DOI: 10.34641/ctestem.2022.452

Precoding skills - Teaching computational thinking to preschoolers in Singapore using unplugged activities

Vidhi Singhal (Founder, Kinder Koder), Singapore, vidhi@kinderkoder.com

ABSTRACT

Computational Thinking (CT) by now is widely recognized as an important skill in K-12 education. Research suggests that, during children's early formative years, certain types of experiences, including exploration, exposure to basic skills, and practice with rich communication, among others, are critical to support typical development (Ramey and Ramey 1999). Exposure to these experiences cultivates school readiness, which in turn supports children's later achievement. The same holds true for CT. Exposing children to different kinds of CT-oriented problem-solving ideas and strategies, paired with thoughtful guidance, will allow preschool-aged children to practice CT over a wide variety of contexts. Kinder Koder was started in 2020 with the aim of teaching CT to preschoolers through unplugged games and activities. This paper shares how CT is being taught in Kinder Koder's enrichment classes in Singapore by using a framework more suited for early childhood education. This will help preschool teacher's integrate teaching CT in their day to day classrooms.

KEYWORDS

computational thinking, unplugged, kindergarten, play based learning

1. INTRODUCTION

Coding is, "telling the computer what to do and how to do it." Before you can think about coding, you need to work out exactly what you want to tell the computer to do. Thinking through problems this way is Computational Thinking. It allows us to take complex problems, understand what the problem is, and develop solutions. So CT is the step that comes before you actually do coding and hence we call it precoding skills to make it simpler for parents to understand. Figure 1 shows this relationship between CT and coding.

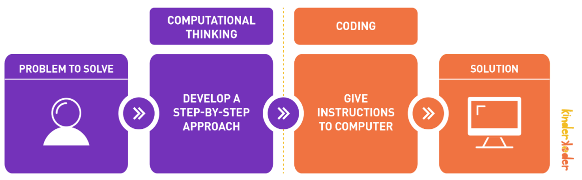

Figure 1. CT vs Coding

2. FRAMEWORK

Computational thinking involves a broad set of approaches and skills. As per ISTE's operational definition Computational thinking (CT) is a problem-solving process that includes (but is not limited to) the following characteristics:

•Formulating problems in a way that enables us to use a computer and other tools to help solve them.

•Logically organizing and analyzing data

•Representing data through abstractions such as models and simulations

•Automating solutions through algorithmic thinking (a series of ordered steps)

•Identifying, analyzing, and implementing possible solutions with the goal of achieving the most efficient and effective combination of steps and resources

•Generalizing and transferring this problem solving process to a wide variety of problems

Abstraction as a concept is difficult for preschoolers. Computational thinking is itself a very abstract idea for kids and it can be made less abstract by teaching them unplugged, out of the screen, into the physical world. Giving children something to hold with their hands, like blocks or cards. This stimulates active engagement and allows children to experience the material. Instead of speculating about what would go wrong, they can try it themselves and experience the consequences of certain actions. To make learning for age appropriate we've come up with the following framework (Figure 2) for early childhood i.e. 3-6 years.

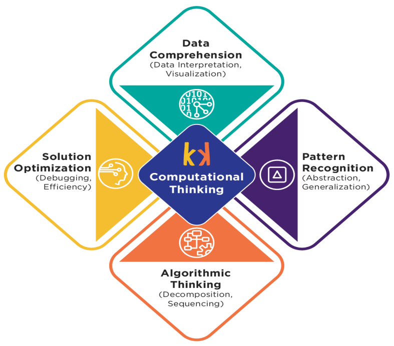
Data Comprehension (Data Interpretation, Visualization)
Solution Optimization (Debugging, Efficiency)
kk Computational Thinking
Pattern Recognition (Abstraction, Generalization)
Algorithmic Thinking (Decomposition, Sequencing)

Figure 2. CT Framework

3. OUR APPROACH

For each of the skills we have identified age appropriate learning outcomes and we develop unplugged activities to support that learning as shown in Table 1.

