A·P·U
ASIA PACIFIC UNIVERSITY
OF TECHNOLOGY & INNOVATION

Introduction to Visual and Interactive Programming

CT803-4-0-OIVIP

Topic 2+

Problem solving, Algorithm and Debugging

Topic Learning Outcomes

At the end of this topic, you should be able to:

- Define algorithm

Contents & Structure

- What is a problem? – Type of problems
- Problem solving in everyday life
- What is problem solving strategies
- Algorithm

Introduction

- Computers everywhere

Ovens, washing machines and toys. Banks and Hospitals, Transport Reservations, Signaling Aircraft and industrial plant controllers, Missiles and satellites, and many more.

Six Basic Computer Operations

- A computer can
  - *receive* information. → input
  - *produce* information. → output
  - *perform arithmetic.*
  - *assign a value* to a piece of data.
  - *compare two pieces of information*

    ` and select one of two alternative actions.
  - *repeat* a group of actions.

process

General Problem-Solving Concepts : Problem Solving in everyday life

What is Problem ?

A Problem is a state of difficulty that need to be resolved

While solving a problem there is a desire to attain some specific goal .

Here is some examples we face in day-to-day life can be important or can be least importance .

- Will I get proper transport to go to my workspace ?
- Should I wear shoes today ?
- Should I go to a movies ?
- Which Cell phone should I buy?

Types of Problem

Problem Based on algorithmic solution :

Sequence of instruction .

For solving some problem , series of actions are taken to reach the solution.

Problem based on heuristic Solutions :

The Solution that can not reached through direct set of steps are called heuristic solutions .

With heuristic solution problem solver must follow six steps of problem solving for more than once .

Problem solving with computer

Solution : Instruction that must be followed to produce the best result .

Results: It is an outcome or the completed computer -assisted answer .

Program : Computer program are set of instructions executed to obtain solution to certain problem. These programs are written in some specific programming language.

Computer deal with the problem having algorithmic solution. The field of computer that deal with solving the heuristic problems is called artificial intelligence .

Difficulties with problem solving

We do not understand the problem correctly .

Sometimes we do not define the problem correctly or adequately

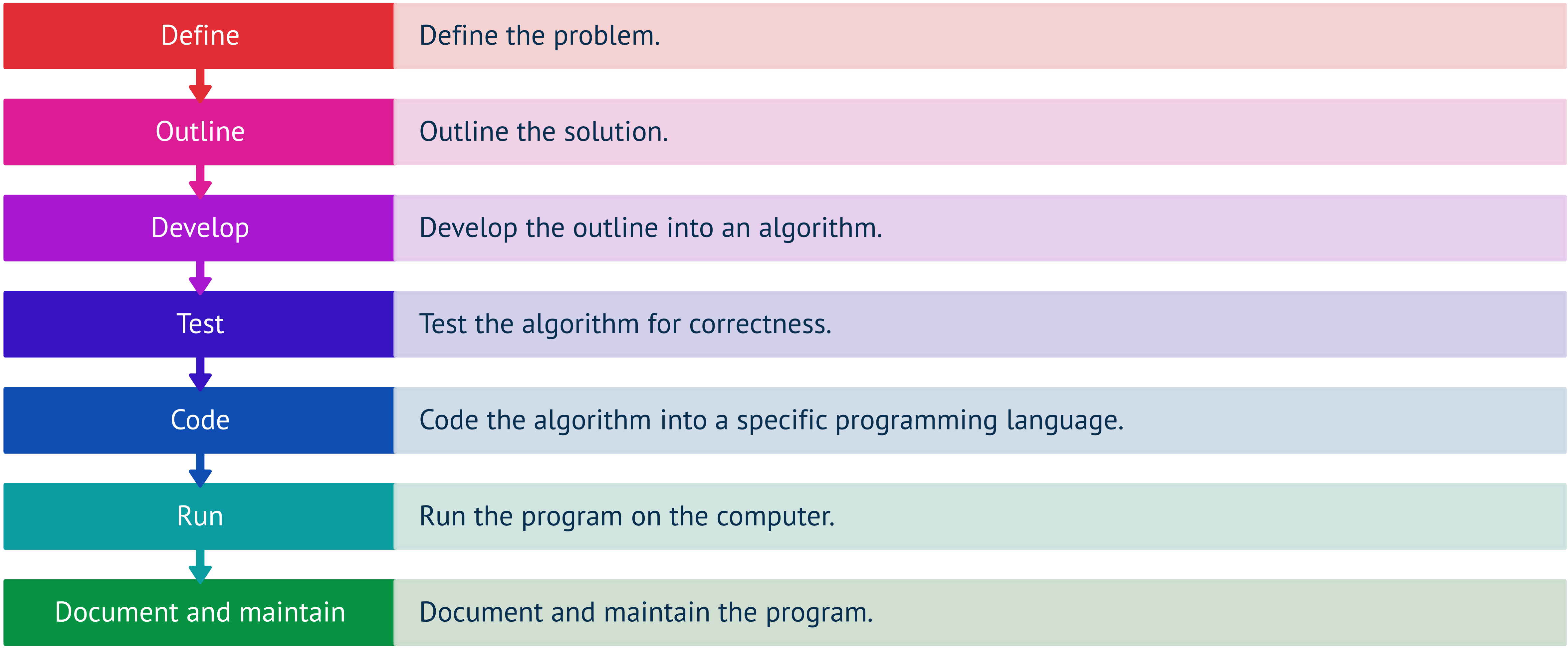
People get afraid of taking decisions while solving the problems

Sometimes the list of alternatives is incomplete .

The sequence of solutions to the problems is not logical many times .

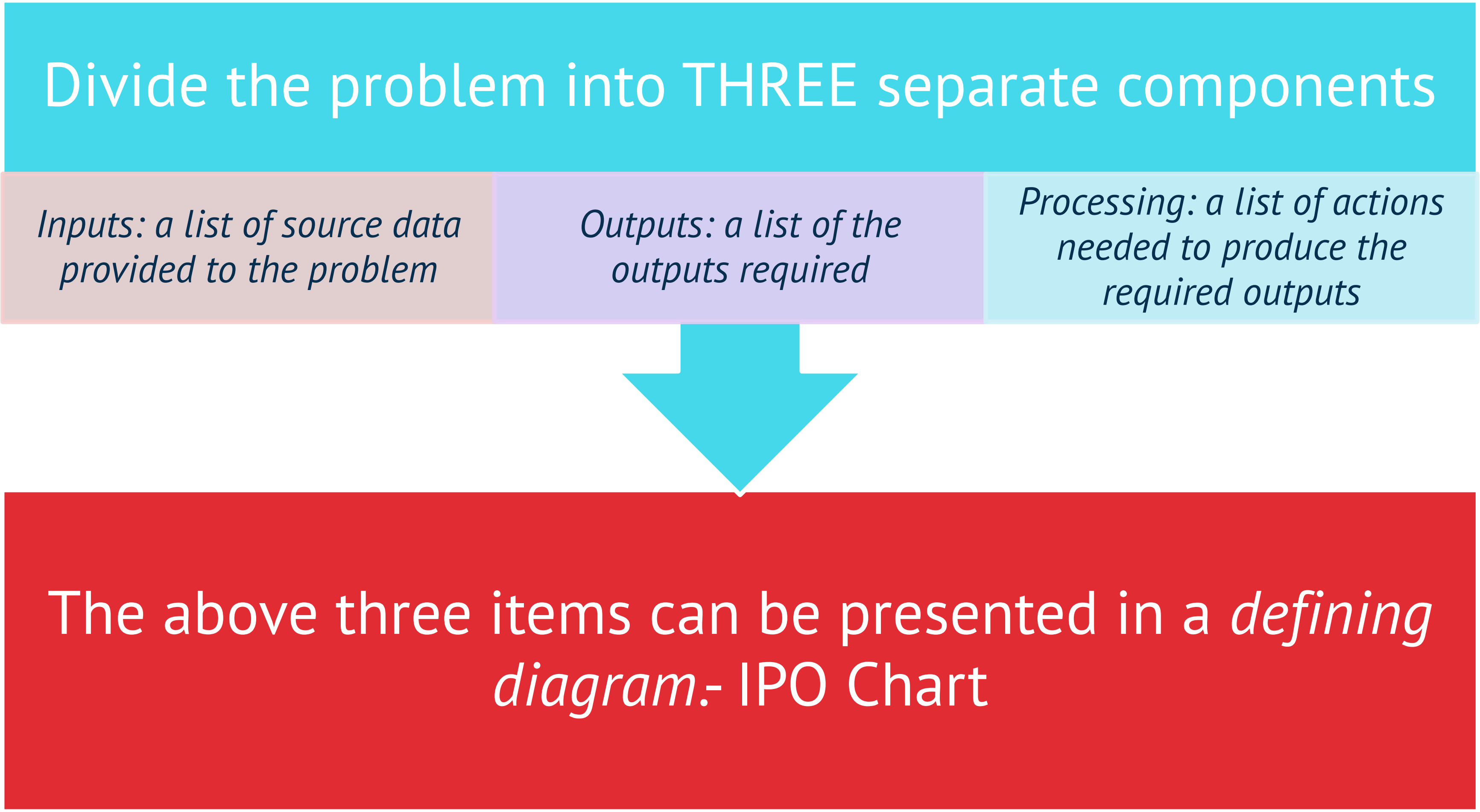
When solving problem on the computer , one of the most difficult tasks for the problem solver is writing the instructions

Steps to Developing a Program



Define	Define the problem.
Outline	Outline the solution.
Develop	Develop the outline into an algorithm.
Test	Test the algorithm for correctness.
Code	Code the algorithm into a specific programming language.
Run	Run the program on the computer.
Document and maintain	Document and maintain the program.

