Goal Framework Questionnaire. Elliot & McGregor [21] follow in the footsteps of the achievement-goal tradition by offering an extension of the mastery-performance dichotomy into a 2x2 framework: mastery-approach goal, performance-approach goal, mastery-avoidance goal, performance-avoidance goal. This theory of motivation and avoidance in the pursuit of goals offers a different approach to competence-based self-regulation with each of the four parts of the 2x2 framework providing distinct profiles. Their paper details three individual studies that confirm the validity of their framework, with the first one being conducted in a classroom setting via a questionnaire (n=180). This questionnaire was used by only one study from our literature review: Ilves et al. [28].

5 APPROACHES, TOOLS, AND PEDAGOGICAL IMPLICATIONS

This section synthesizes papers classified as *depth* based on the instructional and methodological approaches taken by the authors of those papers. Each subsection is organized thematically into paragraphs corresponding to our research goals. First, we present a description of the approach and the papers that used it, including an exemplar paper. Second, we discuss future work recommended by the authors. We then discuss the pedagogical implications identified by these papers. We intend for this section to provide researchers and educators with useful information for their own work.

5.1 Reflective activity after assignment/exam

One popular approach for improving student metacognition or self-regulation skills that we found in our dataset involved a reflective activity after an assignment or exam [16, 39, 41, 45, 46, 65, 69]. Examples include reflection on study habits and preparation before and after an exam, interviews after an in-class assignment, and reflective writing assignments about students' time management. VanDeGrift et al. [69] report on an intervention in which 236 CS1.5 students were asked to make plans for working on their programming assignments and to reflect on areas for improvement, the effectiveness of their plans, and their development throughout the term. They found that most students do provide plans in the areas in which they were prompted (planning, coding, and testing) but most of the improvements they identify were in planning or coding.

Many researchers suggest that their approaches could be extended by changing the context [41], type of assignment [39, 45], or shifting the focus of the intervention on a different part of the problem-solving process [46]. For instance, the classroom intervention by VanDeGrift et al. [69] around self-regulation of homework could easily be replicated and extended by utilizing it in a different course. Stephenson et al. replicated the work of Craig et al. [16] and found that exam wrappers may not be as useful as they first appeared [65]. Thus, more work on exam wrappers is needed to understand how they affect metacognition.

VanDeGrift et al. argue that it is not the job of CS faculty to teach only programming but also to teach metacognitive skills that will support future learning [69]. Their report can inspire any instructor wanting a classroom intervention design to train students in self-reflection. Although exam wrappers [16, 65] have varied effectiveness, one positive pedagogical outcome is that both studies found exam wrappers increase student engagement with their exams and increase students' perceptions of fairness in grading practices. Finally, Mani & Mazumder [39] suggest that reflective activities around exams and homework assignments can help students more effectively spend their time when studying and can help instructors more effectively choose which learning strategies to prioritize when learning objectives are not being met.

5.2 Reflective activity as structured assignment

Rather than using reflective activities after an assignment or exam, eight of the *depth* papers utilized reflective activities as structured assignments [23, 29, 36, 46, 54, 66, 69, 70]. These assignments include explaining code [70], reflective writing about an activity [66], a survey asking students about improvements to their process [69], thinking aloud while completing a task [36, 46, 54], and a 3 year 'meta-course' that utilized reflection seminars [29]. Falkner et al., for instance, studied students enrolled in a course that contained two structured reflective exercises requiring students to describe their software development processes and how they have changed [23]. Analyzing these reflections, the authors found that students utilized a variety of self-regulated learning strategies, including general strategies, strategies that are adapted and articulated within the context of CS, and CS-specific strategies.

The authors of these papers identified several areas for future work. Falkner et al. expressed intent to explore explicit scaffolding of strategies to support understanding and assessing problems, prototyping and experimentation, design as decomposition, and explicit time management [23]. Others suggest that future work should seek methods to integrate reflective writing throughout the undergraduate curriculum [66]. Prather et al. identify a set of metacognitive difficulties students encounter and call for developers of automated assessment tools to use their findings to inform future development to address the lack of cognitive scaffolding provided to users [54]. Additionally, Parham et al. suggest that future studies examine differences between novice and expert processes [46].

The positive results of this work on reflection as structured assignments have strong pedagogical implications. For instance, Stone & Madigan suggest students who are consistently asked to reflect on their experiential learning will have an advantage as they begin their career [66]. Papers in this category identify a need for explicit scaffolding of strategy development [23] and a need for incorporating deliberate planning exercises in introductory courses to encourage reflective behavior [69]. Kann et al. confirmed that having students discuss their reflections improves the effectiveness of those reflections, suggesting that some form of communication should be incorporated into reflective activities [29]. Loksa & Ko suggest instructors need to address flaws in students' existing self-regulation behaviors identifying a lack of disciplined self-regulation during problem-solving and few reflections on cognition [36]. Additionally, they believe the timing of teaching self-regulation skills should be considered, suggesting that without first having adequate programming knowledge students may become frustrated.

5.3 Visualizations

Many novices do not have well-developed content knowledge or cognitive control in programming, so they lack even a basic understanding of their progress through the programming problem-solving process. To scaffold the process of learning this skill, several researchers in our dataset chose an approach that involved visualization of student progress [28, 33, 57, 73]. For example, Yan et al. developed Pensieve, a tool that provides an interactive visualization of students' problem-solving processes [73]. Pensieve captures the visual output of students' programs at each compile and uses it to provide a visualization of the progress that was being made during development. Pensieve is used to facilitate a conversation about process and metacognition between a teaching assistant and the student rather than directly assisting students.

