Computational Thinking begins at an early age: Ideation about Computational Thinking.

The University of Maryland, College Park March 18, 2021

Author Note:
This paper was prepared for INST 652 Design Thinking and Youth taught by Dr. Mega Subramaniam

Overview

Today, we live in a society where algorithms are essential: from managing financial investments to managing daily schedules to diagnosing cancer. In today's complex and technological society, someone who can understand and trust algorithmic solutions, as well as participate in the design and development of such solutions, will have more success in every aspect of daily life.

Computational thinking has become a popular term for describing and promoting new ways of thinking. But the term remains elusive to even those in the field of computer science. My aim in this paper is to describe the challenges in current methods for teaching computational thinking and to propose a solution to instill CT skills from an early age. Computational thinking is problem-solving, where one analyses the problem and finds the best possible solution. This can be done by breaking down the problem into parts or recognizing patterns in the issue. CT involves defining, understanding, and solving problems; reasoning at multiple levels of abstraction; understanding and applying automation; and understanding the dimensions of scale.

Computational thinking includes these four major elements: problem decomposition, pattern recognition, abstraction, and algorithms.

Problem Decomposition

It is the process of breaking down. A problem can be decomposed into parts that make it easier to solve by breaking it down. A computer programmer uses decomposition in the same way as a consumer when choosing a solution to a problem. As a programmer breaks the program's requirements into smaller chunks, he or she considers the goals of the program. A programmer creating a video game would need to think about what the game's world will be like, which characters the game will include, and what plot will be in the game. Each of these subparts of the game could then be further broken down. The decomposition can also save developers time. Detecting errors in smaller amounts of code is much easier than in longer code. You can also reuse smaller blocks of code more easily. Decomposing problems will help us think about them logically.

Pattern Recognition

Patterns are regularities found in the world. When we recognize patterns, we can predict what will happen in the future. When we learn to read and recognize people we know, we recognize patterns. Nature is full of patterns as well. In computer programming, pattern recognition is extremely important. Identifying patterns allows programmers to develop

short segments of code that can easily be repeated. One of the most famous codes in history is the Enigma code, which was used by Germany during World War II. The Germans used a special machine to translate messages into code. It was possible to encode messages in 158 quadrillions of ways (158,000,000,000,000) every night. The British cracked the Enigma code using pattern recognition after working hard to do so. Observing patterns in the messages sent by the Germans, the British noticed that messages often began with the same word and ended with the phrase "Heil Hitler". The Enigma code was cracked by the British within five and a half months of discovering these patterns. In his time, Alan Turing, a mathematician at the University of Cambridge, was one of the most important code breakers.

Abstraction

The purpose of abstraction is to make solving a problem easier by filtering out certain details. The key is to figure out which details aren't as important so that the solution is simpler. As you search for patterns, you might begin to notice that some details are more pertinent than others. It is vital that children understand what information is relevant and what information is irrelevant so that they can solve problems correctly and efficiently.

Writing Algorithms

An algorithm is a set of rules that describes in detail the steps needed to complete a task. Algorithms are sometimes called programs. As a matter of fact, the biggest difference between an algorithm and a program is that a program is typically written using some kind of programming language, while an algorithm is a detailed set of instructions for performing some kind of task. Rather than starting to write code before clearly defining the steps to complete a task, good programmers outline the process for each step before they begin. The ability to think through algorithms prior to writing computer code can save a programmer a lot of time and anxiety when writing programs.

Despite the fact that the concept originated from computer science, students can utilize CT with or without a computer. CT draws upon a rich history of studies of human cognition, including systems thinking, problem-solving, and design thinking.

Computer-based thinking (CT) parallels the core practices of STEM (science, technology, engineering, and mathematics) education and supports a student's knowledge of science and math concepts in an effective way. Even so, despite the synergies between CT and STEM education, integrating the two to support synergistic learning remains a difficult

task. There is relatively little known about how conceptual understanding develops in such learning environments and students' difficulties working with integrated curricula.

I believe that efforts must be made to lay the foundations of CT long before students experience their first programming language. First, we posit that programming is to Computer Science, what proof construction is to mathematics, and literary analysis is to English. Hence by analogy, programming should be the entrance into higher CS, not the student's first encounter in CS. We argue that in the absence of programming, teaching CT should focus on establishing vocabularies and symbols that can annotate and describe computation and abstraction, suggest information and execution, and provide notation around which mental models of processes can be used be built. Lastly, we conjecture that students with sustained exposure to CT in their formative education will be better prepared for programming and the CS curriculum. Furthermore, they might choose to major in CS not only for career opportunities but also for its intellectual content. The following questions guide our work:
1. How might we broaden youth's understanding of computational thinking concepts?
2. How might we help youth to connect computational thinking concepts to real-life situations?
3. How might we create a space that promotes the learning of CT concepts through collaboration?
4. How might we build a learning platform that explains CT concepts in a simpler way and is suitable for youth ages 7-13?

Methods

This study aimed to utilize the video interviewing technique to interact with youth about their experience with learning computational thinking concepts in school. In doing so, we sought to explore the methods that youth used to learn computational thinking concepts. At the same time, the study also aimed to investigate the challenges in these methods and evaluate their effectiveness.

Data Collection

One researcher undertook the interviews. Each lasted approximately 20 minutes to 30 minutes and they were conducted online using video conferencing tools. Guidelines for the interview were developed, focusing mainly on two themes (in addition to a set of background questions):

1. Didactic practices and strategies employed when teaching computational thinking concepts, including a focus on the tools and approach used.
2. To affirm if children should be introduced to CT concepts at an early age

The interviews were recorded with the consent of the interviewees and were later fully transcribed. Please find the interview guide below:

Starter questions to make participants feel comfortable:

1. Which grade do you currently hold (or are you currently enrolled in college)?
2. In which class do you excel? Why?
3. In which class do you struggle? Why?
4. Do you like college? Why or why not?
5. What were your aspirations in choosing your major/ field of study?

Questions specific about CT:

1. Did you face any difficulties in recognizing patterns in problems?
2. How did you learn to separate information that is relevant and important from extraneous details while solving problems?
3. How were you taught algorithmic thinking at school/college?
4. Did you take any individual effort to improve algorithmic thinking?
5. Did any specific game interest you in problem-solving?
6. Do you remember how and when you were introduced to computational thinking concepts?
7. What kind of resources did you use to improve your problem-solving skill?
8. Do you prefer working alone or with a group while solving problems?
9. Are there any examples of when you learned using the project-based learning approach?
10. Did you have any fears or anxieties when you first came into this? If so, what do you think caused them?
11. What approach would you use if you were asked to teach the problem-solving skill?
12. Would you have benefited from being introduced to CT earlier in school?
13. What were your parent's contributions in helping you towards improving computational thinking?
14. How do you stay connected with your child’s progress towards learning computational thinking at school?

15. What kind of activities do you perform at home to inculcate your child's CT skills?

In choosing to use interviews to collect data, I sought to capture "the user's needs, values, and beliefs" (IDEO, 2015, p. 34). I used this method to ensure I was collecting the most information possible from the participants, despite some warnings regarding interviewees saying what they think the research wants to hear and doing something different than they would normally say. A total of 4 participants were interviewed. Among the participants, two were 20-year-olds enrolled in a STEM program. Another participant works as an engineering manager for a software company while the last participant is a 15-year-old currently learning to program at school. I also tried to alleviate some of the pressure brought on by an interview by starting the process by asking questions about their day and background. The interview protocol included 20 questions including some follow-up questions. The interview took place in a semi-structured format, such that I was able to anticipate the responses and thoughts of the youth and ask them probing questions in order to get a more complete understanding of their answers. As a user researcher, I also made sure to focus on participants' nonverbal behaviors such as posture, hand gestures, and facial expressions to glean more behavioral and emotional information. When I was interviewing one of the STEM undergraduates I was able to capture the passion that person had towards programming from his vigorous flailing of hands while talking about it.

Interviews

According to IDEO (p. 10, 2015), "Design Thinking begins with in-depth interviews." To gain a clear understanding of what kids have to say about computational thinking, I started my interview process with two participants from the STEM program. I chose to interview 20-year-olds majoring in STEM fields because they must know CT concepts well in order to succeed. In doing this, I hoped to understand their experience learning computational thinking and how it benefited their problem-solving ability. A follow-up interview was also conducted with each of them to determine how far they understood the major CT concepts (Decomposition, Abstraction, Pattern recognition, and algorithmic thinking). My third participant is a high school student who is enrolled in a program that includes an introduction to programming, so I wanted to know about her experience. Moreover, she has experience in learning with Scratch from her primary grades, which makes her a great candidate. The purpose of interviewing her is to also determine whether early exposure to CT concepts has benefitted her in her current course. Lastly, I chose the parent who is the engineering manager of a company as my final participant. It was important for me to understand how parents are involved in the learning process,

and, since this person is an SME, I also wanted to understand his perspective on how CT can be applied effectively in real-life situations.

Participant 1
The first participant was a 20-year-old enrolled in a STEM program. In addition to his understanding of problem-solving, he was enthusiastic about the program. Because he wasn't taught about CT in school, he had difficulty grasping the concepts when he enrolled in the course in college. However, his understanding of CT was unclear. In his eyes, CT and coding are similar.

Participant 2
The second participant was a 21-year-old enrolled in a STEM program. During the program, he learned how to program through game-based learning. Unfortunately, he also had similar misconceptions about CT. He believed that CT could only be achieved through programming. He was unaware that he had unwittingly applied concepts from CT to his coursework and internships. These examples show that students have not understood the correct definition of CT.

Participant 3
This participant is a high school student who is currently studying introduction to programming. She was introduced to programming in her elementary grades through a visual programming tool named Scratch. She mentioned that she has difficulty with recognizing patterns in her current learning since it was difficult to get her mind to work that way.

Participant 4
This participant is a parent who works as the Engineering Manager for a company. According to him, he does not have a lot of time to help his kid during his daily routine. But he strongly believes that CT concepts need to be taught at an early age in order to help shape kids' thinking. He believes that parents' involvement can help in teaching the concepts in a simple way to children from a very early age.