1. Define The Problem



Divide the problem into THREE separate components

Inputs: a list of source data provided to the problem

Outputs: a list of the outputs required

Processing: a list of actions needed to produce the required outputs

The above three items can be presented in a *defining diagram.*- IPO Chart

Analyse the problem

Understand

Thoroughly understand the problem

Understand

Understand problem requirements

- Does program require user interaction?
- Does program manipulate data?
- What is the output?
- Are all possible circumstances handled?

Divide

If the problem is complex, divide it into subproblems

- Analyze each subproblem as above

Analyse the problem

Purpose:

To describe in detail a solution to a problem and information needed in solving the problem.

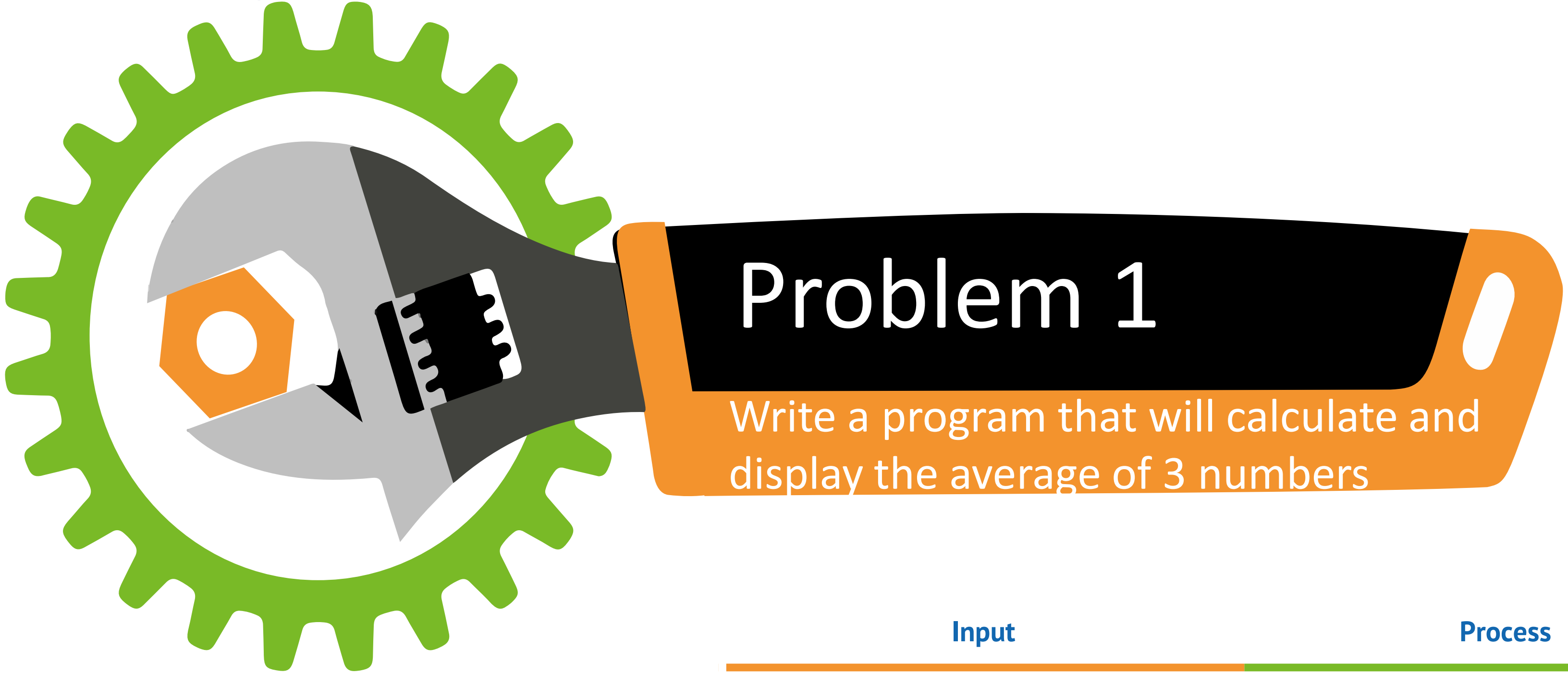
How

- Study and understand the problem
- Identify
  - The needed input.
  - The required output.
  - The needed process.
    - For example: scientific formula or hypothesis.

Defining Diagram

Input	Processing	Output

Analyse the problem

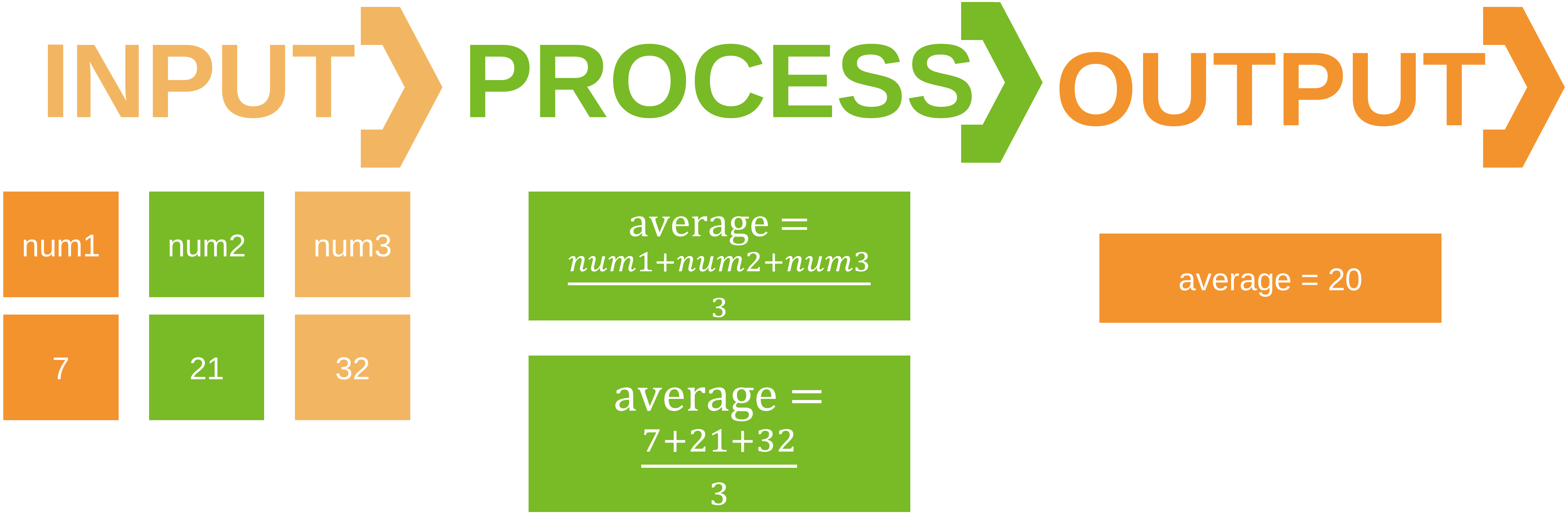


Problem 1

Write a program that will calculate and display the average of 3 numbers

Input	Process	Output
num1, num2 and num3	average = $\frac{num1+num2+num3}{3}$	average