Avenues for use of visualization tools are expansive. Pensieve is a proof-of-concept that would benefit from implementation and use at a large scale [73]. Additionally, several key concepts implemented in Pensieve could be implemented in automated assessment tools such that students would have immediate access. Ilves, Leinonen, & Hellas used the questionnaire by Elliot & McGregor [21] and call for future work assessing the link between different goal orientations and course performance needs [28]. They also describe future work investigating why one type of visualization, in which students could compare their progress to others in the course, was more successful than the other type.

There are several important pedagogical insights from visualization of student progress. Two studies found the lowest-performing students benefit the most from visualizations without harming the highest-performing students [28, 73]. However, Ilves et al. also found that certain kinds of visualizations could be more harmful to students than not seeing a visualization at all [28]. Yan et al. discuss how visualizations provide a unique opportunity to discuss the shape of a student's code at every snapshot to help them understand their problem-solving process [73]. These types of discussions, the authors argue, put the human element back into CS program grading, which is often automated. Finally, Kurvinen et al. demonstrate how using a problem-solving model and explicitly scaffolding metacognition and self-regulation can improve problem-solving outcomes. They also provide a model for how one might structure a collaborative activity to support the development of metacognitive and self-regulatory skills [33].

5.4 In-process scaffolding

Five of the *depth* papers gave descriptions of approaches for scaffolding metacognition or self-regulation while students are in the process of completing a task [33, 36, 41, 48, 53]. A notable paper in this category is the work by Prather et al. who investigated the effects of an intervention to assist novice programmers in overcoming metacognitive difficulties [53]. In this paper the authors provided an explicit metacognitive prompt prior to solving a programming problem. They found that students were more successful at solving the problem with the prompt than without it and said that the prompt helped them think through the problem and avoid wrong ideas early in the problem-solving process. Additionally, students who received prompts and submitted a correct solution tended to verbalize more metacognitive behaviors than those who did not.

Prather et al. recommend future work investigating how solving randomly generated test cases before coding might impact students' metacognitive skills [53]. They also seek to understand the relationship between the number of times a student re-reads a problem prompt and metacognition. Peteranetz et al. stated that in future work they will evaluate a stand-alone computational creativity course in an in-person and online format [48].

Two papers in this category identified important pedagogical implications. The results of Peteranetz et al.'s experiment support findings from previous correlational studies and computational creativity exercises (CCEs) as valuable additions to CS courses [48]. The authors suggest that instructors should explicitly connect the CCEs to the course by 1) discussing the exercises in class with students, 2) explicitly mapping CCEs to class topics, and 3) relating the activities to real-world problems. Kurvinen et al. investigated the effects of computer-assisted learning at the primary school level. They describe their environment as a collaborative education platform that enables instructors to create virtual assignments, keep track of students' progress, and conduct automated assessments [33].

5.5 Active learning

Active learning refers to a variety of approaches in which students take a more direct and participatory role in their learning, compared with traditional instruction in which the student role is more passive. Two of the *depth* papers used active learning approaches to support and study metacognition [4, 31]. Bagley & Chou investigated the effects of collaboration on students learning Java [4]. Among other findings, they found that brainstorming and devising strategies in pairs lead to significant differences in perceptions and outcomes between genders, with females believing testing produces a better solution and receiving significantly higher scores.

Of the two papers in this category, only the Kirkpatrick et al. paper [31] identified opportunities for future work. They explored the integration of Team-Based learning techniques and the $RSQC^2$ into an operating systems course, finding improvements in student outcomes and readiness to participate. They refer to these results as initial evidence of efficacy, suggesting similar approaches could be a model for future researchers. Additionally, they discuss their implementation failures and provide a rationale about why later offerings of their course used modified versions of their approach.

While neither paper in this category explicitly identifies future work, Kirkpatrick et al. encourage instructors interested in adopting active learning techniques to follow their methods. They emphasize that their approach requires only very modest effort on the part of the instructor, and yet their results show great promise.

5.6 Awareness

Only one of our *depth* papers focused on metacognitive awareness. Martin et al. describe three course interventions targeting student procrastination, which were designed to be scalable for large classrooms: reflective writing assignments about time management, schedule sheets to force students to plan their time, and email alerts that contain student progress compared to their peers and tips for how to improve if they are behind [41]. They found that only the email alert intervention provided a substantial effect.

Martin et al. describe several avenues of future work. First, improving the depth and quality of the feedback that students receive in their email alerts might further reduce procrastination. Second, they suggest that their other two interventions may have seemed to fail because they only collected data on students' code when they turned it in. A finer-grained measurement may provide different results. Finally, a system that could monitor student progress and compare it to several years worth of data might yield even better insights into supporting students' metacognitive awareness.

Pedagogically, Martin et al. found that the email alerts reduced the frequency of late submissions and increased the frequency of early submissions. This could be extended to other types of communication from an instructor and reinforces earlier findings that increased instructor presence has a positive impact on students [41]. However, they reported that students subjectively found these email alerts to be annoying and a waste of time.