Table 1. Example of age appropriate learning outcomes

CT Skill	Age (3-6 years)
Data comprehension	• Understanding directions (left, right, forward) • Sorting of numbers (1-10) • 1:1 correspondance

Pattern recognition	• Grouping of similar objects basis shape, size, color • Being able to identify, extend and create simple patterns
Algorithmic thinking	• Follow instructions to complete simple tasks (drawing, coloring) • Understanding what algorithms are • Focus on sequence of steps
Solution Optimization	• Understanding a problem can have multiple solutions and some better than the other • Understanding what is an error

4. IMPLEMENTATION

4.1 Data comprehension

Logically organizing, representing and interpreting data. Understand instructions and numbers. E.g. Figure 3:

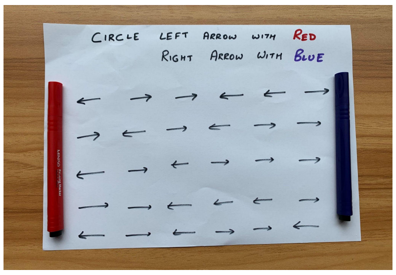

Figure 3. Sample activity for data comprehension

4.2 Pattern Recognition

Pattern recognition is the process of identifying, defining, extending, and creating patterns. This forms the foundation for higher order thinking skills such as abstraction (hiding the complexities of one pattern from another) and generalization (spotting things that are common between patterns). E.g. Figure 4:

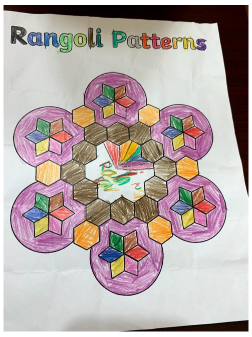

Figure 4. Sample activity for pattern recognition

4.3 Algorithmic Thinking

To break down a problem into smaller sub-problems. This is known as Decomposition. Then finding a step by step solution to a sub problem.E.g. Figure 5:

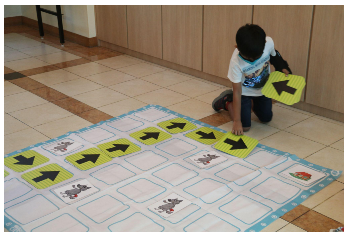

Figure 5. Sample activity for algorithmic thinking

4.4 Solution Optimization

To be able to identify gaps in solutions, evaluate, resolve inconsistencies, optimize and come up with an efficient solution. E.g. Figure 5:

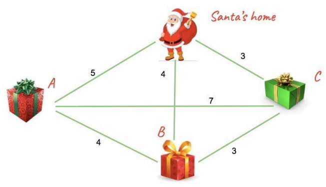

Figure 5. Sample activity for solution optimisation

5. CONCLUSION

Over 300 students so far have benefited from this approach. While we lack quantitative data to measure their learning outcomes, the qualitative responses have been very promising. Both parents and kids have found this way of teaching very engaging and have repeatedly asked for more content. Feedback from a parent below:

"Amazing, Amazing, Amazing. Ma'am, your ideas are so unique. They are just lovely. Thank you for sharing them." Parent of a preschooler.

This unplugged approach introduces CT concepts to kids in a non daunting way making them more prepared for future learning. As a next step we would look for ways to measure the learning outcomes.

6. REFERENCES

CAS-UK.Computing at School Working Group http://www.computingatschool.org.uk

Computer Science Unplugged https://www.csunplugged.org/en/

ISTE and CSTA(2007). Operational definition of computational thinking for K-12 education.

Ramey, Craig T., and Sharon L. Ramey (2004). "Early Learning and School Readiness: Can Early Intervention Make a Difference?" Merrill-Palmer Quarterly 50, no. 4: 471–91

Wing,J.M.(2006).Computational thinking Communications of the ACM, 49(3), 33-35