The Rationale of Data Collection Methods

As described previously, youth interviews and observations were selected as the main data source because they would be the quickest and more direct manner through which I could learn what my participants understood about computational thinking. I chose to also interview adults, even parents or teachers who might be "experts" because I wanted

to understand how involved the parents or teachers are in the learning journey of the kids. Also, observations are important because it helps negate many types of bias and much of the subjective interpretation that comes with researchers self-reporting “facts.” Basic psychology reveals that people remember and relay things in different ways. This method is much better than questionnaires where responses are limited to answers to predetermined questions. In this method, I was able to come up with context-specific follow-up questions based on the response of the participants. In the future, I would love to perform an ethnographic study where I can perform a direct observation of users in their natural environment. The objective is to gain insights into how kids interact with students and parents in their natural environment and learn computational thinking concepts.

Characteristics of Stakeholders

In this project, the main stakeholders are the youth creators. The tool or system will be used by them and they will do the learning. However, other individuals who have direct ties to the tool or system will be included as well. Other stakeholders may be teachers or parents who will be guiding the youth through the learning process depending on where the tool is being used; the tool was designed to be versatile enough to be used in a variety of different situations. This tool can also be used by professionals in the STEM field to brush up on their basics of CT concepts.

While anyone can who is interested in learning the CT concepts can use the tool, it is important to me that I focus on youth ages 7- 13 (2nd-8th grades) from varying backgrounds who are fluent in the English language. I have selected this age group for a few reasons. Previous work on computational thinking in young elementary school-age children can be found in the research literature on constructionist programming environments (Repenning, Webb, & Ioannidou, 2010; Resnick et al., 2009). Wing (2006) describes computational thinking as a fundamental skill for everyone, not just for computer scientists. Young children are often seen using recycled materials to build cities and bridges and becoming "little engineers" (Bers, 2008b) so I chose second-grade students as the lower-bound. For the upper bound, I chose to end at 8th grade because high school students are already proficient users of some too existing tools. Certainly, middle school students could benefit as well, but since not all students will be able to utilize more complicated software to its full extent, the need for a simplified tool continues all the way through middle school.

The solution will not be restricted to a particular race or region, since I want it to be universal, and applicable to anyone fluent in English. The users of the solution should be comfortable with using technology to access the solution. As a practical matter, kids in the specified age range might not be as technologically savvy as expected, so adult supervision is recommended to facilitate their learning. I selected this age range after talking to the participant who is a high school student who was introduced to CT concepts in the fourth grade and is now well versed in programming. The participant mentioned that they used the Scratch tool to learn programming and CT concepts.

Adults between 30 and 50 years of age will be able to guide the kids through their learning and help them learn how to use the solution. The adults should be technically proficient so they can assist the kids. Considering English is the primary language of communication, it is expected of adults to be fluent.

Data Analysis

As part of the interview, careful notes were taken to document the important points. Once the interviews were done, the recorded interviews were transcribed for further processing.

Findings

This paper presents the initial findings from the study of using video interviewing tools to understand the challenges in the learning methods of computational thinking. These initial findings stem from analyses of four selected interview videos as well as field notes regarding the effectiveness of the activity.

One participant had used Scratch as a starting point for CT learning. Using Scratch, kids can create programs by connecting up blocks that represent various concepts like expressions, control structures, and programming statements. As the blocks are formed, you can see where they fit together, and the drag-and-drop system prevents connecting blocks in ways that would not have any effect on the programming logic. Block-based programming offers the advantage of being easy to understand since the blocks are described in a common language. Coupled with drag-and-drop interactions and the convenience of browsing programming languages, its mastery is achievable. With Scratch's features and capabilities, it is logical to assume that institutions would include it in their teacher training plans.

Even students who are currently training to become engineers do not understand what computational thinking is. They have failed to realize that it is a concept that is ingrained in our day-to-day life. As a result, they keep getting confused between coding and

problem-solving skills. The approach to teaching this essential skill needs to be changed. As pointed out by the interviewees, we need measures to help students understand the concept and apply it to real life rather than memorizing it just for good grades.

Further, CT skills need to be taught early in a child's life rather than starting in high school. Also, there is a greater focus needed on improving collaboration skills. They should not see it as a competition when they work with others. Instead, they should help each other in finding efficient solutions.

The research data shows that parents don't take any extra effort to teach CT concepts at home. Parents should become more involved in their children's learning of computational thinking. Parents' involvement can help in teaching the concepts in a simple way to children from a very early age.

User Tasks

The challenges that were identified in the learning methods of the CT concepts are:

1. Children often focus on memorizing the syntax of programming languages rather than learning the concepts needed to be useful in the future.
2. Computational thinking is a way of thinking rather than a specific body of knowledge about a device or language. Primary-grade children don't shy away from taking risks. Instead of introducing these concepts in high school, it is better to take advantage of young children's natural inclination to explore and play and encourage problem-solving skills to improve students' thinking. It cultivates playful thinking while providing structure so that the skills that students are learning will be able to be transferrable to more complicated tasks in the future.

Thus, primary grade children will be utilizing the solution to learn about the CT concepts in collaboration with parents and teachers. The solution will be presented in a real-life setting, so the knowledge can easily be applied in the real world. This content will be designed in a way that is easily understood by elementary school students so that they can apply the knowledge in the future. The process of teaching programming in the later grades will thus be easier.

While using the solution, adults play an equally important role. It is important that they help the children in their learning process. They should serve as mentors and assist in the use of solutions and clarification of concepts.

Social and Technical Systems

As with any work, this designed solution will be within the existing social and technical systems. The solution may intersect with a variety of social systems, including those between youth, within families, within communities, in schools, and in homes. Most likely, when youth are using this design, they will be doing so with teachers or parents in the same creative environment. This age group thrives off social interaction and even when they work independently, they are interacting with one another. So, the solution would be designed in such a way that it prompts the kids to collaborate with adults during the learning process. It is imperative that this tool be built within the current technology systems in order to be available to all students. Almost every student has access to a device, most likely a smartphone, whether it belongs to them personally or to a parent. I propose to design an app that would be easy to download on their phones and can be used in various settings including the home and school. Using the social media sharing option, the solution would enable the children's learning progress to be shared easily with their parents and teachers.

Evaluation Criteria

Anything designed for kids should help them to learn and apply concepts in computational thinking to their daily lives, not just memorize them for a test. I want the design to:

- Stress the relevance of computational thinking to real-life situations
- Be user friendly for youth
- Encourage youth to view computational thinking as more than just programming
- Versatile enough to be used in different areas including, but not limited to, classrooms, after school activities, and homes
- Be easily comprehendible by elementary school students

Implications

Computing Thinking builds on Computer Science concepts, but the two are not the same. A computer science degree teaches students about computers and computing systems, but computational thinking refers to the thinking processes involved in solving complex problems and generalizing this problem-solving process to a variety of problems. The research data showed that computational thinking involves much more than just programming. Many students have the misconception that computational thinking is the same as coding because they have all been introduced to CT concepts via programming. This notion should be broken by introducing CT concepts to elementary school children in a way that lets them understand that CT concepts are intrinsic to their everyday lives.

Students were introduced to CT using Scratch, however, that also gives more importance to programming rather than concepts such as pattern recognition, abstraction, etc. For example, the high school student that I interviewed who was taught CT through Scratch mentioned having difficulty with recognizing patterns since it was difficult to get her mind to work that way. I want to help children begin to alter their way of thinking as early as age 7 because they will have a curiosity for learning and that can be used to shape their thoughts. My objective is to design a solution to teach CT concepts in a real-life setting so that whatever students learn gets retained and their knowledge can be successfully applied in real life. As they are introduced to programming in high school, they don't feel intimidated as they are required to apply CT concepts, but rather the concepts learned at early age should kick in naturally and enable them to code effectively and efficiently. Finally, I hypothesize that students whose learning has been sustained by early exposure to CT will be better prepared for programming and CS coursework and that they may choose to major in CS for both career opportunities and intellectual growth. It is important that I design the solution in a way that appeals to children aged 7 to 13 years. Additionally, I will ensure that the content is designed to draw on their own experiences so that they can transfer the concepts effectively. Moreover, I will design the solution's content so that it encourages collaboration with adults. As a result, I will ensure that learning outcomes demonstrate a thorough understanding of CT concepts.

References

1. IDEO. (2015). Design Thinking for Libraries: A toolkit for patron-centered design (Web-version) Chapter 2, p. 25-48. Retrieved from http://designthinkingforlibraries.com/

2. Wing, Jeannette M. (2006). Computational thinking. Communications of the ACM, 49(3), 33-35. Retrieved June 10, 2010, from http://www.cs.cmu.edu/afs/cs/usr/wing/www/publications/Wing06.pdf

3. Bers, Marina U. (2008b). Engineers and storytellers: Using robotic manipulatives to develop technological fluency in early childhood. In Olivia N. Saracho & Bernard Spodek (Eds.), Contemporary perspectives on science and technology in early childhood education (pp. 105-125). Charlotte, NC: Information Age.