Analyse the problem



INPUT

PROCESS

OUTPUT

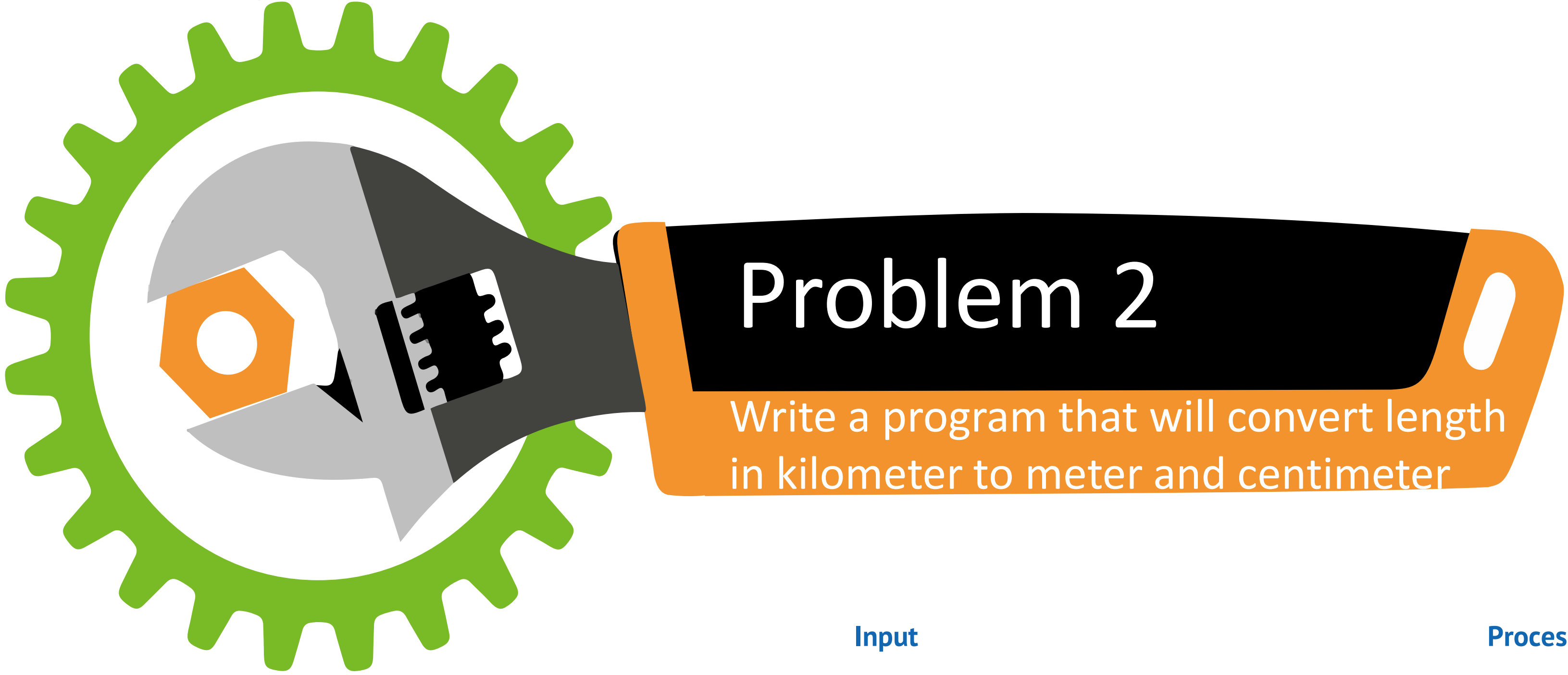
num1	num2	num3
7	21	32

$$\text{average} = \frac{num1+num2+num3}{3}$$

$$\text{average} = \frac{7+21+32}{3}$$

average = 20

Analyse the problem



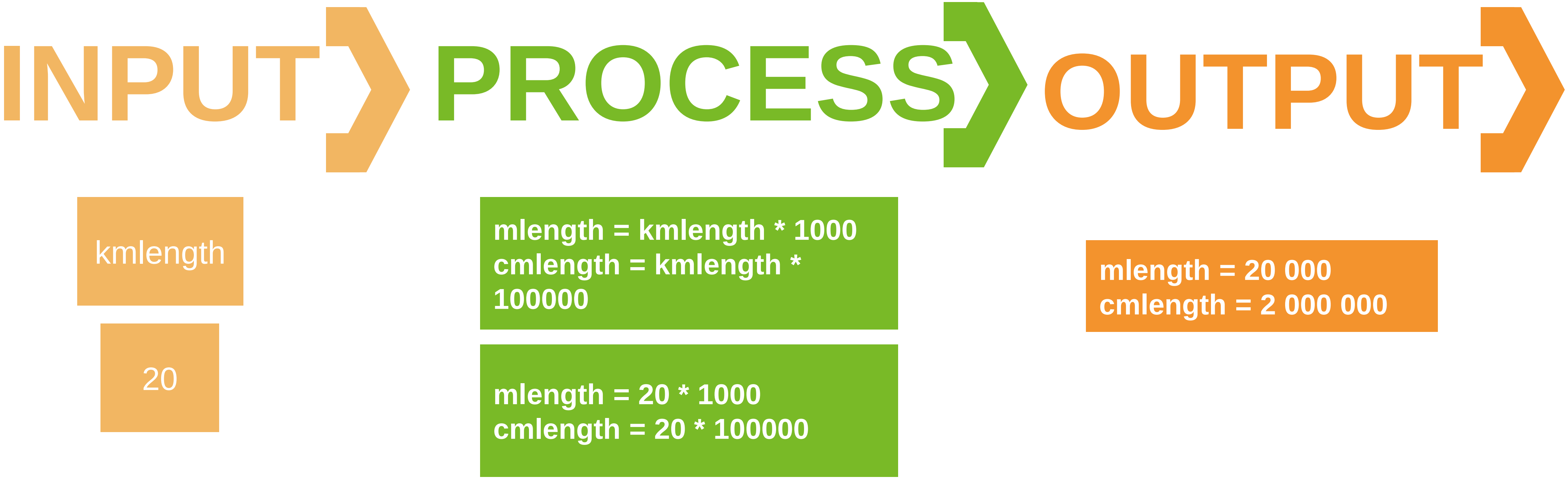
Problem 2

Write a program that will convert length in kilometer to meter and centimeter

Input	Process	Output

kmlength	mlength = kmlength * 1000
	cmlength = kmlength * 100000
		mlength and cmlength

Analyse the problems



INPUT

PROCESS

OUTPUT

kmlength

20

mlength = kmlength * 1000
cmlength = kmlength * 100000

mlength = 20 * 1000
cmlength = 20 * 100000

mlength = 20 000
cmlength = 2 000 000

2. Outline The Solution

During this stage, certain details are identified from the problem by analyzing it further, such as:

- major processing tasks involved.
- major subtasks (if any)
- major control structures.
- major variables
- mainline logic

Strategies

- Ask questions!
  - What do I know about the problem?
  - What is the information that I have to process in order to find the solution?
  - What does the solution look like?
  - What sort of special cases exist?
  - How will I recognize that I have found the solution?

3. Develop The Algorithm

A detailed step by step algorithm is written out.

Often use one of three tools:

- Pseudocode
- Flowcharts
- Nassi-Schneiderman diagrams – will not be covered in this module

Before a computer can perform a task, it must have an algorithm that tells it what to do.

Algorithmic thinking definition

- Identifying the steps involved in solving a problem.
  - What are the steps involved with making a cup of tea?
  - What are the steps involved in the calculation 200÷12?
  - What are the steps involved in checking if a player has hit a target in a game?
  - What are the steps involved in getting dressed for school?

What is Algorithm?

Algorithms are the practical application of algorithmic thinking.

Algorithms are simply steps/procedures used to get the intended result.

Algorithms state the steps required to get to the desired result.

Creating algorithms is all about thinking logically and so to think computationally is to think in a structured, organized and logical manner.

Algorithms are vital to programming. Without one, coding a solution is almost impossible…but with one, coding becomes an easy job (if you know your programming syntax that is!)

Algorithm in a real world

Search engines such as Bing or Google use algorithms to put a set of search results into order,

Your Facebook news feed is derived from your friends' status updates and other activity, but it only shows that activity which the algorithm thinks you'll be most interested in seeing.

The recommendations you get from Amazon, Netflix and eBay are algorithmically generated, based in part on what other people are interested in.

Writing Algorithms

Algorithms are independent of any language.

Writing an algorithm in a specific language:

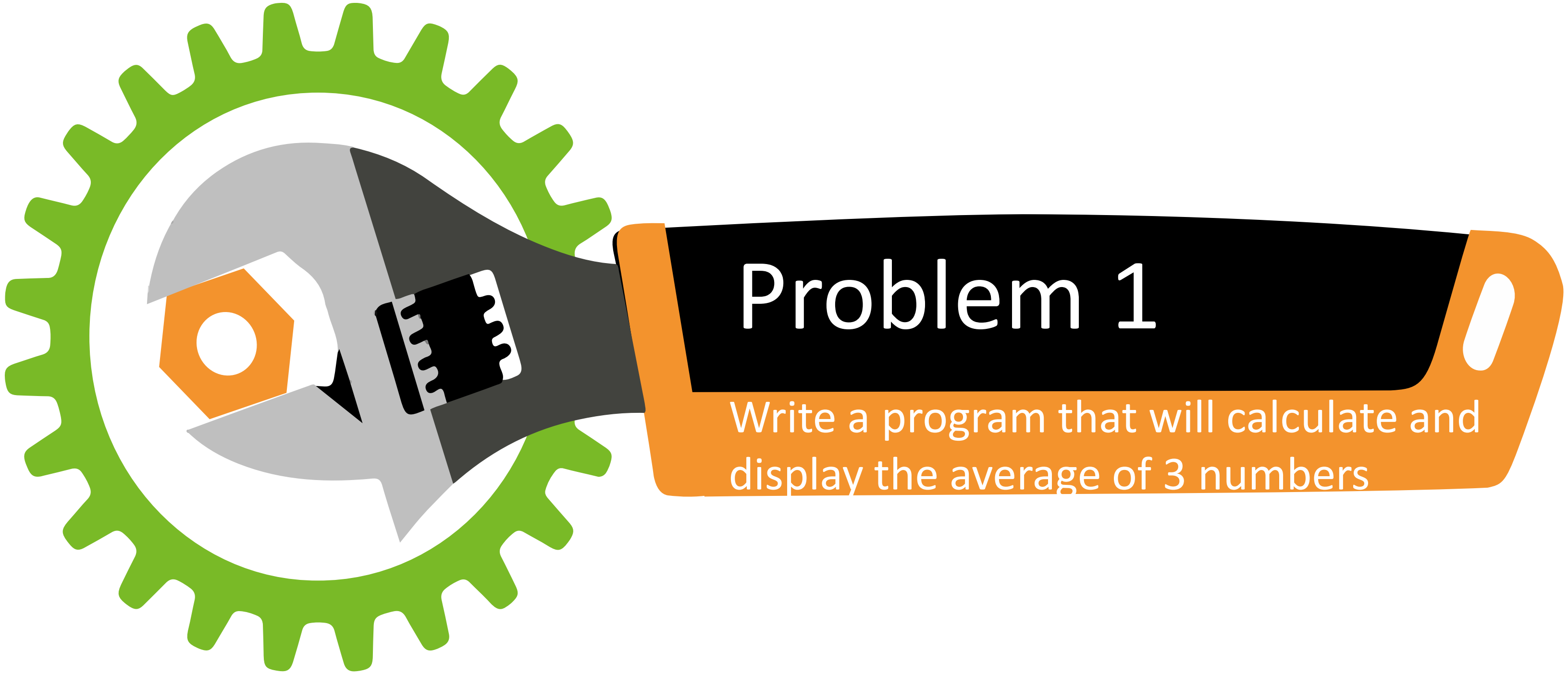
- Too **time consuming**
- **Pointless** – could just code
- Cannot be taken by a **programmer using a different language**
- **Too comple**x – need knowledge of syntax (code specific to a language)

Writing an algorithm in everyday language:

- Too **time consuming**
- Open to interpretation – thus resulting in different outcomes

So, we use Flowcharts and pseudocode to write algorithms so that they are concise, accurate and easy to understand so that a programmer of any language could understand the steps required to solve a task

Program Design

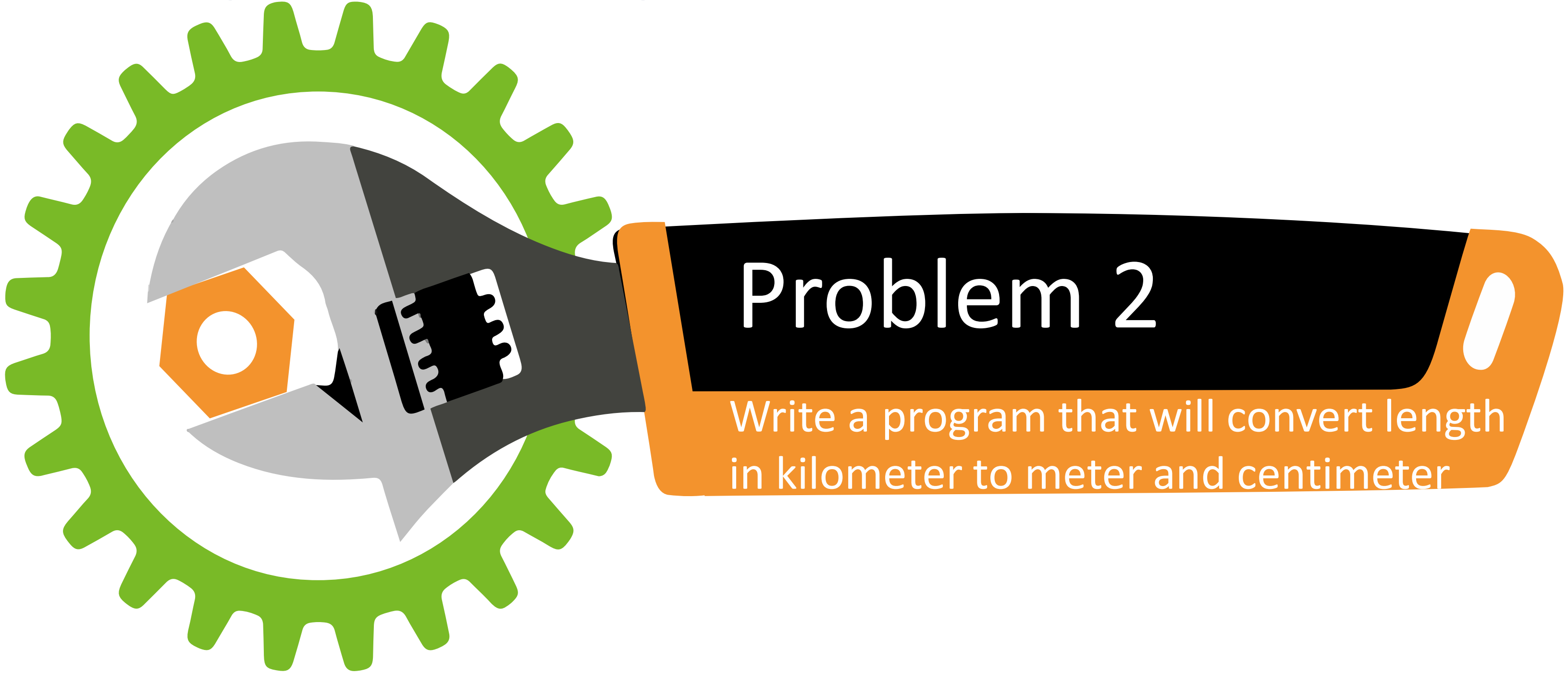


Problem 1

Write a program that will calculate and display the average of 3 numbers

ALGORITHM

Program Design



Problem 2

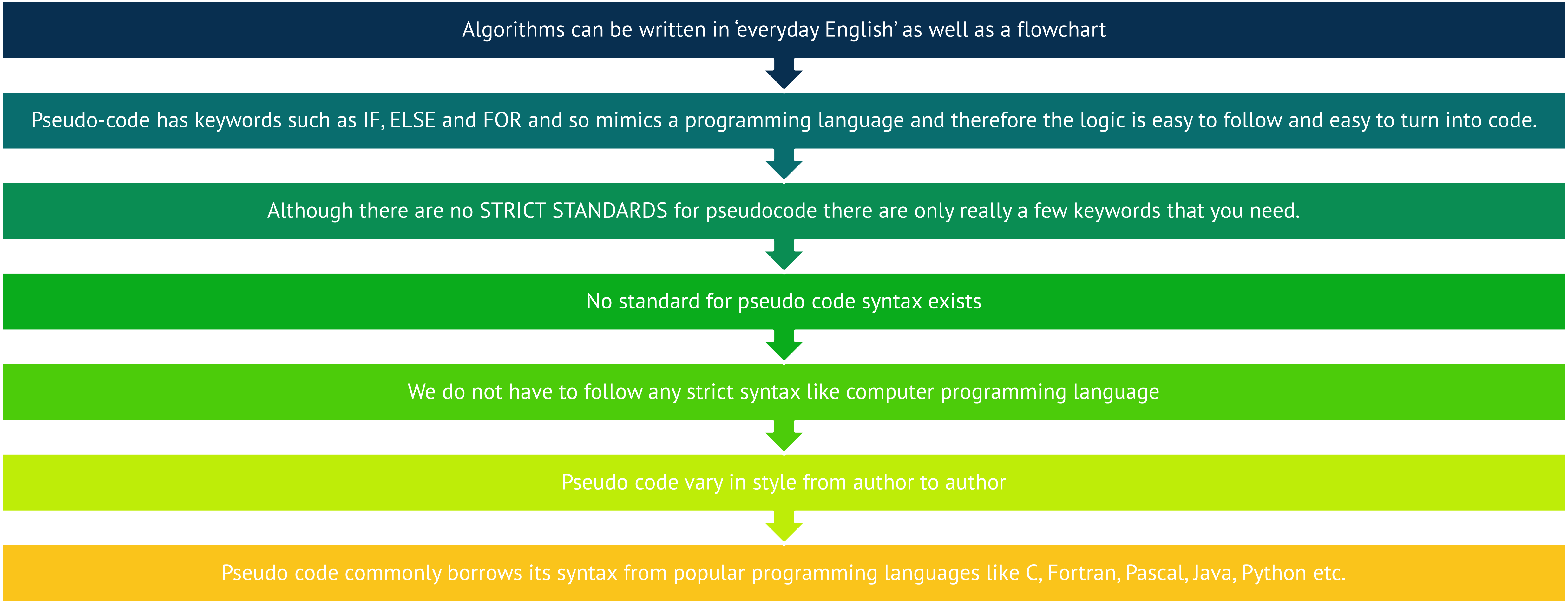
Write a program that will convert length in kilometer to meter and centimeter

ALGORITHM

Pseudo code - About

- No standard pseudocode.
- A pseudo code is an informal way to describe a program
- Pseudo code is not a computer program
- Pseudo code can use natural language or compact mathematical notation
- It is a rough sketch of the actual program

Pseudo code - Syntax



Algorithms can be written in ‘everyday English’ as well as a flowchart

Pseudo-code has keywords such as IF, ELSE and FOR and so mimics a programming language and therefore the logic is easy to follow and easy to turn into code.

