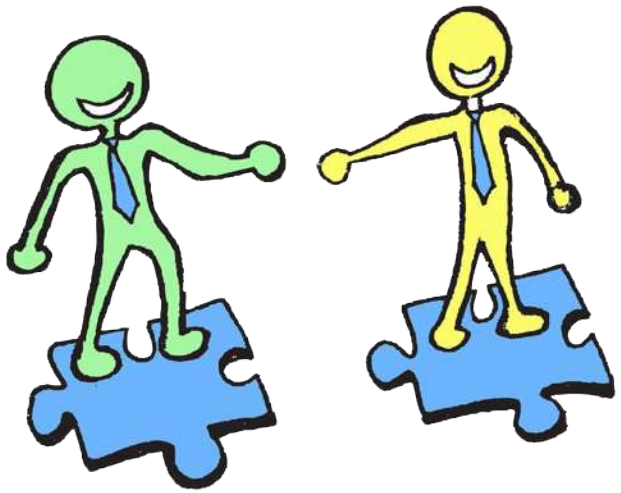
Why Computational Thinking?



Objectives

Student

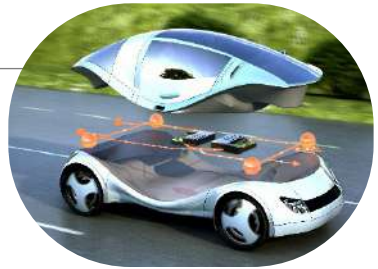
- **Obtained the skills for 21st Century learning**
- **life-long learner**
- **Problem solver**
- **Solution seeker**
- **Digital maker**

What is the Future?

Technology is everywhere

Contact Lens

Medical – Deep Brain Stimulation



Transportation

Future Advancement

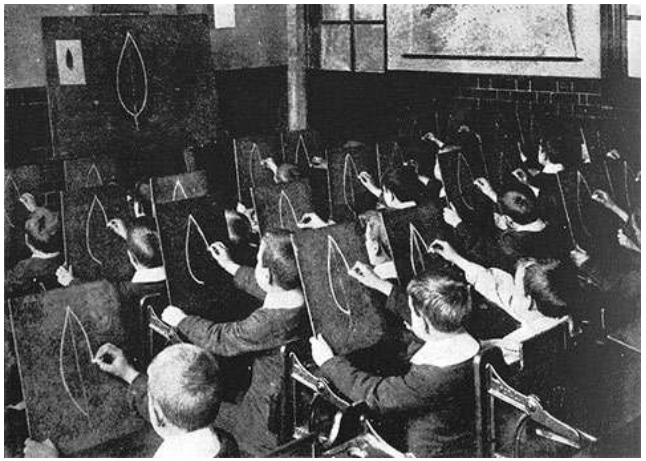
Screenless Display

Holographic Television

Entertainment

NOW

Scenario 1: Current educational system → Prepare for jobs that do not exist yet?



18th Century

Year 2017

NOW

Scenario 2: Job Skill ↓

The Education System is failing to produce highly skilled workers
- 2012

NOW

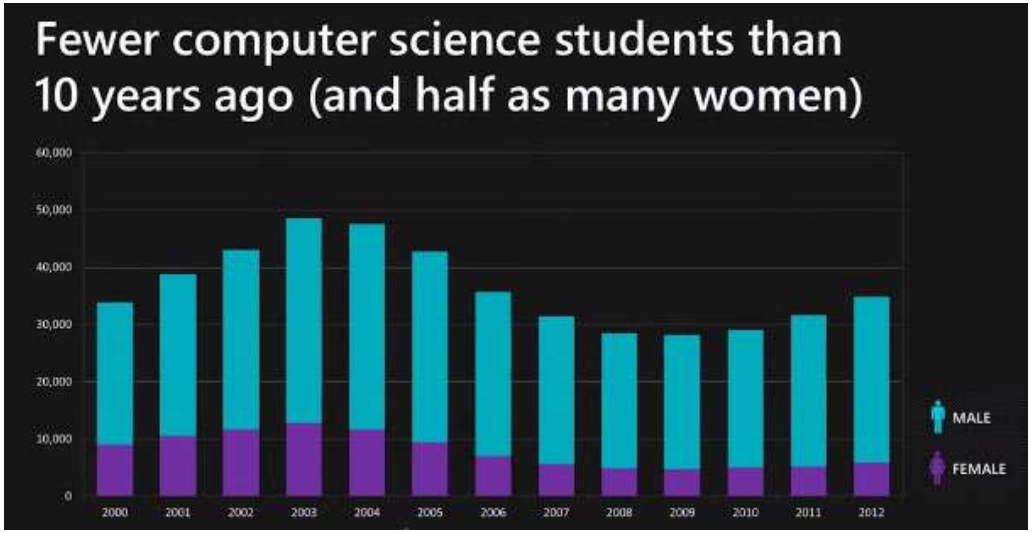
Scenario 3: Not producing independent learners

