Improving Computer Science Curricula for Accessibility and Higher Engagement

Sofia Alvarez

ABSTRACT

This paper contributes to the academic discussion regarding standardized K-12 computer science education. Economic barriers often preclude the expansion of computer science education in K-12 schools, especially in low-income communities. For this reason, the proposed curriculum utilizes low-cost, accessible materials to serve a wider range of schools across the nation. With this foundation, the curriculum was designed to employ alternative methods of teaching to promote higher engagement in student learning, including year-long projects and social constructivist approaches. Students learn fundamental non-technical skills alongside foundational concepts to prepare for career opportunities in the field.To further promote accessibility, course descriptions discuss topics covered, teaching methods, course structure, and resources for teachers to closely follow. The Computer Science Teachers Association (CSTA) K-12 CS Standards are referenced numerous times to ensure the curriculum is aligned. Its design is meant to support feasibility for CS K-12 teachers to implement and encourage students to take interest in computer science and envision themselves as members of the field.

1 INTRODUCTION

Access to computer science education has become increasingly important in today's technology-driven society, yet many students find themselves confronted with systematic barriers that obstruct their opportunities for academic and career success. These barriers are labelled under the digital divide, defined as the unequal access to technology across the country, particularly affecting low-income communities. The consequences of these inequities manifest in the K-12 education sector, preventing students from receiving quality computer science education on a nationwide scale. First, the standardization of CS K-12 education is offset by the lack of consolidated teaching resources. An adequate curriculum must offer an affluent collection of quality resources for teachers and students to use in schools. Moreover, CS education expansion requires that all schools, regardless of socioeconomic status, have the means to offer quality computer science education. However, low-cost materials that offer a comprehensive understanding of computer science are in high demand.

Second, technological infrastructure is limitedly distributed across rural and low-income communities. This environment impedes digital literacy and exposure to technology in these regions, thus precluding students from taking interest in computer science. Research shows that these barriers hold a greater effect on minority students, with Black and Hispanic students being less likely than White students to use a computer at home everyday or know an adult who works in CS [18]. Impersonal attitudes towards computer science that arise in these regions, especially from students, fight against the efforts by advocacy groups to implement a standardized computer science curriculum.

The expansion of computer science education is beneficial for all students, regardless of interest. The labor market has proved that workers with information and communication technology (ICT) skills often qualify for higher economic growth. In addition, CS classes help students develop problem solving and computational skills that can be applied to various subjects [17]. Moreover, the integration of computer science in K-12 schools is an essential step towards accessibility. Towards this goal, different teaching methods and low-cost resources were compiled towards creating a curriculum that could be implemented in K-12 schools. The following sections discuss its purpose in offering students a broad exploration in computer science for higher engagement while remaining feasible for teachers to utilize or submit to their school board.

2 RELATED WORK

The 2022 State of Computer Science Education is an annual report written by the Code.org Coalition. It recounts nationwide progress towards quality CS education and lends itself as a tool for CS education advocacy groups developing policy plans [3]. However, its overview fails to mention local challenges that hinder computer science as as a subject in K-12 schools. These limitations include accessibility, quality resources, and a need for established curricula for teachers to follow. Providing a closer look into these matters in future reports could invite the CSEd community to develop effective academic programs that address local school districts' needs. In this spirit, a K-12 computer science curriculum will be presented to contribute to the discourse. It comprises online resources for accessibility and covers a wide range of CS topics to familiarize young students with the field and its influence. In addition to contributions, it is essential for the computer science education community to evaluate and advance the current methods of teaching and implementation.

Currently, the most widely integrated K-12 curriculum for computer science is the AP Computer Science Principles (CSP) course developed by Collegeboard. It is split into five components: creative development, data, algorithms and programming, computer systems and networks, and the impact of computing. The course joins these concepts to prepare students for a career in STEM and more advanced computer science studies [8]. Usually, the course is taken between the second and fourth years of high school. However, as CS education policies take root across the country, educators are reconsidering the most advantageous age for students to begin learning computer science. As the next sections will cover, the proposed curriculum begins as a math-integrated subject in the seventh grade. Using programming and computers as an accessory to mathematics promotes advanced digital literacy at an age when students are using technology daily. The curriculum touches on each of the topics in the AP CSP course at a slower pace, spanning four years. In addition, the classes involve non-technical skills that are important for computer science, such as group collaboration and creativity.