Although there are no STRICT STANDARDS for pseudocode there are only really a few keywords that you need.

No standard for pseudo code syntax exists

We do not have to follow any strict syntax like computer programming language

Pseudo code vary in style from author to author

Pseudo code commonly borrows its syntax from popular programming languages like C, Fortran, Pascal, Java, Python etc.

Pseudo-code Start and End “Key Words”

- Pseudocode begin with a START and ends with END
- The algorithm goes in between.
- You will need to DECOMPOSE the problem set in the question to work out what comes in between
- Pseudocode and their statements

```
START
.................................
END
```

Pseudo-code Input / Output "Key Words"

At times, your program will most certainly ask the user for inputs and output values too.

Inputs and Outputs (like "Name?" or "...display age") are indicated using the following words.

Usually, a programmer will choose one and stick with it throughout their algorithm.

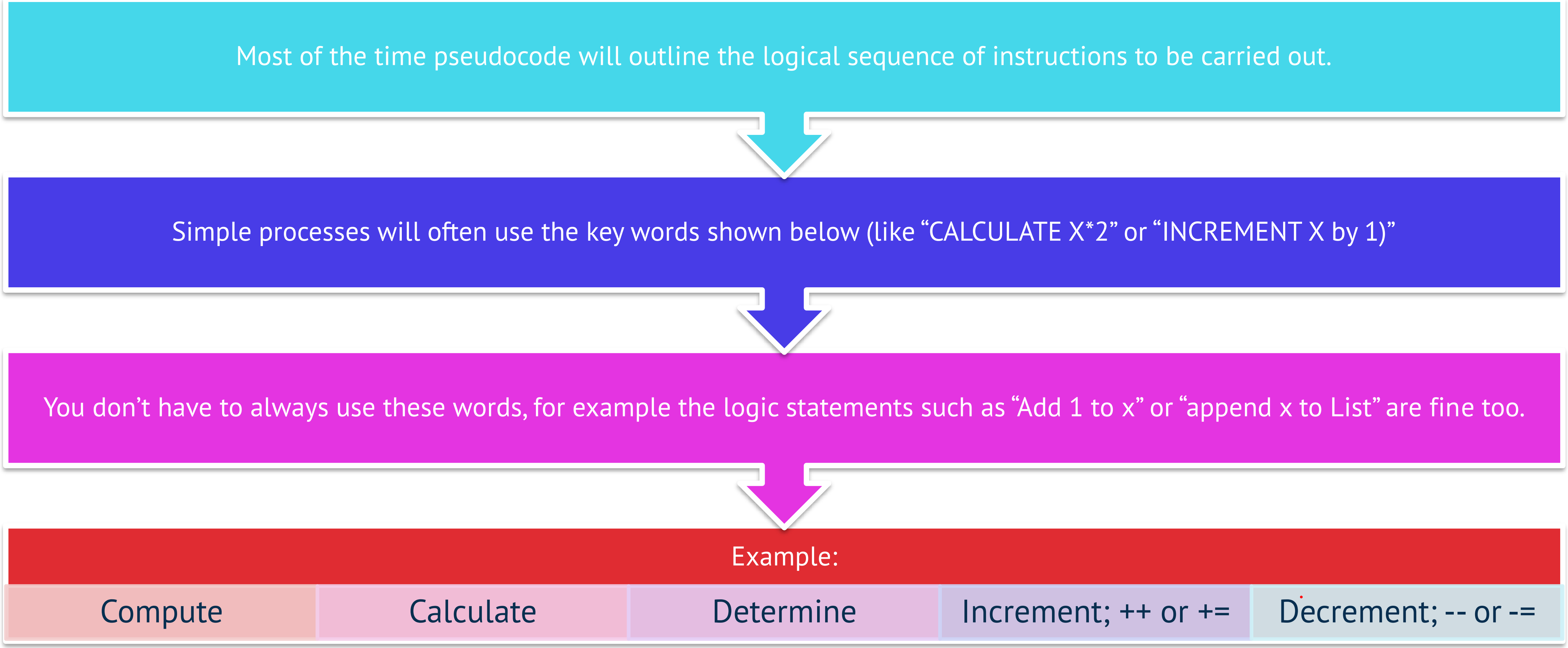
INPUTS:

READ
OBTAIN
GET
INPUT

OUTPUTS:

PRINT
DISPLAY
SHOW
OUTPUT

Pseudo-code Process “Key Words”



Most of the time pseudocode will outline the logical sequence of instructions to be carried out.

Simple processes will often use the key words shown below (like “CALCULATE X*2” or “INCREMENT X by 1)”

You don’t have to always use these words, for example the logic statements such as “Add 1 to x” or “append x to List” are fine too.

Example:

Compute	Calculate	Determine	Increment; ++ or +=	Decrement; -- or -=

Pseudo-code More Keywords

Variable Assignment “Key Words”

- At times, your program will assign values to variables.
- In pseudocode this is done using the following key words.
  - SET

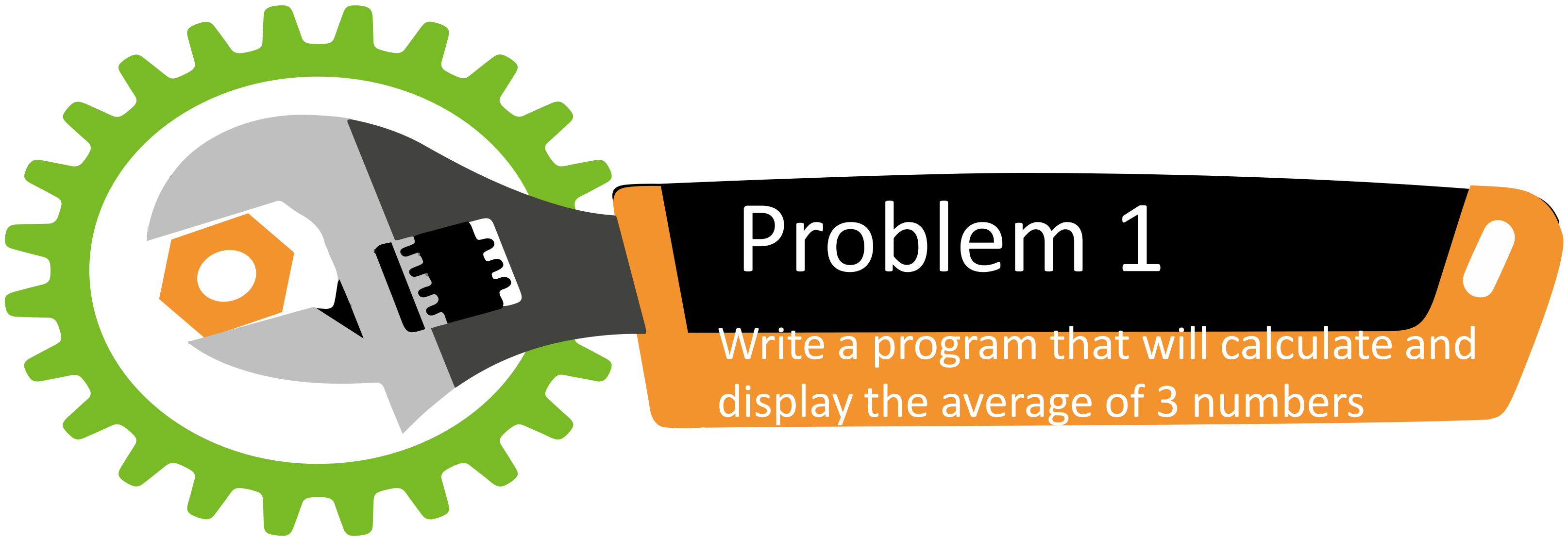
Decision/Selection “Key Words”

- At times, your program will be programmed to make a decision based on certain conditions.
- Decisions (like “IF X = 3, THEN …”) are shown using, the following key words.
  - IF
  - THEN
  - ELSE
  - ELSE-IF
  - ENDIF

Loops / Iterations “Key Words”

- Programs will often loop in places while certain conditions occur (infinitely) or for a set number of times (finitely).
- Loops use the following key words:
- FOR
- WHILE / ENDWHILE
- REPEAT / UNTIL

