PROGRAMMING LIFE CYCLE

INTRODUCTION

Computer Programming is the art of making a computer do what you want it to do.

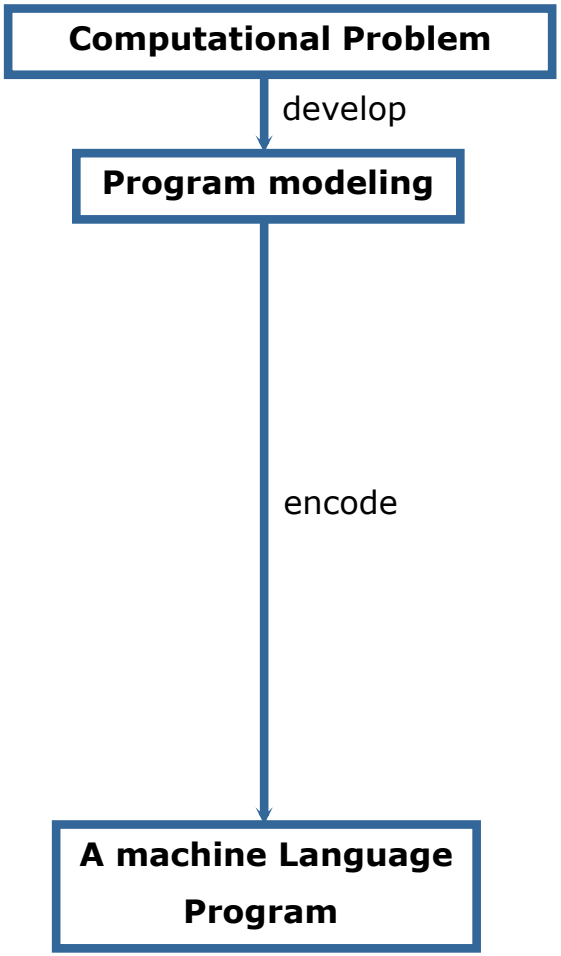
✓ At the very fundamental level, it consists of issuing a sequence of commands to computers to achieve an objective.

✓ A computer program is simply a set of instructions computers follow to perform a specific task.

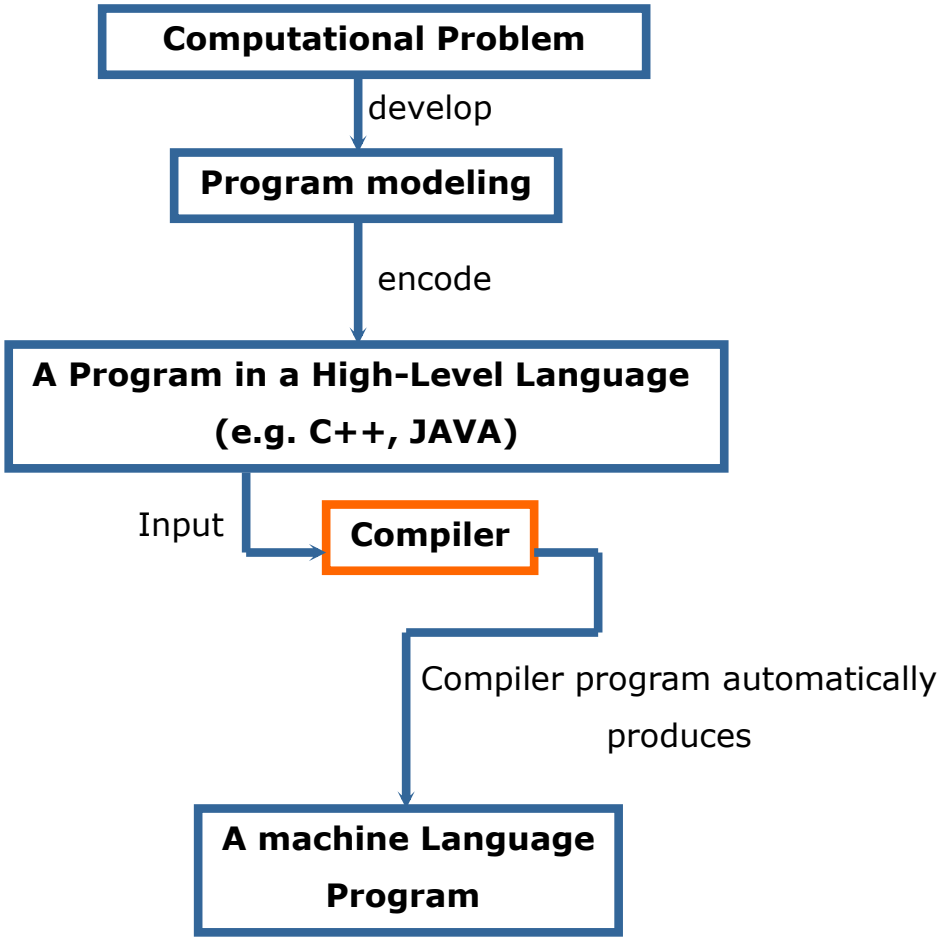
✓ A program to computers is similar to a recipe to cooks to make a dish.

✓ It describes the ingredients (Data/Input) and the sequence of steps (Process) needed to convert the ingredients into a dish of food (Result/Output).

HOW CAN WE PROGRAM OUR COMPUTER MACHINE



Software Development
Using machine language
(Low-Level language)



Software Development
Using a High-Level language

PROGRAMMING LIFE CYCLE

The development of the system/program

- The general steps programmers go through to develop the system

1. Problem Analysis
2. Program Modeling
3. Coding and Program's source code
4. Testing and debugging
5. Documentation
6. Maintenance

PROGRAMMING LIFE CYCLE

Problem Analysis

1. Human

- Problem solving is the logical process of identifying a problem and find out its solution. Users (as human) use intuition, experience, and knowledge to solve the problems.
- The following scenario illustrates how humans find a solution to a problem

PROGRAMMING LIFE CYCLE

Problem Analysis

EXAMPLE:

Problem: *Who get the highest score in class?*

Data/Input: **i. Student name**

ii. score

Table 1: Midterm Score

ID.	STUDENT NAME	SCORE
1	Ahmad	62
2	Zulkifli	60
3	Suraya	65
4	Siti	61

PROGRAMMING LIFE CYCLE

Problem Analysis

Process:

1. Look at the table
2. Find the maximum score by scanning, now you get **65**
3. Look at the left side and you know the student name **Suraya**.
4. Answer **Suraya**

Result/Output: *Suraya*

PROGRAMMING LIFE CYCLE

Problem Analysis

- However, if the given data is as below:

Table 2: Other Midterm Score

NO.	STUDENT NAME	SCORE
1	Anuar	62
2	Zakaria	60
3	Salmi	62
4	Norliza	61
5	Shamsudin	59
6	Khairul	65
7	Sazali	62
8	Milah	65
9	Maria	66
10	Emy	62
11	Johari	65
12	Azizan	63

PROGRAMMING LIFE CYCLE

Problem Analysis

- By the same processes above, you need to spend more time to find out that **Maria** has the highest score of **66**.