Academic papers contribute towards achieving a standardized computer science curriculum for K-12 schools by developing curricula, educational models, and teaching methods. One such paper by Hazzan et al. presents a model for high school CS education in the United States inspired by that of Israel [10]. Their model is comprised of four pieces from Israel's education system to foster CS education: mandatory CS teaching licenses, CS curricula and syllabuses, CS teacher preparation programs, and research in CS education. They explain how these components can be implemented and how they interact with each other. In particular, they outline a two- to three-year CS curriculum for high school students. However, throughout this paper, there is a greater focus on explaining how the system would work rather than how each component operates individually. As a result, there are no specific resources, such as books or websites, or structures discussed with the curriculum. For the proposed curriculum in the following sections, methods of teaching and class structure are recommended for each course, along with explicit resources for the material covered. By doing so, the curriculum stands as a framework for instructors to model their classroom with.

3 PROPOSED CURRICULUM FOR GRADES SEVEN THROUGH TEN

An objective with this curriculum is to use low-cost, accessible materials. The main resources used are the CS Discoveries Courses from Code.org and the Bootstrap:Algebra Pyret materials. Bootstrap offers free access to lesson plans and Code.org requires an free online account by the teacher. Another resource used is CodeHS digital textbooks and free curricula online. In the following courses that involve mathematics, these topics are in alignment with the Mathematics Standard of Learning from the Virginia Department of Education.[15] [14] Each grade level's course will be compared to the Computer Science Teachers Association (CSTA) K-12 CS Standards in commitment towards creating a quality curriculum for computer science.

