Module IV - COMPUTATIONAL THINKING

Learning Objective	Learning Outcome
• To help the students to develop their ideas into a robust step-by-step solution using adequate technology • To introduce them to the concept of Computational Thinking - its key techniques/approaches • To give them an understanding of algorithms, flowcharts, sensors, development boards, programming through hands-on/DIY activities and exposure to various resources	• Would have understood the concept of Computational Thinking and its application for problem solving • Would know how to develop their ideas into flowcharts, algorithms and convert them into programming language • Would have undergone real-time hands-on projects using sensors, coding while understanding their multiple applications

About Computational Thinking

Computers can be used to help us solve problems. However, before a problem can be tackled, the problem and the ways in which it could be solved needs to be understood. Computational thinking allows us to do this. It allows us to take a complex problem, understand what the problem is and develop possible solutions. These solutions can then be presented in a way that a computer, a human, or both, can understand.

Thinking computationally is not programming. It is not even thinking like a computer! Simply put, programming tells a computer what to do and how to do it. Computational thinking enables us to work out exactly what to tell the computer to do.

For example, if you agree to meet your friends somewhere you have never been before, you would probably plan your route before you step out of your house. You might consider the routes available and which route is 'best' - this might be the route that is the shortest, the quickest, or the one which goes past your favourite shop on the way. You'd then follow the step-by-step directions to get there. In this case, the planning part is like computational thinking, and following the directions is like programming!

Being able to turn a complex problem into one we can easily understand is a skill that is extremely useful.
In fact, it's a skill you already have and probably use every day.

Let's take a look at two more examples:

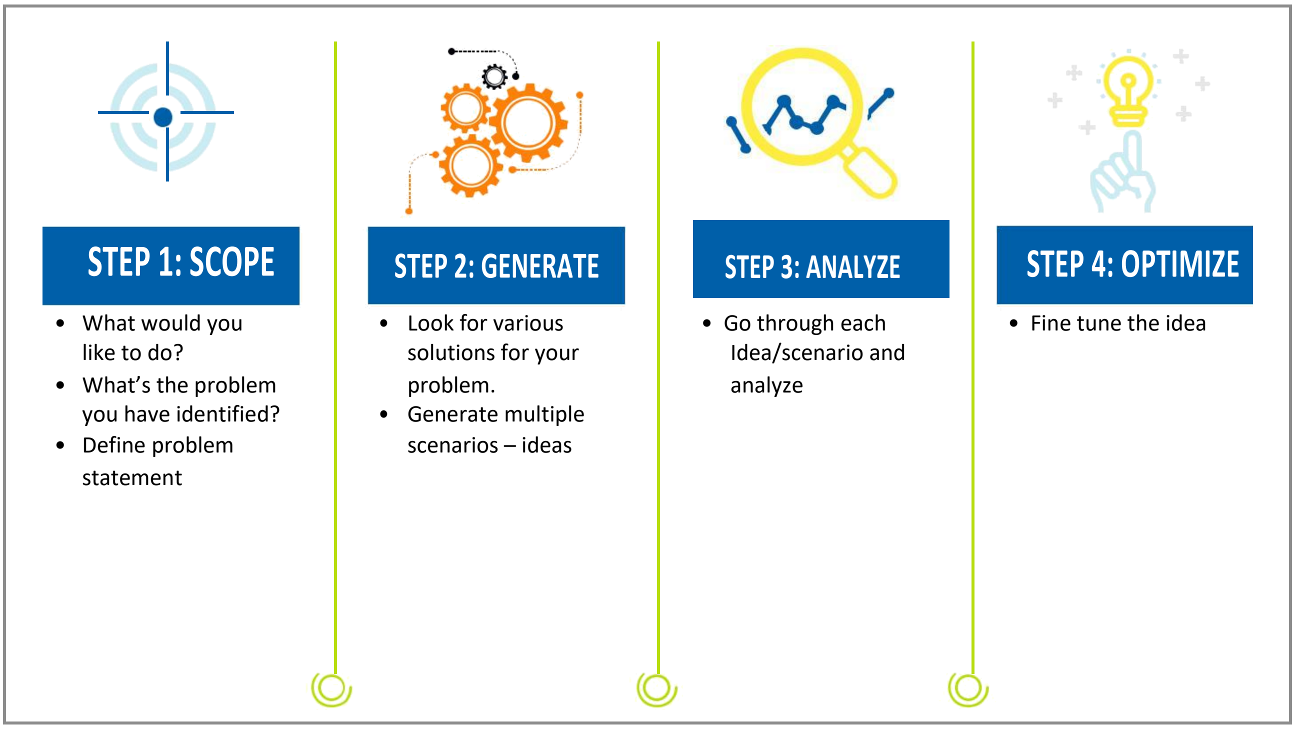
Example 1	Example 2
Let's say you need to decide what to do with your group of friends. If all of you like different things, you would need to decide: • What you could do • Where you could go • Who wants to do what • What have you previously done that has been a success in the past • How much money you have and the cost of any of the options • How much time you have From this information, you and your friends could decide more easily where to go and what to do – in order to keep most of your friends happy. You could also use a computer to help you to collect and analyze the data to devise the best solution to the problem, both now and even in a future scenario.	Another example might occur when playing a videogame. Depending on the game, in order to complete a level you would need to know: • What items you need to collect, how you can collect them, and how much time do you have to collect them • What is the best route to complete the level in the quickest time possible • What kinds of enemies are there and their weak points From these details you can work out a strategy for completing the level in the most efficient way.