5.7 Pre- and post-behavior surveys

So far, this section has discussed instructional approaches to cognitive control research. Now we shift to methodological approaches taken by researchers. One popular method of inquiry is conducted via surveys given before and after an intervention, measuring the change in metacognitive and self-regulatory behavior [13, 33, 35, 45, 57]. This is often appropriate for longitudinal studies seeking to measure shifts in student behavior over the course of a semester [45]. A specific instrument for this kind of intervention, used by two papers in this category, is the MSLQ discussed in Section 4.3 [13, 35]. As an exemplar, Murphy & Tenenburg investigated whether students are able to accurately estimate their own knowledge of a specific programming subject by having students predict their scores before and after an assignment [45]. They found that students estimations of their own knowledge correlate to their performance.

Authors in this category identify several areas for future work. Murphy & Tenenburg suggested replicating and extending their work to determine whether their findings were due to failure to learn the material or lack of retention [45]. Campbell et al.'s results support prior work that instructor presence is key for increasing self-efficacy, but they recommend investigating more fully for online CS1 course offerings [13]. Finally, Lishinski et al. call for more work to better determine how the self-efficacy feedback loop operates differently in CS [35]. This involves investigating whether there are additional cognitive factors affecting the self-efficacy feedback loop, and investigating the complex relationship between cognitive and motivational processes that occur in CS programming projects.

Multiple pedagogical implications arise from the work of these authors. As introduced above, the results of Campbell et al. highlight the importance in face-to-face and online learning of increased instructor presence, regularly offering practice exercises, and having regular get-togethers aimed at increasing intrinsic value to encourage students [13]. Lishinski et al. suggest that increasing self-efficacy, particularly among women, can potentially reduce the gender gap in computing education [35]. Finally, Murphy & Tenenburg most notably propose that it may not be helpful for the professor to ask students what they think the professor should review, because the bottom half of students simply don't know what they don't know [44]. However, having students estimate their quiz grades before and after taking them seemed to have the largest impact on the students performing in the lowest quartile. This practice may be a way to boost the lowest performing students by helping them develop a better understanding of what they do and do not know, thereby increasing metacognition about the subject.

5.8 Standardized tests/instruments analysis

Using standardized tests or comparing results between two different instruments was a method used to investigate metacognition and self-regulation by six of the papers in the *depth* category [8, 13, 34, 35, 57, 71]. Watson et al. used the MSLQ as one of seven instruments to compare test-based (eg. learning styles, learning strategies, academic experience, etc.) and data-driven predictors of programming performance [71]. The authors found that of the "traditional category" of test-based performance predictors, only self-efficacy was significantly correlated with programming performance while 11 of the 12 identified programming behaviors were found to be significantly related to performance.

Future work for papers in this category identify opportunities for new tools to support learners and to further understand the relationship between metacognition or self-regulation and learning programming. Watson et al. identifies that using programming behavior metrics could be used to build tools to identify and provide automatic assistance to weaker or struggling students [71]. Ruf et al. expressed interest in researching tools that focus on visualizing the flow of a program [57]. Alternatively, Leppänen et al. proposed future work focusing on understanding when and why students take pauses while working on programming tasks [34].

The nature of this category of work means direct pedagogical implications are limited. However, Bergin et al. suggest that tools designed specifically to help students self-regulate and encourage the development of intrinsic goal orientation and higher task values might enable them to achieve higher results [8]. Ruf et al. also recommend choosing an environment that visualizes program structure and fosters intrinsic motivation in students [57].

5.9 Analysis of secondary data

A few *depth* papers approached student metacognition and self-regulation through analysis of large sets of secondary data [30, 34, 47]. Secondary data are collected primarily for a purpose other than research. For instance, Kim & Ko reviewed four online coding tutorials, mining pedagogical insights [30], and Parker et al. examined the log files of student and teacher usage of an eBook [47]. Leppänen et al. examined keystroke-level data of student progress on programming assignments. They examined the effect of pauses on student completion rates, progress, and correlated that data with exam scores and final grades [34]. They found that student self-regulation behaviors, as assessed by the MSLQ, did not correlate to pause-taking behaviors while working on assignments. Short pauses, from 10 seconds to 4 minutes, had a negative correlation to exam scores, while longer pauses had no correlation.

Leppänen et al. identify five areas of future work [34]. Most notably, they call for identifying other background variables affecting student pausing behaviors. They also speculate that student pausing behavior might change closer to the deadline. Another fruitful avenue is to determine what students are doing during their pauses (i.e. are they reading code or surfing the web?). Kim & Ko found that although many coding tutorial systems did generate some kind of feedback, which is critical for successful metacognitive monitoring, it was rarely customized with respect to the prior knowledge of the learner [30]. They call for future work investigating personalized support in online tutorials with precise, contextualized feedback.

Pedagogically, these authors make several recommendations from their research. Parker et al. propose a number of design guidelines for student eBooks, including the need for more scaffolding and more effective incentives for spaced practice, which is an effective learning strategy exhibited by expert learners and requiring metacognitive control [47]. After examining the most successful pedagogies in online coding tutorials, Kim & Ko have multiple recommendations, including that educators make use of educational games and interactive tutorials over other tutorial types [30].

6 CONCLUSION

Metacognition and self-regulation are critical processes for successful learning that have been studied for decades in various academic disciplines. As interest in cognitive control within computing education research increases, our goal was to synthesize the studies related to learning programming, summarize theories and measurements that researchers can use, and promote future research directions and pedagogical implications. Based upon our review, we would like to highlight themes from the content of this research.