4. Voogt, J., Fisser, P., Good, J., Mishra, P., & Yadav, A. (2015, June 17). Computational thinking in compulsory education: Towards an agenda for research and practice - education and Information Technologies. SpringerLink. Retrieved

March 22, 2022, from https://link.springer.com/article/10.1007/s10639-015-9412-6

5. University, J. J. L. E., Lu, J. J., University, E., George H.L. Fletcher Washington State University, Fletcher, G. H. L., University, W. S., University, M. S., Technology, G. I. of, University, X., Columbia, U. of B., & Metrics, O. M. V. A. (2009, March 1). Thinking about computational thinking: Proceedings of the 40th ACM technical symposium on computer science education. ACM Conferences. Retrieved March 22, 2022, from https://dl.acm.org/doi/abs/10.1145/1508865.1508959?casa_token=bGtE6BqtC94AAAAA%3AjecXq-YCrpgaJd8CyaU7TRpjuQ0y3w4g_wOUHHOrTa2S41P-EjpLgdNSrEEy0FTolE1qdGlTKKMBVw

6. Montiel, H., & Gomez-Zermeño, M. G. (2021, May 21). Educational challenges for computational thinking in K–12 education: A systematic literature review of "scratch" as an innovative programming tool. MDPI. Retrieved March 22, 2022, from https://www.mdpi.com/2073-431X/10/6/69

7. Ching, Y.-H., Hsu, Y.-C., & Baldwin, S. (2018, April 30). Developing computational thinking with educational technologies for Young Learners - TechTrends. SpringerLink. Retrieved March 22, 2022, from https://link.springer.com/article/10.1007/s11528-018-0292-7

8. Angeli, C., & Giannakos, M. (2019, November 1). Computational thinking education: Issues and challenges. Computers in Human Behavior. Retrieved March 22, 2022, from https://www.sciencedirect.com/science/article/pii/S0747563219303978?casa_token=qDuWvhEYt5wAAAAA%3AWRsKuwtIW9S01Qx_2UHOnu6le1Rd7bXlzwGH5rftTwJGro2LuurEZoihJro2bj9F8hcIeMBN6o8#bib6

9. Basu, S., Biswas, G., Sengupta, P., Dickes, A., Kinnebrew, J. S., & Clark, D. (2016, May 21). Identifying middle school students' challenges in computational thinking-based Science Learning - Research and practice in technology enhanced learning. SpringerOpen. Retrieved March 22, 2022, from https://telrp.springeropen.com/articles/10.1186/s41039-016-0036-2#Sec10

10. Turchi, T., Fogli, D., & Malizia, A. (2019, February 4). Fostering computational thinking through collaborative game-based learning - multimedia tools and applications. SpringerLink. Retrieved March 22, 2022, from https://link.springer.com/article/10.1007/s11042-019-7229-9

11. University, D. J. P. T., Portelance, D. J., University, T., University, M. U. B. T., Bers, M. U., Arkansas, U. of, Contributor MetricsExpand All Dylan J Portelance Tufts University Publication Years2015 - 2015Publication c, & Dylan J Portelance Tufts University Publication Years2015 - 2015Publication counts1Available for Download1Citation count20Downloads (cumulative)566Downloads (6 weeks)13Downloads (12 months)81Average Citation per Art. (2015, June 1). Code and tell: Proceedings of the 14th International Conference on Interaction Design and Children. ACM Conferences. Retrieved March 22, 2022, from https://dl.acm.org/doi/pdf/10.1145/2771839.2771894

12. Hsu, T.-C., Chang, S.-C., & Hung, Y.-T. (2018, July 3). How to learn and how to teach computational thinking: Suggestions based on a review of the literature. Computers & Education. Retrieved March 22, 2022, from https://www.sciencedirect.com/science/article/pii/S0360131518301799?casa_token=CNjlKqTbOg4AAAAA%3AQBFQXwn8gsX9bkUXg4SFb-DZHHYfafGIMZ90yLuBfDaQTErBbhkaGHRPb0e12pt5TdljKzlk3Wc

Computational Thinking begins at an early age: Ideation & Iteration about Computational Thinking.

The University of Maryland, College Park May 5, 2021

Author Note:

This paper was prepared for INST 652 Design Thinking and Youth, taught by Dr. Mega Subramaniam.

Introduction

Almost every aspect of our lives is influenced by computer technology. A growing number of ordinary objects are being designed to operate via computer programs (Hartigan 2013). 21st-century skills aim to teach creativity, critical thinking, clear communication, teamwork, and effectively solving complex problems with collaboration. Computational thinking is of paramount importance.

Using computational thinking, we can view systems to consider how computers can be used to solve problems, model data, and generate tangible solutions. Wing (2011) defined computational thinking as the thought process of finding a solution to a problem step by step, in a logical and organized manner, much like a computer. CT is viewed as the first step in the learning process before students learn computer programming skills. While this is true, computational thinking has a much broader scope. CT develops skills and approaches that are universal and transferrable and are extremely apt and valuable in the age of computers. Although a future coder certainly needs CT, it does not follow that everyone who has learned CT eventually should learn to code. The development of a think-like-a-computer capability will benefit professionals in any field.

Considering that we need a workforce with skills in computational thinking, how can the next generation be prepared? A way of thinking takes a lot of time to develop. For future professionals to fully master and use CT, it is crucial to introduce CT concepts to kids at an early age and help them stay involved with it throughout their academic careers Yadav, Mayfield, Zhou, Hambrusch, and Korb, 2014).

In this project, I have partnered with kids from 7 - 13 years of age to devise solutions that will help them learn and apply computational thinking to everyday life. I conducted interviews with kids to understand the challenges in the current learning methods of computational thinking. The biggest challenge is that many students haven't been exposed

to computational thinking early, causing difficulty understanding its concepts. Furthermore, since computational thinking is a mental process, they find it difficult to adapt. The second biggest challenge is that many students have not been able to connect CT to everyday life since they have been trained to memorize it just for good grades. The following questions guided my work:

1. How might we develop a computational way of thinking in youth?
2. How might we help youth connect computational thinking concepts to real-life situations?
3. How might we build a learning platform that explains CT concepts more straightforwardly and is suitable for youth from an early age?
4. How might we create a space that promotes the learning of CT concepts through collaboration?

Together with my design partners (kids aged 7 - 13 years), I designed solutions to overcome the above challenges and create spaces that help them understand CT concepts and apply the learning to real-world scenarios through collaborative learning.

A. User Tasks

Kids don't shy away from taking risks. It is better to take advantage of young children's natural inclination to explore and play and encourage problem-solving skills to improve students' thinking. It cultivates playful thinking while providing structure so that the skills that students are learning will be transferable to more complicated tasks in the future. Thus, primary grade children will be utilizing the solution to learn about the CT concepts in collaboration with parents and teachers. The solution will be presented in a real-life setting, so the knowledge can easily be applied in the real world. This content will be designed in a way (game-based

learning, image-based learning) that is easily understood by elementary school students so that they can apply the knowledge in the future. The process of teaching programming in the later grades will thus be more effortless. While using the solution, adults play an equally important role. They must help the children in their learning process. They should serve as mentors and assist in using solutions and clarification of concepts.

Requirements

This designed solution will be within the existing social and technical systems, as with any work. The solution may intersect with various social systems, including those between youth, families, communities, schools, and homes. When youth use this design, they will most likely be doing so with teachers, parents, or other kids in the same creative environment. This age group thrives off social interaction, and even when they work independently, they interact with one another. So, the solution would be designed so that it prompts the kids to collaborate with other kids and adults during the learning process. I have created two different design solutions to the problems discussed. These learning tools must be built within the current social and technological systems to be available to all students. Almost every student has access to a device, most likely a smartphone, whether it belongs to them personally or to a parent. Thus, the first design solution will be integrated into the existing ubiquitous devices. The second design solution is unplugged and old school, which does not use computer technologies. Considering the disadvantages of technology, which include health problems and dangers of browsing (Charles, 2021) and families with limited access to technology (Swenson et al., 2020), I believe the second design solution is beneficial. Hence, there is a requirement to provide solutions that are universal and accessible to all.

A. Design solution 1 - Arcade app

The arcade game utilizes the game-based learning approach predominantly. The focus of game-based learning is not only to design games for students to play but to design activities that help students learn by incrementally introducing ideas and gradually moving them toward their goals (Pho et al., 2015). This learning app shows a list of games that teach various computational thinking concepts. Before the kids start playing the puzzle-type games, they are prompted to generate their plan of problem-solving, which is an essential step in problem-solving. Upon successfully completing the game, they learn about the concepts they employed through a visual learning approach. Visual learning is a style that utilizes visual tools, such as images, graphics, colors, and maps, so students can comprehend ideas and thoughts effectively. This app can be used in a collaborative setting where kids connect with their friends to solve puzzles. In the end, they learn about where the concept will be used in the real world, which will prompt them to have structured, useful conversations.

B. Design solution 2 - Design your own board game (unplugged)

This design solution is specifically designed for students who don't have technology access and for parents who wish to cut down the computer time for their kids. This solution is a fun learning experience without technology. Kids do not need to use technology every time they need to learn CT. I call this method Unplugged CT because it enables kids to learn concepts without using a computer. Kids can play this game with their own imaginations and a variety of craft materials. During these activities, CT skills, attitudes, and approaches are naturally

discussed and remembered. Taking inspiration from creating your own puzzles, this particular idea is about giving the kids the supplies they need to design their own games. Children can learn how to design, work in teams, and think critically by creating a board game. The game begins with the first dice roll. Depending on the number on the dice, that player must provide one idea for the game. Every time a dice is thrown, each player must build upon the existing idea. By using the art supplies, the children can explain their ideas in a more visual way. In the end, they could make their prototype out of the art materials or even digitize the design using online tools.

Design Methodology

After the inspiration phase, in which I interviewed kids and their current learning methods of computational thinking concepts, I identified the following challenges:

1. Students face difficulty adapting to CT when they haven't been exposed to it early, causing them to have difficulty understanding its concepts.
2. Students have not been able to connect CT to everyday life since they have been trained to memorize it just for good grades

The ideation phase had two parts: In the first design session, my aim was to generate ideas about how kids want to learn CT concepts. Based on the data collected, I created some prototypes. In the second design session, I reviewed those prototypes with the kids and made changes to them.

A. Design Session 1

The first design session focused on generating ideas for teaching computational thinking to kids between ages 7 and 13. Because the kids were connecting from across the country, the session was conducted online. The design sessions were

conducted with kids aged seven years, 13 years, 12 years, and eight years. I conducted the session via Zoom and obtained consent from the parents and kids to record the session. I also provided the kids with access to Jamboard, which was used during the design session. The session lasted from 50 mins to 60 mins. Since multiple themes emerged from the initial inspiration phase, I decided to combine two to form a refined How Might We question, which is How might we help children connect computational thinking with the real world at an early age?