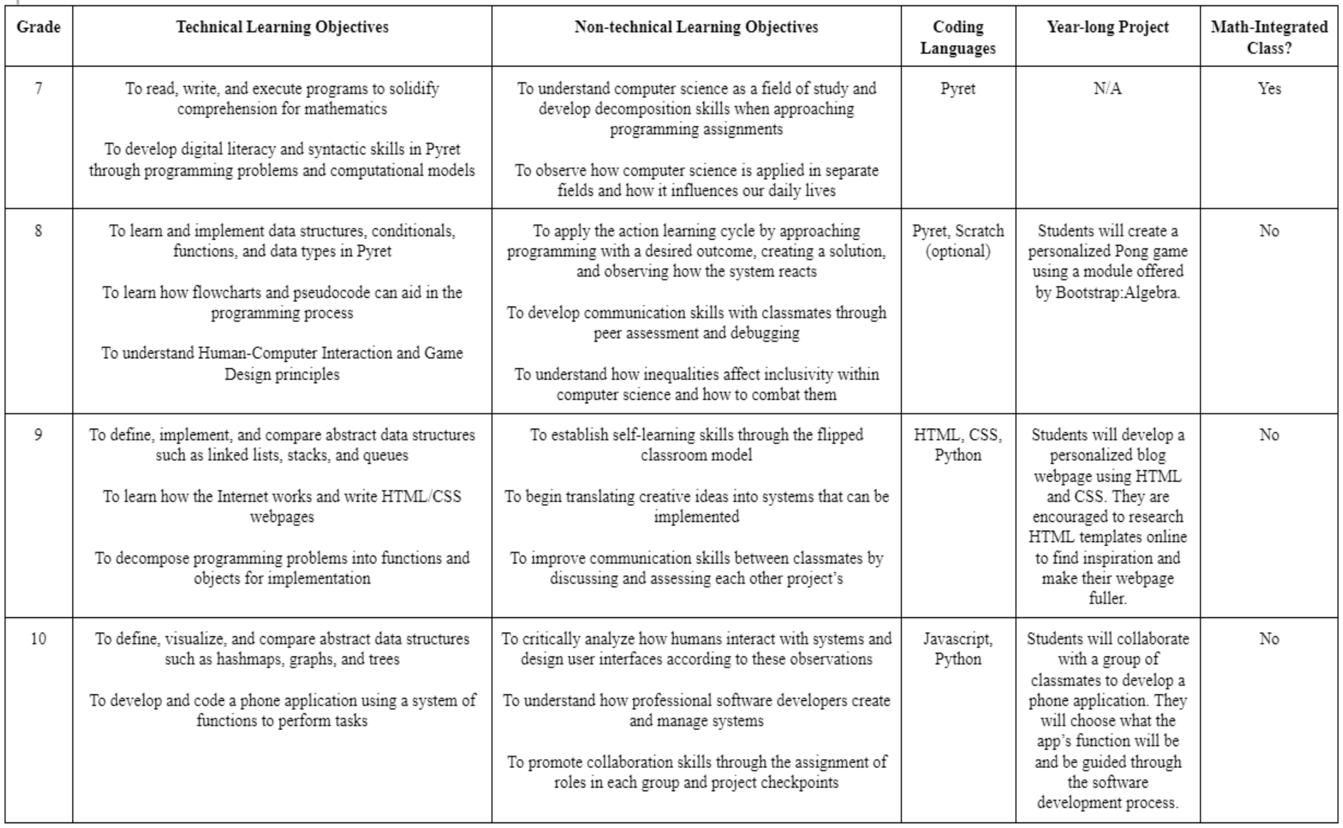
Seventh Grade: This course is an introduction to programming and computing. It is designed as an integrated subject with grade seven mathematics. The purpose of this course is to present students with real world applications with programming in mathematics. Topics covered will be taught alongside the corresponding math topic in the order of the standard curriculum. Students will learn what a programming language is, how to read and execute functions, variable initialization, and syntactic skills. Using the language Pyret to apply these topics, teachers help solidify students' understanding of subjects through coordinate systems and geometrical figures. This programming language was chosen for its aim to be an introductory language for promising computer scientists [2]. It is used throughout the Bootstrap:Algebra materials for its online compiler and powerful run-time.

By offering an overview of how programming works and its utility in the context of mathematics, the course is directly aligned with multiple clauses of the CSTA K-12 CS Standards. First, the use of Pyret programs that solidify comprehension for mathematic concepts teaches students to create and operate on variables, aligning with clause 2-AP-11 [1]. Next, encouraging students to experiment and modify programs, such as an activity where students can plot linear functions to visualize relationships between variables, teaches to collect and transform data. In addition, they examine computational models and programing problems, allowing the course to meet clauses 2-DA-08 [1], 2-DA-09 [1], and 2-AP-13 [1].

Eighth Grade: This is the first independent computer science course in the curriculum. After establishing a foundation in coding in seventh grade, students will now pursue their first year-long project. The Bootstrap:Algebra module "Making Pong" is an online walk-through project that allows students to build a Pong game [2]. For this, there are prerequisite modules that are necessary to learn before the project is feasible. These include modules for data structures, conditionals, functions, and data types, directly satisfying CSTA K-12 CS Standards 2-DA-07 [1], 2-AP-12 [1], and 2-AP-14 [1]. There are a variety of worksheets offered by Bootstrap:Algebra to enforce their learning of these essential CS concepts. Students are encouraged to use diagrams to aid their learning such as flowcharts and pseudocode as recommended by CSTA K-12 CS Standard 2-AP-10 [1]. Leading up to the Pong project, the course will also cover game design principles and human-computer interaction (HCI), fulfilling the CSTA K-12 CS Standard to teach students how to analyze and improve user interfaces [1]. A recommended resource is Scratch for educators, a free programming language that students can use online and practice programming for video games [16]. While it is a separate programming language from Pyret, it works in an abstract manner for simplicity. Students should learn an overview of how games are defined, components of video games, and the game development process.