PROGRAMMING LIFE CYCLE

Problem Analysis

- You might be in a worse scenario with the given data below:

Table 3: Midterm Score for Entire Grade

NO.	STUDENT NAME	SCORE
1	Anuar	62
2	Zakaria	60
3	Salmi	62
4	Norliza	61
5	Shamsudin	59
6	Khairul	65
7	Sazali	62
8	Milah	65
9	Maria	66
10	Emy	62
11	Johari	65
12	Azizan	63
.	.	.
.	.	.
.	.	.
.	.	.
100	Zaki	60

PROGRAMMING LIFE CYCLE

Problem Analysis

- Once the table size becomes bigger and bigger, you may need more concentration, with additional tool: notepad and pencil, or calculator to solve this problem. This example exhibits one of the human limitations to work with a large volume of data. It is impractical for humans to solve the massive and complex problem without using tools.

PROGRAMMING LIFE CYCLE

Problem Analysis

2. **Computer**

- Computers consist of two main components: hardware and software. These two components work collaboratively to carry out tasks users required to accomplish. Hardware is controlled by software, and software is written by humans; therefore, hardware is controlled by humans.

PROGRAMMING LIFE CYCLE

Problem Analysis

- To make computers work effectively, humans (in this case, programmers) must think logically in order to solve the problem.
- Problem Analysis consists of **input, process** and **output**

PROGRAMMING LIFE CYCLE

Problem Analysis

- Define the purpose of the program

- Identify the problem
    i. Input
    ii. Process
    iii. Output

PROGRAMMING LIFE CYCLE

Problem Analysis

- Input data: a list of the source data provided to the problem.

  - Gather input data;
    - i. from keyboard
    - ii. from files or disk drives

  - Define input data type;
    - i. Number
    - ii. Character

PROGRAMMING LIFE CYCLE

Problem Analysis

- Process: a list of actions needed to produce the required output.

  - Manipulate data to perform information

  - Process the input data

  - It is about steps to be taken in order to get the output

PROGRAMMING LIFE CYCLE

Problem Analysis

- Output: a list of the outputs required.
  - Information

  - Display the results as output;
    - i. Send it to the screen
    - ii. Write to a file

PROGRAMMING LIFE CYCLE

Example 1:

- Problem:

  - Create a program to read 10 numbers from keyboard. Program will calculate the average of that numbers. Print the number and the average.

PROGRAMMING LIFE CYCLE

Example 1:

- Problem Analysis:

1. INPUT: 10 number

2. PROCESS: Calculate the average by using the formula; average=total/10

3. OUTPUT: 10 number and average

PROGRAMMING LIFE CYCLE

Example 2:

- Problem: **Add three numbers**

A program is required to read three numbers from a user, add them together, and print their total on the screen.

PROGRAMMING LIFE CYCLE

Example 2:

By looking at the problem above, you know that the program needs three numbers to be the input and the output is the sum total.

- Problem Analysis:
  1. INPUT: number1, number2, number3

  2. PROCESS: add numbers together

  3. OUTPUT: total

PROGRAMMING LIFE CYCLE

Example 3:

- Problem:

- Grading system has a following policy:

Score from 0 – 39; the grade is “F”

Score from 40 – 49; the grade is “D”

Score from 50 – 64; the grade is “C”

Score from 65 – 79; the grade is “B”

Score from 80 - 100; the grade is “A”

The program accepts “Score” input (score must be 0 – 100), then displays the grade that the student earned.

PROGRAMMING LIFE CYCLE

Example 3:

- Problem Analysis:

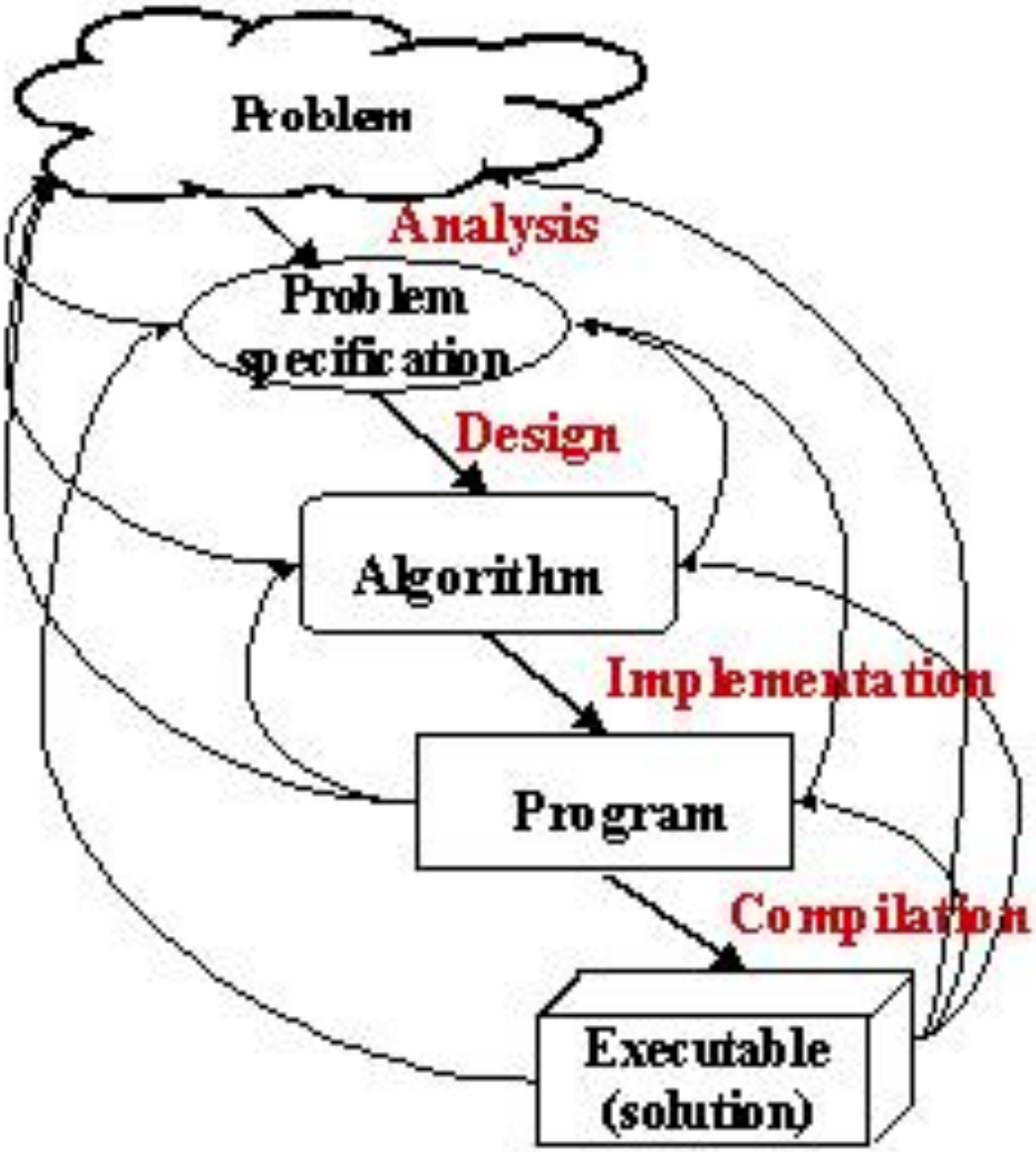
1. INPUT: score
2. PROCESS: get the grade for the score base on policy
3. OUTPUT: grade