Introduction phase (10 mins)

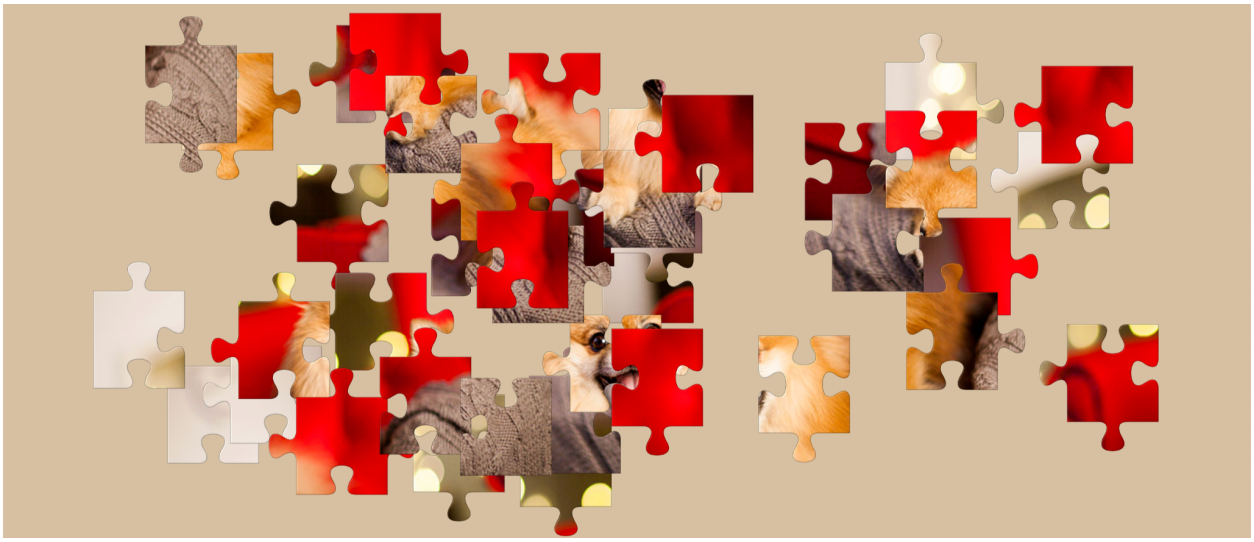
During this phase, I introduced myself to the participants and conducted some ice breaker activities. As an ice-breaker activity, I selected Show and Tell as a fun way to get children talking about their favorite possessions. Children were asked to bring a favorite stuffed animal that they bring everywhere they go so they can talk about it. I offered the older kids the option of bringing their favorite book if they did not have a toy which the 13-year-old kid took up. The ice-breaker activity was effective in getting the kids to start talking without feeling shy and helped me discover their favorite things as well. It allowed me to craft a puzzle with their favorite characters in mind. In order to get the kids to talk about their toys, I used the following question prompts.

Puzzle-solving Phase (10mins)

During the phase, I introduced the kids to the concept of problem-solving. It consists of finding solutions to problems. A problem is a situation that needs to be changed.

Before they started with the puzzle, I posed the question of the day: What is your process for solving a problem that has been given to you? And asked them to think about this question while solving the puzzle. I provided the kids with the following puzzles and allowed them to take their time to complete the activity.

Puzzle for 7 - 9-year-old kids



Puzzle for 10 - 13-year-old kids

Afterward, the kids solved the puzzles and used the Google Jamboard to reflect on their experience by writing answers on sticky notes. Some kids also sketched some of their design ideas on the jam board.

Sticky note critiquing (20mins - 30mins)

I used the sticky notes technique for this. In this technique, I aim to evaluate the puzzle-solving activity, and this would provide feedback and directions for future improvements on how I can best assist kids in learning how to solve problems (Fails et al., 2013).

In each category, kids recorded their ideas or observations on a separate sticky note. A parent or adult lent a helping hand for the children to express themselves. Below are the screenshots of the activity:

What would you do differently next time?

- have only one person touching the screen
- more people accessing the same puzzle and making it a competition.
- First I will concentrate what is the picture
- have a more laid out plan with turns and less free for all
- I want to solve more puzzles

What did you learn?

- More patience need to solve this
- have a more laid out plan
- I learnt that I can solve a problem by breaking it down to smaller parts

How would you like to learn?

- More game based learning because its fun
- have games be twisted in with problem solving
- I like to play more games to solve the puzzle quickly
- imagine a real world problem and approach it with what would you do and what supplies would you need and what steps would you take.
- I want problem solving top relate more to math as it's my favorite subject and its fun because it goes everywhere.

How did you work it out? or How do you know that?

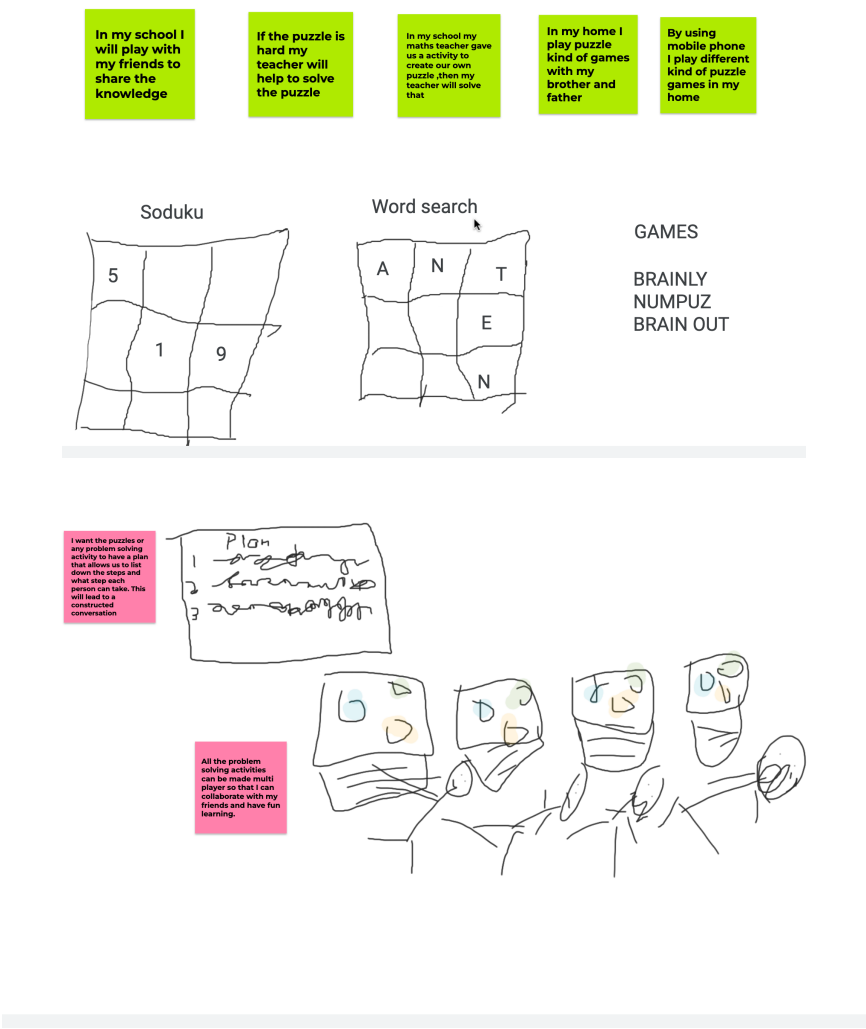
- A shape fits inside another shape that is identical. For example: a circle goes within a cirle.
- did the border first then filled it in
- wanted to do the hard ones first
- you know how big it is going to be so you can judge where to put it
- I was thinking about the shapes and then colors !!!
- I thought about the colors next. So you matched the color and then u dragged and dropped it.

What was hard?

- Different sizes were confusing.
- having to find the place for the ones that have no extra detail
- The shapes are uneven to identify correct location
- Characters means tell a story - a problem might be in it.
- Many colors means fun and puzzle might be hard
- Puzzles could have been more complex
- Few more shapes to make it fun

What was easy?

- Drag and drop was fun
- separating the edges from the middle
- I started with center so I can solve it
- At first it was very confusing for me
- I needed the background picture picture helps me to solve it
- I concentrated with the colours and corners



Sticky note evaluation

Debrief

I saved the Jamboard and recorded the session, which helped me review how the kids performed in each activity. I collected the sticky notes and examined them for patterns that indicated what areas could be emphasized in future work. I noted down all the "Big Ideas" brought to light during the session.

Some of the "Big Ideas" that I got from the design session were:

1. When children are given problems or puzzles based on their favorite things, they relate to the problem much better. They love visual stimulation, and it's also helpful to have a backstory on puzzles, as it gives them the sense that they are solving a real-world problem.
2. The kids felt that they preferred to solve problems as a team rather than individually. They mentioned that if they teamed up with friends, the problem-solving activity would be more fun. Another child's sketch shows how she prefers her friends to join the activity on their own phones so that everyone can view the same screen at the same time.
3. Another idea was to allow kids to design their own puzzles so that they can both create the problem and think about various different solutions, which they felt was a good strategy to improve problem-solving skills.
4. One of the kids had the idea of showing real-world applications after solving various puzzles so that they could relate what they learned.

B. Design Session 2

I utilized the second design session to present kids with initial prototypes of the games I designed based on the first design session. I conducted the design session with the same kids who participated in my first design session. The session was 60 minutes long, and three kids aged 7-14 participated.

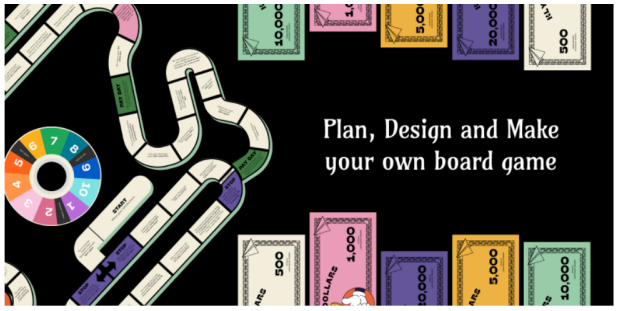
Introduction phase (10mins)

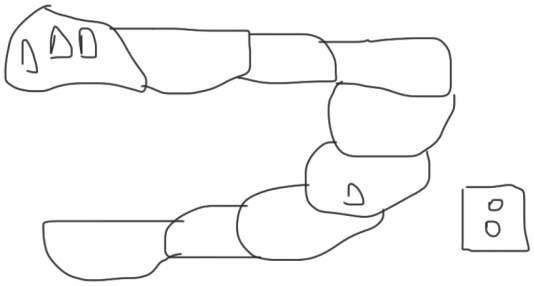
I eliminated the introduction activity since I had the same set of kids, and we had introduced ourselves in session 1. As an alternative, I chose to talk to them about their day before we began. I began the session by posing the question of the day,

"Would you prefer to play a given game or design your own?". As discussed in Beth Bonsingore's lecture (ELMS, Module 5, Lecture) and the resources provided in it (Fails et al., 2012; Poole & Peyton, 2013 in ELMS, Module 5, Lecture), the question of the day helps to transition the kids from the introduction phase of the session into the design phase. The goal is to introduce the kids to the session's main idea before jumping right into it. The children had a different set of answers. A few felt that designing their own game would make them masters and allow them to use their own imagination to come up with interesting games. On the other hand, some kids felt that they preferred to play games designed by others because they had no experience designing games themselves. A few kids said they would be open to it if they were provided with an instruction sheet or template to get them started.

