SEEE1022 INTRODUCTION TO SCIENTIFIC PROGRAMMING

CH1
Overview of Programming Languages

Dr. Mohd Saiful Azimi Mahmud (azimi@utm.my)
P19a-04-03-30, School of Electrical Engineering, UTM

www.utm.my
innovative • entrepreneurial • global

univteknologimalaysia utm_my utmofficial

OBJECTIVES

1. To understand and differentiate the terms of programming, algorithm, computer program and scientific programming.
2. To introduce students to several scientific programming languages.
3. To understand the basic concept of programming methods.

OVERVIEW

WHAT IS PROGRAMMING?

LET'S DEFINE A COMPUTER

- One that computes: a programmable electronic device that can store, retrieve and process data.

WHAT IS PROGRAMMING?

- Much of our behavior is characterized by logical sequences.
- E.g. Turning a page by hand
  1. Lift Hand
  2. Move hand to the right side of book
  3. Grasp top-right corner of page
  4. Move Hand from right to left until page is positioned
  5. Let go of page.

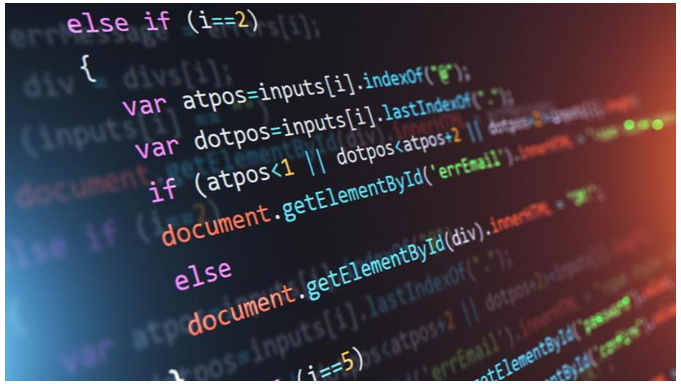
WHAT IS PROGRAMMING? (CONT.)

WHAT IS PROGRAMMING? (CONT.)

- Many things are done is a certain order of sequence. E.g. walking, baking a cake, manufacture a pen etc.
- Solving math problems are done in a sequence.
- Describing the order and sequence of operations in a process is called programming.
- In this course, we are going to learn about computer programming.

WHAT IS A COMPUTER PROGRAM?

- It is a list of sequences or steps the computer performs to solve a problem.
- The process of describing (or writing) these steps is called programming.
- The list itself is called a computer program, or just program.



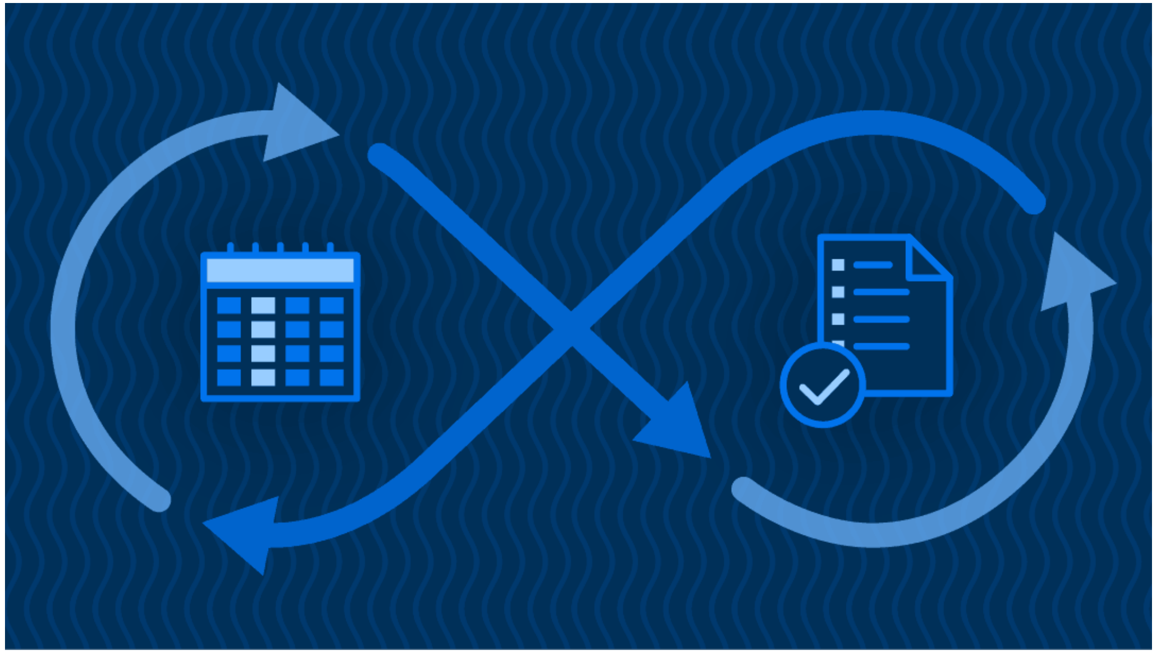
PROGRAM DEVELOPMENT

INTRODUCTION

- A computer is not intelligent – it cannot analyse a problem and come up with a solution.
- Human must analyse the problem, develop the sequence of instructions to solve the problem and describe it to the computer – We as programmers.

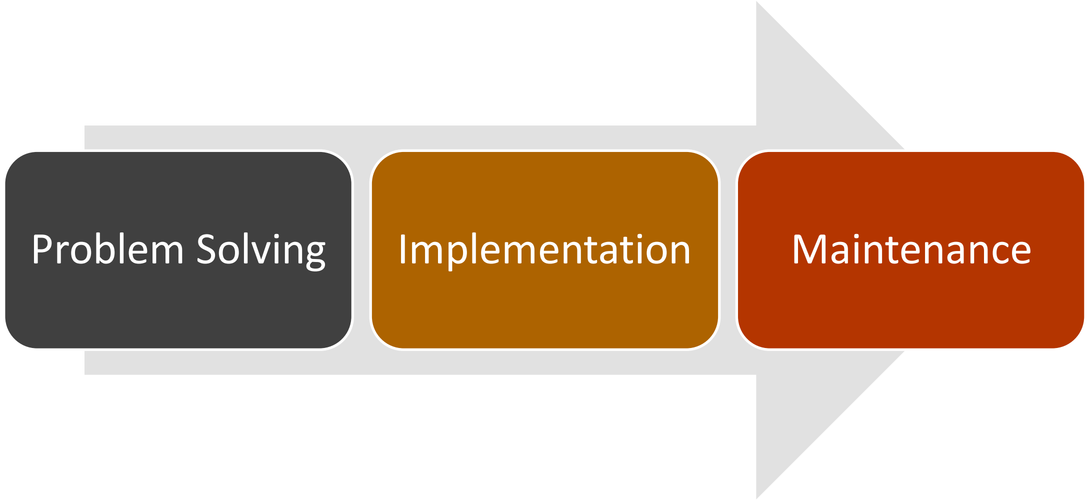
ADVANTAGES OF USING A COMPUTER?

- Computer can repeat the solution very quickly
- Can perform consistently again and again
- Frees people from repetitive, boring tasks.



PROGRAM DEVELOPMENT (CONT.)

PROGRAM DEVELOPMENT IS A THREE-PHASE PROCESS



A. PROBLEM SOLVING PHASE

1. Analysis and specification
   - Understand (define) the problem and what the solution must do
2. General solution (algorithm)
   - Develop a logical sequence of steps that solves the problem
3. Verify
   - Follow the steps exactly to see if the solution really does solve the problem

PROGRAM DEVELOPMENT (CONT.)

PROGRAM DEVELOPMENT IS A THREE-PHASE PROCESS

B. IMPLEMENTATION PHASE

1. Concrete solution (program)
   - Transfer the algorithm into a programming language
2. Test
   - Run the program, manually check the results. If there are errors, analyse the program, algorithm and determine the source of errors and make corrections.

