The Effectiveness of an Accessible Computing Curriculum

Abdu Arslanyilmaz
YSU, United States
aarslanyilmaz@ysu.edu

Margie Briley
YSU, United States
mlbriley@ysu.edu

Gregory Boerio
YSU, United States
gvboerio@ysu.edu

Mary Beth Loto
Potential Development,United States
marybeth@potentialdevelopment.org

Katie Petridis
Potential Development,United States
katie@potentialdevelopment.org

Abstract: There has been a limited number of studies in which a computing curriculum is designed and developed for students with Autism Spectrum Disorders (ASD), and there has been no study to test the effectiveness of an accessible computing curriculum for students with ASD. Therefore, the objectives of this study are 1) to implement an accessible computing curriculum at an inner-city school for seventh-grade students with ASD, and 2) evaluate the effectiveness of the accessible curriculum in improving students with ASDs' learning of computational thinking concepts (CTCs) (sequences, loops, parallelism, conditionals, operators, and data) and their development of fluency in computational thinking practices (CTPs) (experimenting, iterating, testing, debugging, reusing and remixing, abstracting, and modularizing) by comparing two groups of twenty-two students; one group taught utilizing the adjusted curriculum and the other utilizing the original curriculum. Students' CTCs were measured by analyzing both groups' pretest and posttest scores, and their CTPs were measured by their artifact-based interview scores.
Keywords: accessible,computing curriculum,computational thinking,students with autism spectrum disorders,computational thinking concepts,computational thinking practices

Introduction

Computational thinking (CT) as an essential 21st-century skill and knowledge will be instrumental to new discovery and innovation in all fields of endeavor [1], and therefore, CT should be taught to all students alongside reading, writing, and arithmetic [2]. It has been shown that CT increases students' interest, knowledge, and skills in computing [3]–[7]; in quantitative and critical thinking [8]; in abstracting, generalizing, and writing convincing arguments [9]; in building different dimensions of CT [10]; in near and far transfer of problem-solving skills and improvement of spatial relations ability [11]; in understanding algorithmic flow [12]; and in predicting future academic success [13], [14]. Accordingly, many prior studies have implemented CT into various subjects at different grade levels [5]–[9], [11]–[14]. Additionally, a CT curriculum was designed and developed making a CT curriculum for mainstream students accessible to students with ASD [15], [16]. However, no accessible CT curriculum was investigated to measure its effectiveness for students with ASD. Therefore, an effective intervention strategy is to implement the accessible CT curriculum and test its effectiveness. This study implemented the accessible CT curriculum at an inner-city school for seventh-grade students with ASD, and evaluated its effectiveness in improving students with ASDs' learning of CTCs (sequences, loRCA, parallelism, conditionals, operators, and data) by comparing two groups of students; one group taught utilizing the adjusted curriculum and the other utilizing the original curriculum.

Literature Review

Many studies have been conducted to show successful and effective implementation of CT curriculums into various subjects at different grade levels. It was reported that CT performance in college freshman courses could predict future academic success [13], there is a strong correlation between a student's academic performance and his/her ability to compute [14], an introduction to CT course increases students' interest in computing [6], CT could improve quantitative and critical thinking [8], CT can help to abstract, generalize, and write convincing arguments [9], and CT focused curriculum improves students' impression of, interests in, and knowledge about computing [5], [7]. Also, coding was reported to be useful for building different dimensions of CT in nine and ten-year-old students [10], programming was found to lead to near and far transfer of problem-solving skills and improvement of spatial relations ability in fifth and sixth-grade students [11], and integrating CT into middle and high school classes increases knowledge in algorithmic flow [12] and middle school students' CT skills [3].

The Problem and Objective

However, there has only been few studies in which a CT curriculum was designed and developed for minority students, and no study was conducted to test the effectiveness of an accessible CT curriculum for students with ASD. One of these rare studies was conducted by [17] targeting high school students formally diagnosed with a learning disability or attention deficit disorder (more commonly known as ADHD), where challenges to teaching a CS course to students with ADHD were identified, adjustments were proposed and tested in an attempt to make an accessible CS course for these students. Some of these adjustments that were successfully made and tested were reported to be related to barriers for students in language, reading, written expression, math, and attention [17, p. 47]. Another rare study was conducted by [15], [16], where adaptations and accommodations were designed and developed to make an original CT curriculum accessible to students with ASD. ASD is a neurodevelopmental condition characterized by impairments in social interaction, communication, creativity, imagination, and organization of daily activities [18].