Storyboarding phase (20mins)

Storyboarding (Fails, Guha & Druin, 2013) was used to present a digital mockup of the game prototypes to the children. I showed them the mockup on the Google Jamboard and made sure that they had access to the link in the meeting. In addition, I used Comicboarding since the 7-year-old had difficulty using the mouse to draw and instead described his ideas for me to illustrate (Fails et al., 2013). So using this method, the kids reviewed the existing sketches and illustrated their suggestions.





What do you think about this game?

Are the materials enough to help you DYO game?

What do you think about the directions of the game?

What would you change in this game? What did you like and dislike?

I have not designed game before so I don't know how to do it.

Good idea - can be more creative

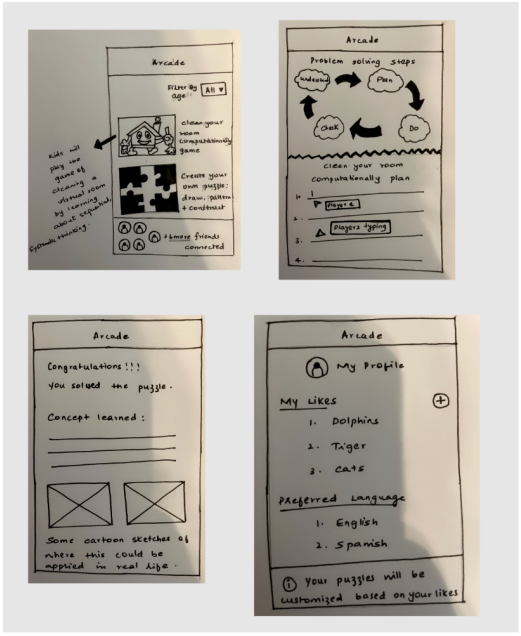
No need a template or an instruction sheet

Can make a board game about making a board agme

Need more directions because I do not know how to start thinking about this.

I like this idea because we need not have to rely on internet and phones all the time

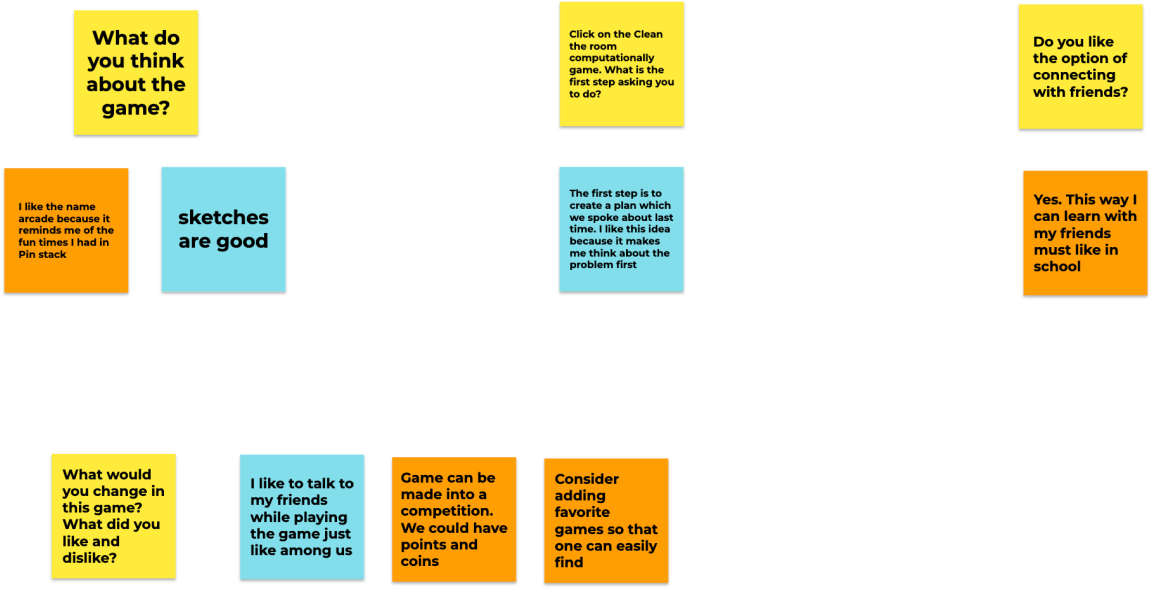
A colorful board game and dice about designing our own board game.



Storyboarding of the Arcade app and DYO Board Game

Sticky notes Evaluation (20mins)

I used the sticky notes technique to pose questions about the different features of the designs to the kids, which, according to Subramaniam (2016), is a straightforward process for youth to partake in. The sticky notes technique complemented the previous storyboarding approach because the sticky notes helped the kids talk about their likes and dislikes and draw their design ideas. The children were asked to use separate sticky notes for each idea (Subramaniam, 2016). Sticky notes seemed like the best strategy to me because I wanted the children to stay organized and provide informative feedback on the mockups I presented, and it is an excellent strategy for participants to reflect on (Knudtzon et al., 2003).



Evaluation of mockups with sticky notes

Debrief

I saved the Jamboard and recorded the session which helped me review how the kids performed in each activity. After gathering the sticky notes, I examined them to identify all the "Big Ideas" and make those changes in the mockups.

Design Summary

At first, I planned to create a library program similar to early math and early literacy programs. Including CT concepts in the library programs would help kids learn the concepts as they grow. However, I did not pursue this for two reasons. In order to implement this solution, existing library employees would have to be trained on CT concepts, a 21st-century skill, and this is an additional overhead. In addition, I found that the children were most interested in game-based learning methods after my design sessions with them. Also, if this were taught in an academic setting, they would treat it as if it were schoolwork and do it out of obligation. When I realized this, I modified my approach and decided to design something that makes learning CT fun. Based on interviews, children said that learning is not fun, but it could be if they could play a game. The reason why games are so popular with kids is that they are used constantly. It can also be controlled or managed by adults and includes a learning experience.

Designs

My design for game-based learning consisted of two different solutions. The first solution requires that students access it via technology. My goal was to make the solutions universal and accessible to everyone. Taking into account the negative impacts of technology and the limited availability of technology in remote areas, I have devised the

second solution. In the second solution, I came up with a design that can be played anywhere without any technology involvement.

A. Arcade game, a mobile app to learn CT

Overview

The arcade game utilizes the game-based learning approach predominantly. The focus of game-based learning is not only to design games for students to play but to design activities that help students learn by incrementally introducing ideas and gradually moving them toward their goals (Pho et al., 2015). This learning app shows a list of games that introduces various concepts of computational thinking. Before the kids start playing the puzzle-type games, they are prompted to generate their own plan of problem-solving, which is an essential step in problem-solving. Upon successful completion of the game, they learn about the concepts they employed through a visual learning approach. Visual learning is a style that utilizes visual tools, such as images, graphics, colors, and maps, so students can comprehend ideas and thoughts effectively. This app can be used in a collaborative setting where kids connect with their friends to solve puzzles. In the end, they learn about where the concept will be used in the real world, which will prompt them to have structured, useful conversations.