Let's Stop Spoon-Feeding the Students

By Judit Neurink 16/11/2013

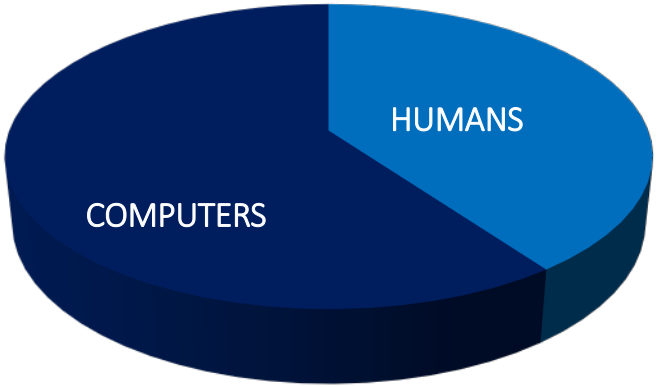
NOW

Scenario 4: Computer Scientist ↓



A Global Problem

By 2030



Computers will displace humans in 60% of current occupations

National Resource Council Study, US

Which country will have the workforce prepared to innovate?

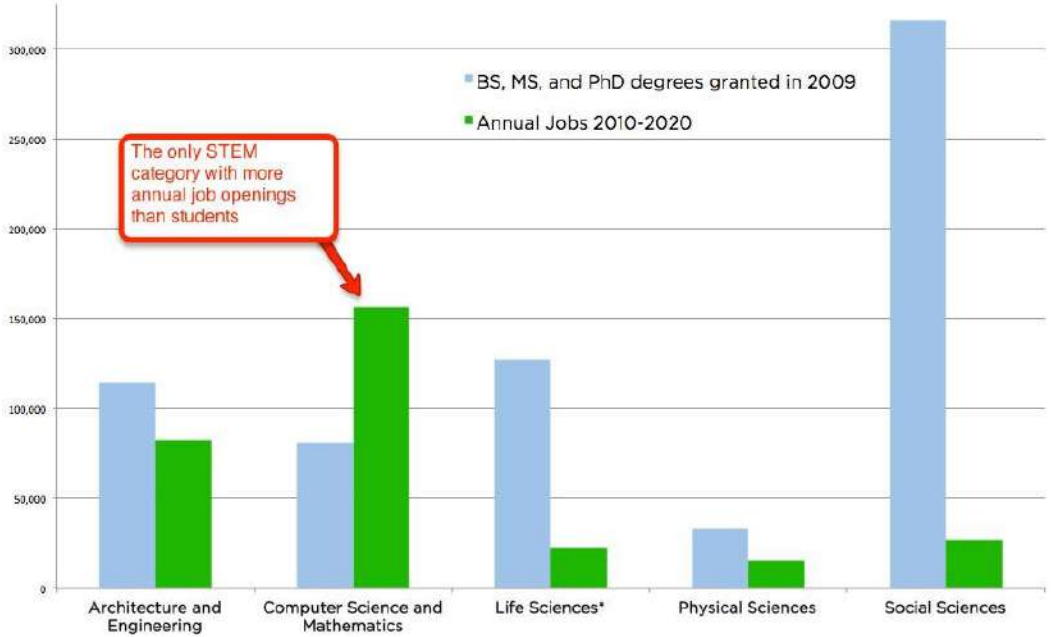
Solution – Bridging the Gap

Computational Thinking

Hadi Partovi:
“Understanding how technology works, how the Internet works, and learning to solve problems with computational thinking, these skills are as important as learning how electricity works, how digestion works, or solving problems using algebra.”

Solution – Bridging the gap

STEM Profession needs CS the most



- The job gap and growth opportunity is in computer science, not in STEM
- Unfilled jobs are in computer science
- The biggest are of economic growth is in computer science
- The foundational field isn't being taught in our schools is computer science.

Introducing Computational Thinking

What is CT?

What is Computational Thinking

Overview

*Computational thinking is the **thought processes** involved in **formulating problems and their solutions** so that the solutions are represented in a form that can be effectively carried out by an **information-processing agent*** (Cuny, Snyder, & Wing, 10).

Informally, computational thinking describes the mental activity in formulating a problem to admit a computational solution. The solution can be carried out by **a human or machine, or more generally, by combinations of humans and machines**.

Computational thinking is the new literacy of the 21st century. – Wing, 2010

https://youtu.be/C2Pq4N-iE4I

The Six Concepts

HTTPS://WWW.YOUTUBE.COM/WATCH?V=QBNTZCJ0UGI&T=6S

HTTPS://WWW.YOUTUBE.COM/WATCH?V=SVVB5RQFYXK

What is Computational Thinking

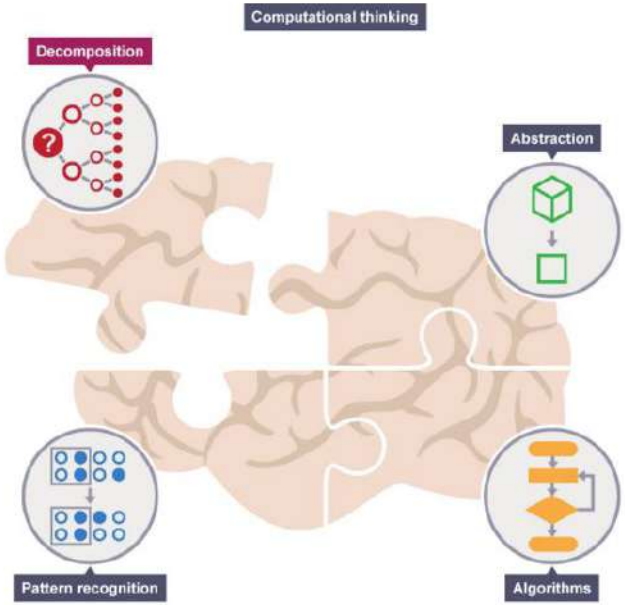
Concepts

- Decompose
- Patterns
- Abstractions
- Algorithms
- Logical Reasoning
- Evaluation