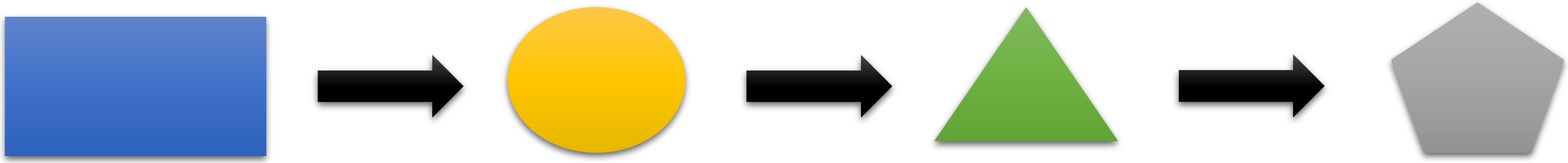
C. MAINTENANCE PHASE

1. Use
   - Use the program
2. Maintain
   - Modify the program to meet changing requirements, or the correct errors that may show up

WHAT IS AN ALGORITHM?

INFORMAL REVIEW

- Programming process is begun by first analysing the problem and developing the general solution.
- The general solution is called algorithm.
- A computer program and an algorithm look similar - because all programs are algorithms!
- Recap:
  - A program is simply an algorithm that has been written for a computer!
- An algorithm is a verbal or written description of a logical sequence of actions.
- E.g. recipes, instructions, directions



EXAMPLES OF ALGORITHMS

ALGORITHM TO START YOUR CAR

1. Insert the key
2. Depress the brake pedal
3. Make sure the transmission in Park (or neutral)
4. Turn the key to start position
5. If the engine starts within six seconds, release the key to ignition position
6. If the engine doesn’t start, release the key and gas pedal, wait for ten seconds and repeat steps 3 to 6 again but not more than five times.
7. If the car doesn’t start, call the garage.

EXAMPLES OF ALGORITHMS (CONT.)

HOW TO MAKE PEANUT AND JELLY SANDWICH

1. Put the bread, peanut butter, jelly, knife and plate onto the workspace.
2. Place two slices of bread on the plate
3. Using the knife, spread peanut butter on one slice.
4. If you want jelly, using the knife, spread jelly on the other slice.
5. Slap the two slices together, sticky side in.
6. Repeat steps 2 through 5 for each sandwich needed.
7. Eat the sandwiches

EXAMPLES OF ALGORITHMS (CONT.)

ALGORITHM FOR A SIMPLE ATM MACHINE

1. Insert bank card and get the password from the user
2. If the password is invalid, display error message and skip to step 6
3. Get the inputs:
    1. Get the transaction type (deposit or withdrawal) and the amount from the user
    2. Get the current balance from the bank
4. If the transaction is withdrawal, check the current balance:
5. If amount is greater than the current balance, show error message and skip to step 6
6. If amount is equal or less, subtract the amount from the current balance
7. Output error message, or the cash, and the current balance.
8. As the user whether to repeat steps 3 through 6 for another transaction?

WHAT IS AN ALGORITHM?

FORMAL DEFINITION

CAN YOU FIGURE OUT FROM THE PREVIOUS EXAMPLE?

An algorithm is an **ordered set** of **unambiguous, executable steps** that **defines a terminating process**

CHARACTERISTICS OF AN ALGORITHM

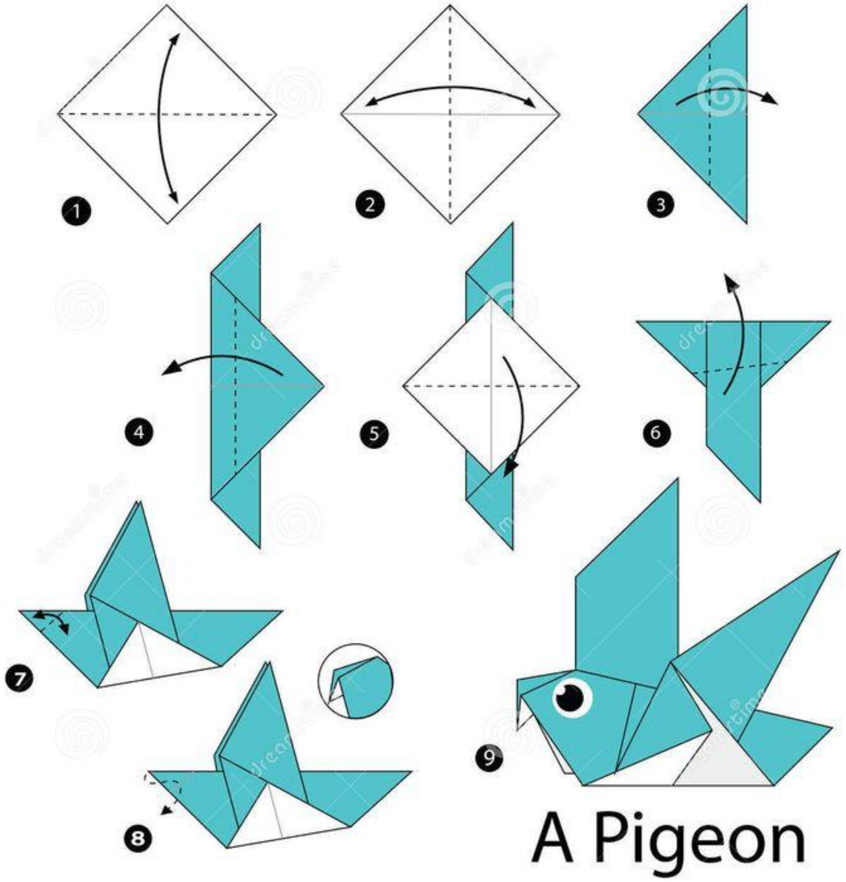
ALGORITHM CHARACTERISTICS

1. Each and every instruction should be precise and unambiguous.
2. An algorithm should have finite number of steps.
3. An algorithm should produce correct result.
4. An algorithm should not use particular programming language.

History: Algorithm was first proposed by Al-Khawarizmi, Persian mathematician, in the 9th century!

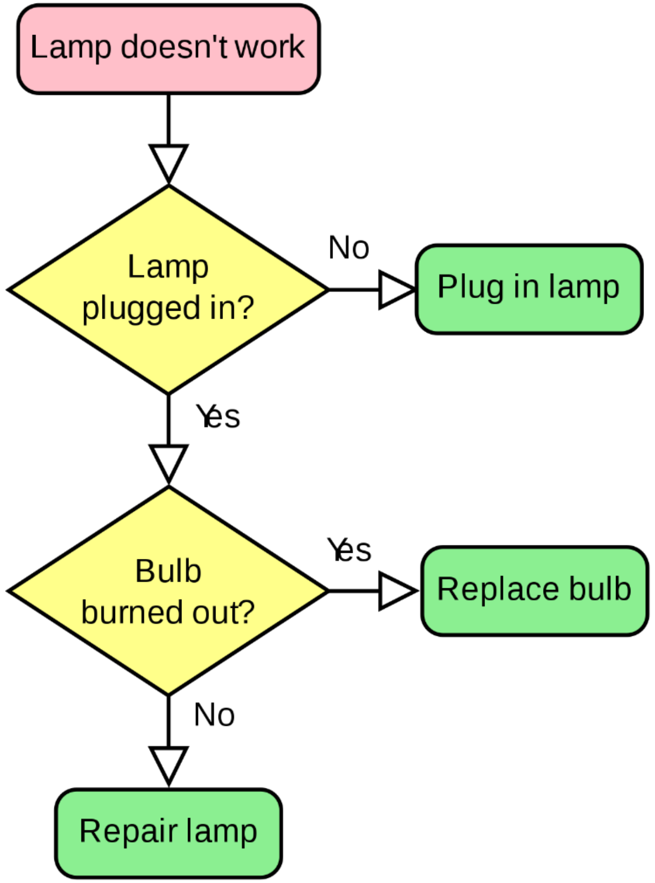
HOW ALGORITHM ARE DESCRIBED?