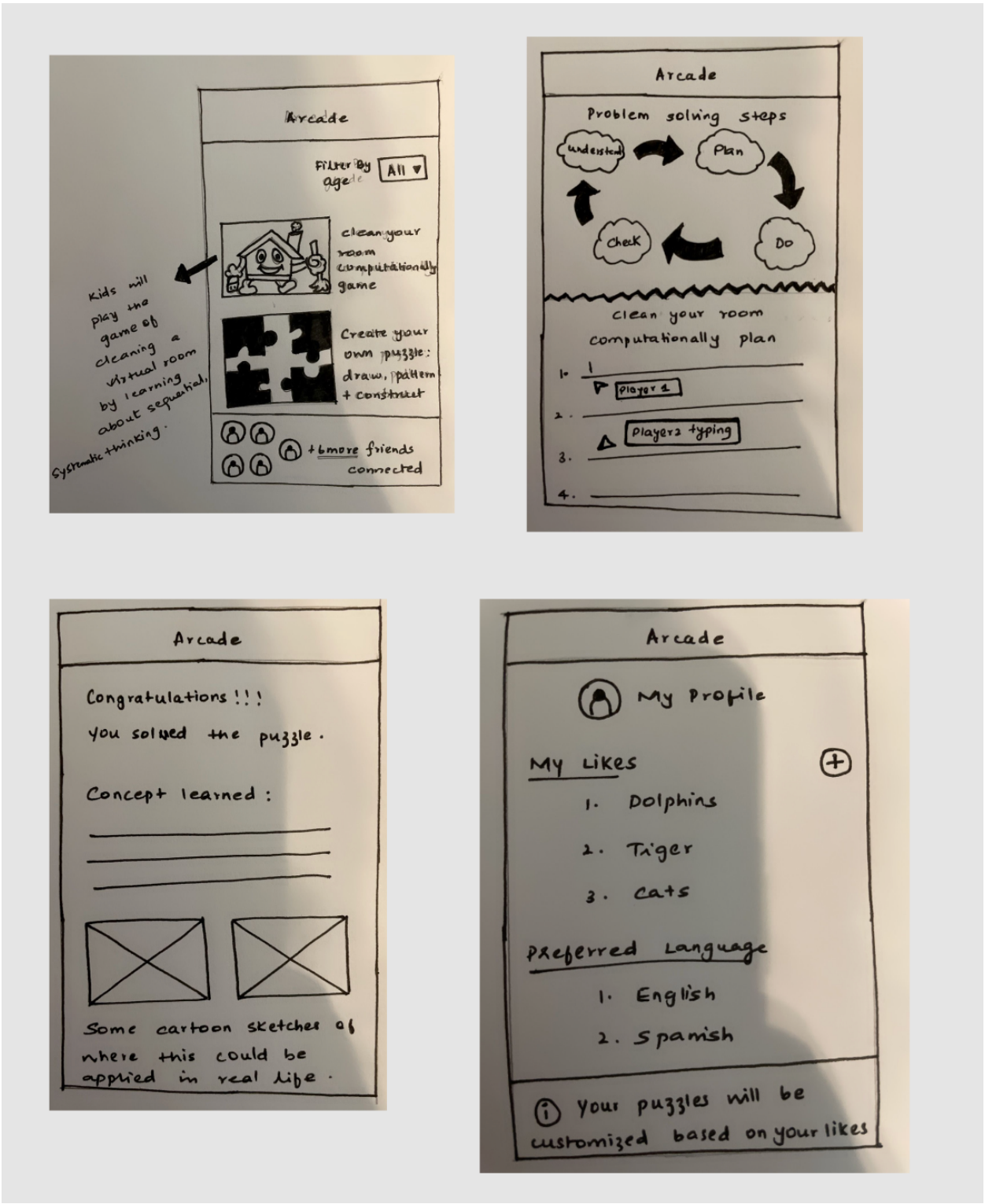
Illustrations

This app would contain a variety of problem-solving games that would help kids master different concepts of computational thinking. This app would contain games for different age groups. Game-based learning appealed to me because the kids mentioned most of what they learn at school is theoretical, and they always get excited when they learn through games because it is fun. I also learned that it is

important to customize game-based learning activities based on kids' age and interests. Based on the children's interests in the show and tell activity, I customized the puzzles in the design session. For instance, a girl brought her dolphin stuffed animal and shared the fact that animals are her favorite thing. In the puzzle-solving phase, I provided a dog jigsaw puzzle. This triggered an incredibly positive response from the kid. All the kids confirmed that they love activities that include their favorite things. This idea is used in the app in which kids add information about their likes, which will be used to customize puzzles.

To teach kids to first create a plan of how they are going to solve problems before starting the game, the first step before they begin the game would be to have them write down their plan. Once they complete the activity, they will be provided with the outcomes of the learning, and they will also be shown sketches of where this concept will be applied in real life (visual learning).

The following mock-ups were created after the first design session:



Sketches of Arcade app

After my second design session, the following big ideas emerged:

1. Children wanted the game to provide the ability to compete with their friends and earn coins.
2. Kids wanted easy access to their favorite games.
3. They also wanted an option to chat with friends from the app.

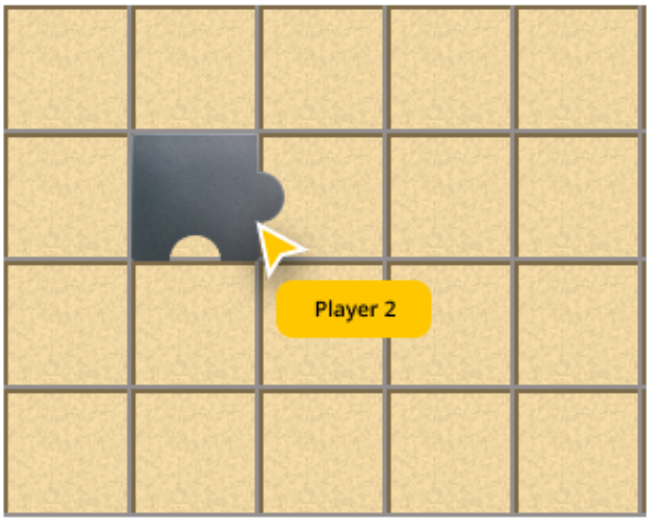
The following mockups were created after the second design session:

CT games list screen

9:41

Jigsaw Puzzle

Lorem ipsum dolor sit amet Lorem ipsum dolor sit



9:41

Profile

Stefani Wong

Novice

Edit

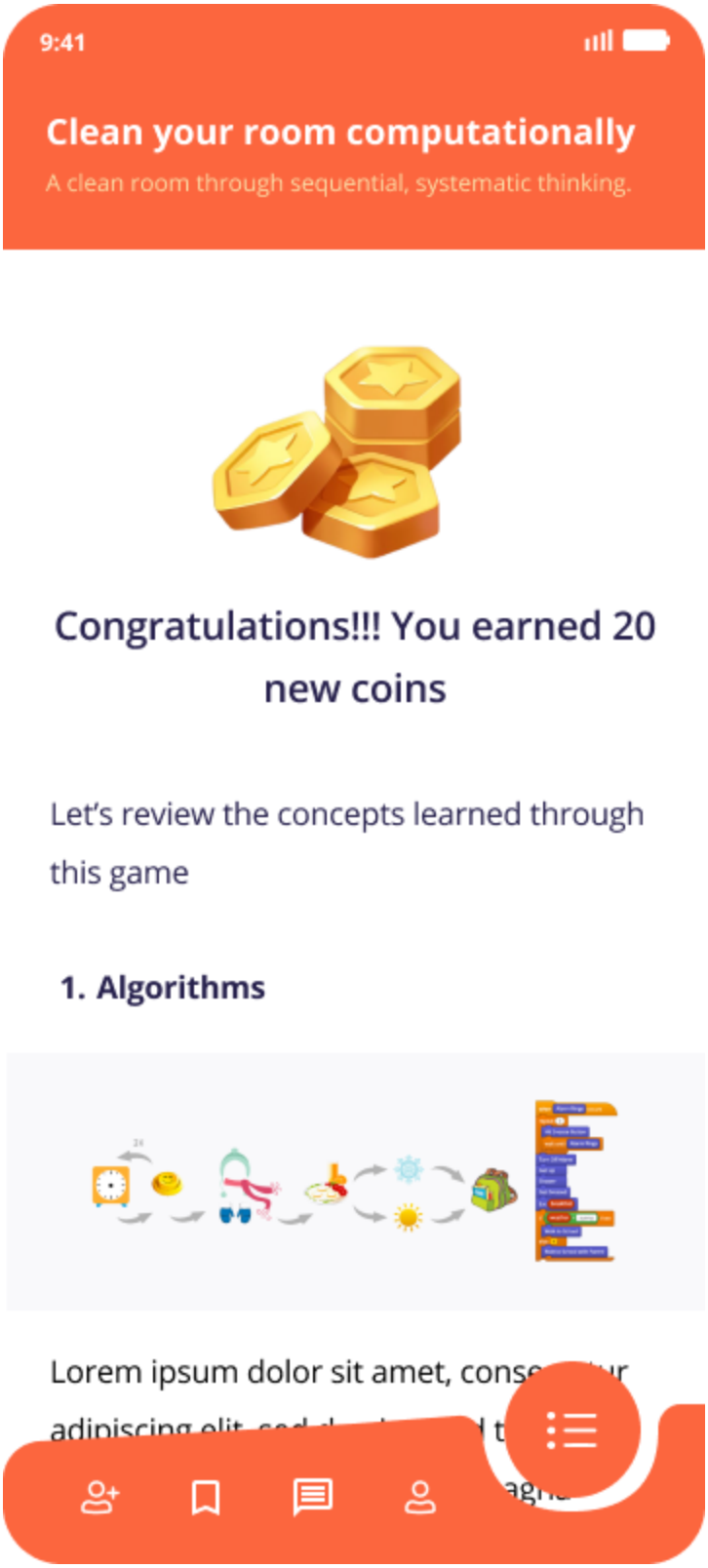
Coins Earned	120 coins
Current Rank	#20087

My Likes

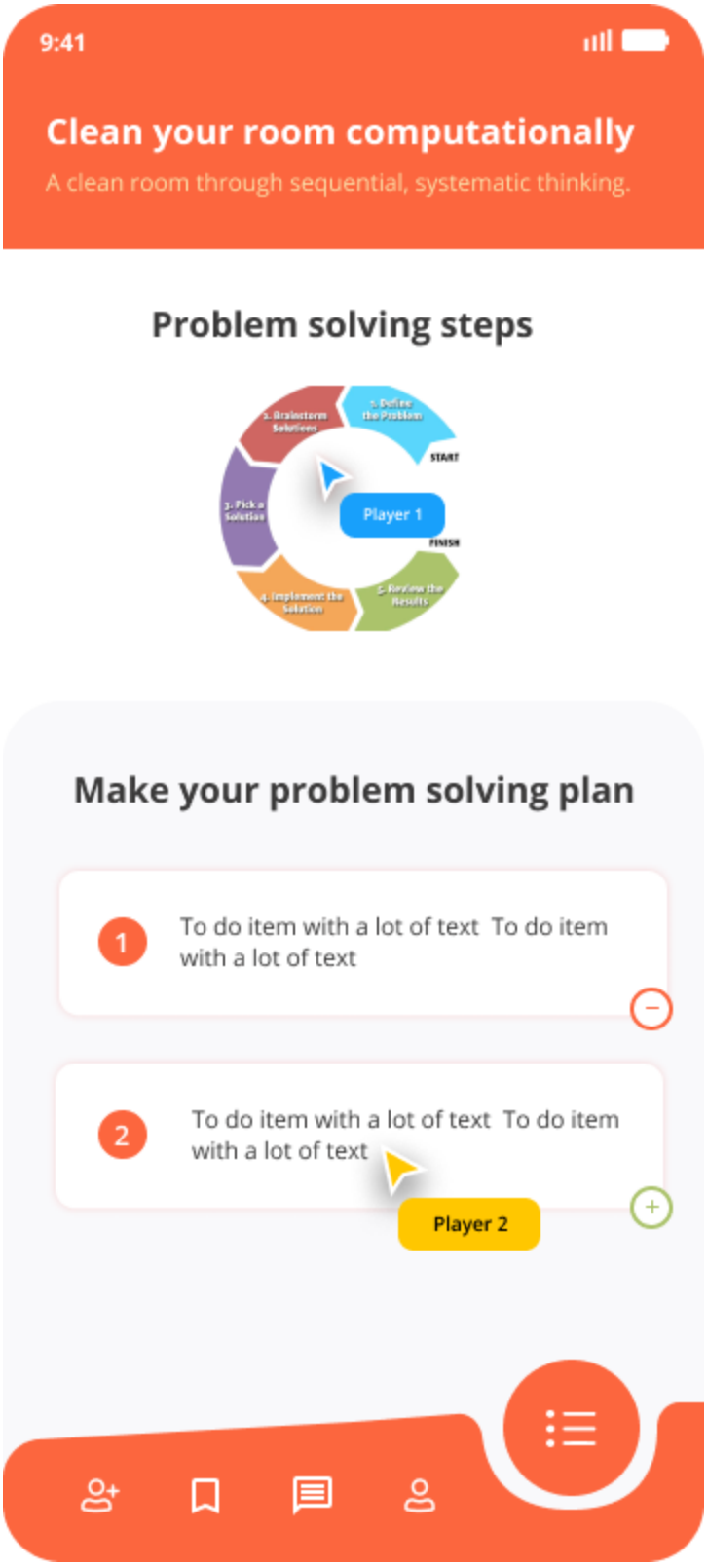
Animals	Dolphins, Dogs, Tiger	>
Animals	Dolphins, Dogs, Tiger	>
Animals	Dolphins, Dogs, Tiger	>