(1) Metacognitive knowledge is difficult to achieve in domains about which the learner has little content knowledge.
(2) Metacognition and self-regulation are often directly related to self-efficacy, and self-efficacy is often directly related to performance. However, the direct link between cognitive control and performance is weak.
(3) Self-report measurements of cognitive control, such as the MSLQ, often measure what students think they do, rather than what they actually do.

In addition, based on themes from the methodology of this research, we would like to propose recommendations for future work.

(1) Provide a clear definition of metacognition or self-regulation as it pertains to your study. The field does not have common definitions for these terms.
(2) Consider which related concepts might affect cognitive control in your context (e.g., cognitive load or motivation) and select theories that include or complement those concepts.
(3) Theories about cognitive control include several aspects of metacognition and self-regulation. If a study addresses a particular aspect, be specific about the scope of the research to make clear contributions to the literature (e.g., an intervention that supports the planning aspect only).
(4) Given that cognitive control processes are difficult to observe, using multiple measurements can better triangulate effects and increase validity. For example, data collected during an activity likely provides different information than that collected after an activity.

We intend for this paper to serve as a resource for the computing education research community to help researchers who are new to this area to find relevant work and help all researchers make clear contributions to a literature that extends into many fields.

ACKNOWLEDGMENTS

We would like to thank Catherine Mooney and Ray Pettit for their early contributions, and Simon for his insight.

REFERENCES

[1] Onni Aarne, Petrus Peltola, Juho Leinonen, and Arto Hellas. 2018. A Study of Pair Programming Enjoyment and Attendance Using Study Motivation and Strategy Metrics. In *Proc. of the 49th ACM Technical Symposium on Computer Science Education* (Baltimore, Maryland, USA) *(SIGCSE '18)*. ACM, New York, NY, USA, 759–764. https://doi.org/10.1145/3159450.3159493

[2] Khuloud Ahmad and Paul Gestwicki. 2013. Studio-Based Learning and App Inventor for Android in an Introductory CS Course for Non-Majors. In *Proceeding of the 44th ACM Technical Symposium on Computer Science Education* (Denver, Colorado, USA) *(SIGCSE '13)*. ACM, New York, NY, USA, 287–292. https://doi.org/10.1145/2445196.2445286

[3] Thomas A. Angelo and K. Patricia Cross. 1988. *Classroom Assessment Techniques: A Handbook for Faculty.* Office of Educational Research and Improvement, NCRIPT, Ann Arbor, Michigan.

[4] Carole A. Bagley and C. Candace Chou. 2007. Collaboration and The Importance for Novices in Learning Java Computer Programming. *ACM SIGCSE Bulletin* 39, 3 (jun 2007), 211. https://doi.org/10.1145/1269900.1268846

[5] Albert Bandura. 1986. The Explanatory and Predictive Scope of Self-efficacy Theory. *Journal of social and clinical psychology* 4, 3 (1986), 359–373.

[6] Albert Bandura and Richard H Walters. 1977. *Social Learning Theory.* Vol. 1. Prentice-hall Englewood Cliffs, NJ.

[7] Brett A. Becker, Paul Denny, Raymond Pettit, Durell Bouchard, Dennis J. Bouvier, Brian Harrington, Amir Kamil, Amey Karkare, Chris McDonald, Peter-Michael Osera, and et al. 2019. Compiler Error Messages Considered Unhelpful: The Landscape of Text-Based Programming Error Message Research. In *Proc. of the Working Group Reports on Innovation and Technology in Computer Science Education* (Aberdeen, Scotland Uk) *(ITiCSE-WGR '19)*. ACM, New York, NY, USA, 177–210. https://doi.org/10.1145/3344429.3372508

[8] Susan Bergin, Ronan Reilly, and Desmond Traynor. 2005. Examining the Role of Self-Regulated Learning on Introductory Programming Performance. In *Proc. of the First International Workshop on Computing Education Research* (Seattle, WA, USA) *(ICER '05)*. ACM, New York, NY, USA, 81–86. https://doi.org/10.1145/1089786.1089794

[9] Aaron E. Black and Edward L. Deci. 2000. The Effects of Instructors' Autonomy Support and Students' Autonomous Motivation on Learning Organic Chemistry: A Self-determination Theory Perspective. *Science Education* 84, 6 (2000), 740–756. https://doi.org/10.1002/1098-237X(200011)84:6<740::AID-SCE4>3.0.CO;2-3

[10] Benjamin Samuel Bloom. 2001. *A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives.* Longman.

[11] Monique Boekaerts and Lyn Corno. 2005. Self-Regulation in the Classroom: A Perspective on Assessment and Intervention. *Applied Psychology* 54, 2 (2005), 199–231. https://doi.org/10.1111/j.1464-0597.2005.00205.x arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1464-0597.2005.00205.x

[12] J. Bransford, A.L. Brown, and R.R. Cocking. 2000. *How People Learn: Brain, Mind, Experience, and School: Expanded Edition.* The National Academies Press, Washington, DC. https://doi.org/10.17226/9853

[13] Jennifer Campbell, Diane Horton, and Michelle Craig. 2016. Factors for Success in Online CS1. In *Proc. of the 2016 ACM Conference on Innovation and Technology in Computer Science Education* (Arequipa, Peru) *(ITiCSE '16)*. ACM, New York, NY, USA, 320–325. https://doi.org/10.1145/2899415.2899457

[14] Geraldine Clarebout, Jan Elen, Lieve Luyten, and Hadewych Bamps. 2001. Assessing Epistemological Beliefs: Schommer's Questionnaire Revisited. *Educational Research and Evaluation* 7, 1 (2001), 53–77. https://doi.org/10.1076/edre.7.1.53.6927

[15] Savia Coutinho. 2008. Self-efficacy, Metacognition, and Performance. *North American Journal of Psychology* 10, 1 (2008).