Program Design – Pseudo-code



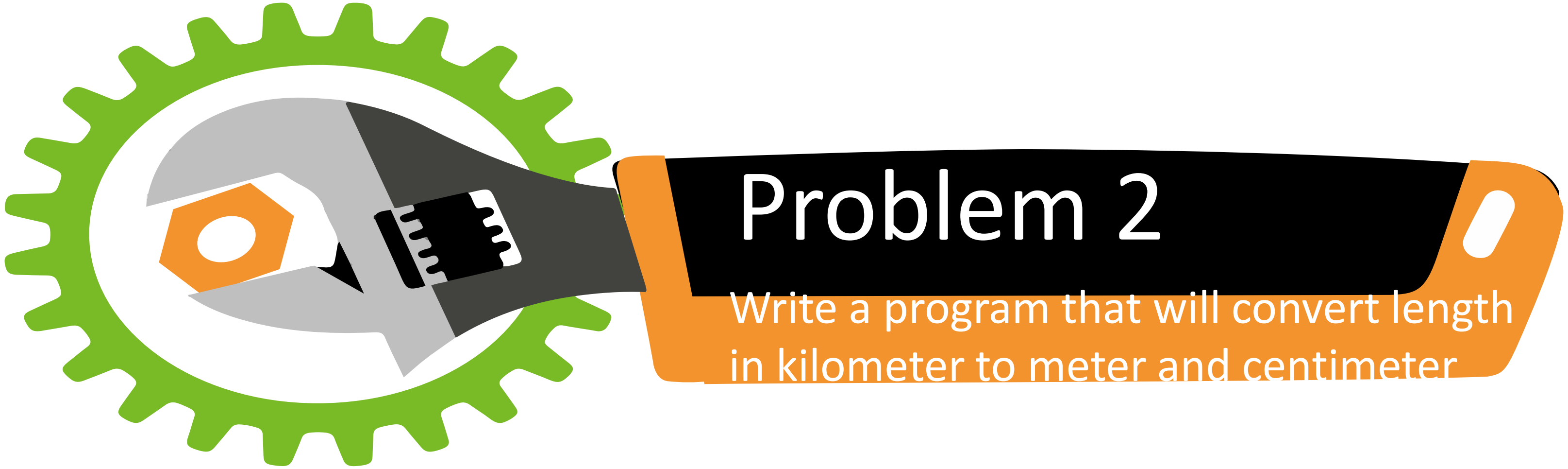
Problem 1

Write a program that will calculate and display the average of 3 numbers

Pseudo code

```
Start
Ask num1,num2,num3
Set average to (num1+num2+num3)/3
say average
End
```

Program Design – Pseudo-code



Problem 2

Write a program that will convert length in kilometer to meter and centimeter

Pseudo code

```
Start
Set mlength to 0
Set cmlength to 0
Ask kmlength
mlength = kmlength * 1000
cmlength = kmlength * 100000
output mlength, cmlength
End
```

Program Design – Pseudo-code

- Example 3:
  - A central heating system will try to keep the temperature between 2 values (19 and 21)
  - If the temperature falls below 19 It will turn the heating system on
  - If the temperature rises above 21 it will switch the heating system off.

Program Design - Flowchart

Flowchart is a pictorial way to express algorithm or process.

Visual representation of the logic of a program

Limited range of symbols to describe processes

Arrows to show the order of instructions

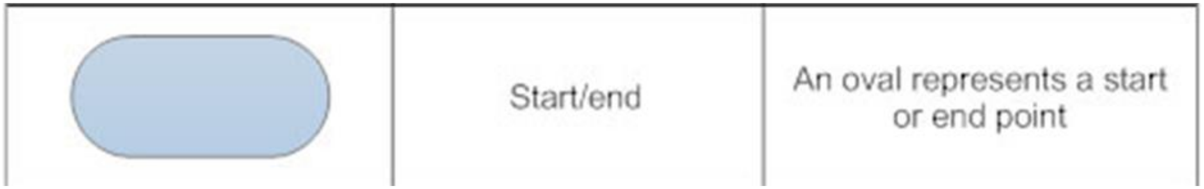
Easier to follow and identify issues than in pseudocode.

So, instead of writing down the algorithm in some programming language like Snap!, C, C++, Java, C#, PHP, Python, Ruby etc. Use flowchart to express the algorithm which gives us a general view about the algorithm.

Flowchart as the name indicates, is about the flow of execution of our algorithm.

Flowchart and their symbols

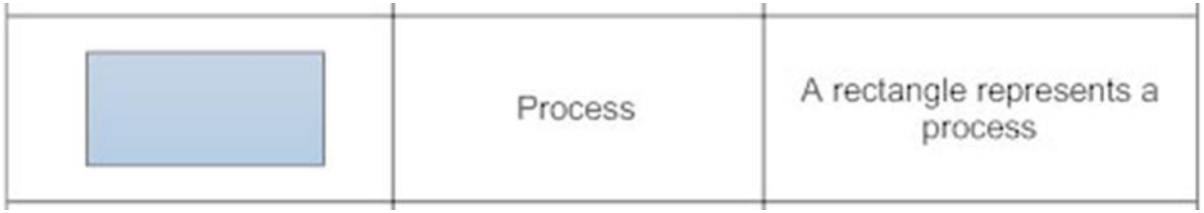
- Start and Stop Symbols
  - All flow charts begin with a Start Symbol and at the end of the flow chart (or at various end points of the chart) we place a Stop Symbol.
  - There are drawn as a rectangle with curved ends



⬭	Start/end	An oval represents a start or end point

Flowchart and their symbols

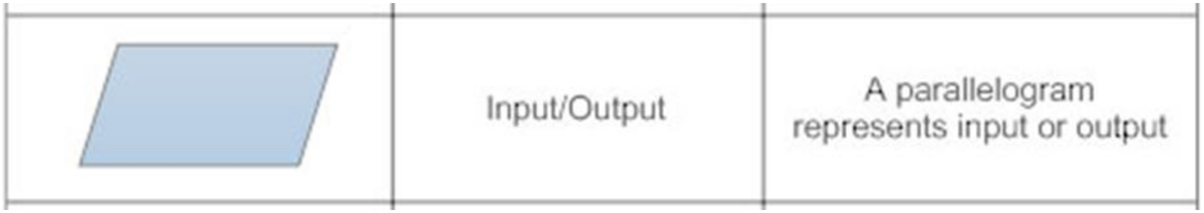
- Process Symbols
  - Most of the time a flow chart will demonstrate the sequence of instructions to be carried out.
  - Simple processes (like “Add 1 to x” or “append x to List”) are shown using a standard rectangle.



▭	Process	A rectangle represents a process

Flowchart and their symbols

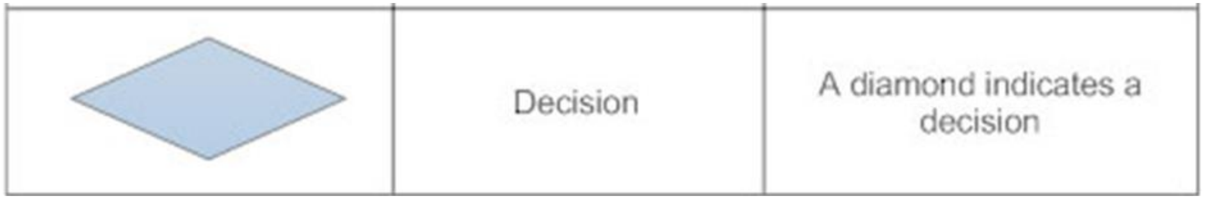
- Input / Output Symbols
  - At times, your program will most certainly ask the user for inputs and output values too.
  - Inputs and Outputs (like “Name?” or “…display age”) are shown using a parallelogram.



▱	Input/Output	A parallelogram represents input or output

Flowchart and their symbols

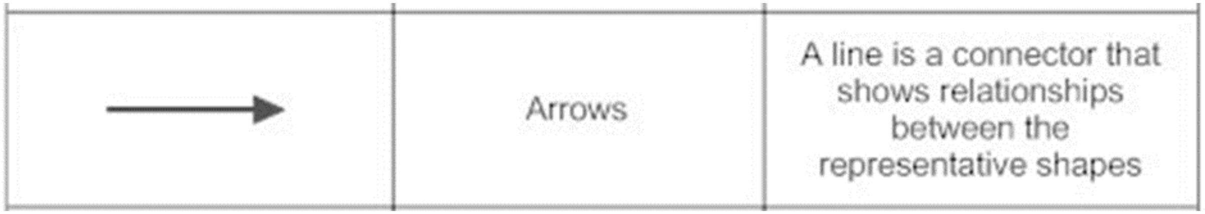
- Decision Symbols
  - At times your program will be programmed to make a decision based on certain conditions.
  - Decisions (like “IF X = 3” or “While Y > 3”) are shown using a diamond.



◇	Decision	A diamond indicates a decision

Flowchart and their symbols

- Flow Symbols
  - Show direction of flow.