Preferred Language

English | Spanish



After completing the activity, they are given the opportunity to learn more



Before playing the game, kids are asked to create a plan for solving problems

Assessment

A. Advantages

- Children with more experience in online spaces may prefer this mobile app to learn computational thinking through solving puzzles.
- Game-based learning methods make this app appealing to children since they all enjoy playing computer games.
- As a fun and educational after-school activity, parents can use this game when their children are bored.
- Through the app's Connect With Friends feature, kids can work together to solve puzzles in a structured manner. This is a great way to promote teamwork, which is a critical soft skill in the real world.
- The puzzles have been designed so that they take components from the real world and present them in a fun way. As an example, the tidy your room computational challenge teaches them sequential, systematic thinking.
- Incentives like coins and rank encourage children to focus more on the game and therefore learn more effectively.
- The games are customized according to the likes and dislikes of the children in order to keep their interest.
- Fun sketches after the game make learning about the topic more enjoyable.

B. Disadvantages

- Technology's negative effects might inhibit the use of the app for learning purposes.

- Since this solution is heavily reliant on smartphones and working internet connections, kids need to be familiar with online spaces and technically proficient. Parental control is much needed to get the most use out of the app.
- The inclusion of coins and ranks can sometimes have a negative impact, like making the children too competitive instead of encouraging them to work as a team.

C. Requirements met

The main requirement is to design a solution that helps kids learn computational thinking through games. Since this game utilized real-world components it would help kids in learning CT concepts and applying them in daily life. There are many educational games available that aim to improve CT in students like CodeSpell (Esper et al., 2014) but they help kids gain knowledge of theoretical concepts. These games are more focused on teaching CT through text-based programming. However, my design solutions help kids in developing a computational way of thinking in a real-world setting. The solution satisfies most of the requirements of a socio-technical system. As a result of the feedback from the second design session, the kids wanted to stay in touch with each other while solving the challenge. The prototype does not have the capability for kids to be connected via a phone call. It is currently possible for them to send one another messages. However, I would like to reserve this functionality for the future.

B. Design Your Own Board Game, an unplugged method of learning CT

Overview

This design solution is specifically designed for students who don't have technology access and for parents who wish to cut down the computer time for their kids. This solution is a fun learning experience without technology. Kids do not need to use technology every time they need to learn CT. I call this method Unplugged CT because it enables kids to learn concepts without using a computer. Kids can play this game with their own imaginations and a variety of craft materials. During these activities, CT skills, attitudes, and approaches are naturally discussed and remembered. Taking inspiration from creating your own puzzles, this particular idea is about giving the kids the supplies they need to design their own games. Children can learn how to design, work in teams, and think critically by creating a board game. The game begins with the first dice roll. Depending on the number on the dice, that player must provide one idea for the game. Every time a dice is thrown, each player must build upon the existing idea. By using the art supplies, the children can explain their ideas in a more visual way. In the end, they could make their prototype out of the art materials or even digitize the design using online tools.

Illustrations

As part of the first session of design, one of the kids shared an experience from one of their classes where they had to create their own puzzle and the teacher rated the activity as she solved the puzzle. Inspired, I devised a solution that would give kids the opportunity to learn and play without using any technology.

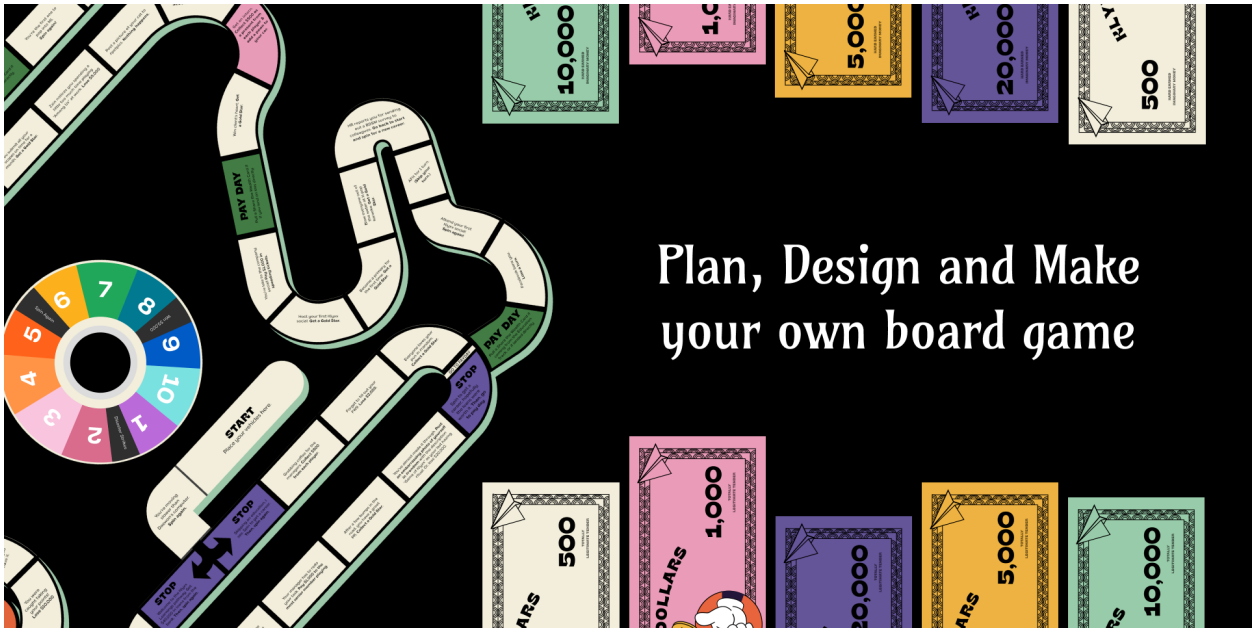
The following was the initial idea of the game developed after the first design session:

Materials

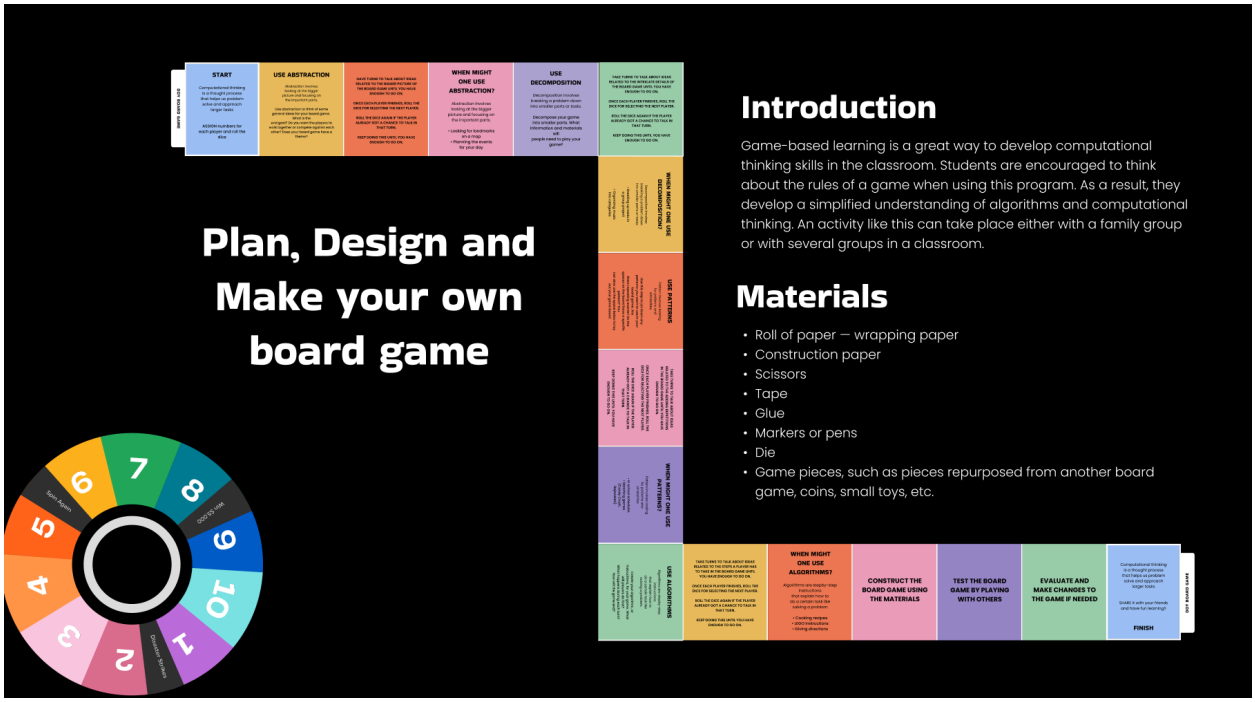
- Roll of paper — wrapping paper
- Construction paper
- Scissors
- Tape
- Glue
- Markers or pens
- Die
- Game pieces, such as pieces repurposed from another board game, coins, small toys, etc.