THROUGH STEP-BY STEP DIAGRAMS



HOW ALGORITHM ARE DESCRIBED?

THROUGH FLOWCHARTS



HOW ALGORITHM ARE DESCRIBED?

THROUGH LIST OF INSTRUCTIONS OR STEPS

1. Insert the key
2. Depress the brake pedal
3. Make sure the transmission in in Park (or neutral)
4. Turn the key to start position
5. If the engine starts within six seconds, release the key to ignition position
6. If the engine doesn't start, release the key and gas pedal, wait for ten seconds and repeat steps 3 to 6 again but not more than five times.
7. If the car doesn't start, call the garage.

HOW ALGORITHM ARE DESCRIBED?

THROUGH PSEUDOCODE

```
procedure IntegerDivision
input : n, d
output: result
begin
    $result \leftarrow 0$
    while $n \geq d$
        $n \leftarrow n - d$
        $result \leftarrow result + 1$
    end
    return result
end
```

ASPECTS OF COMPUTERS AND THEIR OPERATION

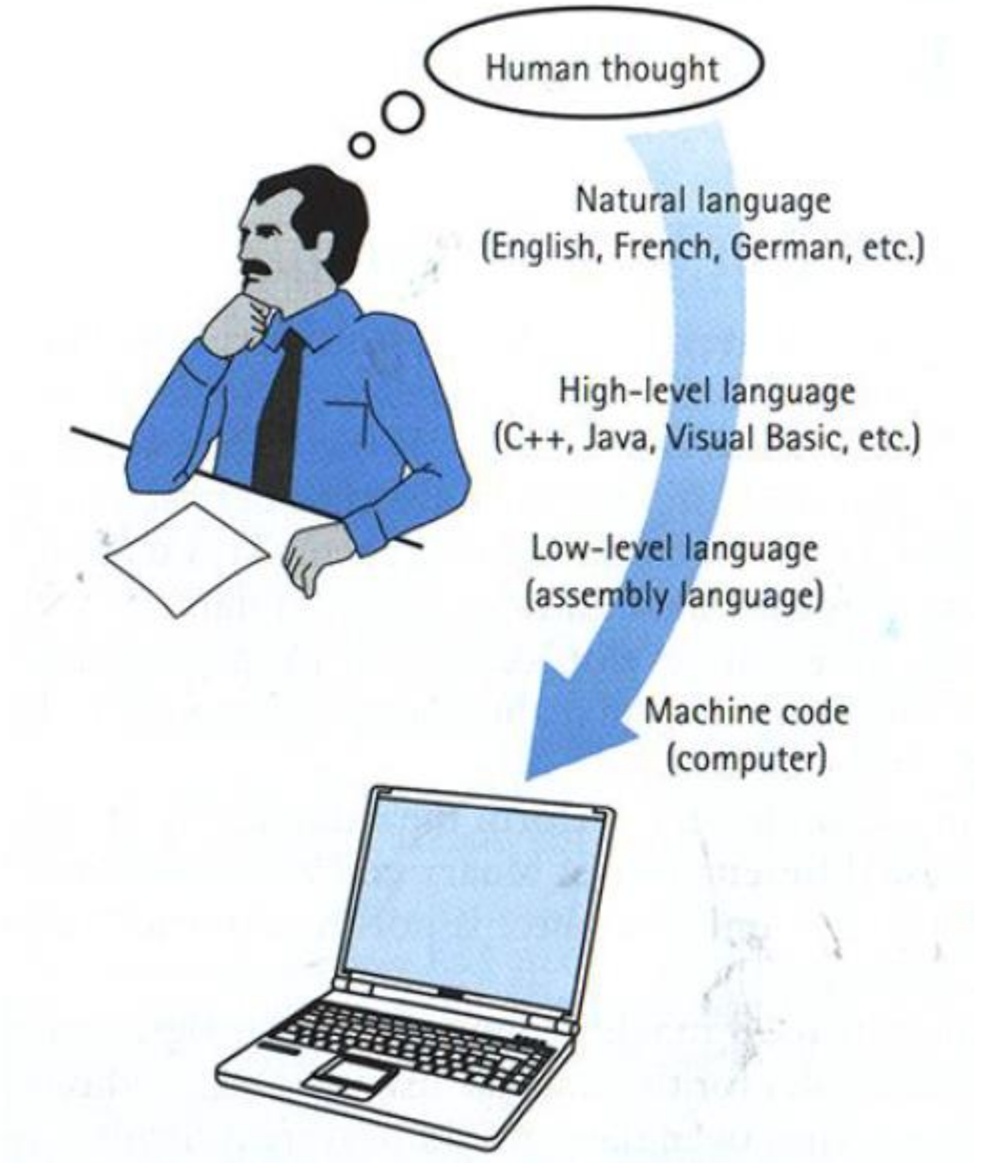
- A computer typically has memory, processing unit, and input and output.
- Memory – Store information and program (RAM, Hard drive, flash drive etc.)
- Processing unit – Execute (run) the program and produce results (CPU, GPU)
- Input/Output – Obtain input from user and produce output (keyboard, mouse, printer, screen)

PROGRAMMING LANGUAGES

PROGRAMMING LANGUAGES

DEFINITION

- A programming language is a simplified form of English with math symbols that adheres to strict set of grammatical rules.
- e.g. C, C++, Python, Javascript, Java



SCIENTIFIC PROGRAMMING LANGUAGES

DEFINITION

- Is a programming language optimized for the use of mathematical formulas and matrices.
- Thus it is more suitable for performing scientific/numerical computing
- Math operations can be performed with any programming language, however it is easier to be done in a scientific programming language.
- E.g. MATLAB, Mathematica, FORTRAN

MATLAB

INTRODUCTION

- Originally a simple language for matrix arithmetic
- Can now do most numerical scientific calculations
- Very heavily used for scientific/numerical computing
- Matlab has lots of specialist toolboxes
- Cost is high
- Free version available – Octave, Scilab
- Benefit : Easier than other languages, easy to prototype

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ \vdots \\ m \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & \dots & n \end{array} \\ \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ a_{31} & a_{32} & \dots & a_{3n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix} \end{array}$$

GNU Octave

Scilab

PYTHON

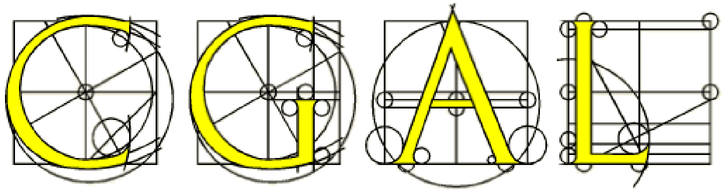
INTRODUCTION

- A very simple, high-level interpreted language
- Much easier and better engineered than most It traps most user errors, including numeric ones
- Best for scripting, system interfaces
- Scientific programming needs numpy
- Unclear whether numerically robust or how reliable
- Benefit: As easy as MATLAB. If you're not doing numeric computing, then Python is better

python NumPy

INTRODUCTION

- Originally for C programmers to a higher level
- Designed for functionality more than error prevention
- Not really very good for scientific programming
- Language is very complicated, and hard to learn well
- Needs a lot of third-party libraries to do scientific computing, e.g. Boost, CGAL etc.
- Benefit: Fast, high efficiency, can create own data structures



FORTRAN : FORMULA TRANSLATION

INTRODUCTION

- Simpler than C++
- Comparable power to C++
- One of the oldest language.
- Benefit : Easy to code in parallel, portable, high efficiency, support for matrices

