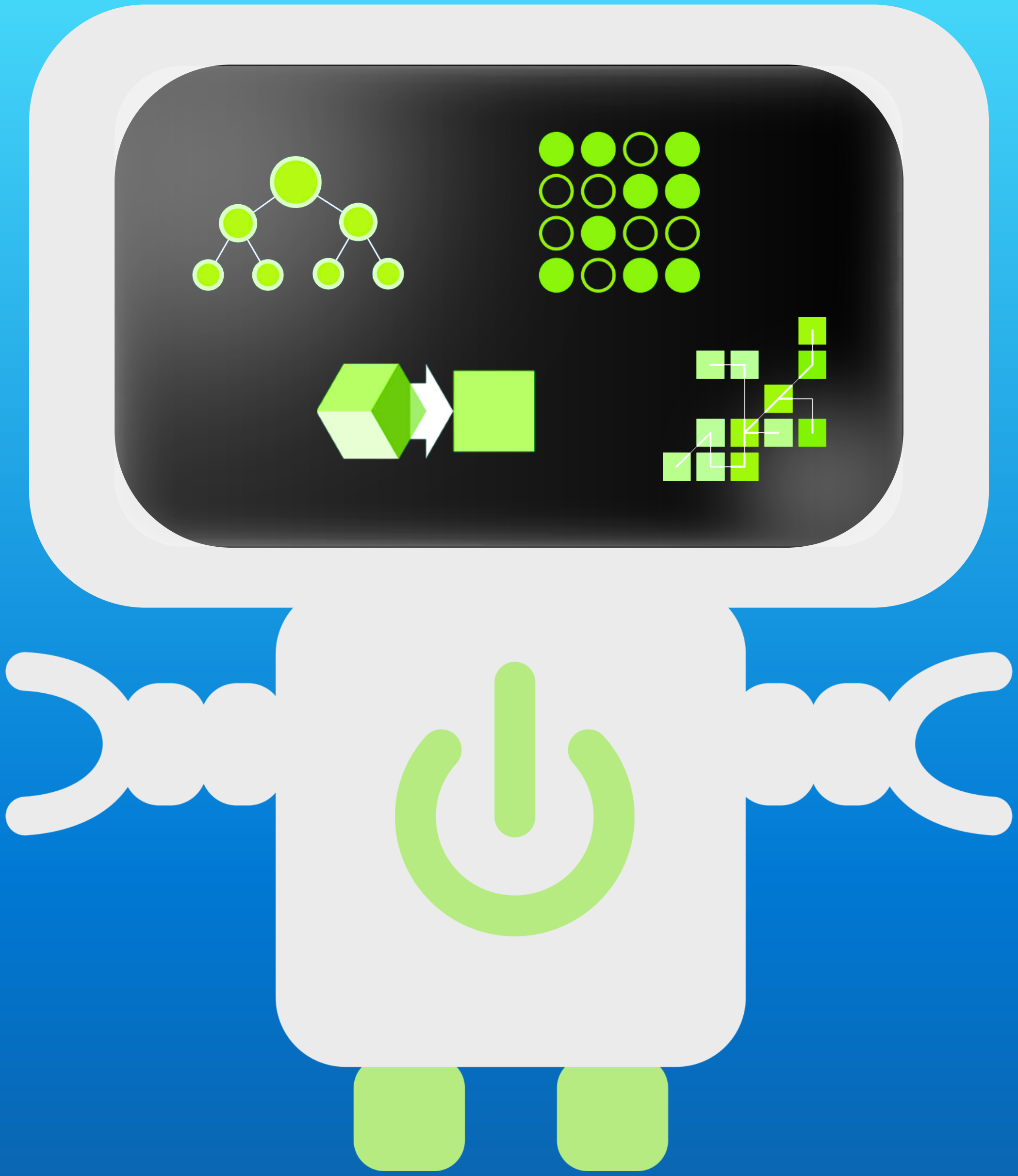

Computational Thinking

Microsoft

What is computational thinking?

Computational thinking (CT) is a problem-solving process that involves breaking down complex problems into smaller, more manageable parts. It involves using a set of problem-solving methods that involve expressing problems and their solutions in ways that a computer could also execute. CT is a way of thinking that draws on concepts fundamental to computer science, such as abstraction, data representation, and logically organizing data. It involves automation of processes but also using computing to explore, analyze, and understand processes (natural and artificial). CT is a set of cognitive skills and problem-solving processes that include (but are not limited to) the following characteristics: **decomposition**, **pattern recognition/data representation**, **abstraction**, and **algorithms**.

The universal components of computational thinking

Decomposition

Decomposition is the process of **breaking down a complex problem** into smaller, more manageable parts. For example, when planning a birthday party, one can break down the problem into smaller parts such as invitations, decorations, food and drinks, and entertainment.

Pattern recognition

Pattern recognition involves **looking for repeating or predictable behaviors or structures** within a problem or system. For example, recognizing patterns in inputs when planning a birthday party can help make decisions about who to invite and what decorations to buy.

Abstraction

Abstraction involves **focusing on the essential details** while ignoring irrelevant details. For example, when planning a birthday party, abstraction can be used to focus on essential details such as the guest list.

Algorithms

Algorithms are **step-by-step procedures for solving problems**. They are used to automate processes and solve complicated problems of scale. For example, when planning a birthday party, algorithms can be used to create specific rules for solving problems such as how to create and send invitations.

Why computational thinking is important

Computational thinking is important because it enables **real-world problem solving**. It teaches students how to take large problems and break them into simpler steps that can be tackled individually. CT helps students navigate complex information and think in a way that compliments technological processes. It encourages students to reflect clearly on the problem they are solving and intentionally develop solutions. CT teaches tenacity, tolerance for ambiguity and complexity, teamwork, and communication skills.

Computational thinking is a valuable skill for everyone to practice

Computational thinking is not just for computer scientists but for students in all fields. It encourages us to reflect clearly on the problem we are solving and intentionally develop solutions. CT teaches tenacity, tolerance for ambiguity and complexity, teamwork, communication skills, and how to navigate complex information. Whether you are making a pitch or looking to build an organization, computational thinking through abstraction is a great capacity to practice.

The difference between computational thinking and coding

Computational thinking (CT) and coding are two different skill sets. CT is a way of thinking that involves breaking down complex problems into smaller, more manageable parts. It involves using a set of problem-solving methods that involve expressing problems and their solutions in ways that a computer could also execute.

CT is a way of thinking that draws on concepts fundamental to computer science, such as abstraction, data representation, and logically organizing data. It involves automation of processes but also using computing to explore, analyze, and understand processes (natural and artificial). CT is a set of cognitive skills and problem-solving processes. Coding, on the other hand, is the process of writing instructions for a computer to execute. It involves using programming languages to create software applications, websites, and other digital products. Coding is like writing in a language that computers can understand. While CT is a way of thinking about problems, coding is the process of creating solutions to those problems.

How does computational thinking differ from design thinking?

Design thinking and computational thinking are two different problem-solving approaches. Design thinking is a user-centered approach to problem-solving that involves understanding the intent or problem before looking at any solution. The design thinking process contains the following steps: empathize, define, ideate, prototype, ideate, and test (plus improve). On the other hand, computational thinking is a problem-solving process that involves breaking down complex problems into smaller, more manageable parts. It involves using a set of problem-solving methods that involve expressing problems and their solutions in ways that a computer could also execute. CT is a way of thinking that draws on concepts fundamental to computer science, such as abstraction, data representation, and logically organizing data.