Directions

This game is best played in teams of at least six players. The game begins with the first dice roll. Depending on the number on the dice, that player must provide one idea for the game. Every time a dice is thrown, each player must build upon the existing idea. By using the art supplies, the children can explain their ideas in a more visual way. In the end, they could make their prototype out of the art materials, or even digitize the design using online tools. In my opinion, using this technique, you can create a fun "Make your own board game" activity.



After the second design session, the kids provided feedback on the initial idea. They wanted the DYO (Design Your Own) board game to be played like a board game with their friends so that they have a set of instructions or a guide that they can utilize to have structured conversations and make their own games. They wanted the board game to be made colorful so that they are appealing. The following prototype was designed after the second design session.

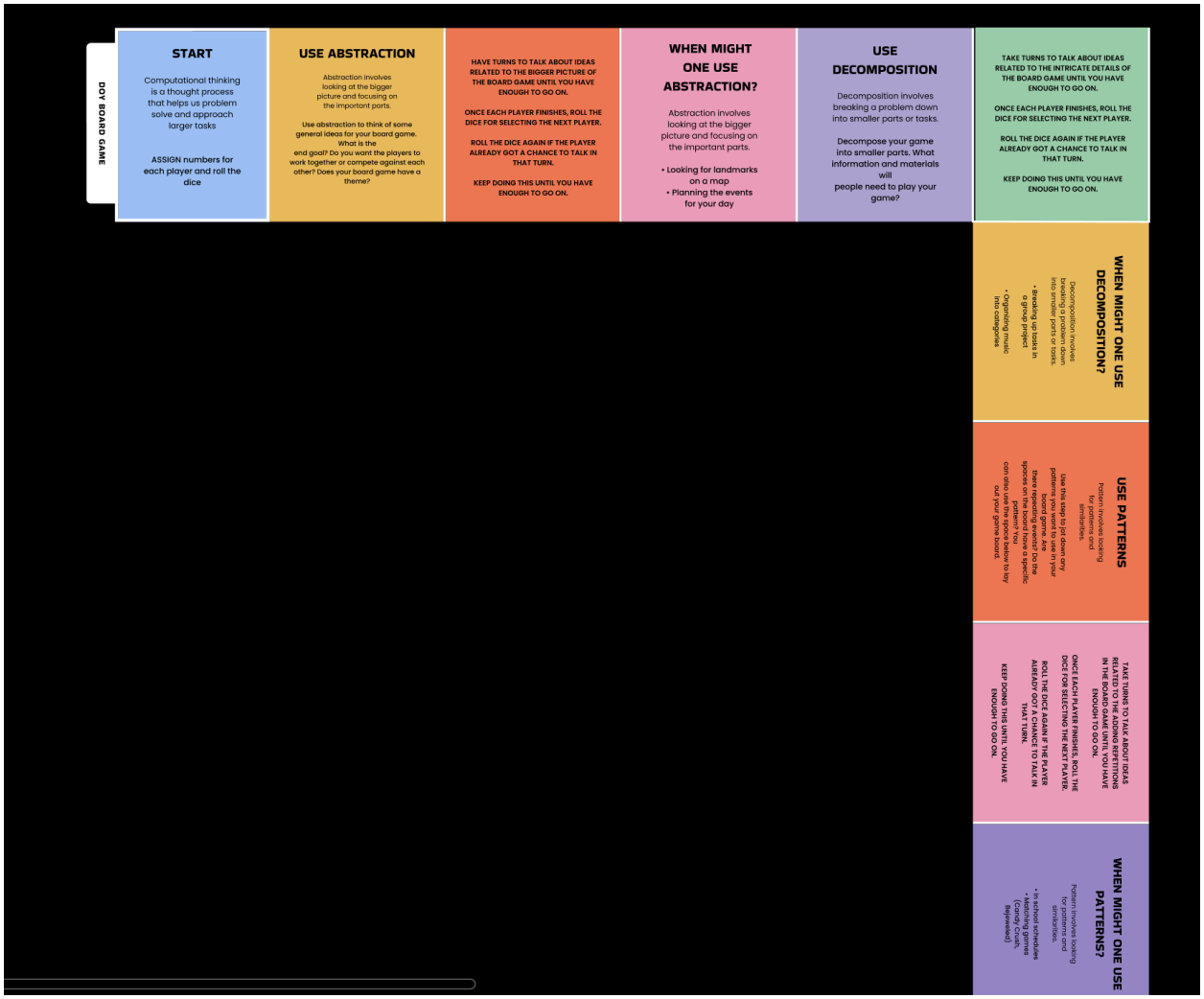


Introduction

Game-based learning is a great way to develop computational thinking skills in the classroom. Students are encouraged to think about the rules of a game when using this program. As a result, they develop a simplified understanding of algorithms and computational thinking. An activity like this can take place either with a family group or with several groups in a classroom.

Materials

- Roll of paper — wrapping paper
- Construction paper
- Scissors
- Tape
- Glue
- Markers or pens
- Die
- Game pieces, such as pieces repurposed from another board game, coins, small toys, etc.



Zoomed version of the board game

Zoomed version of the board game

Assessment

A. Advantages

- The kids are asked to come up with rules for their own game as a result of which they develop an understanding of algorithms and CT concepts.
- By taking turns discussing ideas, this game would help kids develop strong communication and collaboration skills.
- They also learn to listen to others and respect other players.
- During each step in the board game, they are introduced to a CT concept, then asked to design their own game based on that concept, and then, in the next step, they learn about how it would be applied in real life. The following images show an example of “abstraction”, a CT concept from the board game.

- The board game uses straightforward language, so kids can easily understand the concepts.

B. Disadvantages

- My guess is that this game will be viewed as a one-time activity since I am not certain they will continue to play it once they have created their own game. I feel that there is no feature that has the potential to hold their attention.
- The board game can be cumbersome to carry around on the go.

C. Requirements Met

The main requirement is to design a solution that helps kids learn computational thinking through games. I believe the game will effectively teach CT since it relies on a task-based approach. Adding examples after they have completed each step further strengthens the learning process. I intend to evaluate the game using the User Engagement Scale (UES) as per (Brien et al., 2018) and enhance the game so that kids are able to get as much out of it as possible.

Requirement Changes

Anything designed for kids should help them to learn and apply concepts in computational thinking to their daily lives, not just memorize them for a test. I want the design to:

- Stress the relevance of computational thinking to real-life situations
- Be user-friendly for youth.
- Encourage youth to view computational thinking as more than just programming.
- Versatile enough to be used in different areas, including, but not limited to, classrooms, after-school activities, and homes.
- Be easily comprehendible by elementary school students through the use of simple language, colorful interfaces, etc.
- Learn CT concepts in a collaborative environment
- Learning activities must be fun and engaging for the kids.
- Be universal and accessible to kids coming from different backgrounds.

The above list represents the requirements and usability criteria established during the design process. The criteria that were determined in Part I of the project remained. The criteria were gathered from user interviews conducted in the first half of the semester. This set of criteria was validated during the design sessions. I decided to add some more criteria to the list, which are highlighted in the list above. Participatory design sessions proved vital in making these requirements a reality. Children themselves suggested making puzzle-solving activities more collaborative since the more problems they solve together, the more fun it becomes. When the adopted learning methods include games, the kids seem to naturally be interested in using them. As a result, it was necessary for the design solutions to be fun and engaging. The last requirement was not explicitly stated, but since kids come from a variety of backgrounds and because there was little

technological access and the negative impact of technology, I decided to provide a more traditional game-based solution that the kids loved.

References

- Hartigan, M. (2013, August 27). 10 everyday objects that can be programmed to run code. Retrieved from https://edtechbooks.org/-cK
- Wing, J.M. (2011), Research Notebook: Computational thinking -what and why? The Link Magazine, 20-23. https://www.cs.cmu.edu/link/research-notebook-computational-thinking-what-and-why
- The impact of technology on children - cerritos.edu. (n.d.). Retrieved May 12, 2022, from https://www.cerritos.edu/hr/_includes/docs/August_2021_The_Impact_of_Technology_on_Children_ua.pdf
- People in low-income households have less access to internet ... - aspe. (n.d.). Retrieved May 13, 2022, from https://aspe.hhs.gov/sites/default/files/private/pdf/263601/internet-access-among-low-income-2019.pdf
- *Game-based learning - american library association*. (n.d.). Retrieved May 13, 2022, from https://acrl.ala.org/IS/wp-content/uploads/2014/05/spring2015.pdf
- Fails, J. A., Guha, M. L., & Druin, A. (2013). Methods and techniques for involving children in the design of new technology for children. Foundations and Trends in Human–Computer Interaction,6(2), 85-166. Available at http://www.cs.umd.edu/hcil/trs/2013-23/2013-23.pdf

- Subramaniam, M. (2016). Designing the library of the future for and withteens: Librarians as the‘Connector’ in Connected Learning. Journal of Research on Libraries and Young Adults,7(2),1-18. Available at: http://www.yalsa.ala.org/jrlya/2016/06/designing-the-library-ofthe-future-for-and-with-teens-librarians-as-the-connector-in-connected-learning/
- Knudtzon, K., Druin, A., Kaplan, N., Summers, K., Chisik, Y., Kulkarni, R., et al. (2003). Starting an intergenerational technology design team: A case study. In Proceedings, Interaction Design and Children 2003, ACM Press.
- O’Brien HL, Cairns P, Hall M (2018) A practical approach to measuring user engagement with the refined user engagement scale (ues) and new ues short form. Int J Human-Comput Stud 112:28–39
- Esper, S., Foster, S. R., Griswold, W. G., Herrera, C., & Snyder, W. (2014). CodeSpells: Bridging educational language features with industry-standard languages. In Proceedings of the 14th Koli calling international conference on computing education research (pp. 05–14). New York: ACM. https://doi.org/10.1145/2674683.2674684