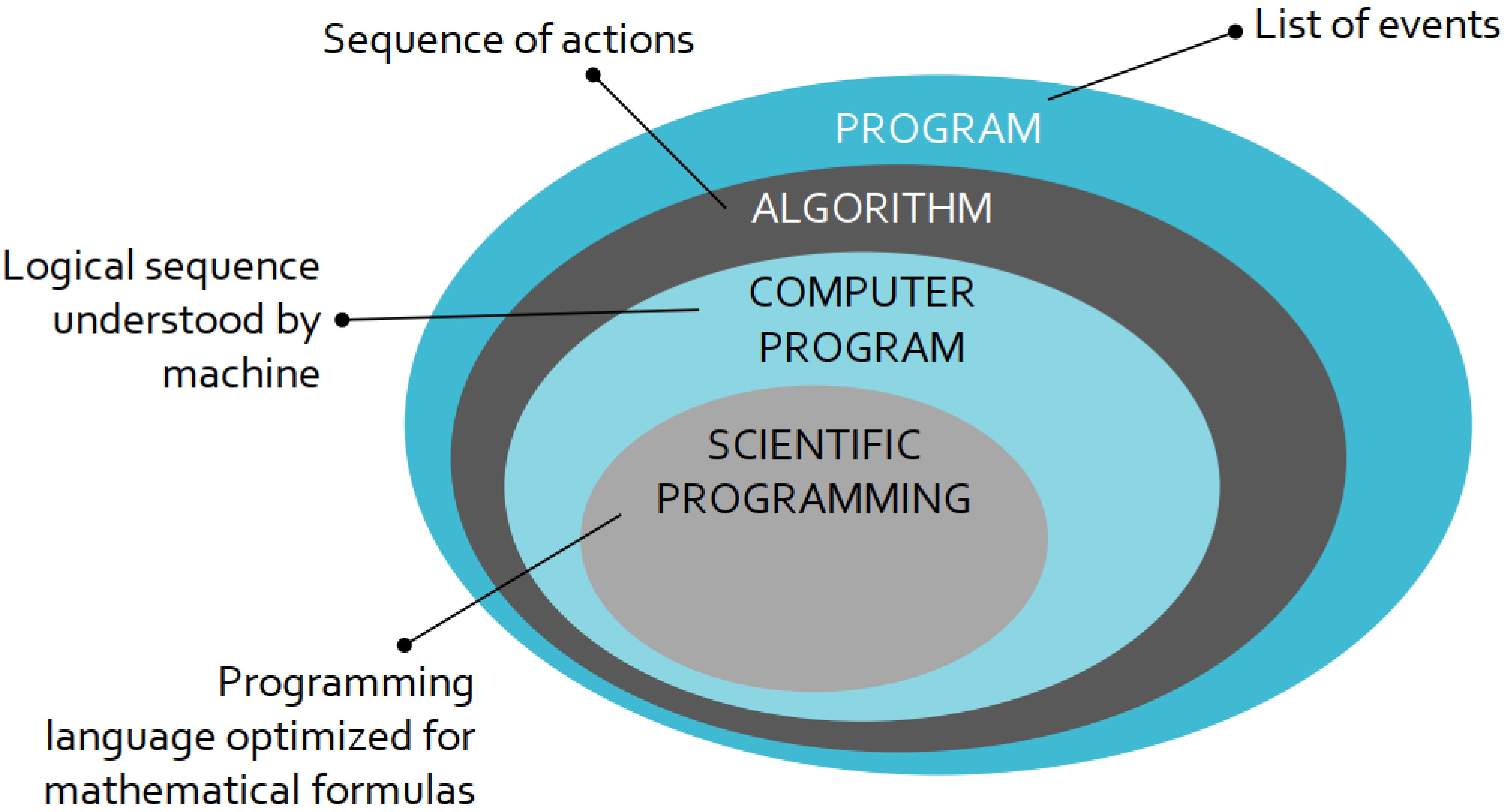
Fortran

ADVANTAGES/DISADVANTAGES

HOW TO CHOOSE?

- Ease of use : Python > MATLAB >Fortran
- Prototyping: MATLAB > Python
- Debuggability: Python > MATLAB > Fortran
- Performance : Fortran > C++ > Python > MATLAB
- Parallelism : Fortran
- Array Handling : Fortran > MATLAB > Python
- Text handling : Python >> C++ > Fortran
- Computer Science : C++ > Python > Fortran
- System interfaces : Python > C++ > Fortran

SUMMARY

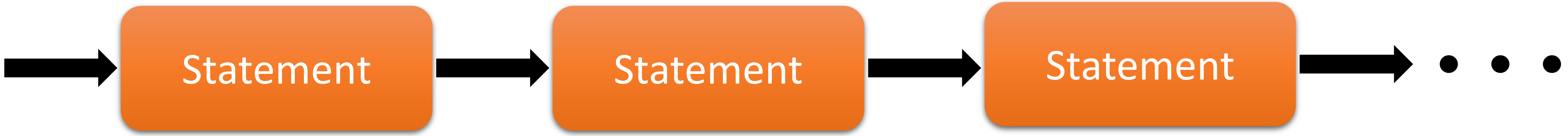

List of events
Sequence of actions
PROGRAM
ALGORITHM
Logical sequence understood by machine
COMPUTER PROGRAM
SCIENTIFIC PROGRAMMING
Programming language optimized for mathematical formulas

UTM

PROGRAMMING BASIC

BASIC OF PROGRAMMING

SEQUENCE

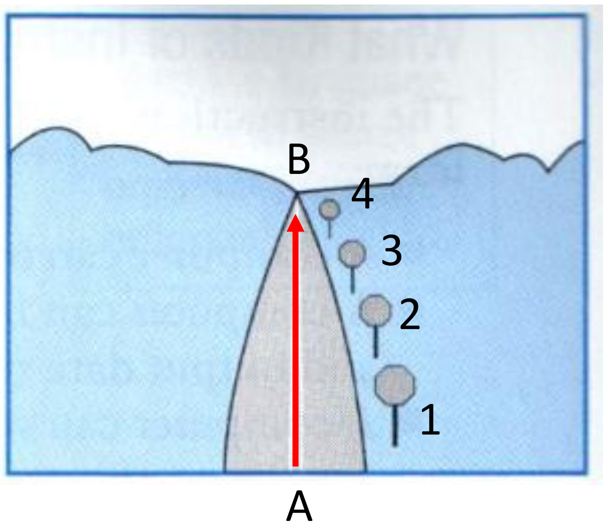


Statement

Statement

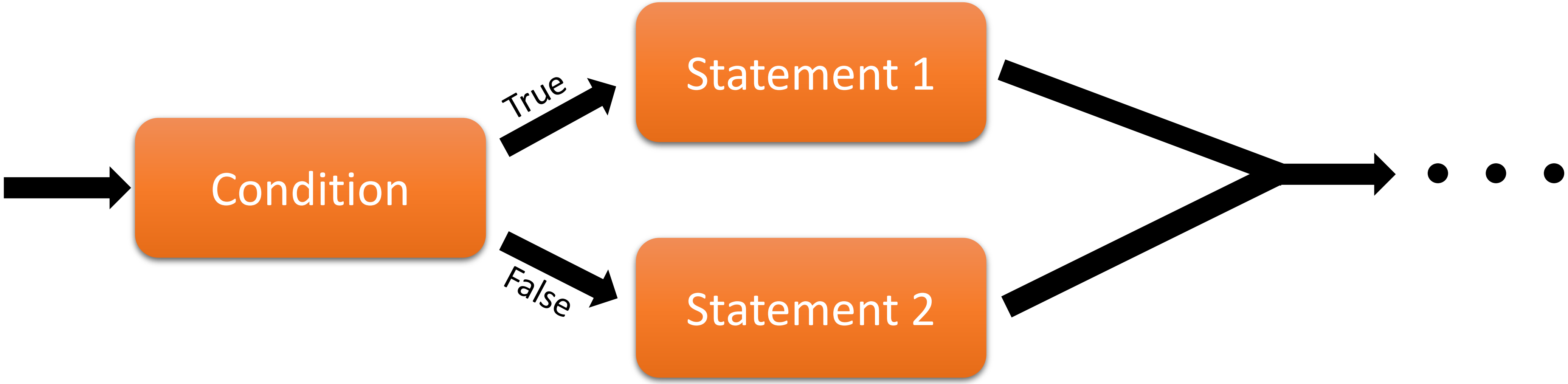
Statement

• • •



BASIC OF PROGRAMMING (CONT.)