[16] Michelle Craig, Diane Horton, Daniel Zingaro, and Danny Heap. 2016. Introducing and Evaluating Exam Wrappers in CS2. *SIGCSE 2016 - Proc. of the 47th ACM Technical Symposium on Computing Science Education* (2016), 285–290. https://doi.org/10.1145/2839509.2844561

[17] John Dalbey and Marcia C Linn. 1985. The Demands and Requirements of Computer Programming: A Literature Review. *Journal of Educational Computing Research* 1, 3 (1985), 253–274.

[18] Daniel L. Dinsmore, Patricia A. Alexander, and Sandra M. Loughlin. 2008. Focusing the Conceptual Lens on Metacognition, Self-regulation, and Self-regulated Learning. *Educational Psychology Review* 20, 4 (01 Dec 2008), 391–409. https://doi.org/10.1007/s10648-008-9083-6

[19] Adam Eck, Leen-Kiat Soh, and Duane F. Shell. 2016. Investigating Differences in Wiki-Based Collaborative Activities between Student Engagement Profiles in CS1. In *Proc. of the 47th ACM Technical Symposium on Computing Science Education* (Memphis, Tennessee, USA) *(SIGCSE '16)*. ACM, New York, NY, USA, 36–41. https://doi.org/10.1145/2839509.2844615

[20] Anastasia Efklides. 2008. Metacognition: Defining its Facets and Levels of Functioning in Relation to Self-regulation and Co-regulation. *European Psychologist* 13, 4 (2008), 277–287.

[21] Andrew J Elliot and Holly A McGregor. 2001. A 2× 2 Achievement Goal Framework. *Journal of personality and social psychology* 80, 3 (2001), 501.

[22] Peggy A. Ertmer and Timothy J. Newby. 1996. Expert Learner: Strategic, self-regulated, and reflective. *Instructional Science* 24, 1 (1996), 1–24.

[23] Katrina Falkner, Rebecca Vivian, and Nickolas J.G. Falkner. 2014. Identifying Computer Science Self-Regulated Learning Strategies. In *Proc. of the 2014 Conference on Innovation Technology in Computer Science Education* (Uppsala, Sweden) *(ITiCSE '14)*. ACM, New York, NY, USA, 291–296. https://doi.org/10.1145/2591708.2591715

[24] John H Flavell. 1979. Metacognition and Cognitive Monitoring: A New Area of Cognitive-Developmental Inquiry. *American psychologist* 34, 10 (1979), 906.

[25] Allyson Fiona Hadwin, Sanna Järvelä, and Mariel Miller. 2011. Self-regulated, Co-regulated, and Socially Shared Regulation of Learning. *Handbook of self-regulation of learning and performance* 30 (2011), 65–84.

[26] Dean Hendrix, Lakshman Myneni, Hari Narayanan, and Margaret Ross. 2010. Implementing Studio-Based Learning in CS2. In *Proc. of the 41st ACM Technical Symposium on Computer Science Education* (Milwaukee, Wisconsin, USA) *(SIGCSE '10)*. ACM, New York, NY, USA, 505–509. https://doi.org/10.1145/1734263.1734433

[27] Christopher Hundhausen, Anukrati Agrawal, Dana Fairbrother, and Michael Trevisan. 2010. Does Studio-Based Instruction Work in CS 1? An Empirical Comparison with a Traditional Approach. In *Proc. of the 41st ACM Technical Symposium on Computer Science Education* (Milwaukee, Wisconsin, USA) *(SIGCSE '10)*. ACM, New York, NY, USA, 500–504. https://doi.org/10.1145/1734263.1734432

[28] Kalle Ilves, Juho Leinonen, and Arto Hellas. 2018. Supporting Self-Regulated Learning with Visualizations in Online Learning Environments. In *Proc. of the 49th ACM Technical Symposium on Computer Science Education* (Baltimore, Maryland, USA) *(SIGCSE '18)*. ACM, New York, NY, USA, 257–262. https://doi.org/10.1145/3159450.3159509

[29] Viggo Kann and Anna-Karin Högfeldt. 2016. Effects of a Program Integrating Course for Students of Computer Science and Engineering. In *Proc. of the 47th ACM Technical Symposium on Computing Science Education* (Memphis, Tennessee, USA) *(SIGCSE '16)*. ACM, New York, NY, USA, 510–515. https://doi.org/10.1145/2839509.2844610

[30] Ada S. Kim and Amy J. Ko. 2017. A Pedagogical Analysis of Online Coding Tutorials. In *Proc. of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education* (Seattle, Washington, USA) *(SIGCSE '17)*. ACM, New York, NY, USA, 321–326. https://doi.org/10.1145/3017680.3017728

[31] Michael S. Kirkpatrick and Samantha Prins. 2015. Using the Readiness Assurance Process and Metacognition in an Operating Systems Course. In *Proc. of the 2015 ACM Conference on Innovation and Technology in Computer Science Education* (Vilnius, Lithuania) *(ITiCSE '15)*. ACM, New York, NY, USA, 183–188. https://doi.org/10.1145/2729094.2742594