An objective for this curriculum is for students to feel prepared for advanced computer science studies or entry-level career opportunities in the field. As such, non-technical skills used throughout the profession are encouraged in students in the courses. This is done using a social constructivist model, defined by Machanick as an educational approach that emphasizes the learner's environment and its influence on their learning [11]. The course teaches the action learning cycle, meaning that students will approach programming problems with a desired outcome, create a solution, apply their plan, then reflect on the outcome before starting the cycle again. In addition, peer assessment and debugging will be a primary activity to promote communication skills between students and strengthen their understanding of the problem. Students will also discuss the issue of accessibility and inequality in computer science as a pivotal step in entering the field. These lessons provide a social context behind the subject matter and aligns with clause 2-IC-21 [1] in the CSTA K-12 Standards. In addition, the action learning cycle and peer assessment activity enforce that students develop the skills for problem solving and communication, aligning with clauses 2-AP-13 [1] and 2-AP-15 [1].

Ninth Grade: This year introduces three essential programming languages: HTML, CSS, and Python. First, Python is an essential language for aspiring computer scientists to learn, so their transition from Pyret to Python will fill the first half of the school-year. A recommended IDE for teachers to use with students is Pycharm Community for its range of functionality and low-cost. As students adapt to the environment and learn syntactic skills, the course



Grade	Technical Learning Objectives	Non-technical Learning Objectives	Coding Languages	Year-long Project	Math-Integrated Class?
7	To read, write, and execute programs to solidify comprehension for mathematics To develop digital literacy and syntactic skills in Pyret through programming problems and computational models	To understand computer science as a field of study and develop decomposition skills when approaching programming assignments To observe how computer science is applied in separate fields and how it influences our daily lives	Pyret	N/A	Yes
8	To learn and implement data structures, conditionals, functions, and data types in Pyret To learn how flowcharts and pseudocode can aid in the programming process To understand Human-Computer Interaction and Game Design principles	To apply the action learning cycle by approaching programming with a desired outcome, creating a solution, and observing how the system reacts To develop communication skills with classmates through peer assessment and debugging To understand how inequalities affect inclusivity within computer science and how to combat them	Pyret, Scratch (optional)	Students will create a personalized Pong game using a module offered by Bootstrap:Algebra.	No
9	To define, implement, and compare abstract data structures such as linked lists, stacks, and queues To learn how the Internet works and write HTML/CSS webpages To decompose programming problems into functions and objects for implementation	To establish self-learning skills through the flipped classroom model To begin translating creative ideas into systems that can be implemented To improve communication skills between classmates by discussing and assessing each other project's	HTML, CSS, Python	Students will develop a personalized blog webpage using HTML and CSS. They are encouraged to research HTML templates online to find inspiration and make their webpage fuller.	No
10	To define, visualize, and compare abstract data structures such as hashmaps, graphs, and trees To develop and code a phone application using a system of functions to perform tasks	To critically analyze how humans interact with systems and design user interfaces according to these observations To understand how professional software developers create and manage systems To promote collaboration skills through the assignment of roles in each group and project checkpoints	Javascript, Python	Students will collaborate with a group of classmates to develop a phone application. They will choose what the app's function will be and be guided through the software development process.	No

Figure 1: Summary of learning objectives and features of courses in proposed curriculum

will present an introduction to abstract data structures, including linked lists, stacks, and queues. With this, the course explains these structures, discusses complexity analysis, and the selection of data structures for particular uses. They will be expected to define what a data structure is and implement it in Python. Through doing so, students show the curriculum's alignment with the K-12 CS Standards 3A-AP-14 [1], 3A-DA-10 [1], and 3A-AP-15 [1]. The main resource for this course is the Introduction to Computer Science in Python 3 course in CodeHS [4]. Due to the high amount of coding, the course is designed to operate as a "flipped classroom," meaning that students watch video lectures at home and work on assignments during class with the help of other students and the instructor.