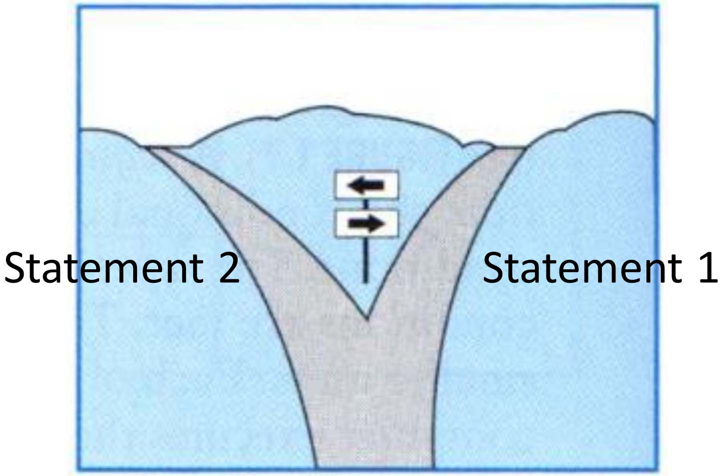
BRANCH/DECISION



SELECTION (also called branch or decision)

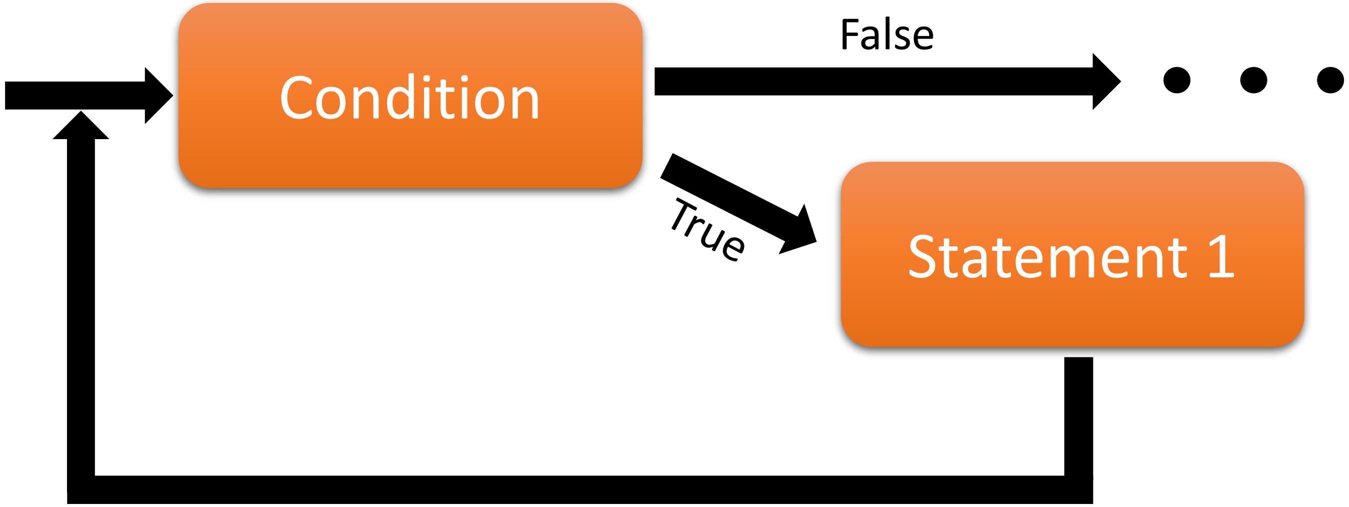
IF condition THEN statement 1

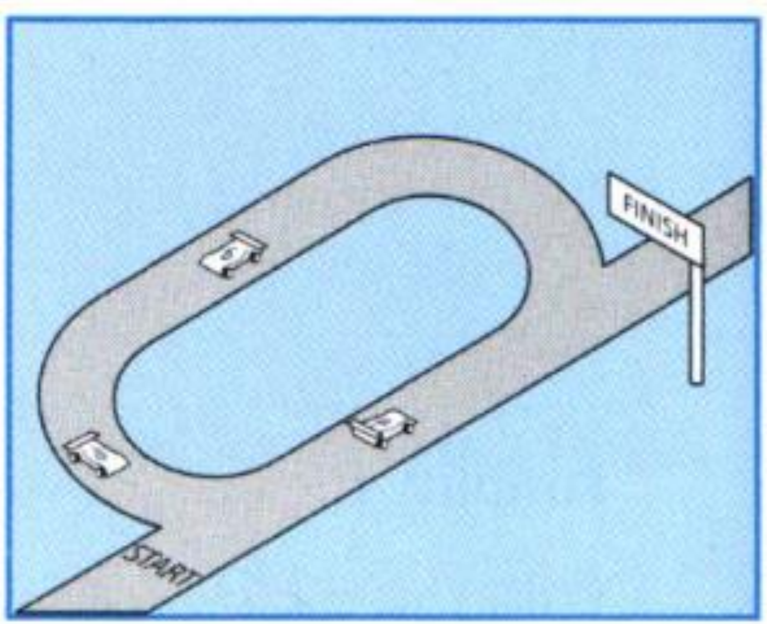
ELSE statement 2



BASIC OF PROGRAMMING (CONT.)

LOOP



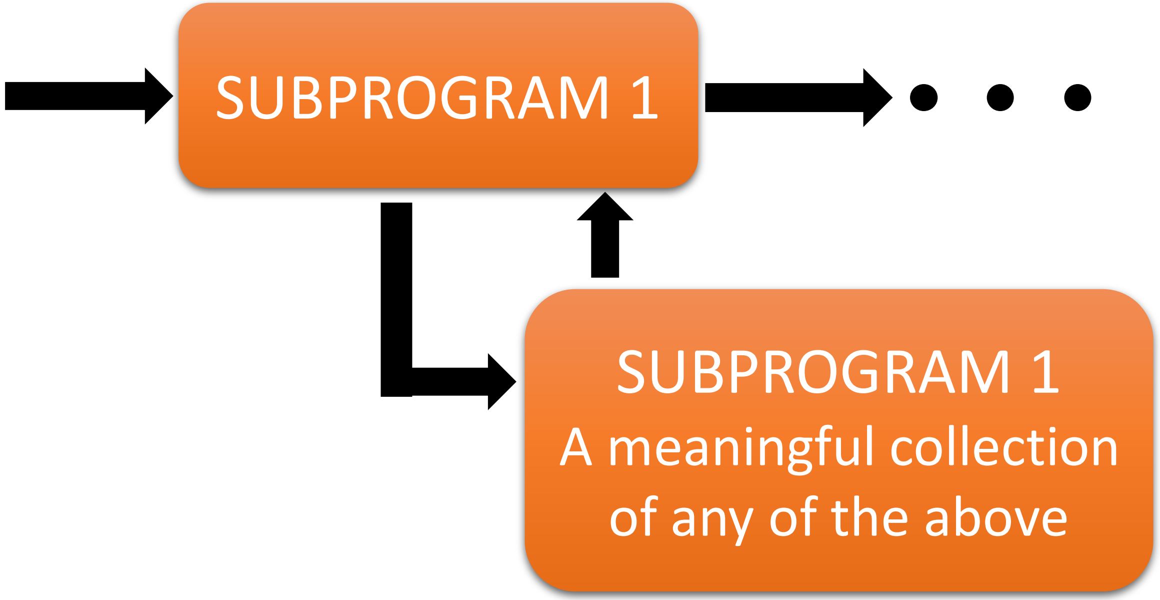


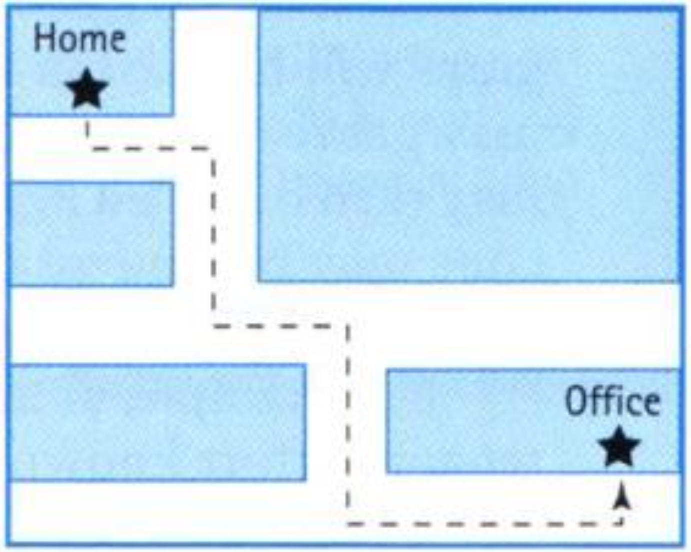
LOOP (also called repetition or iteration)

WHILE condition DO statement 1

BASIC OF PROGRAMMING (CONT.)