The Purpose

The purpose of this study is to report the implementation of an accessible CT curriculum at an inner-city school (PDMS) for seventh-grade students with ASD and report the implementation of the original version of the accessible CT curriculum at another inner-city school for students with ASD (RCA). The aim is to report what worked and what did not in terms of instructions including delivery of instructions, assessment, feedback, and learning environment. The second purpose is to evaluate the effectiveness of the accessible CT curriculum in improving students' learning of CTCs through a two-group repeated measures study.

The Creative Computing Curriculum

The creative computing curriculum [19] developed for mainstream students was chosen to apply the adjustments because 1) the curriculum is based on an easier to learn visual block-based programming environment, Scratch, and students with ASD often demonstrate relative strength in visual-spatial relationships [20, p. 39], and 2) the creative computing curriculum is relatively easy to follow and offers instructional materials and activities to cover all core computing concepts and skills.

The creative computing curriculum [19], [21] offers instructional activities, learning objectives, downloadable libraries of materials uploaded by others, and related instructional materials based on Scratch. The curriculum is designed around three core dimensions of CT: computational thinking concepts (CTCs), computational thinking practices (CTPs), and computational thinking perspectives. CTCs include sequence (S), loRCA (L), parallelism (P), conditionals (C), operators (O), and data (Da) [19], [21]. CTPs include experimenting (E), iterating (I), testing (T), debugging (De), reusing and remixing (R), abstracting (A), and modularizing (M) [19], [21]. CT perspectives dimension of CT is ignored in this study because instructional activities related to this dimension geared toward improving students' understandings of themselves and relationships with others and the technological world around them, which will be too challenging to complete for students with ASD, and Neurological impairments of participating students will prevent us from measuring these.

The original curriculum consists of seven units, each of which has six sessions, for a total of 42 sessions. A total of 36 instructional sessions (6 sessions of 6 units) of this curriculum were adjusted to be accessible to students with ASD.

The Accessible Computing Curriculum

The creative computing curriculum was modified to make it accessible to students with ASD. The accessible computing curriculum consists of six units of six sessions for a total of 36 sessions. Each session involves eight instructional components. These are session schedule, pre-teaching activities, session learning objectives, instructional activities, visual handouts, instructional videos, reflection prompts, and work evaluation rubrics.

Session Schedule. Each of the 36 sessions follows the same order of instructions, even though the depth and breadth change based on the topics covered in each session. Each session starts with a session schedule to help ease any anxiety issue experienced by students. This way, students are informed of what is expected and how much time will be spent on the scheduled items in each session. The session schedule is presented on its own page so the teacher can print and post on the classroom walls and place them on student desks. Time for tasks and breaks are to be individualized based on attention span and behavior needs.

Pre-Teaching Activities. This is the next instructional item after the session schedule for each session in the instructional documents. Students with more severe cognitive function issues, those who are having a harder time comprehending presented information, and those who cannot keep up with the pace of the instructions because of behavioral, psychological, social, or other reasons are given further assistance and prior preparation with these pre-teaching activities. The three instructional elements of pre-teaching activities are the coverage of session topics, terms, and expectations. Specific terms for the sessions that may be unfamiliar to students are presented with their description and visual (symbols) representations (see Figure 12). The reading level of the descriptions as well as all instructional content to be presented to students are matched to the student's reading grade levels. Via the presentation of the expectations, students are informed of what is expected of them in the session to both ease their anxiety and get them ready for the session activities.

Session Learning Objectives. Unlike the objectives of the original curriculum, which presents only one set, two sets of objectives are presented in the adjusted curriculum. One set of objectives is called session objectives, which is to inform the classroom teacher of the instructional target of the session, and the second set of objectives is to inform students on what they will achieve by the end of the session, which is called "learning objectives." The learning objectives are presented at students' grade-level reading. The learning objectives are developed to be measurable, achievable, observable, and to reflect visual, oral, and written comprehension and response.

Instructional Activities. The instructional activities of the original curriculum were simplified and divided into multiple manageable sections. They were modified to be inclusive of students with different characteristics (visual/verbal/kinesthetic, work alone/in small or big groups, visual and/or verbal response, presentation to class/peer/USAT/on a notebook, etc.). One of the instructional activities added includes modeling activities for students to follow along with the classroom teachers to complete session activities. Another instructional activity is the instructions for classroom teachers to pair students having unique characteristics with undergraduate students so they can work one-on-one on session activities. Still another instructional activity integrated is to allow students who prefer studying independently to work alone.