PROGRAMMING LIFE CYCLE

Exercise :

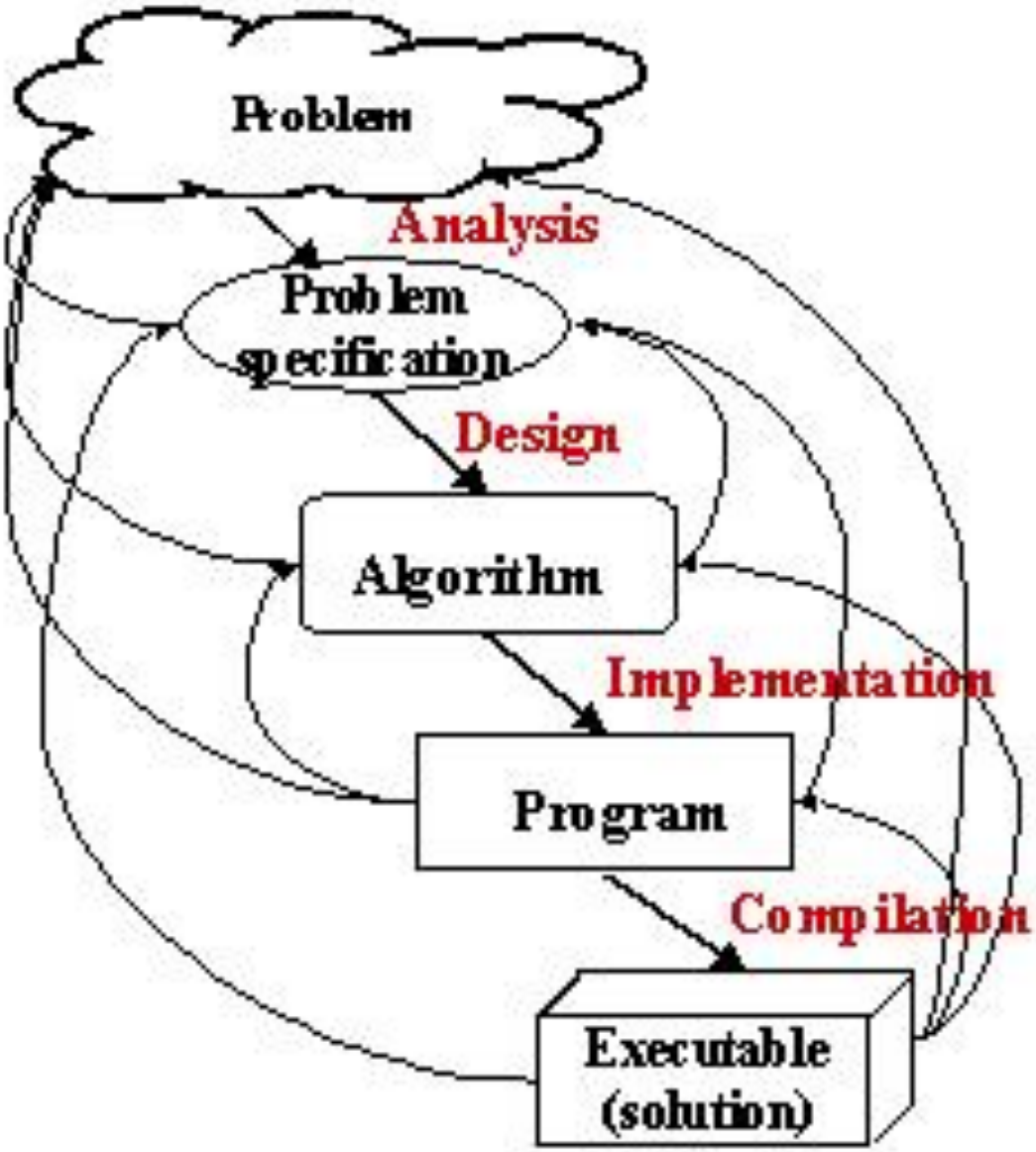
1. The factorial formula is “n! = 1 x 2 x 3 x … x n” (where n is any whole number)Write the program that accepts the input “n” and output the factorial value.
2. Find average score of “n” students in a class. The program will accept the input “n” from the user.
3. Find the minimum score of 10 students in a class.
4. Finding the total of 1 + 2 + 3 + 4 + 5 + … + n (Where n can be any whole number).Program will accept “n” from the user.