SUBPROGRAM





SUBPROGRAM (also called procedure, function, method or subroutine)

FLOWCHART

INTRODUCTION

- Flowchart is a graphical representation of program solving steps so that it makes clear about the program.
- Types of flowchart are:

  a) System Flowchart

  System flowchart gives complete processing mechanism and cannot be converted into program

  Example: Organizational structure of a college.

  b) Program Flowchart

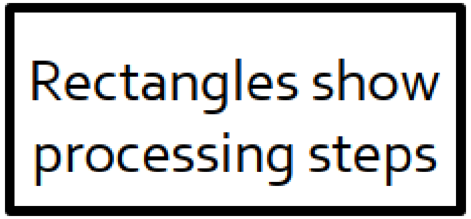
  Program flowchart gives the problem solving method and can be converted into program.

  Example: Problem to calculate area of triangle.

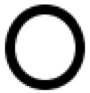
History: Concepts of flowchart was given by John Von Neuman in 1945!

FLOWCHART (CONT.)

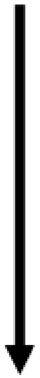
FLOWCHART STRUCTURE



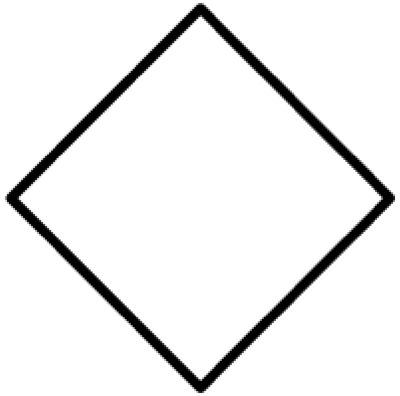
Rectangles show processing steps



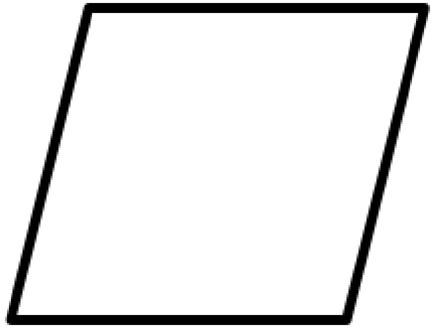
Circles as connectors



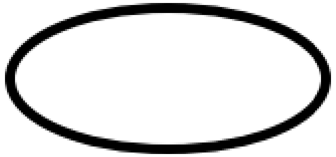
Arrows for connectors



Diamonds indicate points of branch/decision



Parallelograms show input and output steps



Oval for start/finish

FLOWCHART (CONT.)

PROPERTIES OF GOOD FLOWCHART

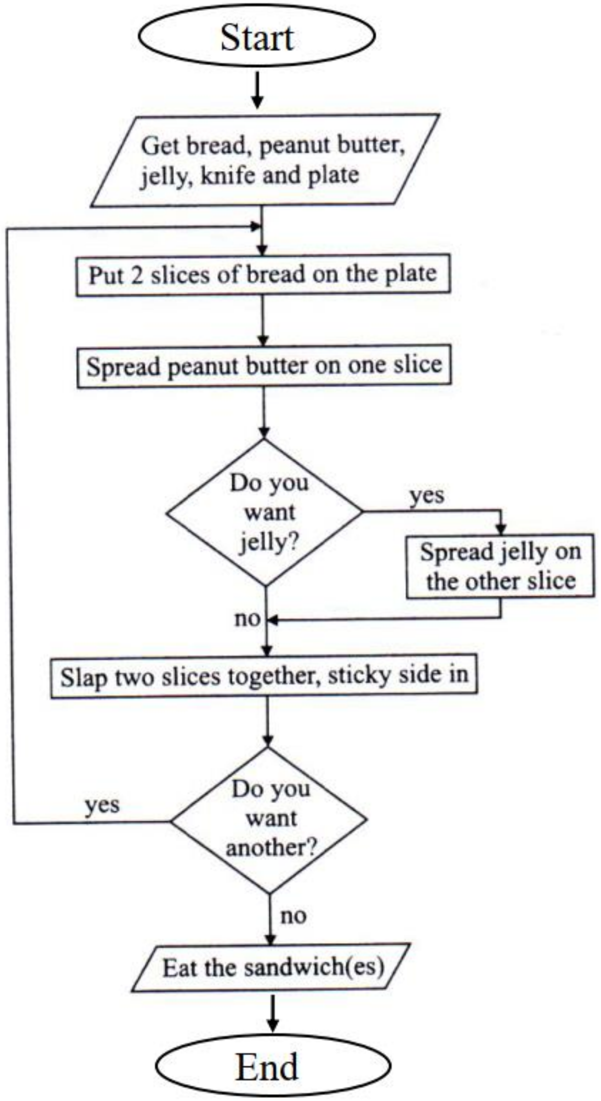
1. The flowchart should have only **one start and ending points**.
2. Flow lines should **not intersect each other**.
3. Flowchart should **not contain the programming language**.
4. The flowchart should be neat and clear for the user.

DISDVANTAGES OF FLOWCHART

1. It takes long time to prepare to proper flowchart.
2. Translation of flowchart into program is sometimes difficult. (that’s why pseudocode is preferred)