Visual Handouts. A total of 27 visual handouts are created. These visual guides are developed for students to follow step-by-step toward completing session projects and/or tasks. These are prepared as standalone PDF documents for teachers to print and post on classroom walls and/or put on student desks.

Instructional Videos. A total of around 60 individual instructional videos were designed and recorded. These are developed for students who prefer visual channels for information presentation. All of these videos are uploaded to the YouTube channel for the accessible curriculum. Instructional documents for the sessions included directions as to how these instructional videos to be utilized.

Reflection Prompts. Reflection prompts are prepared for teachers to help them initiate discussions or to elicit written or visual responses from students toward the session's learning objectives. Each session includes one set of reflection prompts in verbal format accompanied by visual format (symbols) for students with ASD, and each reflection prompt is followed by some space for students' written and/or visual responses.

Work Evaluation Rubrics. A rubric for each session was designed and developed to evaluate students' successes in achieving the session learning objectives. A total of 36 rubrics were created in PDF formats. The rubrics are designed to assess student achievement in three levels of prompts for each item; with physical assistance, with verbal and/or visual cues, and independently.

Research Methods and Procedures

Eleven seventh grade students with ASD in two schools (PDMS and RCA) participated in this study. The participating students were able to use computer applications on tables. The independent variable is the two groups: seventh-grade students with ASD at PDMS as the experimental group, who were exposed to the accessible CT curriculum, and seventh-grade students with ASD at RCA as the control group, who were exposed to the original CT curriculum without the adjustments applied. In this study, the dependent variables are quiz scores and scores of artifacts created by the students, measuring students' learning of CTCs.

Topics in each of the three units were taught in several 45-minute sessions, one session a day for two days a week in 13 weeks (26 sessions in total) from September, 2021 until March, 2022. In addition to the 26 instructional sessions, twelve sessions were allocated to the following assessment activities; two sessions at the end of each unit (except unit #1) to the administration of two quizzes for a total of four sessions and two sessions at the start of the year for the posttest. Therefore, the intervention took 10 weeks of the 26 sessions.

Data Analysis

Several assessment methods were utilized to assess students' learning of CTCs. Assessment instruments utilized, including pretest and quizzes as formative assessments and artifact analysis, built on prior study efforts. One quiz was administered at the end of each of two units except for Unit #1, where students were familiarized with the Scratch environment and block-based programming. The pretest is designed to measure students' ability to formulate and solve problems by relying on CTCs. The test consists of 28 multiple-choice items with one correct and three inaccurate choices. The pretest was given during the first teaching session. Quizzes were developed to evaluate students' comprehension of CTCs as a formative assessment similar to the one used by [12], [22]–[24]. Five different types of questions included in the quizzes are 1) find a solution to a problem by tracing a small sequence of a code block, 2) multiple-choice questions based on small Scratch codes where students would pick the right choice based on either the output of the given code sequence or a blank field in the given code sequence, 3) debug, and 4) identify what CTCs are represented on a shown sequence of a block.

Dr. Scratch was used to analyze students' artifacts. Dr. Scratch is a plugin inspired by Scratch and developed on Hairball [48]. It automatically analyzes Scratch projects to evaluate the development of CTCs and to detect some bad programming habits [49]. This specific plugin was chosen because it reveals information about students' development of CTCs, including abstraction (sequences), parallelism (parallelism), logical thinking (conditionals and operators), synchronization (events), flow control (loRCA), user-interactivity (events), and data representation (data). Dr. Scratch measures and assigns each concept between 0 and 3 points for an overall score. Reports generated by Dr. Scratch will be analyzed.

Results & Discussions

The effectiveness of the adaptations and accommodations in improving students' learning of CTCs were measured by analyzing both groups' pretest, two quizzes, and artifact analysis scores for CTCs. The two groups (seventh-grade students at PDMS and RCA) were the independent variables in this study. Students in both groups were taught the accessible and the original CT curriculums, respectively, and the duration of the intervention were the same.

Table 1 shows the pre-test, two quiz, and artifact scores for the two groups (PDMS and RCA). As seen in the table, both groups have similar pre-test scores. The pre-test was administered before the start of the curriculum implementation to learn about students' prior knowledge in CTCs. The data on table 1shows that the groups were at the same level in terms of their CTCs when they started to participate in this project.