To learn HTML and CSS, the end-of-the-year project will be a blog webpage using the two languages. Consequently, students will learn about web development as an application of computer science. They will learn an overview of what the Internet is and how its network works, following K-12 CS Standard 3A-NI-04 [1]. Code.org offers an Introduction to Web Development unit that covers these topics as well as a number of coding practices for HTML and CSS [6]. Using this resource, students can code with interactive visualizations in order to create their webpage and understand the abstractions required to build it. This learning experience fulfills a variety of K-12 CS Standards that promote an analytical mindset towards overcoming programming challenges and striving for a creative goal, including standards 3A-CS-01 [1], 3A-DA-11 [1], 3A-AP-13 [1], and 3A-AP-16 [1]. In learning the components behind a webpage and how to build it, students will display a proficiency in decomposing challenges through analysis and the use of functions and objects, following K-12 CS Standard 3A-AP-17 [1]. This project was chosen for its exploration into a sector of computer science with in-demand jobs and as an opportunity to teach two essential programming languages.

To embody a social constructivist model, the blog page project is done individually, yet teachers are encouraged to provide group activities that involve students communicating feedback about each other's project. By doing so, students strengthen their knowledge of HCI from eighth grade by analyzing others' work and practice communication skills. Activities such as these also introduce students to the concept of requirements elicitation in the software development process. This foundation is recommended for students in this grade-level by the K-12 CS Standards 3A-AP-21 [1] and 3A-AP-19 [1].

Tenth Grade: This course picks up the discussion of data structures from the previous school year. This time, students will become familiar with hashmaps, graphs, and trees. Similar to the course in ninth grade, the discussion of data structures fulfills K-12 CS Standards 3A-DA-10 [1] and 3A-AP-15 [1]. Students will be expected to define these structures, how they work, and how they compare in terms of efficiency. However, students will not practice implementing these data structures due to their complexity. Students would need to display a proficiency in Python that will be difficult to attain alongside working on their year-long projects. Instead, students are encouraged to sketch diagrams to reflect their understanding of the data structures, such as how elements are inserted or ordering properties. Teachers may provide these overviews through the Introduction to Computer Science in Python 3 and AP Computer Science A digital textbooks offered by CodeHS [5].

This level's year-long project will focus on app development using Javascript. By choosing this as a focal point for students, they will learn the process through which professional software development works while collaborating in a group. It is recommended that the instructor walk students through the software development cycle by teaching how each phase works and providing activities that align with the appropriate work. Roles such as project managers, evaluators, and UI designers will promote group members accepting responsibility over their work and encourage open communication. Fostering these non-technical skills in aspiring computer scientists prepares them for careers in the field, aligning with K-12 CS Standards 3A-AP-22 [1] and 3A-IC-27 [1]. The main resource for teaching Javascript and app development will be the App Development module created by Code.org. This extensive module uses block coding and side-by-side visualization to help students create prototypes throughout their development process [7]. In addition, students refine their ability to decompose procedures in programs and design systems, achieving K-12 CS Standards 3A-AP-13 [1], 3A-AP-16 [1], and 3A-AP-19 [1].

This course is also designed to inspire initiative in students through a series of checkpoints rather than leading them through activities. In this case, teams will be encouraged to display progress to the instructor through work done on their own time. Due to the intensity of coding necessary, the course will follow a "flipped classroom" model, similar to the ninth grade course. Teachers will assign readings or lecture videos to watch at home and hold coding sessions in-class. This guarantees that groups are able to collaborate regularly and reach out for help throughout the project.

4 ADVANTAGES OF CURRICULUM

There are two main objectives ingrained in the design choices behind this proposed curriculum: (1) provide a framework that teachers can implement feasibly and (2) prepare students for career paths in computer science through quality educational experiences. Towards the first purpose, the courses in the curriculum were designed using free online resources. This is an integral feature that benefits the teachers and students involved in the curriculum. For educators, research shows that course material preparation is an obstacle that prevents them from teaching the course efficiently [13]. As such, providing a collection of resources for teachers to utilize reduces this burden. For students, taking a class that requires expensive memberships or programs strain those from low-income families. This barrier discourages this variety of students from participating in CS courses, thereby strengthening the impact of the digital divide. The curriculum's use of cost-efficient materials combats these barriers for both parties and motivates a larger range of students, instructors, and schools to offer computer science courses.