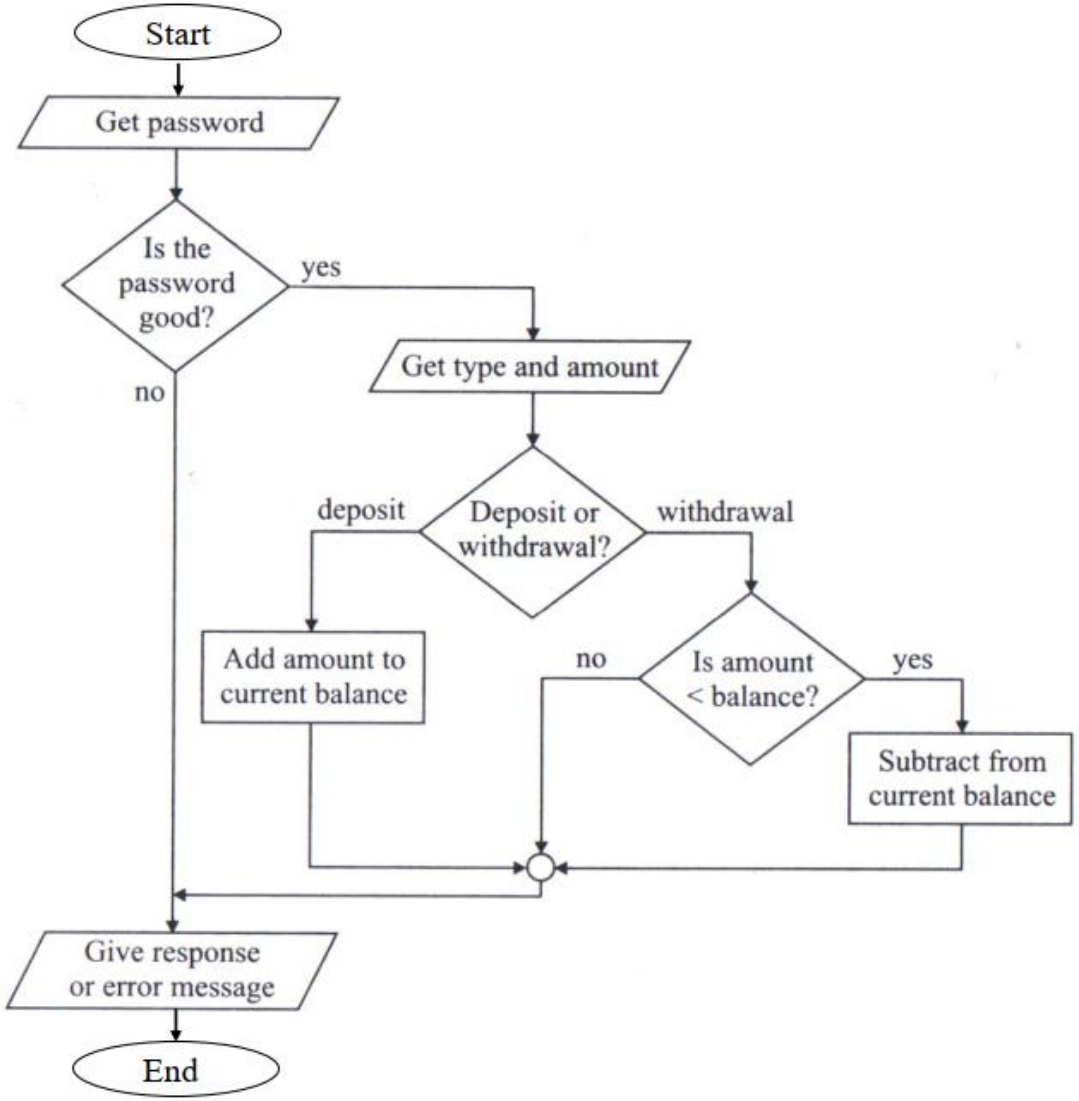
FLOWCHART (CONT.)

EXAMPLE OF FLOWCHART : SANDWICH


Start
Get bread, peanut butter, jelly, knife and plate
Put 2 slices of bread on the plate
Spread peanut butter on one slice
Do you want jelly?
yes
Spread jelly on the other slice
no
Slap two slices together, sticky side in
Do you want another?
yes
no
Eat the sandwich(es)
End

FLOWCHART (CONT.)

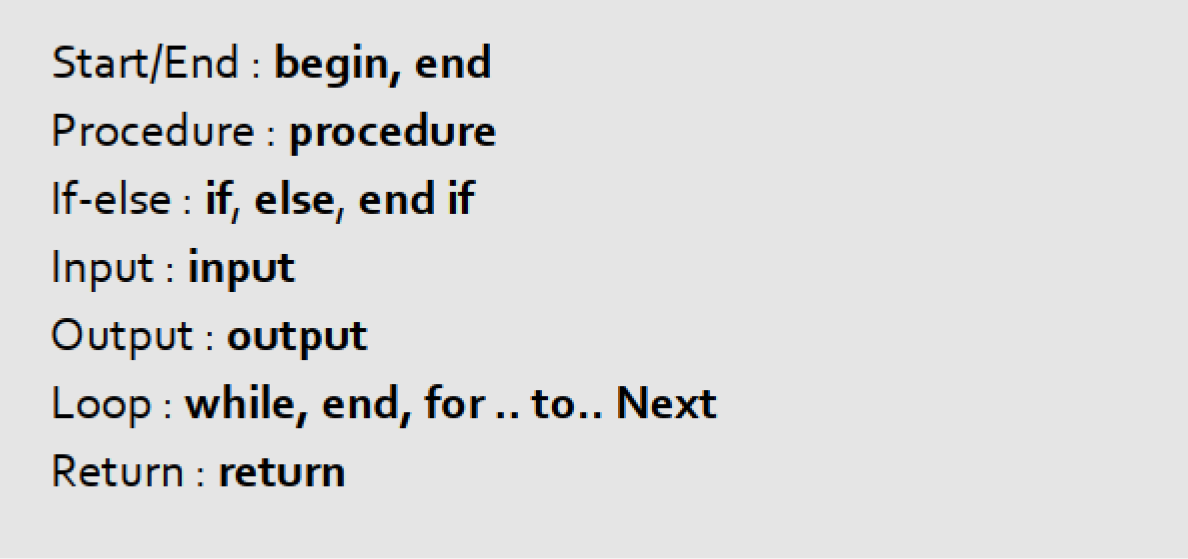
EXAMPLE OF FLOWCHART : ATM



PSEUDOCODE

INTRODUCTION

- Is a notational system in which ideas can be expressed informally during algorithm development.
- Loose representation of formal language (e.g. C++ etc.)
- This is used when target programming language unknown.
- We will use the following notations;



Start/End : **begin, end**

Procedure : **procedure**

If-else : **if, else, end if**

Input : **input**

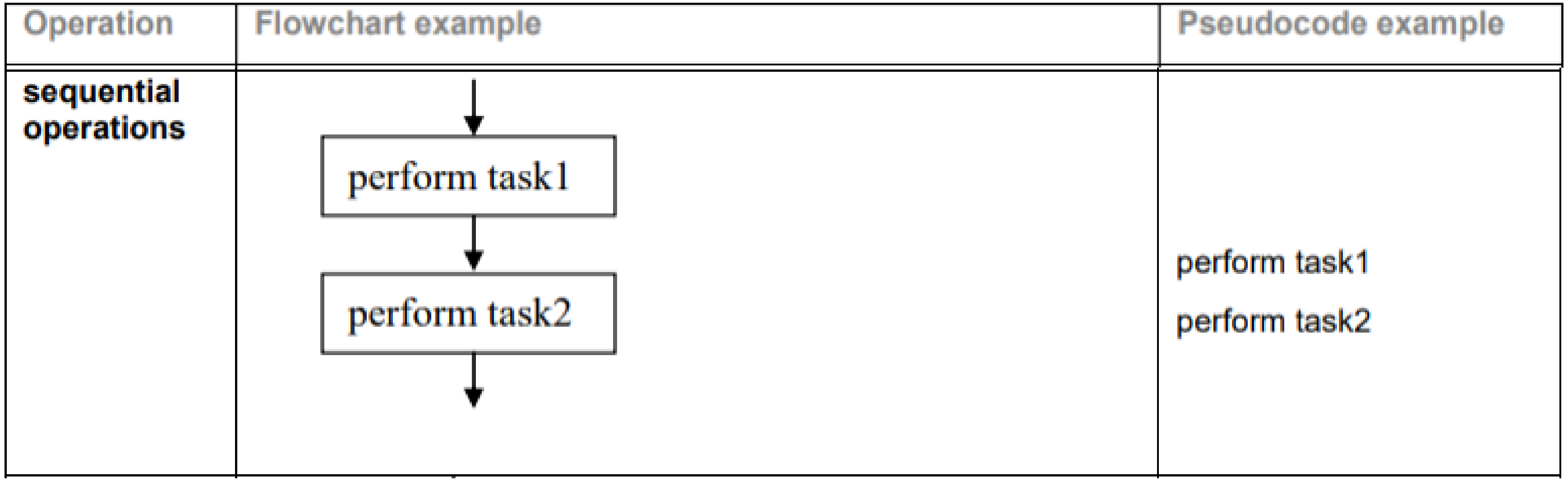
Output : **output**

Loop : **while, end, for .. to.. Next**

Return : **return**

PSEUDOCODE (CONT.)