What is Computational Thinking

Decompose

- **Large task break into minute details**.
- Allow us to clearly explain a process to another person or to a computer, or even to just write notes for ourselves.



What is Computational Thinking

Decompose

Example 1: Brushing our teeth

- Which toothbrush to use
- How long to brush for
- How hard to press on our teeth
- What toothpaste to use

What is Computational Thinking

Decompose

Example 2: Solving a crime

- What crime was committed
- When the crime was committed
- Where the crime was committed
- What evidence there is
- If there were any witnesses
- If there have recently been any similar crimes

What is Computational Thinking

Decompose

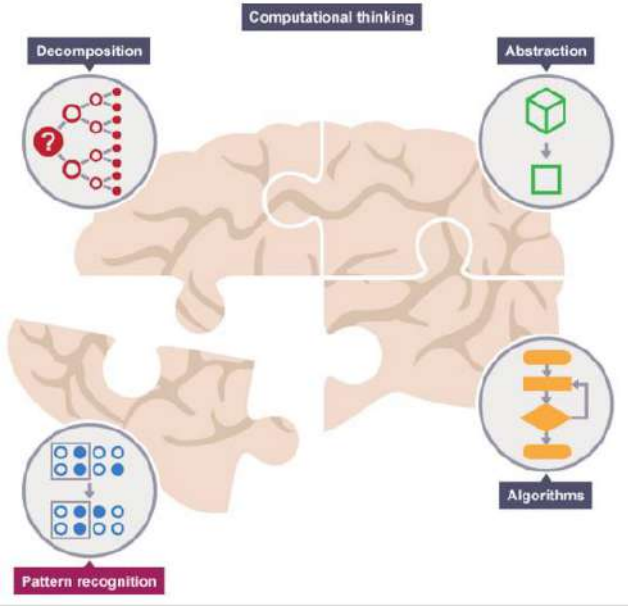
Example 3: Decomposing creating an app

- What kind of app you want to create
- What your app will look like
- Who the target audience for your app is
- What your graphics will look like
- What audio you will include
- What software you will use to build your app
- How the user will navigate your app
- How you will test your app
- Where you will sell your app

What is Computational Thinking

Patterns

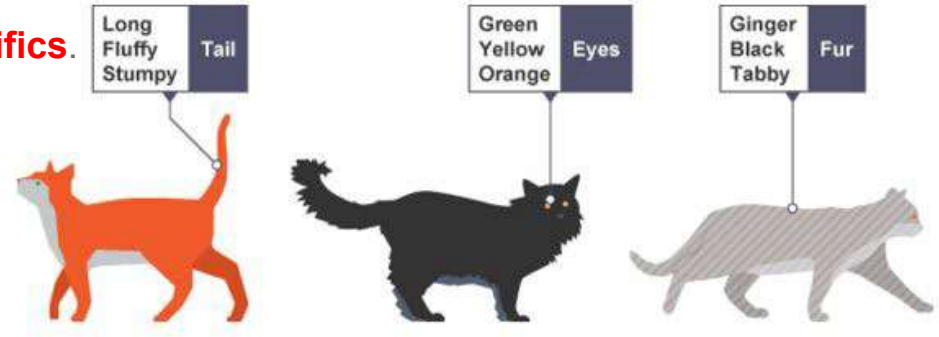
- The ability **to notice similarities or common differences** that will help us make predictions or lead us to shortcuts
- It is frequently the basis for solving problems and designing algorithms.



What is Computational Thinking

Patterns

- Imagine that we want to draw a series of cats.
- All cats share common **characteristics**: they all have eyes, tails and fur.
- Because we know that all cats have eyes, tails and fur, we can make a good attempt at drawing a cat, simply by including these common characteristics.
- Once we know how to describe one cat we can describe others, simply by following this pattern.
- The only things that are different are the **specifics**.



What is Computational Thinking

Patterns

- Finding patterns is extremely important. Patterns make our task simpler. Problems are easier to solve when they share patterns, because we can use the same problem-solving solution wherever the pattern exists.
- The more patterns we can find, the easier and quicker our overall task of problem solving will be.