→	Arrows	A line is a connector that shows relationships between the representative shapes

Flowchart - Rules

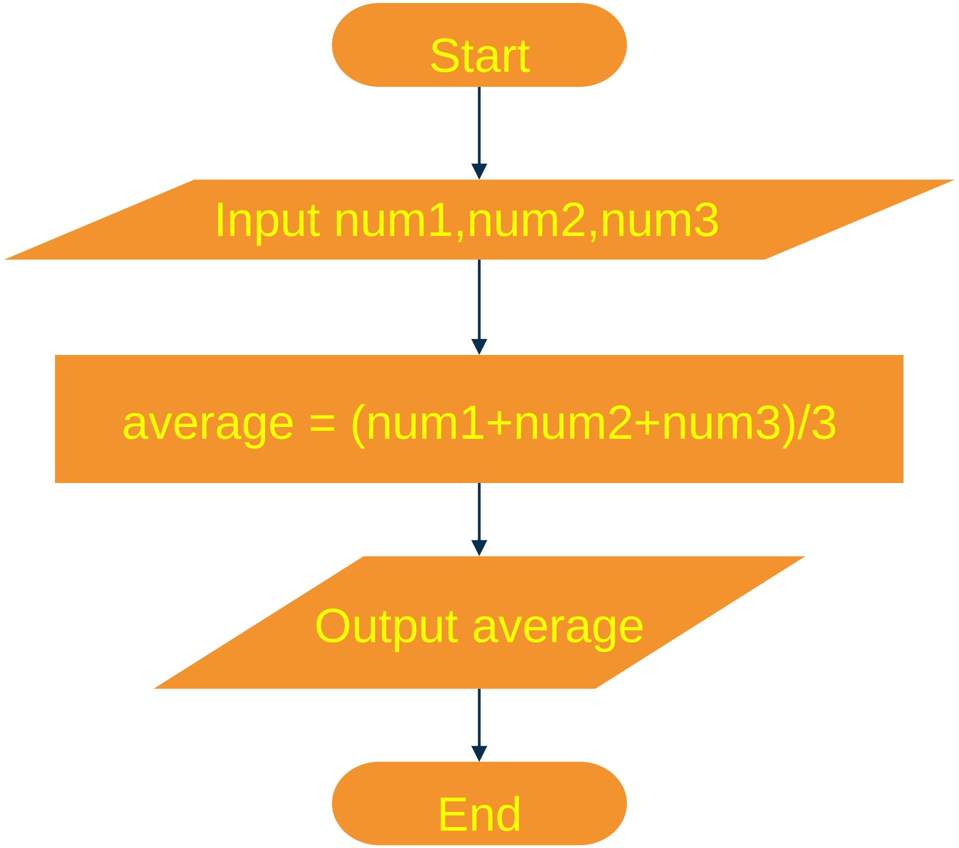
Flowchart is generally drawn from top to bottom

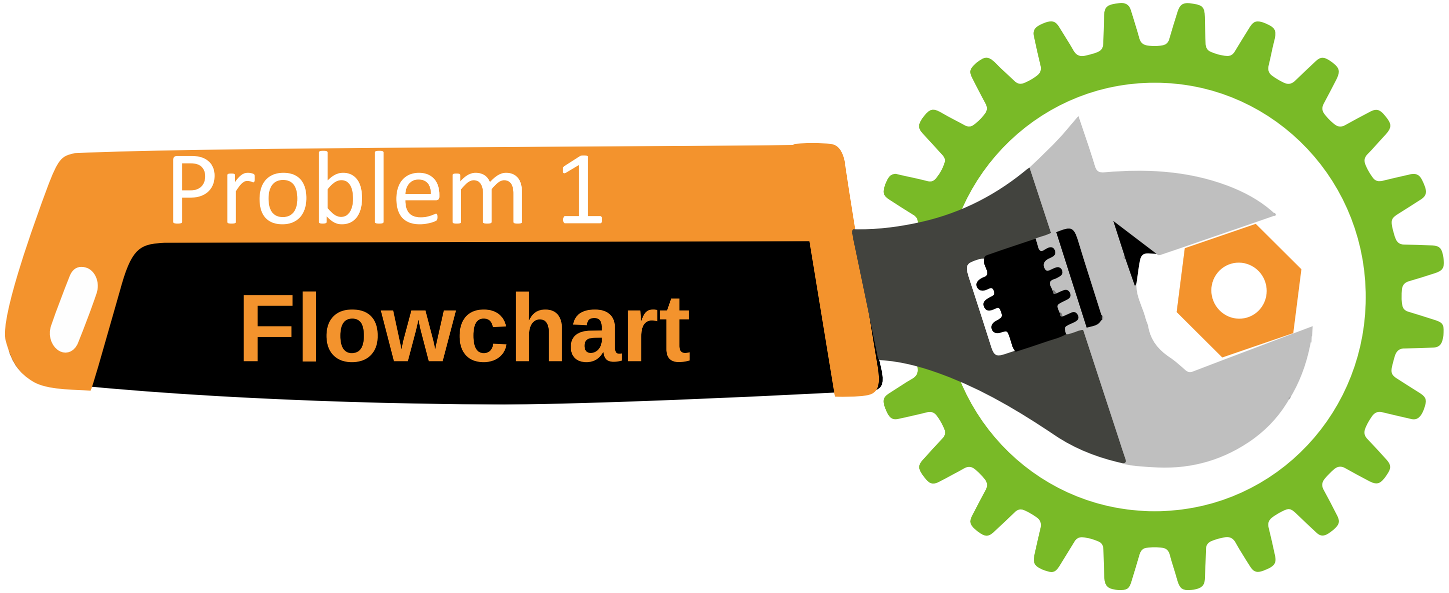
All boxes of flowchart must be connected with arrow.

All flowchart start with a Terminal or Process symbol.

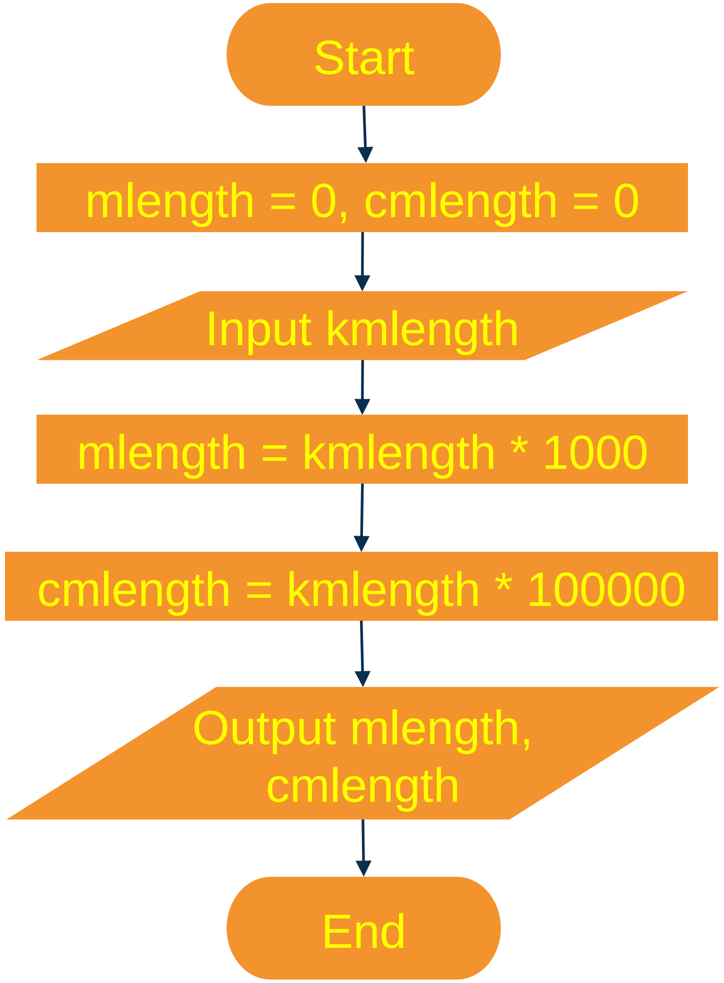
Decision symbol have 2 exit points, one for YES (TRUE) and another for NO (FALSE).

