Computational Thinking

UniSA | Education Futures

SUMMARY

- Computational thinking is a thought process which focuses on the use of algorithms (a set of steps) to explore authentic problems
- It can be conceptualised as not only a problem-solving process but a way to learn
- Computational thinking connects to 21st century competencies such as creativity, critical thinking and problem solving (Paniagua & Istance, 2018)
- The process requires and develops important executive functions such as working memory, response inhibition and planning (Arfé et al., 2019)
- It is often associated with digital technology, but it can be applied to many logical problem-solving situations especially those relate to STEM
- There are clear links between design thinking and computational thinking

Computational Thinking Elements

DECOMPOSITION

Imagining changes in the orientation of shapes and objects

PATTERN RECOGNITION

Sorting the pieces into groups that share features

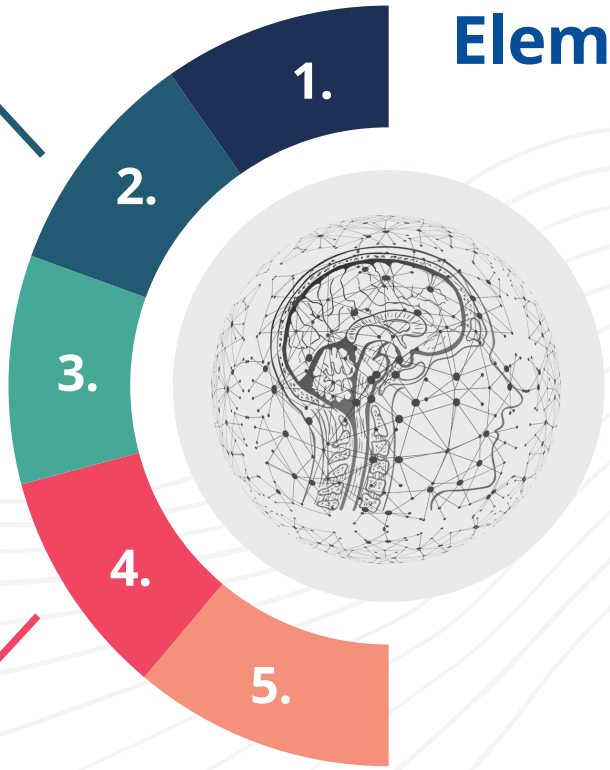
ABSTRACTION

Removing uninportant elements so that you can focus on the key problem

ALGORITHM DESIGN

Imagining changes in the orientation of shapes and objects

DEBUGGING



*Multiple interations may be needed

UniSA Education Futures
edc-enquires@unisa.edu.au

IMPLICATIONS FOR PRACTICE

In order for students to identify and use computational thinking it is important to establish key terms and use them across multiple subject areas

Computational thinking can help students work through problems in many diverse subjects. Using consistent language will help to not only avoid reteaching parts of the process, it will also build students' ability to recognise the importance of thinking carefully through a problem.

The terms used in the infographic are common to many models of computational thinking and are recommended for most students but language you may use with younger students might include:
Break apart, Pieces = Decomposition

- Patterns, matching = Pattern Recognition
- Thoughts, ideas = Abstraction
- Instruction, plan = Algorithm Design

Programming is not the only application of computational thinking, but it is a simple way to introduce the concept

Technology can be a platform for problem solving using a computational thinking process. Programming uses coding knowledge to solve problems within clear parameters and therefore can be used to simplify computational thinking concepts which may be overwhelming in a wider context. Through the use of block and text coding, students can be helped to clearly see the decomposition of a problem, the patterns needed to solve that problem as represented by the individual codes and the codes that are irrelevant or could be supplanted by more efficient codes. They then can carefully consider the sequence of their codes before testing them and debugging.

At a younger age programming can be explored using robots with simple commands that reduce the cognitive load of the task. As children progress in their understanding, they can be exposed to block coding which often has multiple levels of complexity designed to suit the development of the student. Block code can help teach many important computer science concepts such as events, sequences, loops, conditionals, parallelism and variables. Text coding however allows more experienced students to start to use these fundamental concepts in more creative and effective ways.

Computational thinking is an effective scaffold for students in problem based learning

Not all patterns will initially be evident, especially when dealing with multivariate real life problems. It is therefore important that students can develop and test models of the phenomena in order to identify the important variables, establish early patterns and use algorithm design to create solutions. Common models used in schools include graphs, diagrams, maps, role plays, mind maps, and flow charts.

Models as abstractions help students to identify how they can use the identified patterns in new situations. For example, the addition of a new predator to an ecosystem can lead to particular patterns of ecological damage that can be applied to other habitats where a predator is introduced. There will be a certain amount of debugging to identify new variables and adjust the model but the identified patterns can help illuminate new solutions.

Models can also act as important conduits of reasoning when working with others on problems. This can aid in the debugging of malfunctioning systems and the joint identification of patterns. These models can also serve the secondary purpose of effective formative or summative assessment.

RELATED OUTREACH PROGRAMS

Computational Thinking through Mathematical Modelling Python course

To learn more, please contact *edc-enquires@unisa.edu.au*

FURTHER READING AND REFERENCES

Arfé, B., Vardanega, T., Montuori, C., & Lavanga, M. (2019). Coding in Primary Grades Boosts Children's Executive Functions. Frontiers in Psychology, 10, 2713-2713. https://doi.org/10.3389/fpsyg.2019.02713

Grover, S., & Pea, R. (2013). Computational Thinking in K–12:A Review of the State of the Field. Educational Researcher, 42(1), 38-43. https://doi.org/10.3102/0013189x12463051
Kwon, K., Ottenbreit-Leftwich, A. T., Brush, T. A., Jeon, M., & Yan, G. (2021). Integration of problem-based learning in elementary computer science education: effects on computational thinking and attitudes. Educational Technology Research and Development. https://doi.org/10.1007/s11423-021-10034-3

Paniagua, A., & Istance, D. (2018). Teachers as Designers of Learning Environments: The Importance of Innovative Pedagogies. Educational Research and Innovation. 1-204. https://doi.org/10.1787/g272b194c2-en

Wing, J. M. (2006). Computational thinking. Communications of the ACM, 49(3), 33–35. https://doi.org/10.1145/1118178.1118215