What is Computational Thinking

Abstractions

Computational Thinking focuses on the process of abstraction (as in Mathematics):

- Choosing the right abstractions
- Operating in terms of multiple layers of abstraction simulation
- Defining the relationships the between layers

What is Computational Thinking

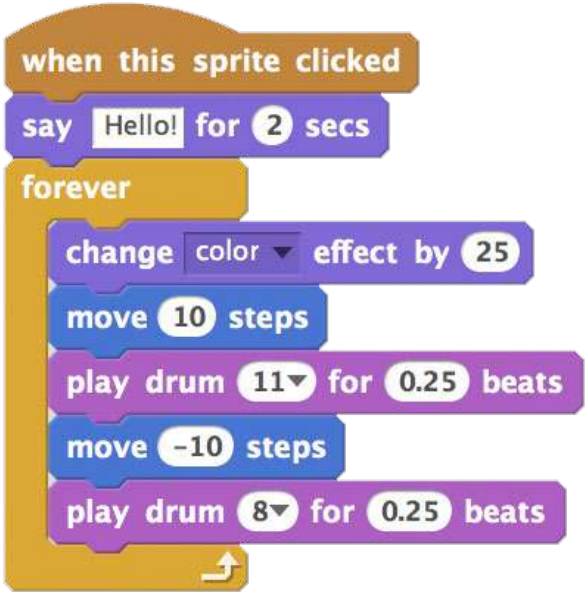
Abstractions

- The ability to filter out information that is not necessary to solve a certain type of problem and generalize the information that is necessary
- It allows us to represent an idea or a process in general terms (variables) so that we can use it to solve other problems that are similar in nature.

What is Computational Thinking

Algorithms

- **The ability to develop a step-by-step strategy for solving a problem**
- It is often based on the decomposition of a problem and the identification of patterns that help to solve the problem
- In CS, it is often written abstractly, utilizing variables in place of specific numbers



What is Computational Thinking

Logical Reasoning

- It helps develop children's ability to **reason logically** and to make deductions from the information they have
- **Make predictions**
- Debug – looking carefully at the code and using logical reasoning to explain what the program is actually doing are good starting points

What is Computational Thinking

Evaluation

- **It is the process that allows us to make sure our solution does the job it has been designed to do and to think about how it could be improved.**
- Once written, an algorithm should be checked to make sure it:
  - Is easily understood – is it fully decomposed?
  - Is complete – does it solve every aspect of the problem?
  - Is efficient – does it solve the problem, making best use of the available resources (eg as quickly as possible / using least space)?
  - Meets any design criteria we have been given
- Failure to evaluate can make it difficult to write a program.

What is Computational Thinking

Evaluation

How do we evaluate our solution?

- Does the solution make sense?

  Do you now fully understand how to solve the problem? If you still don't clearly know how to do something to solve the problem, go back and make sure everything has been properly decomposed.

- Does the solution cover all parts of the problem?

  Does the solution describe everything needed. If not, revisit and keep adding steps to the solution until it is complete.

- Does the solution ask for tasks to be repeated?

  If so, is there a way to reduce repetition? Go back and remove unnecessary repetition until the solution is efficient.

What is Computational Thinking

Evaluation

- Efficiency
  - How fast?
  - How much space?
  - How much power?
- Correctness
  - Does it do the right thing?
    - Does the program compute the right answer?
  - Does it do anything?
    - Does the program eventually produce an answer?
- Ilities
  - Simplicity and elegance
  - Usability
  - Modifiability
  - Maintainability
  - Cost
  - …

TABLE for BAKING CAKE

Cake Making Process	Elaboration
Decompose	• Break the baking process into sub-process • Ingredients – prepare the mix • Oven – pre-heat before use • Pan – need to grease with butter/margarine / flour
Pattern	• Basic ingredients for all types of cakes • Flour, butter, egg, sugar …
Abstraction	• the main criteria of cake making • right ingredients • right recipe • right process
Algorithm	• Step-by-step process of baking cakes
Logical Reasoning	• Reasons and rationale behind the step-by-step process • Modify recipe for different flavor i.e. carrot, chocolate • Modify recipe for different taste i.e. amount of sugar • Modify recipe for different texture i.e. type of flour
Evaluation	• Check the cake if it is cooked properly • Taste the cake when done so that it tastes as intended

Class Activity ~ Breakout Session

Construct a Computational Thinking Stages based on the 6 Concepts on How to Learn a new Programming Language

Decompose

Patterns

Abstraction

Algorithm

Logical Reasoning

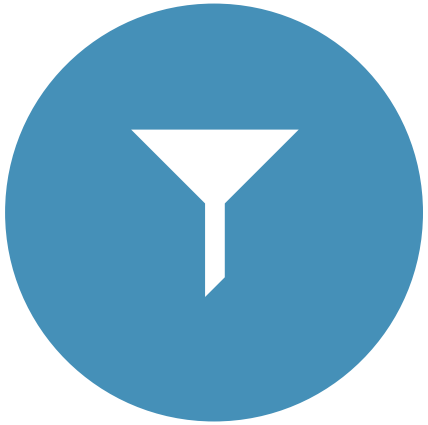
Evaluation

PROBLEM SOLVING METHOD IN PROGRAMMING

PROBLEMS SOLVING METHODS

1. ***Problem Analysis***
   - ***Analyse the problem & identify the specification of the requirements***
     - ***Input & output***
     - ***Process (steps) + Formula***
     - ***Constraints***
2. ***Design the solution (algorithm)***
3. ***Implement the algorithm (Writing a program)***
4. ***Testing & Verification***
5. ***Documentation***

PROBLEM ANALYSIS INVOLVED



DECOMPOSE

PATTERNS

ABSTRACTION

MAPPING TO COMPUTATIONAL THINKING TECHNIQUE

PROBLEM ANALYSIS QUESTIONS

- Understand the problem: What is the problem?
- What input and output?
- Process required (at this stage, no order is necessary)?
- Can you derive the formula?
- Is there a constraint to your possible solution?
- What are the limitation of your solution?