PROGRAMMING LIFE CYCLE



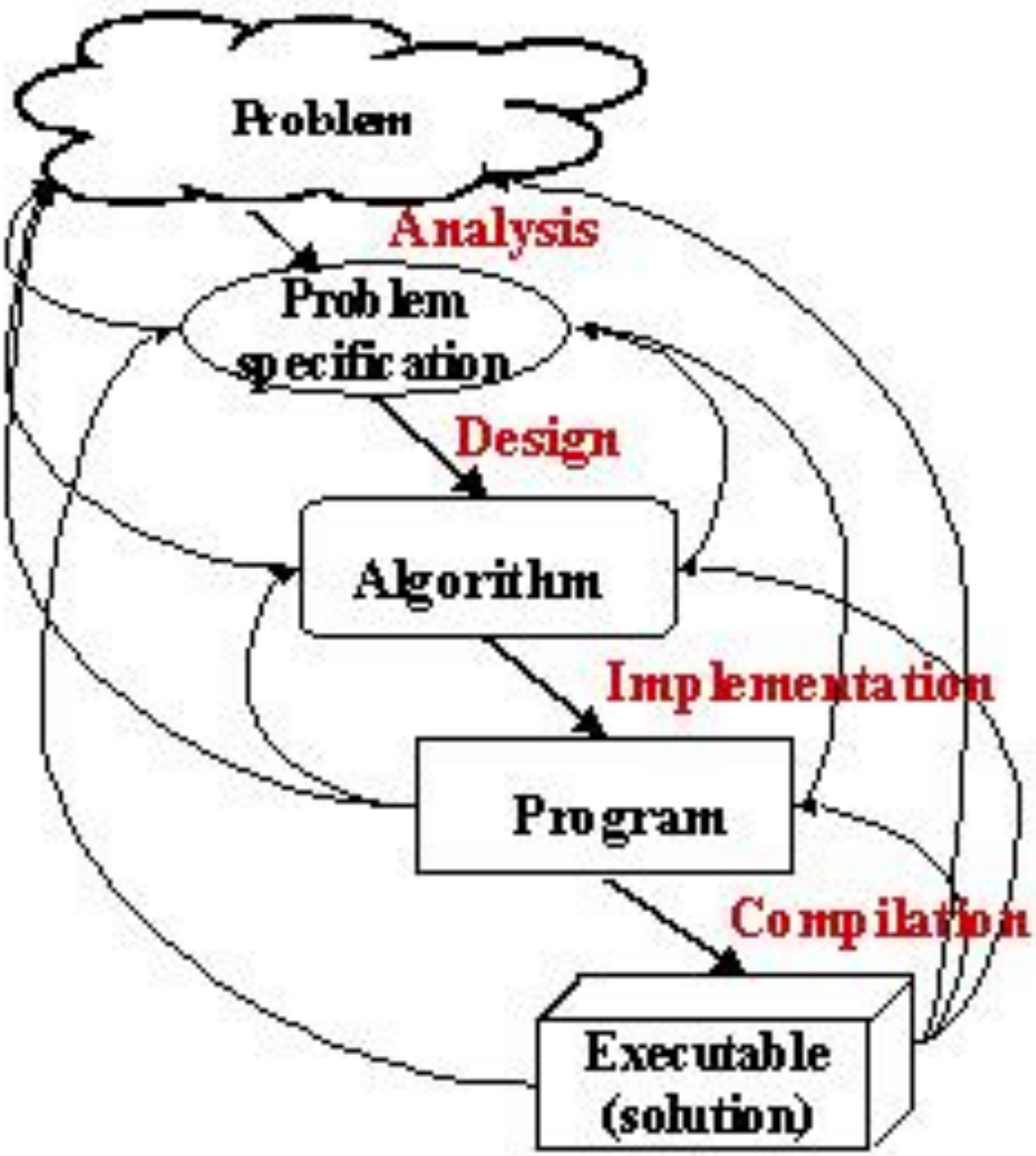
Problem:
Teacher, I want to go home

PROGRAMMING LIFE CYCLE



Problem Analysis:
Input;
student particular
Process;
Felo give approval
Output;
approval

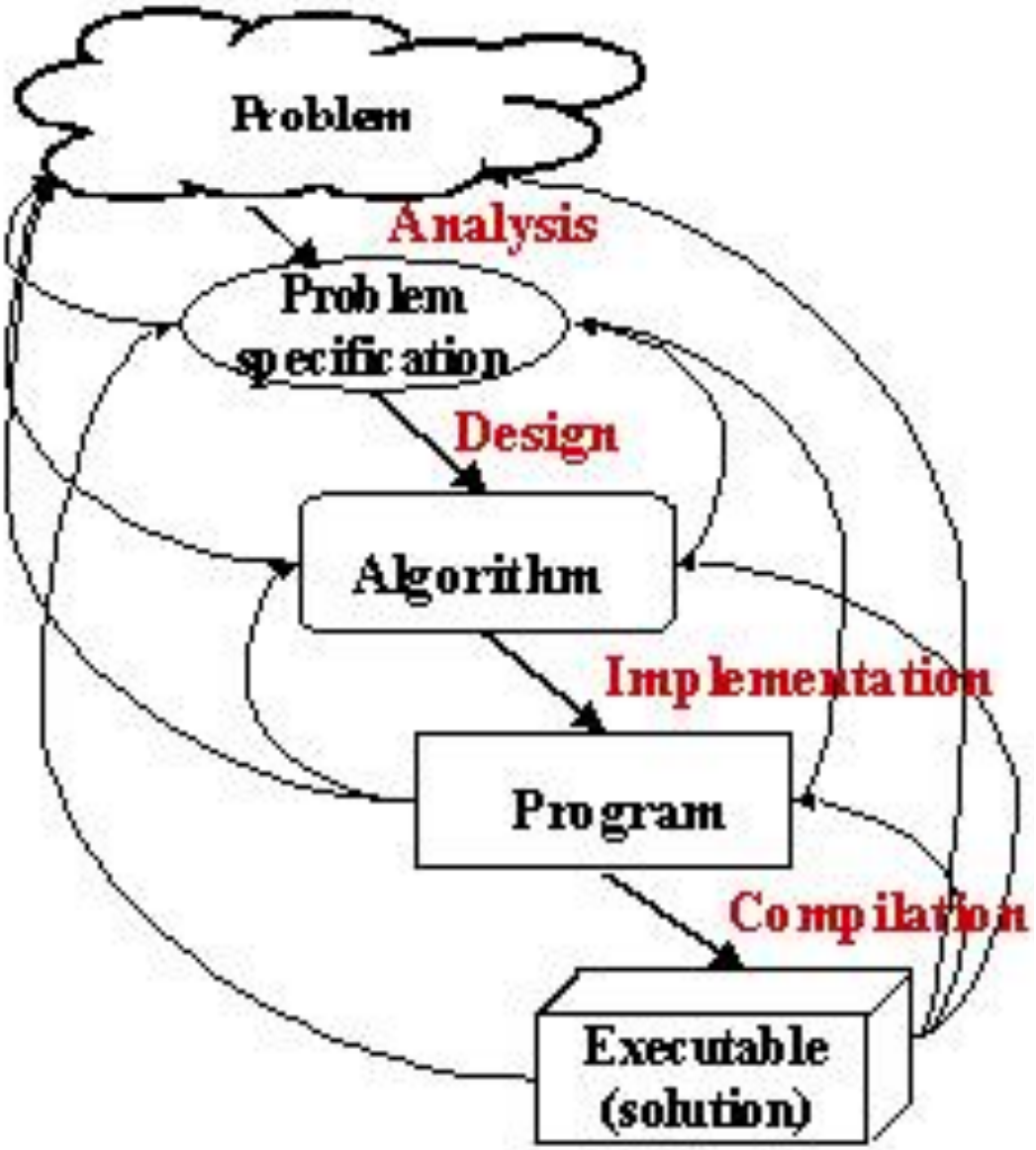
PROGRAMMING LIFE CYCLE



Algorithm:

1. Start
2. Input student particular
3. Felo give the approval
4. If yes, approve
5. If not, not approve
6. Student get the approval
7. End

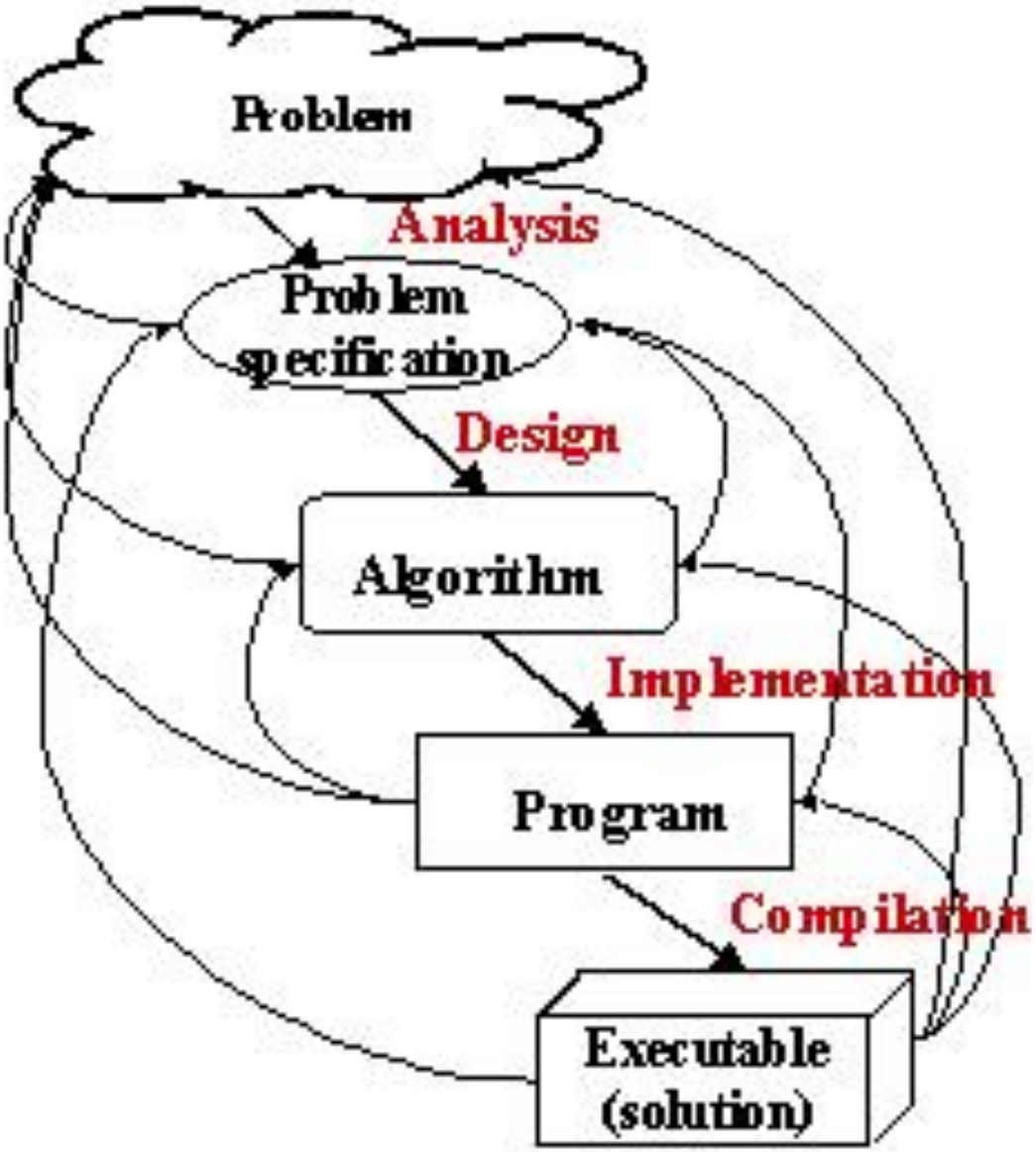
PROGRAMMING LIFE CYCLE



Program:

```
#include <stdio.h>
Main()
{
  char name[];
  char…….;
  cin>>name;
  cin>>…..;
  if …….;
  else ……..;
  cout<<approval;
  return 0;
 }
```

PROGRAMMING LIFE CYCLE



Compilation:

0100000100111111
0000000011111101
0101000101010110
1111101010101010
0110100100101001
1110111010010110
1110101010101010
0101010101010010
1010111010010101
0101010100101011
0101010100101010

PROGRAMMING LIFE CYCLE

Program Modeling

✓ creating model of solution.

✓ **algorithm**, the step-by-step instruction to solve a problem/accomplish a task, then the program will be written based on the algorithm using a programming language.

✓There are many ways to represent an algorithm; however two techniques commonly used are **pseudo code** and **flowchart.**

PROGRAMMING LIFE CYCLE

Program Modeling

1. Pseudo code

   Pseudo code is Structured English used to represent an algorithm. It is easy for both users and programmers to read and write algorithms regardless of the programming experience. There is no standard pseudo code but we should follow some conventional rules below:

   i. Statements are written in simple English.

   ii. Each statement is written on a separate line.

PROGRAMMING LIFE CYCLE

Program Modeling

iii. Statements are starting with the text START.
iv. Statements are ending with the text END.

Example:

1. START
2. *Statement1*
3. *Statement2*
4. *Statement3*
5. END

PROGRAMMING LIFE CYCLE

Example 1:

- Problem:

  - Create a program to read 10 numbers from keyboard. Program will calculate the average of that numbers. Print the number and the average.