Both of the instances above show how computational thinking has been used to solve a complex problem.

This brings us to the four key techniques of computational thinking:

1. **Decomposition** - breaking down a complex problem or system into smaller, more manageable parts (e.g. where to go, how to complete the level)
2. **Abstraction** - focusing on the important information only, ignoring irrelevant detail (e.g. weather, location of exit)
3. **Pattern recognition** – looking for similarities among and within problems (e.g. knowledge of previous similar problems used)
4. **Algorithms** - developing a step-by-step solution to the problem, or the rules to follow to solve the problem (e.g. to work out a step-by-step plan of action)

A closer look will show that you may have worked through some parts of these techniques in Ideation and Design Thinking modules. Here is a quick snapshot:



So, in this module we will concentrate on

1. **Algorithms and flowcharts** - developing a step-by-step solution to the problem, or the rules to follow to solve the problem
2. Getting introduced to **hardware/software** - understand their usage and learn how to put the idea generated through Ideation and detailed through Design Thinking into action by taking it to a computer and learning to give it a real-time shape
3. Learning the basic concept of **Programming**

Understanding Algorithms and Flowcharts

An algorithm is a plan, a set of step-by-step instructions to solve a problem.

If you can tie shoelaces, make a cup of tea, get dressed or prepare a meal, then you already know how to follow an algorithm. In an algorithm, each instruction is identified and the order in which they should be carried out is planned. Algorithms are often used as a starting point for creating a computer program, and they are sometimes written as a flowchart.