DESIGN A SOLUTION (FROM ANALYSIS TO ALGORITHM)

- Analysis information need to be sorted
- 2 options:
  - Pseudocode: NOT a code
  - Flow Charts
- Pseudocode is step by step description/instruction to solve the problem + Logic operator & Arithmetic operator + Control Structure
- Flow chart is a graphical representation using geometric symbols which represents specific meaning/purpose to describe step by step solution

IMPLEMENT THE ALGORITHM

- Require **translation** from **design** stage.
- How to translate from **pseudocode** ~ identify the structure. **But never copy pseudocode into a programming language directly.**
- How to translate from **flowchart** ~ identify the meaning of block diagram.
- In general, there is only the following category from the pseudocode.
  - Sequential structure (statement of facts)
  - Selection statement
  - Repetition/iteration statement

THE QUESTIONS

Design the solution that calculates the sales tax and the price of an item sold in a particular state

The sales tax is calculated as follows: the state portion of sales tax is 4% and the city's portion sales tax is 1.5%. If the item is a luxury item such as a car over $100,000, then there is a 10% luxury tax.

Suppose salePrice denotes the selling price of the item, stateTax denotes the state's sales tax, cityTax denotes the city's sales tax, luxTax deonotes the luxury's tax and finalPrice denotes the final price of the item.

Calculate all taxes and the final price of item (all taxes included).

PROBLEM ANALYSIS

Input/output

- Input: selling price of the item
- Output: sales tax and amount due

Process & Formula:

- Calculate the stateTax = salePrice * 0.04
- Calculate cityTax = salePrice * 0.015
- Determine the luxury item

  if item is a luxury item then calculate luxuryTax otherwise no luxuryTax.
- Calculate total sales tax

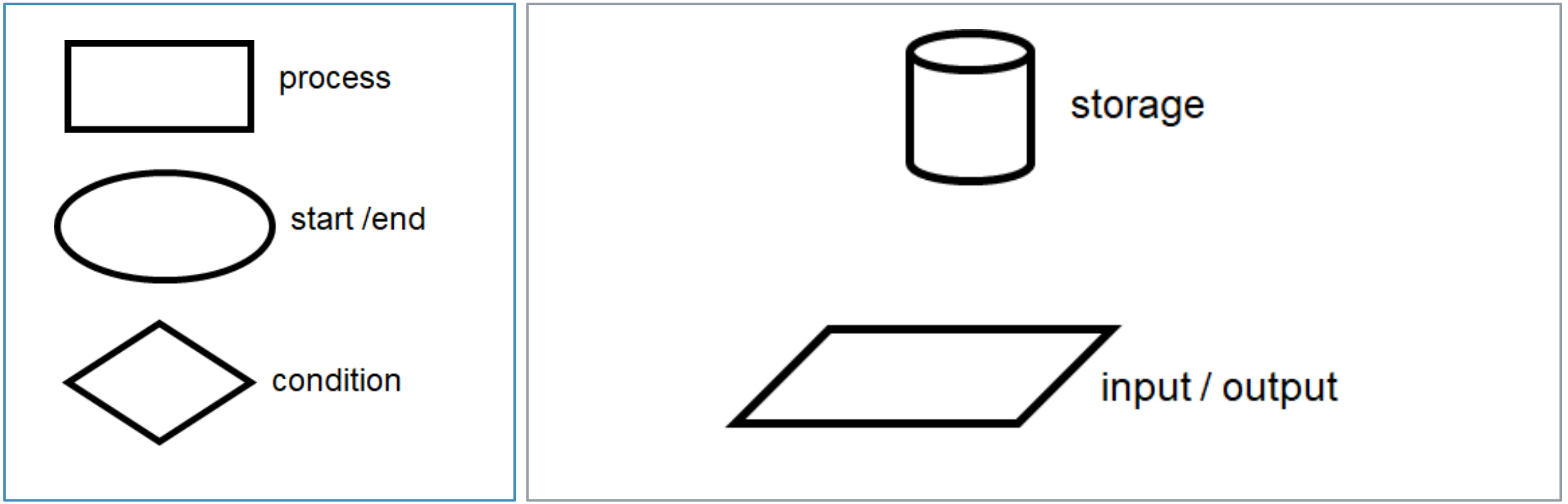
  SalesTax=stateTax + cityTax + luxTax
- Calculate total amount