Their scores on their artifacts were similar as well. This data indicates that students' development of CTCs in terms of abstraction (sequences), parallelism (parallelism), logical thinking (conditionals and operators), synchronization (events), flow control (loops), user-interactivity (events), and data representation (data), have not been affected by the adjusted curriculum.

Students' quiz scores were different both on quiz 1 as well as quiz 2. The mean scores for students in RCA were 3.06 and 2.06 for quiz 1 and 2, respectively, while the mean scores for students in PDMS were 4.59 and 4.00, respectively. PDMS students were 1.5 times better than RCA students on quiz one, and they were 2 times better than RCA on quiz 2. There seems to be growing gap between the two groups as the experiment progresses.

School	Number of Students	Pre-test	Artifact Analysis	Quiz 1	Quiz 2
RCA	11	2.84	5.596	3.06	2.06
PDMS	12	2.82	5.812	4.59	4.00

Table 1. Students' mean scores for artifact analysis, pre-test, quiz 1, and quiz 2.

These results provide us with a preliminary data indicating the direction of the effectiveness of the accessible computing curriculum implemented at PDMS. Specifically, the quiz scores indicate that accessible CT curriculum may be proven effective at the end of the implementation in improving students with ASDs' learning of CTCs.

Acknowledgement

This project is based upon work supported by the National Science Foundation under Award Number (FAIN) 2031427.

References

[1] J. Lockwood and A. Mooney, "Computational thinking in education: Where does it fit? A systematic literary review," *Int. J. Comput. Sci. Educ. Sch.*, vol. 2, Mar. 2017, doi: abs/1703.07659.
[2] J. Wing, "Computational thinking," *Commun. ACM*, vol. 49, no. 3, pp. 33–35, 2006.
[3] C. P. Brackmann, M. Román-González, G. Robles, J. Moreno-León, A. Casali, and D. Barone, "Development of computational thinking skills through unplugged activities in primary school," in *Proceedings of the 12th Workshop on Primary and Secondary Computing Education*, New York, NY, USA, 2017, pp. 65–72. doi: 10.1145/3137065.3137069.
[4] R. Folk, G. Lee, A. Michalenko, A. Peel, and E. Pontelli, "GK-12 DISSECT: Incorporating computational thinking with K-12 science without computer access," in *Proceedings - Frontiers in Education Conference, FIE*, El Paso, TX, USA, 02 2015, vol. 2014, pp. 1–8. doi: 10.1109/FIE.2015.7344238.
[5] D. S. Goldberg, D. Grunwald, C. Lewis, J. A. Feld, and S. Hug, "Engaging computer science in traditional education: The ECSITE project," in *ITiCSE'12 - Proceedings of the ACM Conference on Innovation and Technology in Computer Science Education*, Haifa, Israel, 01 2012, pp. 351–356. doi: 10.1145/2325296.2325377.
[6] S. Hambrusch, C. Hoffmann, J. T. Korb, M. Haugan, and A. l. Hosking, "A multidisciplinary approach towards computational thinking for science majors," *SIGCSE Bull. Inroads*, vol. 41, no. 1, pp. 183–187, 02 2009, doi: 10.1145/1539024.1508931.
[7] T. Burgett, R. Folk, J. Fulton, A. Peel, E. Pontelli, and V. Szczepanski, "DISSECT: Analysis of pedagogical techniques to integrate computational thinking into K-12 curricula," in *Proceedings of Frontiers in Education Conference, FIE*, El Paso, TX, USA, Dec. 2015, vol. 2014, pp. 1–9. doi: 10.1109/FIE.2015.7344241.
[8] H. Qin, "Teaching computational thinking through bioinformatics to biology students," *SIGCSE Bull. Inroads*, vol. 41, no. 1, pp. 188–191, 02 2009, doi: 10.1145/1539024.1508932.
[9] J. T. Jenkins, J. A. Jerkins, and C. l. Stenger, "A plan for immediate immersion of computational thinking into the high school math classroom through a partnership with the Alabama math, science, and technology initiative," in *Proceedings of the Annual Southeast Conference*, Tuscaloosa, AL, USA, 01 2012, pp. 148–152. doi: 10.1145/2184512.2184547.
[10] G. Falloon, "Building computational thinking through programming in K-6 education: A New Zealand experience," in *EDULearn Proceedings*, Barcelona, Spain, 2015, pp. 882–892.
[11] R. B. Miller, G. N. Kelly, and J. T. Kelly, "Effects of logo computer programming experience on problem solving and spatial relations ability," *Contemp. Educ. Psychol.*, vol. 13, no. 4, pp. 348–357, Oct. 1988, doi: 10.1016/0361-476X(88)90034-3.