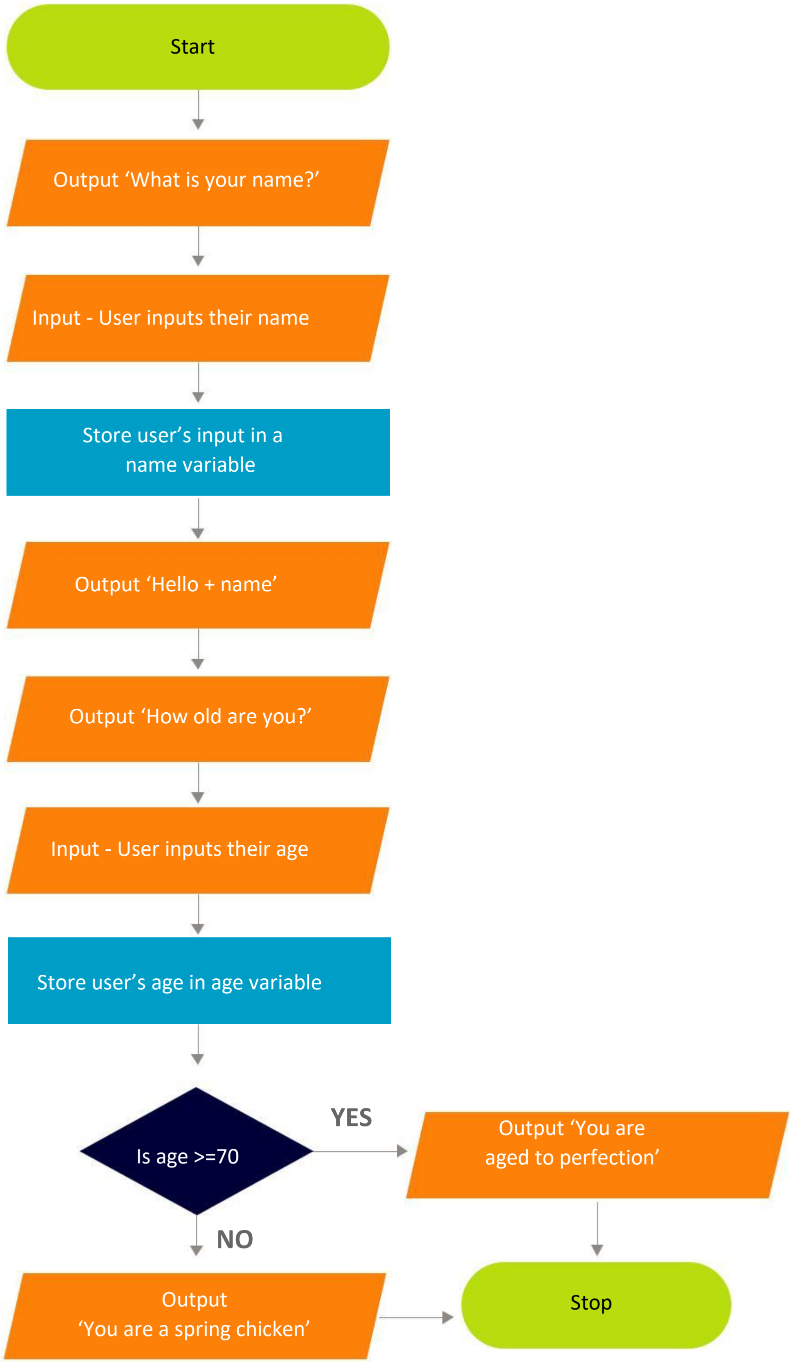
If we want to tell a computer to do something, we have to write a computer program that will give the computer step-by-step instructions on what we want it to do, and how we want it to do it. This step-by-step program will need planning, and to do this we use an algorithm.

Computers are only as good as the algorithms they are given. If you give a computer a poor algorithm, you will get a poor result. Algorithms are used for many different things including calculations, data processing and automation.

Representing an algorithm: Flowcharts

A flowchart is a diagram that represents a set of instructions. Flowcharts normally use standard symbols to represent the different instructions. There are few rules about the level of detail needed in a flowchart. Sometimes flowcharts are broken down into many steps to provide a lot of detail about exactly what is happening. Sometimes they are simplified, so that a number of steps occur in just one step.

A simple program could be created to ask someone their name and age, and to make a comment based on the program represented as a flowchart would look like this:



Source: http://www.bbc.co.uk/education/guides/zpp49j6/revision/3

Note for the facilitator

- Please ensure all the participants/students solve the exercises given at the end of the presentation. Students can be encouraged to come with problems and solutions for the same via flowcharts and algorithms.

Understanding Sensors and Actuators, and their applications

Here you will learn about different kind of sensors and their applications.

Imagine a human body without its five basic senses. It will not be able to interact with its surroundings, and it will not be able to produce any reaction. Similarly in the world of computers we have sensors – to help know the input/output for any selected process/program.

In the broadest definition, a sensor is an electronic component, module, or subsystem whose purpose is to detect events or changes in its environment, and send this information to other electronics, frequently a computer processor. A sensor is always used with other electronics, whether as simple as an electric light or as complex as a computer.

Sensors come in variety of shapes and sizes, and they sense a very large variety of things, from heartbeats to air pressure, from brightness to heat - there is a sensor to measure almost everything. And if there isn't a sensor specific to your requirement, you can mix and match a variety of sensors to accomplish what you need.