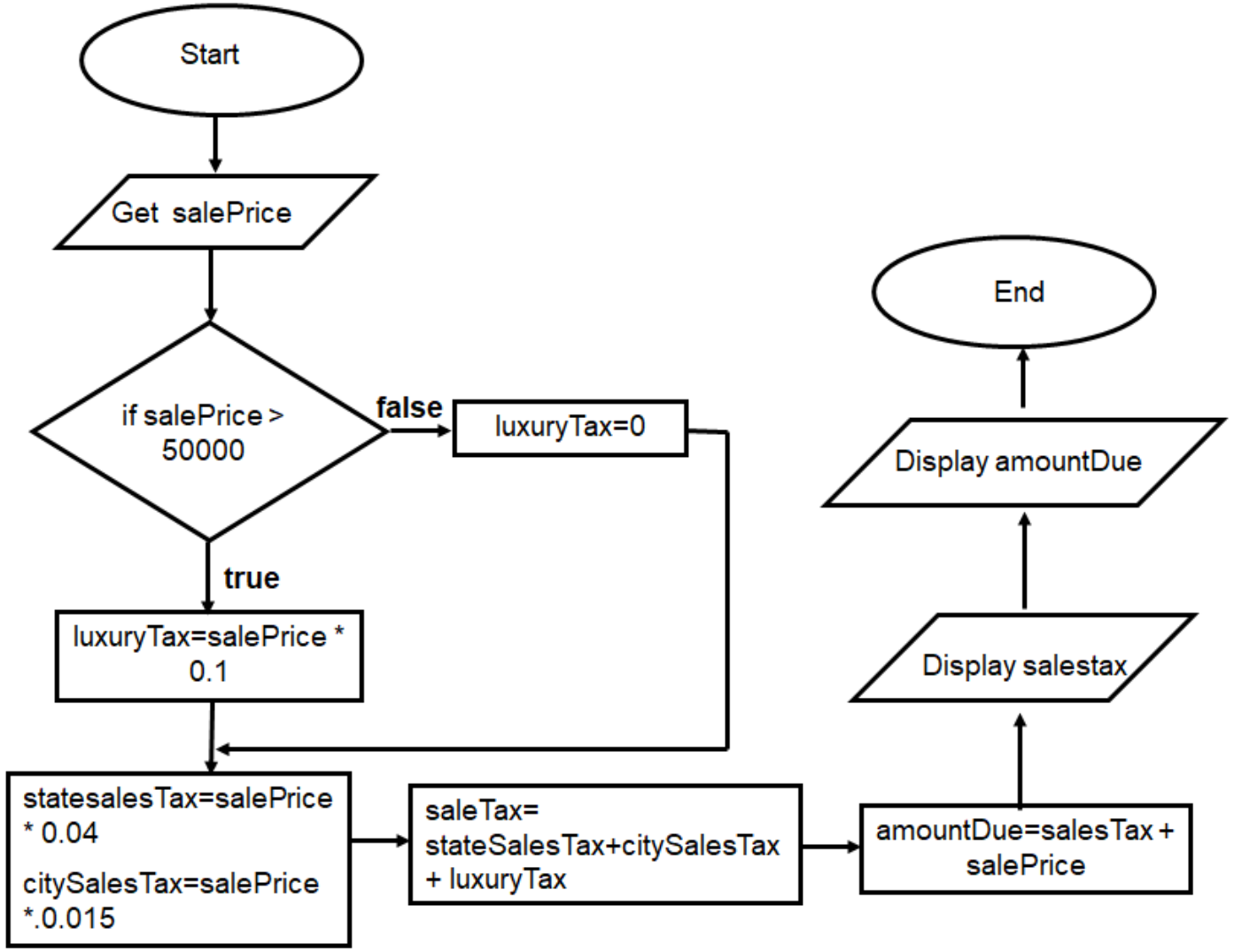
  finalPrice = salePrice + salesTax

PSEUDOCODES

1. get the selling price of the item
2. determine whether the item is a luxury
    2.1 if (item price > 100,000), it is a luxury item
        find luxTax = salePrice * 0.1
    2.2 otherwise (no luxury tax)
        luxTax = 0
3. find the portion of sales tax:
    3.1 stateTax = salePrice * 0.04
    3.2 cityTax = salePrice * 0.015
5. find salesTax
    5.1 salesTax = stateTax + cityTax + luxTax
6. find amountDue
    6.1 finalPrice= salePrice + salesTax
7. display salesTax
8. display finalPrice

FLOWCHARTS




Start
Get salePrice
if salePrice > 50000
false
luxuryTax=0
true
luxuryTax=salePrice * 0.1
statesalesTax=salePrice * 0.04
citySalesTax=salePrice *.0.015
saleTax= stateSalesTax+citySalesTax + luxuryTax
amountDue=salesTax + salePrice
Display salestax
Display amountDue
End

TESTING AND VERIFICATION

When you were implementing the program, be conscious of 3 types of errors in programming:

- **Design error**
  - Also known as logic error and semantic error
  - Occur during analysis and design phase
  - Difficult to detect by compiler
- **Syntax error**
  - Compiler can detect this type of error
- **Execution error**
  - Also known as runtime error

DOCUMENTATION

- Reference for the next programmer
- As the blueprint of the program – use when we need to maintain the program

1. Requirements specification (simple/precise)
2. Input/output, constrains and formulas
3. Pseudocode / flow-chart
4. Source code
5. User Manual