[32] Stan Kurkovsky, Stephanie Ludi, and Linda Clark. 2019. Active Learning with LEGO for Software Requirements. In *Proc. of the 50th ACM Technical Symposium on Computer Science Education* (Minneapolis, MN, USA) *(SIGCSE '19)*. ACM, New York, NY, USA, 218–224. https://doi.org/10.1145/3287324.3287444

[33] Einari Kurvinen, Rolf Lindén, Teemu Rajala, Erkki Kaila, Mikko-Jussi Laakso, and Tapio Salakoski. 2012. Computer-assisted Learning in Primary School Mathematics Using ViLLE Education Tool. In *Proc. of the 12th Koli Calling International Conference on Computing Education Research - Koli Calling '12*. ACM Press, New York, New York, USA, 39–46. https://doi.org/10.1145/2401796.2401801

[34] Leo Leppänen, Juho Leinonen, and Arto Hellas. 2016. Pauses and Spacing in Learning to Program. In *Proc. of the 16th Koli Calling International Conference on Computing Education Research* (Koli, Finland) *(Koli Calling '16)*. ACM, New York, NY, USA, 41–50. https://doi.org/10.1145/2999541.2999549

[35] 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. In *Proc. of the 2016 ACM Conference on International Computing Education Research* (Melbourne, VIC, Australia) *(ICER '16)*. ACM, New York, NY, USA, 211–220. https://doi.org/10.1145/2960310.2960329

[36] Dastyni Loksa and Amy J. Ko. 2016. The Role of Self-Regulation in Programming Problem Solving Process and Success. In *Proc. of the 2016 ACM Conference on International Computing Education Research - ICER '16*. ACM Press, New York, New York, USA, 83–91. https://doi.org/10.1145/2960310.2960334

[37] Dastyni Loksa, Amy J Ko, Will Jernigan, Alannah Oleson, Christopher J Mendez, and Margaret M Burnett. 2016. Programming, Problem Solving, and Self-awareness: Effects of Explicit Guidance. In *Proc. of the 2016 CHI Conference on Human Factors in Computing Systems*. 1449–1461.

[38] Lauri Malmi, Judy Sheard, Päivi Kinnunen, Simon, and Jane Sinclair. 2019. Computing Education Theories: What Are They and How Are They Used?. In *Proc. of the 2019 ACM Conference on International Computing Education Research* (Toronto ON, Canada) *(ICER '19)*. ACM, New York, NY, USA, 187–197. https://doi.org/10.1145/3291279.3339409

[39] Murali Mani and Quamrul Mazumder. 2013. Incorporating Metacognition into Learning. In *Proceeding of the 44th ACM Technical Symposium on Computer Science Education* (Denver, Colorado, USA) *(SIGCSE '13)*. ACM, New York, NY, USA, 53–58. https://doi.org/10.1145/2445196.2445218

[40] Linda Mannila, Lars-Åke Nordén, and Arnold Pears. 2018. Digital Competence, Teacher Self-Efficacy and Training Needs. In *Proc. of the 2018 ACM Conference on International Computing Education Research* (Espoo, Finland) *(ICER '18)*. ACM,