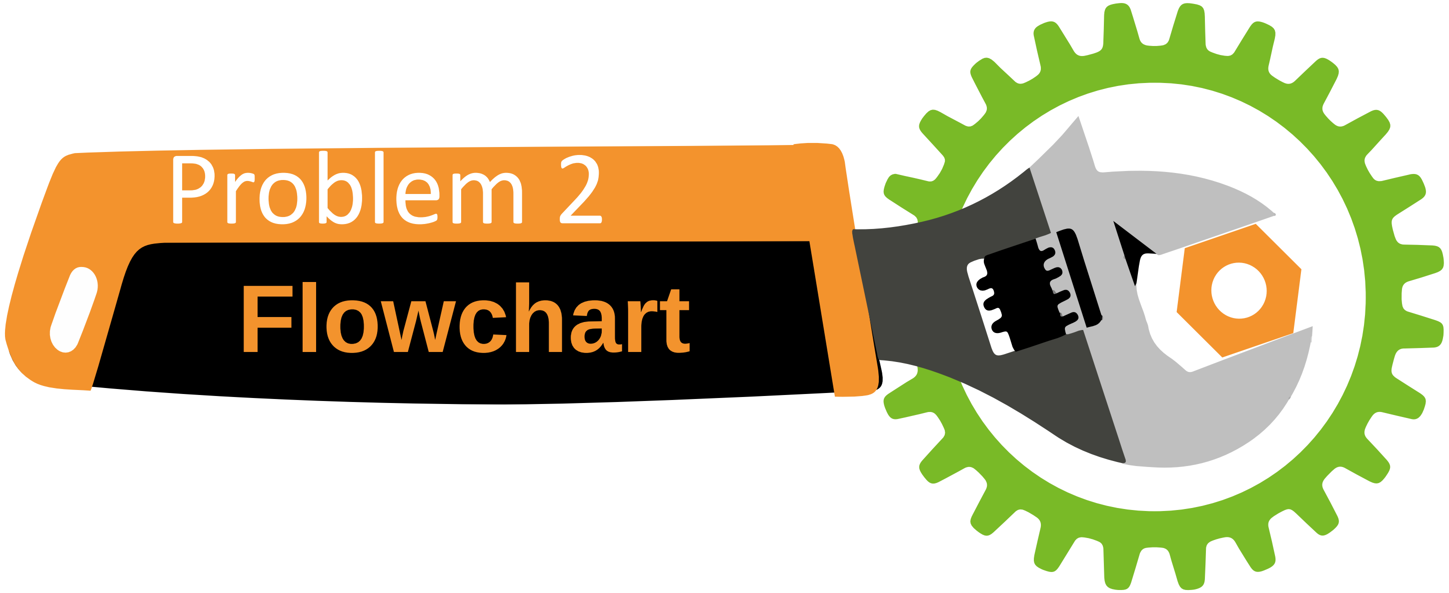
Flowchart


Start
Input num1,num2,num3
average = (num1+num2+num3)/3
Output average
End


Problem 1
Flowchart

Flowchart


Start
mlength = 0, cmlength = 0
Input kmlength
mlength = kmlength * 1000
cmlength = kmlength * 100000
Output mlength, cmlength
End


Problem 2
Flowchart

4. Test Algorithm For Correctness

One of the most important in the development of a program, and yet it is the step most often forgotten.

The main purpose of desk checking the algorithm is to identify major logic errors early, so that they may be easily corrected.

5. Code the Algorithm

Code the algorithm into a specific programming language.

6. Run the Program

Use a program compiler or interpreter and programmer-designed test data to machine-test the code for both syntax and logic errors.

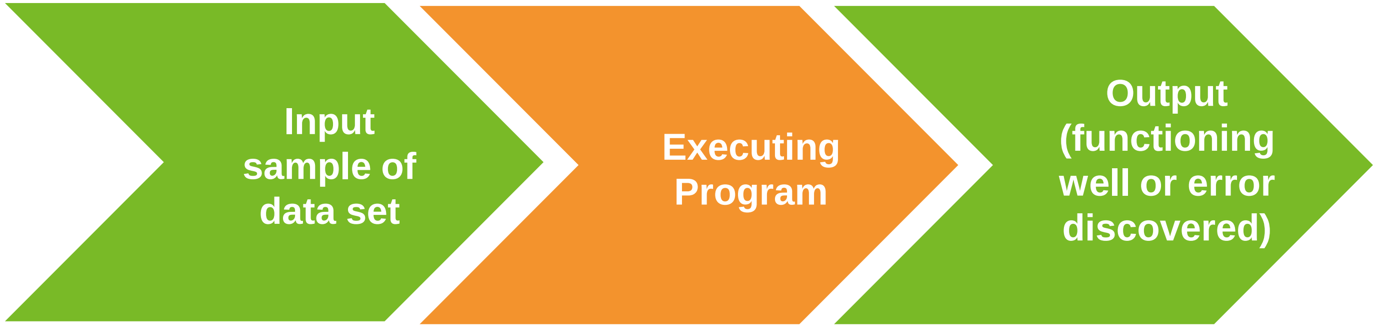
Testing and Debugging

Definition

Using a set of data to discover errors and to ensure accuracy of the program.

Testing Process

Diagram indicates the process of testing.



Testing and Debugging

TESTING

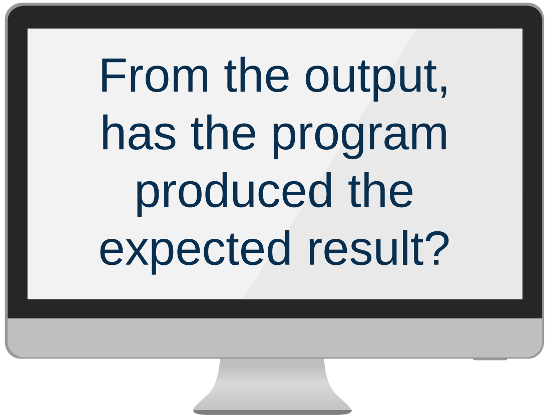
- Assume that a program to find the average of 3 numbers has been coded.
- Then, execute the program.
- Using a few numbers, test the program to verify whether the result is as expected

Testing 1:

Input: 4, 5, 8

Testing 2:

Input: 7, 8, 6



From the output, has the program produced the expected result?

Testing and Debugging

DEBUGGING

Definition

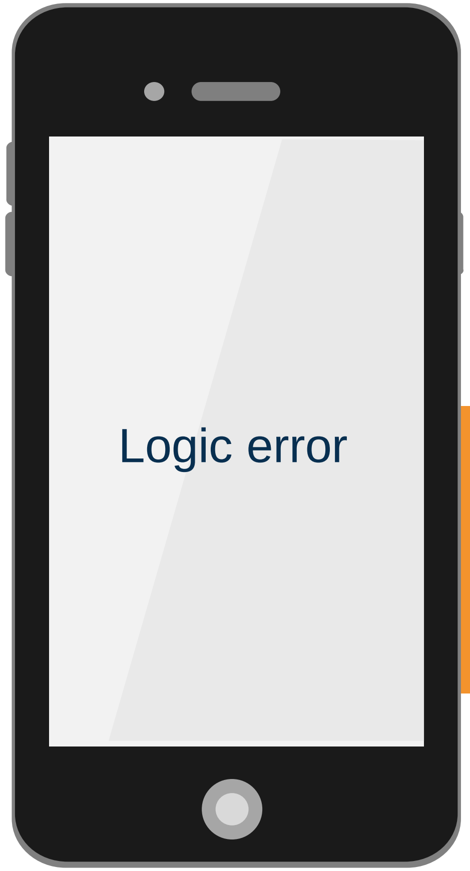
- An error is known as bug
- Debugging is a process of identifying and correcting errors.
- There are 3 types of error:
  - Syntax error - (grammatical rule violations)
  - Logic error – faulty logic
  - Runtime error – causes errors during execution.

Testing and Debugging

- Occurs when the rules of programming language are not applied.
- Correction is done during the program coding.

- The bug can be traced during the compilation process (i.e. when running the compiler on the program)
- Also known as **compile-time error**
- Must be corrected before executing and testing the program

Syntax error (grammatical error)



Logic error

- Logic error is an error that occurs because logical statement in program is wrong
- Cannot be traced by compiler.
- Corrected during the problem solving process

- Also known as **run time error**.
- Example output for average is 4, but when it runs, the output is 2. Why this is happen?

Document and Maintain the Program

Program documentation

- should not be listed as the last step
- Really an ongoing task from the initial definition of the problem to the final test result.
- Involves both external documentation (such as hierarchy charts, the solution algorithm, and test data results) and internal documentation which may have been coded in the program.

Program maintenance refers to changes which may need to be made to a program throughout its life.

Maintenance

DEFINITION

☑ Activity that verifies whether the operational system is performing as planned or an activity to modify the system to meet the current requirement.

☑ The process of changing a system after it has been applied to maintain its ability.

☑ The changes may involve simple changes such as error correcting

Maintenance

How to do maintenance?

Adjustment
Adding needs to new system.

Testing
Test the ability of the system.

Repair
For example: An old system cannot update the new data.

Measurement
Access data time. Example, time to save, print and others.

Updating
Update the database.

Replacement
Replace the old system to new system.

Exercise

- Design the pseudocode and flowchart that
  - Reads two numbers and multiplies them together and print out the result.
  - To find the average of 3 numbers
  - Tells a user that the number they entered is not a 5 or a 6.
  - Performs the following:
    - Ask a user to enter a number.
    - If the number is between 0 and 10, write the word blue.
    - If the number is between 10 and 20, write the word red.
    - if the number is between 20 and 30, write the word green.
    - If it is any other number, write that it is not a correct color option.