PROGRAMMING LIFE CYCLE

Example 1:

- Problem Analysis:

1. INPUT: 10 number

2. PROCESS: Calculate the average by using the formula; average=total/10

3. OUTPUT: 10 number and average

PROGRAMMING LIFE CYCLE

Example 1:

- Program Modeling: Pseudo code

1. START
2. Declare a name for every value to be input -NO1, NO2, NO3, NO4, NO5, NO6, NO7, NO8, NO9, NO10,
Declare a name for the value of sum -TOTAL
Declare a name for a value of average -AVERAGE
Set all = 0
3. INPUT 10 numbers
4. Calculate the total by using the formula; TOTAL=NO1+NO2+… NO10

PROGRAMMING LIFE CYCLE

Example 1:

- Program Modeling: Pseudo code

5. Calculate the average by using the formula; AVERAGE=TOTAL/10
6. PRINT 10 numbers and AVERAGE
7. END

PROGRAMMING LIFE CYCLE

Example 2:

- Problem: **Add three numbers**

A program is required to read three numbers from a user, add them together, and print their total on the screen.

PROGRAMMING LIFE CYCLE

Example 2:

- Problem Analysis:

    1. INPUT: number1, number2, number3

    2. PROCESS: add numbers together

    3. OUTPUT: total

PROGRAMMING LIFE CYCLE

Example 2:

- Program Modeling: Pseudo Code

1. START
2. Declare num1, num2, num3 and Total
   Set num1, num2, num3 and Total = 0
3. INPUT num1, num2, num3
4. Add numbers together by using the
   formula; Total=num1+num2+ num3
5. PRINT total
6. END

PROGRAMMING LIFE CYCLE

Exercise :

1. The factorial formula is “n! = 1 x 2 x 3 x … x n” (where n is any whole number)Write the program that accepts the input “n” and output the factorial value.
2. Find average score of “n” students in a class. The program will accept the input “n” from the user.
3. Find the minimum score of 10 students in a class.
4. Finding the total of 1 + 2 + 3 + 4 + 5 + … + n (Where n can be any whole number).Program will accept “n” from the user.

PROGRAMMING LIFE CYCLE

Program Modeling

2. Flowchart

- ✓ A graphic help us describe the logic of our algorithm
- ✓ Each type of operation, e.g. input, output, calculation/assignment, test a condition (to prepare for a jump), etc.. is represented by a symbol
- ✓ Symbols are connected by arrows to represent the order of the operations
- ✓ Flowchart are useful during algorithm design
- ✓ A flowchart can be relatively easily translated to a programming language

PROGRAMMING LIFE CYCLE

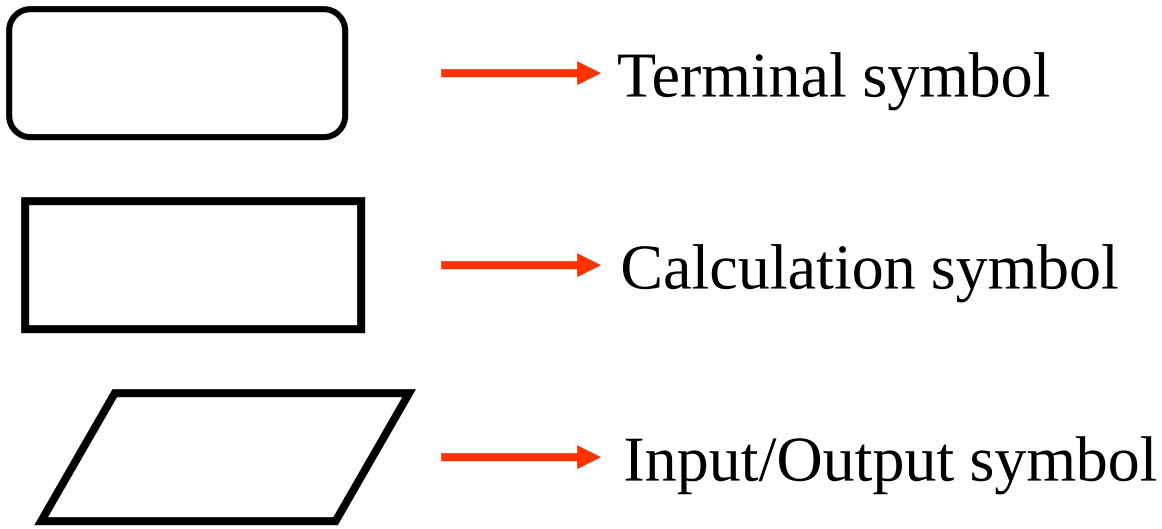
✓ In an algorithm we specify the computation operations (arithmetic, logical, data input, data output) to be done, and their sequence.

✓Each type of computation operation is denoted by a special symbol in the flowchart language

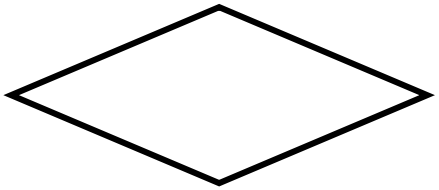
✓ Symbols are connected by arrows to represent the order of the operations (sequence)

PROGRAMMING LIFE CYCLE

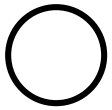
A flowchart is a graphical diagram that define the “flow” of a program using a series of standard geometric symbols and lines, which are connected according to the logic of the algorithm. There are a lot of symbols used in flowcharting but the common five are:



PROGRAMMING LIFE CYCLE



Decision or condition
Testing symbol



Branch symbol

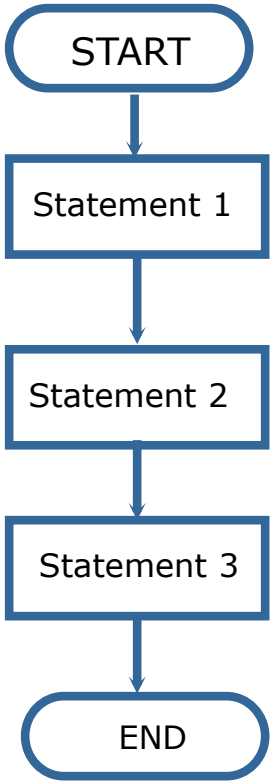
Data flow symbol

PROGRAMMING LIFE CYCLE

Flowcharts Symbols

Terminal: Start/End

- ✓ Every algorithm starts somewhere, and terminates somewhere
- ✓ Every flowchart must have one start symbol and one end symbols
- ✓ A start symbol denotes the start of the algorithm
- ✓ When an end symbol is encountered this indicates the algorithm has terminated.