The duration of this curriculum is another intentional design choice towards promoting accessibility. In deciding what grade should computer science be introduced to students, it was important to consider how they perceived technology at different ages. By middle school, children begin to perform more academic and personal use of technology. This stage of life was chosen to introduce computer science to students at this pivotal age, thereby increasing their level of confidence with the subject [12]. Along with this choice, this first course is integrated with math to display an application of computer science that students can understand and interact with. This is important as a study run by Delen and Bulut has shown direct and indirect links between early exposure to technology and proficiency in mathematics and science [9].

Similarly, the curriculum's last course is intended for tenth grade students, thus ending the teaching path by the midpoint of high school. Through this design choice, the curriculum offers a comprehensive foundation in computer science, along with skills in six different programming languages, by tenth grade. At this point, students are eligible for the AP Computer Science Principles course which can broaden the student's understanding of computer science to a sophisticated level. By graduation, students of this curriculum will have the capability to join the labor market within the computer science sector or pursue higher education. This defeats a large issue of accessibility within CS careers as students will not be required to join a four-year university before having a chance to work in the field. In summary, the courses offered are designed to be accessible for teachers and students so that all schools can open the door for computer science opportunities for its students.

The second fundamental goal of this curriculum is to teach students foundational skills for computer science while providing them with an engaging experience with the subject. With this purpose, the courses were designed to involve projects and assignments that focus on one particular application of computer science for the whole year. These applications were also chosen to relate to students' experiences with technology, involving apps, websites, and video games. This provides students with essential technical skills along with real-life context to help them envision themselves as professionals in the field. In addition, the utilization of social constructivist model was chosen for its ability to promote non-technical skills while teaching practical skills. These are equally important to technical skills such as coding or debugging, but they are often precluded from CS courses. Professional computer scientists require communication and collaboration skills to advance in their careers. As a result, the curriculum's promotion of these skills through peer assessment and group projects presents students with a full engagement with computer science as a subject and as a potential career. Concisely, the curriculum utilizes alternative methods of teaching and learning in order to engage students with a genuine experience in computer science while guiding them through technical concepts.

5 LIMITATIONS

There are numerous barriers that prevent the expansion of computer science K-12 education, including financial strains and lack of resources. For public schools to offer computer science, there is a necessity for low-cost materials that provide quality instruction.

The most essential tool for CS classrooms is a stock of computers or laptops for students to use. For schools in low socioeconomic status communities, this first step may be difficult to implement, therefore preventing any form of digital literacy being taught. In addition, homework assignments and lecture videos throughout the proposed curriculum are intended to be completed at home through the use of a personal computer. This places a burden on families to provide computers and reliable internet in order for the curriculum to take place. Beyond this hurdle, the curriculum was designed with economic status in mind, both for the schools and families involved in the education system. As a result, all materials discussed were chosen for their low- or no-cost prices and compliance with the CSTA K-12 CS Standards.

While the curriculum is designed to start with an application of basic computer science concepts, its integration with math has the potential to interfere with other mathematics courses the school districts offers. For example, a seventh grader may place into an advanced math course but also hold an interest for computer science. For them, they must decide to pursue an advanced mathematics path or enroll in a course that repeats previously learned mathematics. This issue depends on the conditions of each school. In addition, the proposed curriculum's use of the Bootstrap:Algebra requires an instructional workshop to teach the material in a professional manner. There are virtual and in-person workshops offered, but it is important to note that each registration ranges from $1200 to 1400$.