Actuators are basically things that produce an action. They are like the hands and legs of a person, and they come in various forms and shapes. Electromagnets, relays, DC motors, servo motors etc. are a few examples of commonly used actuators.

Note for the facilitator

- Please refer to the Computational Thinking presentation about sensors and actuators provided in the pen drive
- Please ensure all the participants/students solve the exercises given at the end of the presentation. Students can be encouraged to come with problems and solutions for the same via flowcharts and algorithms.

Understanding Circuits

This will help you to understand what are circuits and different kind of circuits.

In electronics, a circuit is a path between two or more points along which an electrical current can be carried.A circuit is a closed loop that electrons can travel in. A source of electricity, such as a battery, provides electrical energy in the circuit. Unless the circuit is complete, that is, making a full circle back to the electrical source, no electrons will move.[3]

[3]http://www.qrg.northwestern.edu/projects/vss/docs/power/2-whats-a-circuit.ht

Resources

To get some more inspiration, you may refer to the following:

Brief Description	Resource URL
Computational Thinking and Thinking About Computing	https://www.cs.cmu.edu/afs/cs/usr/wing/www/talks/ct-and-tc-long.pdf
More about Computational Thinking	Computational Thinking (In depth Explanation) https://www.youtube.com/watch?v=C2Pq4N-iE4I Computational Thinking from code.org https://www.youtube.com/watch?v=injJWiSA0pw Abstraction Computational Thinking https://www.youtube.com/watch?v=jV-7Hy-PF2Q
Know more about Single Board microcontrollers	https://en.wikipedia.org/wiki/Single-board_microcontroller
All about circuits	http://www.build-electronic-circuits.com/free-electronic-circuits/ http://www.dummies.com/programming/electronics/how-to-build-a-simple-electronic-circuit/ Basic Electricity :Current ,Resistance and Ohms law https://www.youtube.com/watch?v=NfcgA1axPLo Basic Tutorial on Introduction to Circuits by Khan Academy (Tutorial) https://www.youtube.com/watch?v=3o8_EARoMtg There is a course on Circuit Physics by Khan Academy. It's Very Good for Beginners. (Playlist for Circuit Physics Course) https://www.khanacademy.org/science/physics/circuits-topic How to use Breadboard https://www.youtube.com/watch?v=6WReFkfrUIk Paper Circuit (How to do it) https://www.youtube.com/watch?v=BwKQ9Idq9FM Build your own USB charger - a step by step guide http://www.build-electronic-circuits.com/wp-content/uploads/2017/lm/How-To-Build-A-Portable-USB-Charger.pdf
All about sensors	https://www.electrical4u.com/sensor-types-of-sensor/ https://www.engineersgarage.com/articles/sensors https://www.youtube.com/watch?v=q1xNuU7gaAQ
What's an Algorithm?	https://www.youtube.com/watch?v=6hfOvs8pY1k https://www.youtube.com/watch?v=Da5TOXCwLSg
Algorithm in Pseudo Code, Flow Diagrams & Programming	https://www.youtube.com/watch?v=HhBrkpTqzqg https://www.youtube.com/watch?v=XDWw4Ltfy5w
An Overview of Flow Chart	https://www.youtube.com/watch?v=uCNliFuKG8I
Real life algorithms – Paper Airplanes	https://www.youtube.com/watch?v=AWqo8Gxtrjs
Electronic Components	Simple Guide to Electronics (Tutorial) https://www.youtube.com/watch?v=6Maq5IyHSuc Basics of Coils (Part 1) (Tutorial) https://www.youtube.com/watch?v=kdrP9WbJlb8 Basics of Coils (Part 2) (Tutorial) https://www.youtube.com/watch?v=XCnI6ZOYKes How Does a Transistor Work? (Tutorial) https://www.youtube.com/watch?v=IcrBqCFLHIY
Making the motors work	D.C Motor Working (Tutorial) https://www.youtube.com/watch?v=LAtPHANEfQo&t=65s Stepper Motor Working (Tutorial) https://www.youtube.com/watch?v=TWMai3oirnM