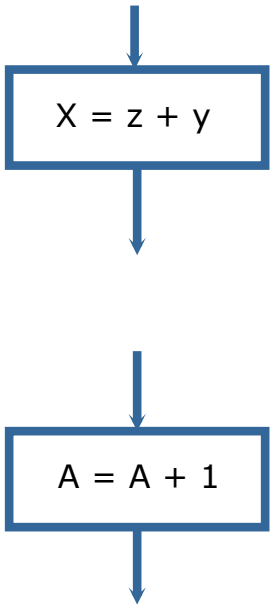


PROGRAMMING LIFE CYCLE

Flowcharts Symbols

Calculation (square or rectangle)

- ✓ You use it to specify a calculation, e.g. arithmetic expression
- ✓ Examples:
  - x = z+y
  - a = pi * r*r
  - z = z + 1 (add one to the current value of z and make that the new value of z)
  - c = SQRT(x*x+y*y)
- ✓ x,z,y,a,c,pi, and r above can be thought of as nick names



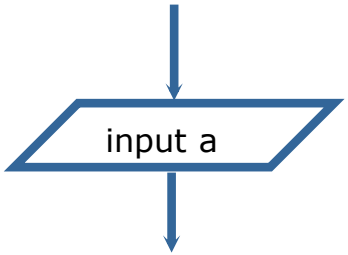
PROGRAMMING LIFE CYCLE

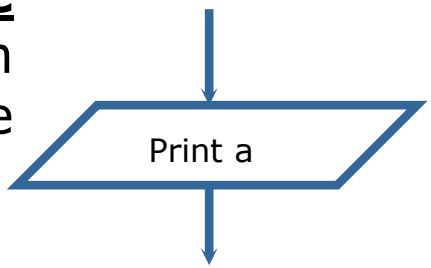
Flowcharts Symbols

Input/Output

- ✓ Use it to specify an input or an output operation
- ✓ By an input operation we mean a **read** operation of data from a peripheral device to Main Memory (e.g. user typing on a keyboard, or reading a byte from the Modem)
- ✓ By an output operation we mean a **write** operation of data to a peripheral device from Main Memory (e.g. data displayed on the monitor, or sent to the printer)
- ✓ Examples:

  Read a value for the radius,r, from the keyboard; print value of the area, a, on the screen





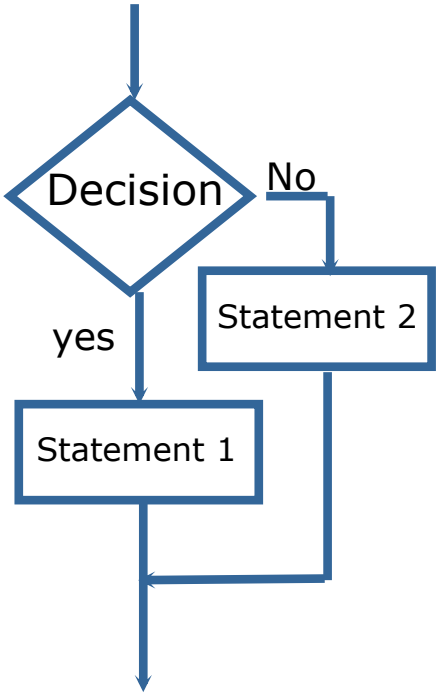
PROGRAMMING LIFE CYCLE

Flowcharts Symbols

Decision or condition Testing

(Diamond)

- ✓ Use it to specify a condition to be tested. Based on the condition being true or false the next operation will be determined



- ✓ A decision is composed of :

1. A condition
2. An operation to be done if condition is true
3. An operation to be done if condition is false

- ✓ We use decisions to enable repeating a set of operations multiple times; we call this construct a loop.

PROGRAMMING LIFE CYCLE

Example 1:

- Problem:

  - Create a program to read 10 numbers from keyboard. Program will calculate the average of that numbers. Print the number and the average.

PROGRAMMING LIFE CYCLE

Example 1:

- Problem Analysis:

1. INPUT: 10 number

2. PROCESS: Calculate the average by using the formula; average=total/10

3. OUTPUT: 10 number and average

PROGRAMMING LIFE CYCLE

Example 1:

- Program Modeling: Pseudo code

1. START
2. Declare a name for every value to be input -a1, a2, a3, a4, a5, a6, a7, a8, a9, a10
Declare a name for the value of sum -TOTAL
Declare a name for a value of average -AVERAGE
Set all = 0
3. INPUT 10 numbers
4. Calculate the total by using the formula; TOTAL=a1+a2+… a10

PROGRAMMING LIFE CYCLE

Example 1:

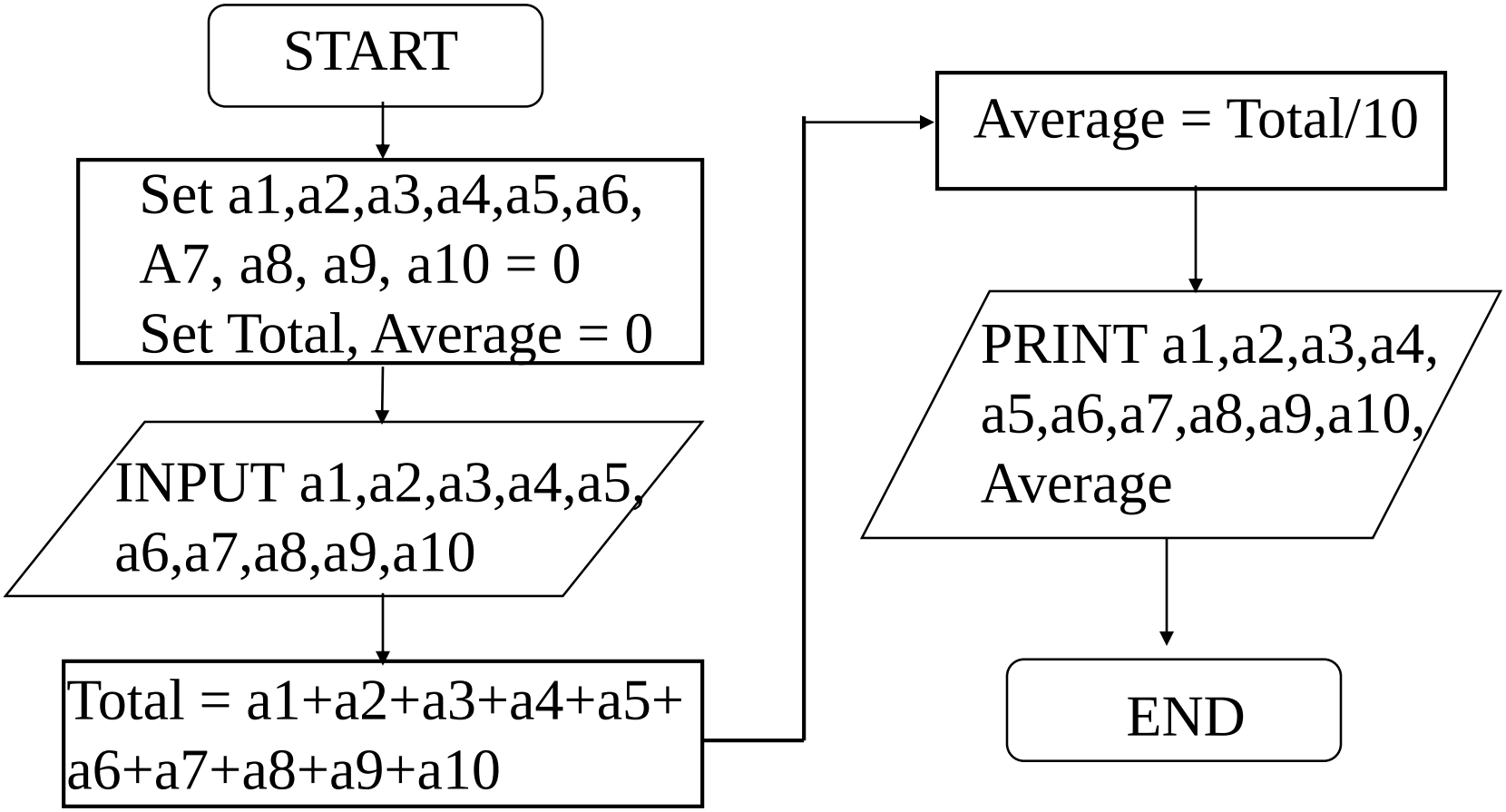
- Program Modeling: Pseudo code

5. Calculate the average by using the formula; AVERAGE=TOTAL/10
6. PRINT 10 numbers and AVERAGE
7. END

PROGRAMMING LIFE CYCLE

Example 1:

- Program Modeling: Flowchart



PROGRAMMING LIFE CYCLE

Example 2:

- Problem: **Add three numbers**

A program is required to read three numbers from a user, add them together, and print their total on the screen.

PROGRAMMING LIFE CYCLE

Example 2:

- Problem Analysis:

    1. INPUT: number1, number2, number3

    2. PROCESS: sum all the number together

    3. OUTPUT: total

PROGRAMMING LIFE CYCLE

Example 2:

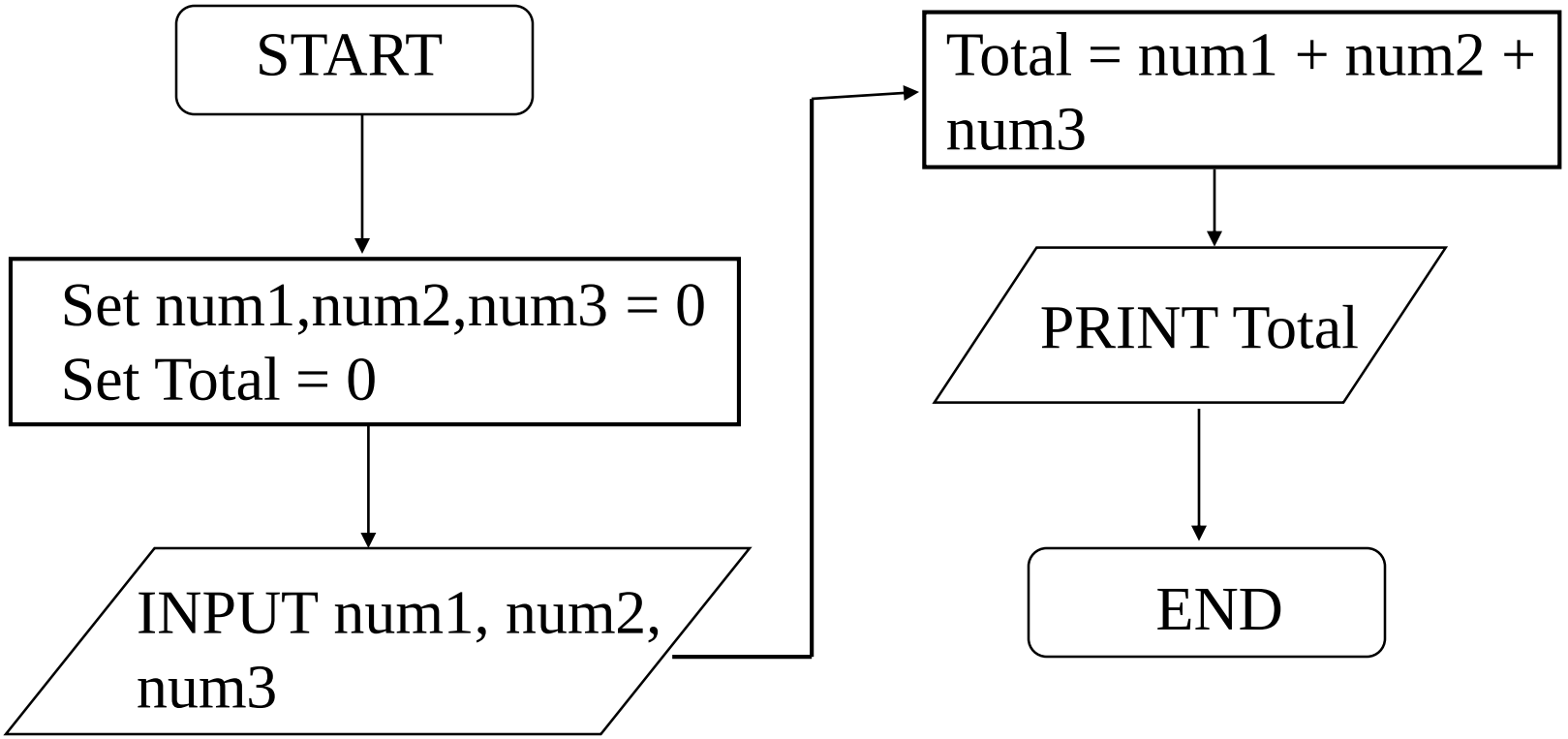
- Program Modeling: Pseudo Code

1. START
2. Declare num1, num2, num3 and Total
   Set num1, num2, num3 and Total = 0
3. INPUT num1, num2, num3
4. Add numbers together by using the
   formula; Total=num1+num2+ num3
5. PRINT Total
6. END

PROGRAMMING LIFE CYCLE

Example 1:

- Program Modeling: Flowchart



PROGRAMMING LIFE CYCLE

Exercise :

1. The factorial formula is “n! = 1 x 2 x 3 x … x n” (where n is any whole number)Write the program that accepts the input “n” and output the factorial value.
2. Find average score of “n” students in a class. The program will accept the input “n” from the user.
3. Find the minimum score of 10 students in a class.
4. Finding the total of 1 + 2 + 3 + 4 + 5 + … + n (Where n can be any whole number).Program will accept “n” from the user.