[12] S. Grover, S. Cooper, and R. Pea, “Assessing computational learning in K-12,” in *Proceedings of the 2014 Innovation and Technology in Computer Science Education Conference*, Uppsala, Sweden, 01 2014, pp. 57–62. doi: 10.1145/2591708.2591713.

[13] R. Haddad and Y. Kalaani, “Can computational thinking predict academic performance?,” in *Proceedings of the 5th IEEE Integrated STEM Education Conference*, Princeton University, 2015, pp. 225–229. doi: 10.1109/ISECon.2015.7119929.

[14] O. L. De Oliveira, L. M. Del Val Cura, and M. Do Carmo Nicoletti, “Quantitative correlation between ability to compute and student performance in a primary school,” in *SIGCSE 2014 - Proceedings of the 45th ACM Technical Symposium on Computer Science Education*, Atlanta, GA, USA, Jan. 2014, pp. 505–510. doi: 10.1145/2538862.2538890.

[15] A. Arslanyilmaz, M. Briley, M. B. Loto, C. Fernberg, G. Beadle, and J. Coldren, “An Accessible Computing Curriculum for Students with Autism Spectrum Disorders - Learning & Technology Library (LearnTechLib),” in *Proceedings of Society for Information Technology & Teacher Education International Conference*, Online, 2021, pp. 17–23. Accessed: Aug. 13, 2021. [Online]. Available: https://www.learntechlib.org/primary/p/219107/

[16] A. Arslanyilmaz, M. Briley, M. Loto, C. Fernberg, and J. Coldren, “Making an Original Computing Curriculum Accessible for Students with ASD,” Nov. 2021. doi: 10.13140/RG.2.2.28368.84485.

[17] S. Wille, J. Century, and M. Pike, “Exploratory research to expand opportunities in computer science for students with learning differences,” *Comput. Sci. Eng.*, vol. 19, no. 3, pp. 40–50, May 2017, doi: 10.1109/MCSE.2017.43.

[18] C. Alvarado, R. Munoz, and R. Villarroel, “ValpoDijo: Developing a software that supports the teaching of chilean idioms to children with autism spectrum disorders,” in *Proceedings of the 12th Latin American Conference on Learning Technologies (LACLO)*, La Plata, Argentina, 2017, p. 1. doi: 10.1109/LACLO.2017.8120905.

[19] K. Brennan, C. Balch, and M. Chung, “Creative computing: Scratch curriculum guide.” Harvard Graduate School of Education, 2014. Accessed: Dec. 31, 2017. [Online]. Available: scratched.gse.harvard.edu/guide

[20] A. A. L. S. Programs, A. Learning, and T. R. Branch, *Teaching students with autism spectrum disorders*. Alberta Learning, Special Programs Branch, 2003. [Online]. Available: https://books.google.com/books?id=-RxSXwAACAAJ

[21] K. Brennan and M. Resnick, “New frameworks for studying and assessing the development of computational thinking,” presented at the 2012 Annual Meeting of the American Educational Research, Vancouver, British Columbia, Canada, 2013.

[22] B. Ericson and T. McKlin, “Effective and sustainable computing summer camps,” in *SIGCSE’12 - Proceedings of the 43rd ACM Technical Symposium on Computer Science Education*, Raleigh, NC, USA, Jan. 2012, pp. 289–294. doi: 10.1145/2157136.2157223.

[23] O. Meerbaum-Salant, M. Armoni, and M. Ben-Ari, “Learning computer science concepts with scratch,” *Comput. Sci. Educ.*, vol. 23, no. 3, pp. 239–264, Jan. 2013, doi: 10.1080/08993408.2013.832022.

[24] I. Zur-Bargury, B. Pârv, and D. Lanzberg, “A nationwide exam as a tool for improving a new curriculum,” in *Proceedings of the ACM Conference on Innovation and Technology in Computer Science Education*, University of Kent, Canterbury, England, Jan. 2013, pp. 267–272. doi: 10.1145/2462476.2462479.