CODE DOCUMENTATION

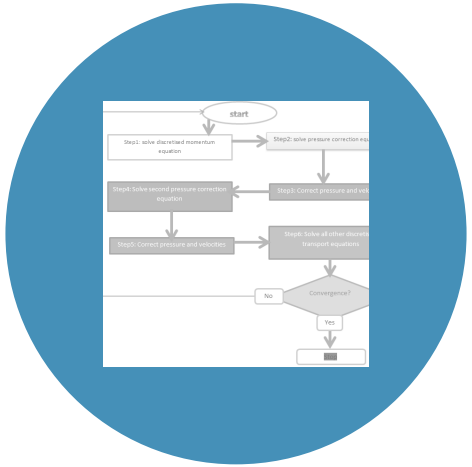
Header

- Name of programmer
- Date created & Date Update
- General purpose of a program

Source code

- Use meaningful program/function and variables name
- Write a meaningful comment. Not each line of the code
- For a function answer the question: what is the function for, what it accept, how it is process and how the answer is given back to calling function and etc.

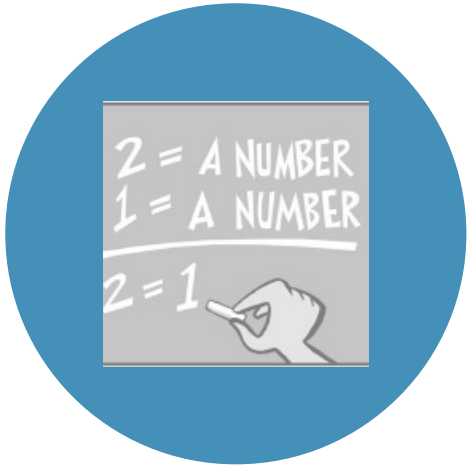
DESIGN THE SOLUTION INVOLVE…



ALGORITHM

MAPPING TO COMPUTATIONAL THINKING TECHNIQUE

IMPLEMENTATION & TESTING INVOLVED



LOGICAL REASONING

EVALUATION

MAPPING TO COMPUTATIONAL THINKING TECHNIQUE

SELF PRACTICE QUESTIONS

Given someone has travel with the speed 120km/h, after 2 hours, how far has he travel?

Based on the formula, Speed = Distance/Time

Therefore, Distance = Speed x Time

Distance = 120 x 2 = 240 km.

Conduct the whole problem solving methods

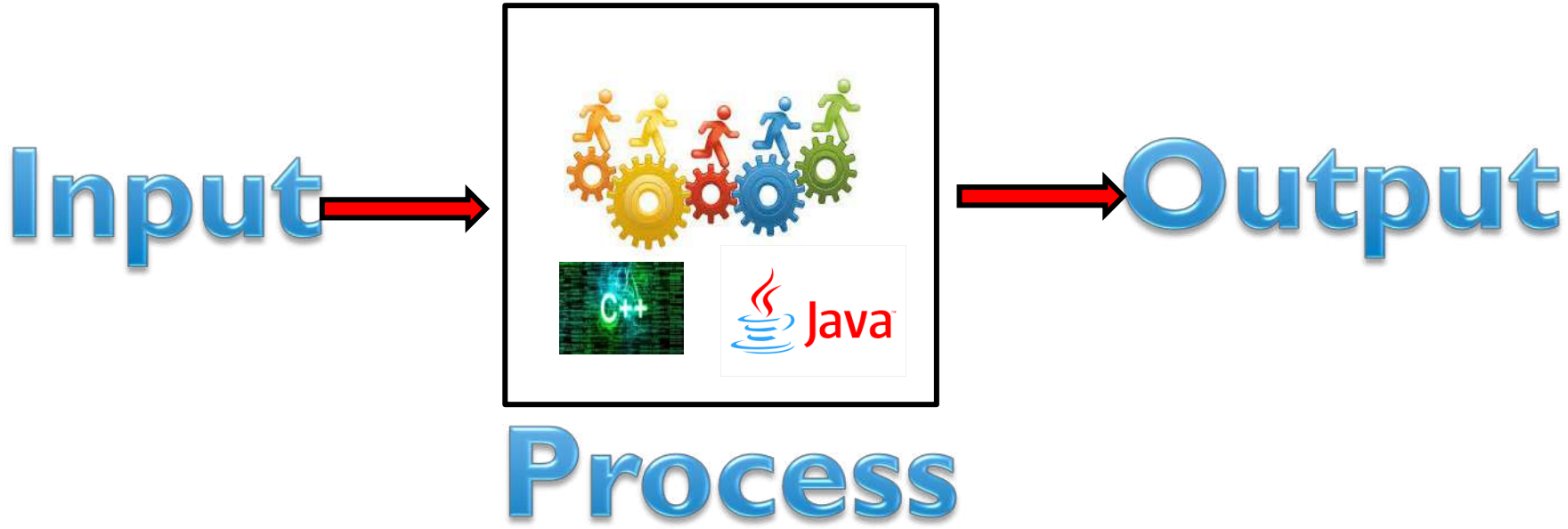
PROBLEMS SOLVING METHODS

1. ***Problem Analysis***
   - ***Analyse the problem & identify the specification of the requirements***
     - ***Input & output***
     - ***Process (steps) + Formula***
     - ***Constraints***
2. ***Design the solution (algorithm)***
3. ***Implement the algorithm (Writing a program)***
4. ***Testing & Verification***
5. ***Documentation***

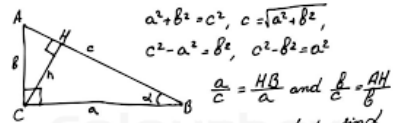
SOLUTION USING PROGRAM!!!