New York, NY, USA, 78–85. https://doi.org/10.1145/3230977.3230993
[41] Joshua Martin, Stephen H. Edwards, and Clfford A. Shaffer. 2015. The Effects of Procrastination Interventions on Programming Project Success. In *Proc. of the Eleventh Annual International Conference on International Computing Education Research* (Omaha, Nebraska, USA) *(ICER '15)*. ACM, New York, NY, USA, 3–11. https://doi.org/10.1145/2787622.2787730
[42] Roger McDermott, Iain Pirie, Åsa Cajander, Mats Daniels, and Cary Laxer. 2013. Investigation into the Personal Epistemology of Computer Science Students. In *Proc. of the 18th ACM Conference on Innovation and Technology in Computer Science Education* (Canterbury, England, UK) *(ITiCSE '13)*. ACM, New York, NY, USA, 231–236. https://doi.org/10.1145/2462476.2465589
[43] Trevor T Moores, Jerry Cha-Jan Chang, and Deborah K Smith. 2006. Clarifying the Role of Self-efficacy and Metacognition as Predictors of Performance: Construct Development and Test. *ACM SIGMIS Database: the DATABASE for Advances in Information Systems* 37, 2-3 (2006), 125–132.
[44] Laurie Murphy, Timothy Fossum, Susan Haller, Kate Sanders, Renée McCauley, Briana B. Morrison, Carol Zander, Suzanne Westbrook, Brad Richards, and Ruth E. Anderson. 2005. A multi-institutional investigation of computer science seniors' knowledge of programming concepts. *Proc. of the Thirty-Sixth SIGCSE Technical Symposium on Computer Science Education, SIGCSE 2005* (2005), 510–514. https://doi.org/10.1145/1047124.1047505
[45] Laurie Murphy and Josh Tenenberg. 2005. Do Computer Science Students Know What They Know?: A Calibration Study of Data Structure Knowledge. *Proc. of the 10th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education* (2005), 148–152.
[46] Jennifer Parham, Leo Gugerty, and D. E. Stevenson. 2010. Empirical Evidence for the Existence and Uses of Metacognition in Computer Science Problem Solving. *SIGCSE'10 - Proc. of the 41st ACM Technical Symposium on Computer Science Education* (2010), 416–420. https://doi.org/10.1145/1734263.1734406
[47] Miranda C. Parker, Kantwon Rogers, Barbara J. Ericson, and Mark Guzdial. 2017. Students and Teachers Use An Online AP CS Principles EBook Differently: Teacher Behavior Consistent with Expert Learners. In *Proc. of the 2017 ACM Conference on International Computing Education Research* (Tacoma, Washington, USA) *(ICER '17)*. ACM, New York, NY, USA, 101–109. https://doi.org/10.1145/3105726.3106189
[48] Markeya S. Peteranetz, Shiyuan Wang, Duane F. Shell, Abraham E. Flanigan, and Leen-Kiat Soh. 2018. Examining the Impact of Computational Creativity Exercises on College Computer Science Students' Learning, Achievement, Self-Efficacy, and Creativity. In *Proc. of the 49th ACM Technical Symposium on Computer Science Education* (Baltimore, Maryland, USA) *(SIGCSE '18)*. ACM, New York, NY, USA, 155–160. https://doi.org/10.1145/3159450.3159459
[49] Paul R. Pintrich. 2000. Chapter 14 - The Role of Goal Orientation in Self-Regulated Learning. In *Handbook of Self-Regulation*, Monique Boekaerts, Paul R. Pintrich, and Moshe Zeidner (Eds.). Academic Press, San Diego, 451 – 502. https://doi.org/10.1016/B978-012109890-2/50043-3
[50] Paul R Pintrich et al. 1991. A Manual for the use of the Motivated Strategies for Learning Questionnaire (MSLQ). (1991).
[51] Paul R Pintrich and Elisabeth V De Groot. 1990. Motivational and Self-regulated Learning Components of Classroom Academic Performance. *Journal of educational psychology* 82, 1 (1990), 33–40.
[52] Paul R. Pintrich, David A. F. Smith, Teresa Garcia, and Wilbert J. Mckeachie. 1993. Reliability and Predictive Validity of the Motivated Strategies for Learning Questionnaire (MSLQ). *Educational and Psychological Measurement* 53, 3 (1993), 801–813. https://doi.org/10.1177/0013164493053003024
[53] James Prather, Raymond Pettit, Brett A. Becker, Paul Denny, Dastyni Loksa, Alani Peters, Zachary Albrecht, and Krista Masci. 2019. First Things First: Providing Metacognitive Scaffolding for Interpreting Problem Prompts. In *Proc. of the 50th ACM Technical Symposium on Computer Science Education* (Minneapolis, MN, USA) *(SIGCSE '19)*. ACM, New York, NY, USA, 531–537. https://doi.org/10.1145/3287324.3287374
[54] James Prather, Raymond Pettit, Kayla McMurry, Alani Peters, John Homer, and Maxine Cohen. 2018. Metacognitive Difficulties Faced by Novice Programmers in Automated Assessment Tools. In *Proc. of the 2018 ACM Conference on International Computing Education Research - ICER '18 (ICER '18)*, Vol. 95. ACM Press, New York, New York, USA, 41–50. https://doi.org/10.1145/3230977.3230981
[55] Keith Quille, Natalie Culligan, and Susan Bergin. 2017. Insights on Gender Differences in CS1: A Multi-Institutional, Multi-Variate Study.. In *Proc. of the 2017 ACM Conference on Innovation and Technology in Computer Science Education* (Bologna, Italy) *(ITiCSE '17)*. ACM, New York, NY, USA, 263–268. https://doi.org/10.1145/3059009.3059048
[56] Vennila Ramalingam and Susan Wiedenbeck. 1998. Development and Validation of Scores on a Computer Programming Self-Efficacy Scale and Group Analyses of Novice Programmer Self-Efficacy. *Journal of Educational Computing Research* 19, 4 (1998), 367–381. https://doi.org/10.2190/C670-Y3C8-LTJ1-CT3P
[57] Alexander Ruf, Andreas Mühling, and Peter Hubwieser. 2014. Scratch vs. Karel: Impact on Learning Outcomes and Motivation. In *Proc. of the 9th Workshop in Primary and Secondary Computing Education* (Berlin, Germany) *(WiPSCE '14)*. ACM, New York, NY, USA, 50–59. https://doi.org/10.1145/2670757.2670772