EXAMPLE (SEQUENTIAL OPERATIONS)

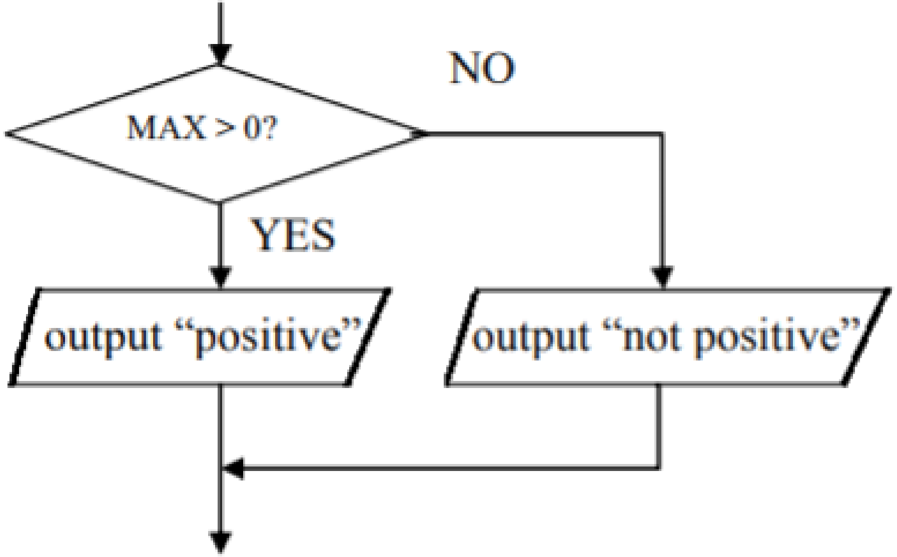


Operation	Flowchart example	Pseudocode example
sequential operations	perform task1 perform task2	perform task1 perform task2

PSEUDOCODE (CONT.)

EXAMPLE (CONDITIONAL OPERATIONS)

conditional operations

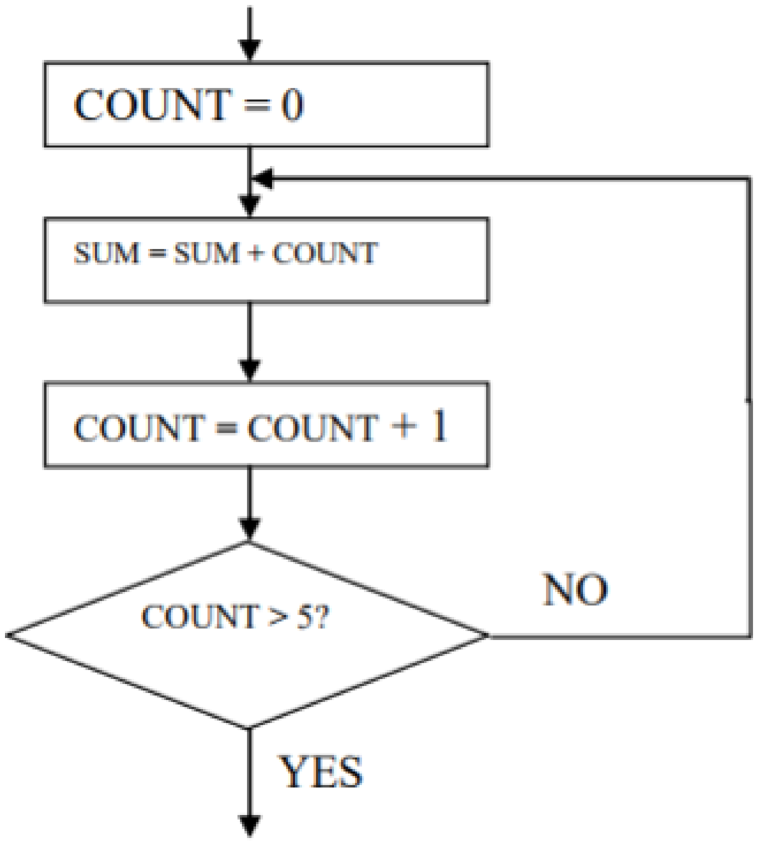


```
if MAX > 0 then
    output "positive"
else
    output "not positive"
end if
```

PSEUDOCODE (CONT.)

EXAMPLE (FOR LOOP)

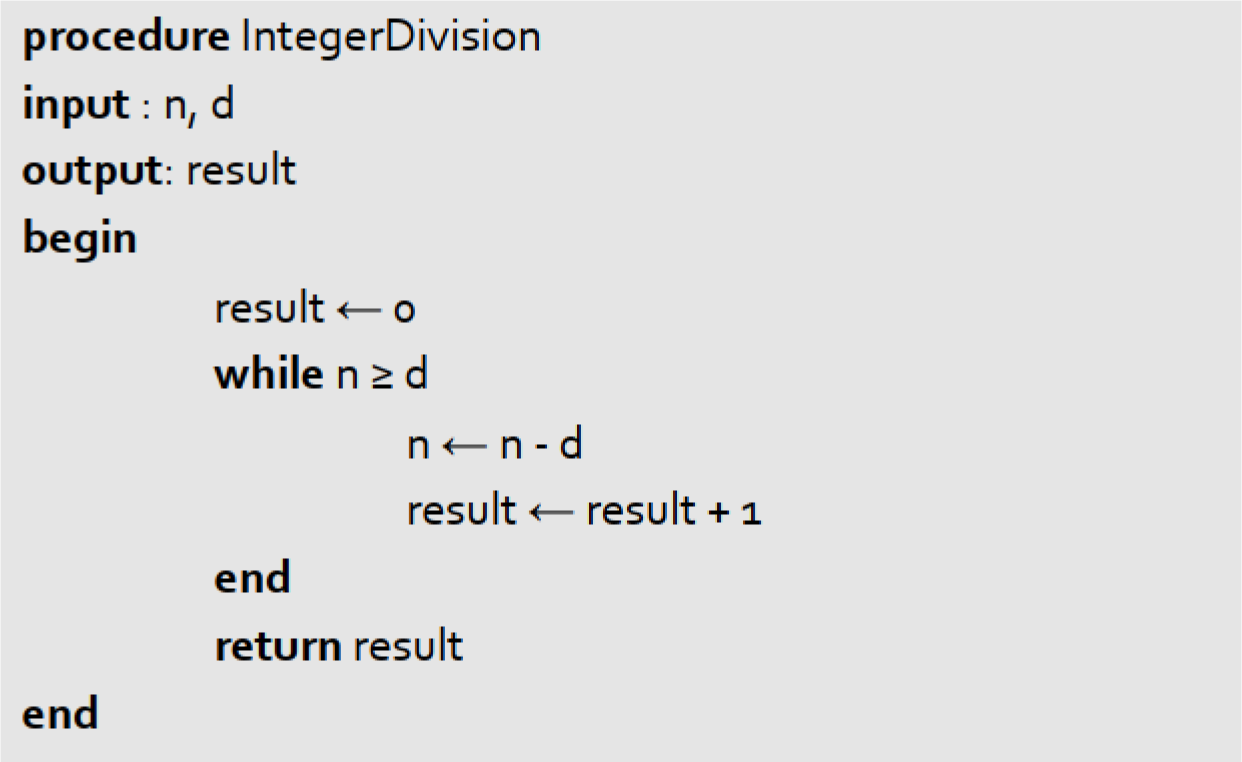
Operation	Flowchart example	Pseudocode example

for loop	 COUNT = 0
	SUM = SUM + COUNT
	COUNT = COUNT + 1
	COUNT > 5?
	NO
	YES
		for count = 0 to 5
		  SUM ← SUM + count
		next count

PSEUDOCODE (CONT.)

ASSESSEMENT

- Given the following pseudocode, draw the flowchart?



```
procedure IntegerDivision
input : n, d
output: result
begin
        result ← 0
        while n ≥ d
                n ← n - d
                result ← result + 1
        end
        return result
end
```

REFERENCES

- Programming and Problem Solving with C++, 6th Edition, Nell Dale and Chip Weems, 2014, Jones & Bartlett Learning

UTM
UNIVERSITI TEKNOLOGI MALAYSIA

univteknologimalaysia utm_my utmofficial

Thank You

www.utm.my

innovative ● entrepreneurial ● global