Sequence of instructions is very important!!!!!

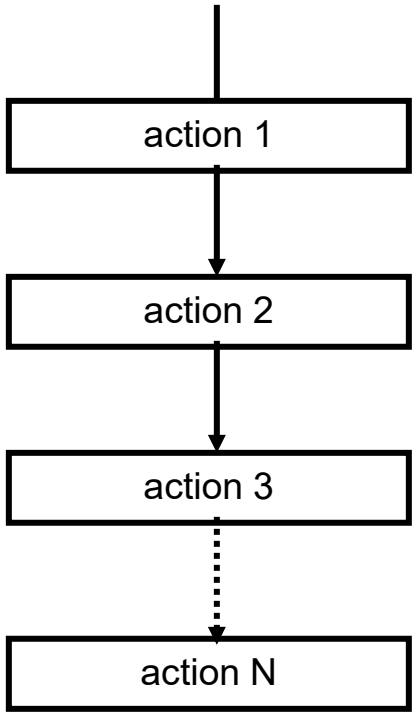
Algorithm → Program



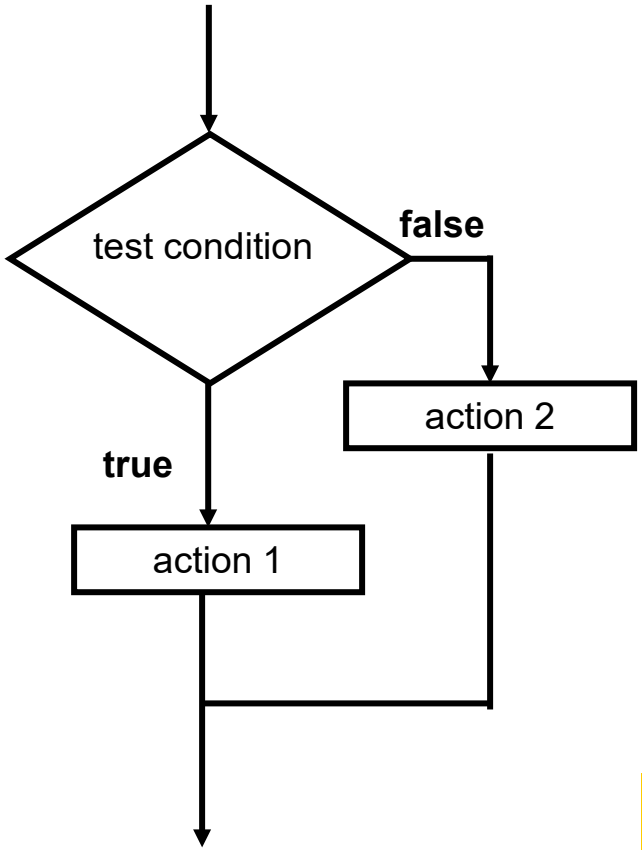
Y= mc



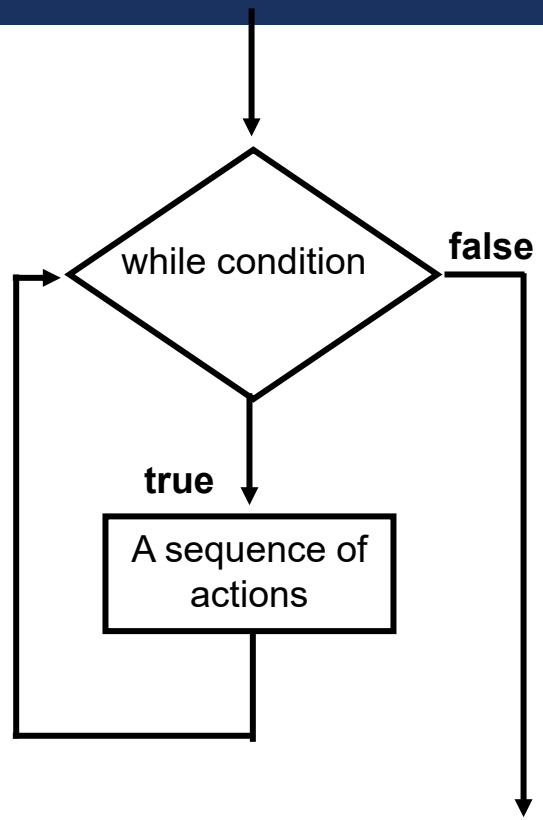
ALGORITHM – BASIC CONTROL STRUCTURES



A. Sequence



B. Selection /Decision



C. Repeation/iterative

THANK YOU

THAT'S ALL FOLKS