[58] Richard Ryan and James Connell. 1989. Perceived Locus of Causality and Internalization: Examining Reasons for Acting in Two Domains. *Journal of Personality and Social Psychology* 57, 5 (1989), 749–761.
[59] Marlene Schommer. 1990. Effects of Beliefs About the Nature of Knowledge on Comprehension. *Journal of Educational Psychology* 82, 3 (1990), 498–504. https://doi.org/10.1037/0022-0663.82.3.498
[60] Dale H Schunk. 2008. Metacognition, Self-regulation, and Self-regulated Learning: Research Recommendations. *Educational psychology review* 20, 4 (2008), 463–467.
[61] Sarah Schwarm and Tammy VanDeGrift. 2003. Making Connections: Using Classroom Assessment to Elicit Students' Prior Knowledge and Construction of Concepts. In *Proc. of the 8th Annual Conference on Innovation and Technology in Computer Science Education* (Thessaloniki, Greece) *(ITiCSE '03)*. ACM, New York, NY, USA, 65–69. https://doi.org/10.1145/961511.961532
[62] Ralf Schwarzer and Matthias Jerusalem. 1995. Generalized Self-Efficacy Scale. In *Measures in health psychology: A user's portfolio. Causal and control beliefs*, J. Weinman, S. Wright, and M. Johnston (Eds.). NFER-NELSON, Windsor, UK, 35–37.
[63] Duane F. Shell, Jenefer Husman, D. M. Droesch, Indira Nath, N. Wall, and Jeannine E. Turner. 1995. *Project CIRCLE: First Year Evaluation Report (Grant #R215D30195)*. Technical Report. U.S. Department of Education: Secretary's Fund for Innovation in Education and University of Texas, College of Education, Learning Technology Center. Washington, DC.
[64] Duane F. Shell, Jenefer Husman, Jeannine E. Turner, Deborah M. Cliffel, Indira Nath, and Noelle Sweany. 2005. The Impact of Computer Supported Collaborative Learning Communities on High School Students' Knowledge Building, Strategic Learning, and Perceptions of the Classroom. *Journal of Educational Computing Research* 33, 3 (2005), 327–349. https://doi.org/10.2190/787L-BCBQ-20FN-FW6C
[65] Ben Stephenson, Michelle Craig, Daniel Zingaro, Diane Horton, Danny Heap, and Elaine Huynh. 2018. Exam Wrappers: Not a Silver Bullet. In *Proc. of the 23rd Western Canadian Conference on Computing Education* (Victoria, BC, Canada) *(WCCCE '18)*. ACM, New York, NY, USA, Article Article 1, 1 pages. https://doi.org/10.1145/3209635.3209655
[66] Jeffrey A. Stone and Elinor M. Madigan. 2007. Integrating Reflective Writing in CS/IS. *SIGCSE Bull.* 39, 2 (June 2007), 42–45. https://doi.org/10.1145/1272848.1272881
[67] Claudia Szabo, Nickolas Falkner, Andrew Petersen, Heather Bort, Kathryn Cunningham, Peter Donaldson, Arto Hellas, James Robinson, and Judy Sheard. 2019. Review and Use of Learning Theories within Computer Science Education Research: Primer for Researchers and Practitioners. In *Proc. of the WG Reports on Innovation and Technology in Comp Sci Education* (Aberdeen, Scotland Uk) *(ITiCSE-WGR '19)*. ACM, NY, USA, 89–109. https://doi.org/10.1145/3344429.3372504
[68] Laura Toma and Jan Vahrenhold. 2018. Self-Efficacy, Cognitive Load, and Emotional Reactions in Collaborative Algorithms Labs - A Case Study. In *Proc. of the 2018 ACM Conference on International Computing Education Research* (Espoo, Finland) *(ICER '18)*. ACM, New York, NY, USA, 1–10. https://doi.org/10.1145/3230977.3230980
[69] Tammy VanDeGrift, Tamara Caruso, Natalie Hill, and Beth Simon. 2011. Experience Report: Getting Novice Programmers to THINK about Improving Their Software Development Process. In *Proc. of the 42nd ACM Technical Symposium on Computer Science Education* (Dallas, TX, USA) *(SIGCSE '11)*. ACM, New York, NY, USA, 493–498. https://doi.org/10.1145/1953163.1953307
[70] Arto Vihavainen, Craig S. Miller, and Amber Settle. 2015. Benefits of Self-explanation in Introductory Programming. In *Proc. of the 46th ACM Technical Symposium on Computer Science Education - SIGCSE '15*, Vol. 68. ACM Press, New York, New York, USA, 284–289. https://doi.org/10.1145/2676723.2677260
[71] Christopher Watson, Frederick W.B. Li, and Jamie L. Godwin. 2014. No Tests Required: Comparing Traditional and Dynamic Predictors of Programming Success. In *Proc. of the 45th ACM Technical Symposium on Computer Science Education* (Atlanta, Georgia, USA) *(SIGCSE '14)*. ACM, New York, NY, USA, 469–474. https://doi.org/10.1145/2538862.2538930
[72] Philip H Winne and Allyson F Hadwin. 1998. Studying as Self-Regulated Engagement in Learning. *Metacognition in educational theory and practice* (1998), 277–304.
[73] Lisa Yan, Annie Hu, and Chris Piech. 2019. Pensieve: Feedback on Coding Process for Novices. *SIGCSE 2019 - Proc. of the 50th ACM Technical Symposium on Computer Science Education* 2 (2019), 253–259. https://doi.org/10.1145/3287324.3287483
[74] Barry J. Zimmerman. 1990. Self-Regulated Learning and Academic Achievement: An Overview. *Educational Psychologist* 25, 1 (1990), 3–17. https://doi.org/10.1207/s15326985ep2501_2
[75] Barry J. Zimmerman. 2000. Chapter 2 - Attaining Self-Regulation: A Social Cognitive Perspective. In *Handbook of Self-Regulation*, Monique Boekaerts, Paul R. Pintrich, and Moshe Zeidner (Eds.). Academic Press, San Diego, 13 – 39. https://doi.org/10.1016/B978-012109890-2/50031-7
[76] Daniel Zingaro. 2014. Peer Instruction Contributes to Self-Efficacy in CS1. In *Proc. of the 45th ACM Technical Symposium on Computer Science Education* (Atlanta, Georgia, USA) *(SIGCSE '14)*. ACM, New York, NY, USA, 373–378. https://doi.org/10.1145/2538862.2538878