Computers & Education 100 (2016) 1–17

Contents lists available at ScienceDirect

Computers & Education

journal homepage: www.elsevier.com/locate/compedu

Instructional supports for students with disabilities in K-5 computing: Findings from a cross-case analysis

Melinda R. Snodgrass [a], Maya Israel [a, *], George C. Reese [b]

[a] University of Illinois, Department of Special Education, USA
[b] University of Illinois, Office of Math, Science, and Technology Education (MSTE), USA

ARTICLE INFO

Article history:
Received 30 October 2015
Received in revised form 20 April 2016
Accepted 21 April 2016
Available online 23 April 2016

Keywords:
Universal design for learning
Students with disabilities
Pedagogy
Supports

ABSTRACT

As computer programming and computational thinking (CT) become more integrated into K-12 instruction, content teachers and special educators need to understand how to provide instructional supports to a wide range of learners, including students with disabilities. This cross-case analysis study examined the supports that two students with disabilities, who were initially disengaged during computing activities, received during computing instruction. Data revealed that students' support needs during computing activities were not CT-specific. Rather, supports specific to these students' needs that were successful in other educational areas were also successful and sufficient in CT. Although additional studies would need to be conducted to ascertain the transferability of these findings to other contexts and students, our results contribute evidence that students with disabilities can and should participate in CT and be provided with the supports they need, just as in all other areas of the curriculum. We present a framework for evaluating student engagement to identify student-specific supports and, when needed, refine the emerging K-12 CT pedagogy to facilitate full participation of all students. We then offer a list of four implications for practice based on the findings.

1. Introduction

1.1. Integrating computing into the general education curriculum

There is growing consensus that computational thinking should be included in K-12 education as one of the science, technology, engineering and mathematics (STEM) areas. One of the main arguments for this integration is that computational thinking can help students learn how to think through unstructured problems, interpret data, and communicate using computers (Lee, Martin, & Apone, 2014). Another argument for teaching computing in K-12 is one of equity, given the historic and persistent underrepresentation of women, people from diverse cultural backgrounds, and people with disabilities in computing fields (NSF, 2015). Kafai and Burke (2015) explained that increasing participation in computing can only occur when a broad range of students have access to effective computing instruction that engages them in personally meaningful ways.

* Corresponding author. Department of Special Education, 288 Education Building, 1310 S. Sixth Street, Champaign, IL 61820, USA.
E-mail address: misrael@illinois.edu (M. Israel).

http://dx.doi.org/10.1016/j.compedu.2016.04.011
0360-1315/

Many terms have been used to describe this academic discipline including *computing*, *computational thinking*, *coding*, and *computer programming*. Although there is no widespread consensus regarding terminology, computational thinking (CT) is generally used to describe the range of computing experiences that include problem solving, designing systems, and finding solutions by thinking in computational ways (Wing, 2006). Aho (2012) further elucidated this definition by stating that CT is "the thought process involved in formulating problems so that their solutions can be represented as computational steps and algorithms" (p. 832).

Given the increased efforts to provide K-12 students with CT experiences, there has been a proliferation of computing software, curricula, and instructional modules targeting young learners. Many of these tools have been designed with a "low floor, high ceiling" (Grover & Pea, 2013, p. 40) so that children can access these software with limited understanding, and as their understanding and proficiency increases, the software are powerful enough to allow for advanced programming. Two common tools used in K-12 CT instruction include *Code.org* and *Scratch*. *Code.org* (https://code.org/) includes leveled modules in which students work through progressively more complex computer programming puzzles that make use of graphically intuitive programming "blocks" in a game-like environment that scaffolds learning about computing and computational thinking, with leveled modules beginning in the primary grades. *Scratch* (https://scratch.mit.edu/) is a programming language that also makes use of graphically intuitive "blocks" wherein students can create stories, animations, and games. It is intended to be an open inquiry computing platform where students can design and share projects. Although not the only computing tools, these two platforms are popular and ones that were used as part of the current study.

Despite the availability of these and other instructional computing tools, there is still little research regarding how to teach CT in K-12, especially with students from diverse backgrounds (Grover & Pea, 2013; Guzdial, 2015). The pedagogical approaches that are available typically come from instruction within higher education. For example, Morrison, Margulieux, and Guzdial (2015) studied the use of worked examples and subgoal labels with students at a technical university and found that these pedagogical approaches were helpful. Such approaches have not been sufficiently studied within the limited K-12 CT literature. Additionally, there is even less focus on investigating how incorporating cognitive accessibility features that have shown to be effective in other content areas (such as Universal Design for Learning [UDL] principles or incorporating explicit instruction alongside open inquiry) can be applied to CT instruction to meet the needs of all students, including students who struggle and students with disabilities (Israel, Pearson, Tapia, Wherfel, & Reese, 2015; Israel, Wherfel, Pearson, Shehab, & Tapia, 2015).

According to the most recent statistics available, approximately 13% of students in K-12 schools have disabilities, 95% of students with disabilities attend regular schools, and a majority of those students spend 80% or more of their school day in general education (U.S. Department of Education, 2015). Consequently, although the Individuals with Disabilities Education Act (IDEA, 2004) affirms that students with disabilities have a right to access the general education curriculum in the least restrictive environment identified by their educational teams, educators have little direction as to how to provide computing instruction and supports that meets the individual needs of students with disabilities in a manner that upholds the intent of the law. The majority of students with disabilities, however, are capable of engaging in computing and computational thinking with appropriate supports and accessible technologies (Ladner & Israel, in press; Stefik & Ladner, 2015).

Education policy within the IDEA as well as the Elementary and Secondary Education Act (otherwise known as the No Child Left Behind Act, 2001) has been moving towards a clear preference towards educating students with disabilities alongside their peers to increase equity and access, which should be extended to computing education opportunities. These policies are consistent with other policies including the United Nations Convention on the Rights of Persons with Disabilities (CRPD) that elucidates that persons with disabilities have a right to non-discrimination, equality of opportunity, and full and effective participation and inclusion in society (Article 3). Ladner (2014) proposed using the CRPD as a blueprint for considering issues related to access of persons with disabilities in computing. The good news is that when students with disabilities receive the proper supports and gain the necessary technical and self-determination skills, their opportunities within the computing fields increase (Burgstahler, Ladner, & Bellman, 2012). Centers, such as Access Computing (http://www.washington.edu/accesscomputing/) and AccessCS10K (http://www.washington.edu/accesscomputing/accesscs10k), have developed programs focused on increasing participation of people with disabilities in computing fields and have illustrated that, with the right supports, students with disabilities can experience a great deal of success learning computing. As computing becomes increasingly integrated into K-12 instruction, there is an opportunity to investigate the experiences and instructional supports that students with disabilities have during CT instruction. Therefore, the purposes of this study were to examine the participation of students with disabilities and their support needs during computing instruction.

1.2. Conceptualizing disability through supports

Disability is often perceived as an inherent, unchangeable characteristic of an individual (World Health Organization, 2013). Through this deficit model, the person with the disability is seen as lacking inherent skills and attributes that are necessary for success. However, while disability certainly involves the presence of health conditions and their effects on the person, the current views of disability acknowledge the social complexities surrounding disability and the influence of contextual factors, which include environmental factors (e.g., social attitudes, accessibility of physical spaces) and personal factors (e.g., gender, race) (WHO, 2002; 2013). Thus, disability can be broadly defined as the "negative aspects of the interaction between an individual (with a health condition) and that individual's contextual factors (environmental and personal factors)" (WHO, 2013, p. 8). By defining it in this way, disability can become a fluid experience, diminishing when supports are