An opposing view may suggest that learning multiple programming languages at once will be too challenging for novice programmers in the context of the ninth grade course. However, the contexts around the use of HTML and CSS versus Python differ greatly. While the former both pertain to the layout and stylization of webpages, the latter is a programming language for developing programs. Due to this differentiation, students will work closely with the translation process between Pyret, from the eighth grade course, and Python for the first half of the school year. After this, the second half will be dedicated to the Code.org Web Development unit for students to learn what HTML and CSS is and how it works. This separation helps ease the process for students to learn both types of languages.

6 CONCLUSIONS

The standardization of CS K-12 curricula is an integral part of bridging the digital divide. By ensuring accessible computer science education, all students are empowered to pursue their interests in technology and the CS community. The contribution presented in the preceding sections is one of many submissions towards the academia surrounding this subject with the hope of selecting one as the standard. In contrast to other frameworks, however, the courses discussed were designed to foster non-technical skills that are essential for computer scientists. This is possible through innovative teaching methods that focus on social and contextual factors when communicating material to students. In addition, the curriculum directly challenges economic barriers that prevent underprivileged communities from enacting computer science education policy. The courses explore the sectors within computer science using low-cost resources to ensure that schools are capable of using the framework. However, a form of accreditation is necessary before it can be considered for implementation. The design of the curriculum reflects a multitude of standards presented by the CSTA, yet it lacks the formal verification from the association. Moving forward, this assessment is necessary before the framework can be submitted to various educational organizations, such as Code.org or CSForALL.

REFERENCES

[1] Computer Science Teachers Association. 2017. K-12 Computer Science Standards. https://csteachers.org/page/standards

[2] Bootstrap. 2022. https://www.bootstrapworld.org/materials/algebra/

[3] Code.org Coalition. 2022. https://advocacy.code.org/stateofcs

[4] CodeHS. 2023. https://codehs.com/curriculum/catalog?tag=Python

[5] CodeHS. 2023. AP Computer Science A Textbook. https://codehs.com/textbook/apcsa_textbook/10.2

[6] Code.org. 2022. https://studio.code.org/s/csd2-2022

[7] Code.org. 2023. https://studio.code.org/docs/ide/applab

[8] Collegeboard. 2020. *AP Computer Science Principles Course and Exam Description.* Collegeboard. https://apcentral.collegeboard.org/media/pdf/ap-computer-science-principles-course-and-exam-description.pdf

[9] Erhan Delen and Okan Bulut. 2011. The Relationship Between Students' Exposure to Technology and Their Achievement in Science and Math. *The Turkish Online Journal of Educational Technology* 10, 3 (Jul 2011).

[10] Orit Hazzan, Judith Gal-Ezer, and Lenore Blum. 2008. A model for High School Computer Science Education. *Proceedings of the 39th SIGCSE technical symposium on Computer science education* 40, 1 (2008), 281–285. https://doi.org/10.1145/1352135.1352233

[11] Philip Machanick. 2007. A Social Construction Approach to computer science education. *Computer Science Education* 17, 1 (Mar 2007), 1–20. https://doi.org/10.1080/08993400600971067

[12] Caroline Miller. 2023. https://childmind.org/article/when-are-kids-ready-for-social-media/

[13] Samim Mirhosseini, Austin Z. Henley, and Chris Parnin. 2023. What is your biggest pain point? *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1* (Mar 2023), 291–297. https://doi.org/10.1145/3545945.3569816

[14] Virginia Department of Education. 2016. Mathematics Standards of Learning for Virginia Public Schools: Grade Eight.

[15] Virginia Department of Education. 2016. Mathematics Standards of Learning For Virginia Public Schools: Grade Seven.

[16] Scratch. [n. d.]. https://scratch.mit.edu/educators#teacher-accounts

[17] Emiliana Vegas, Michael Hansen, and Brian Fowler. 2021. Building Skills for Life: How to expand and improve computer science education around the world. https://www.brookings.edu/essay/building-skills-for-life-how-to-expand-and-improve-computer-science-education-around-the-world/

[18] Jennifer Wang and Sepehr Hejazi Moghadam. 2017. Diversity barriers in K-12 computer science education. *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education* (2017), 615–620. https://doi.org/10.1145/3017680.3017734