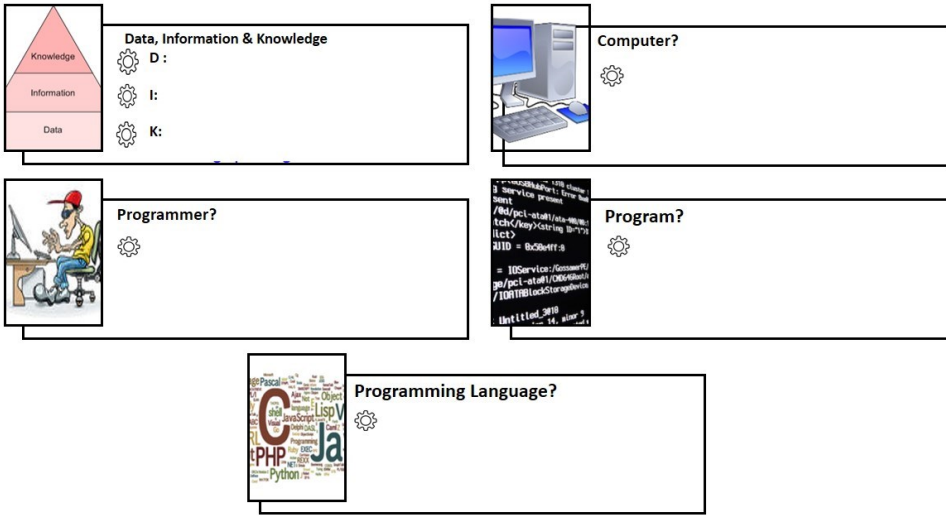


# INTRODUCTION TO PROGRAMMING CONCEPTS

Foundation in Science Technology  
(FiST) Unikl  
Kolej MARA Kuala Nerang  
Nurulhuda Mior Khairuddin

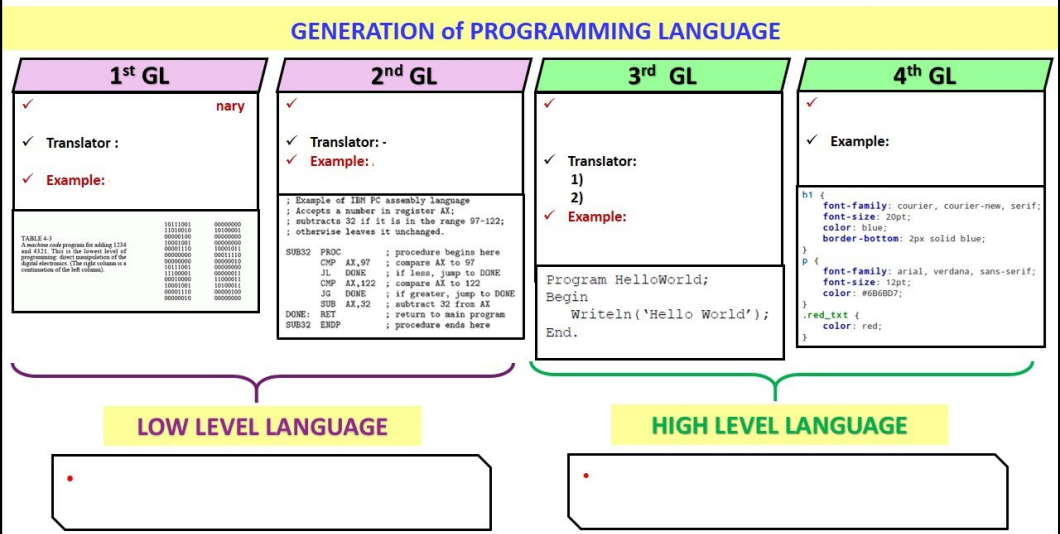
# Introduction to Programming Language

Topic 1&2



# History Of Programming Language

Topic 1&2

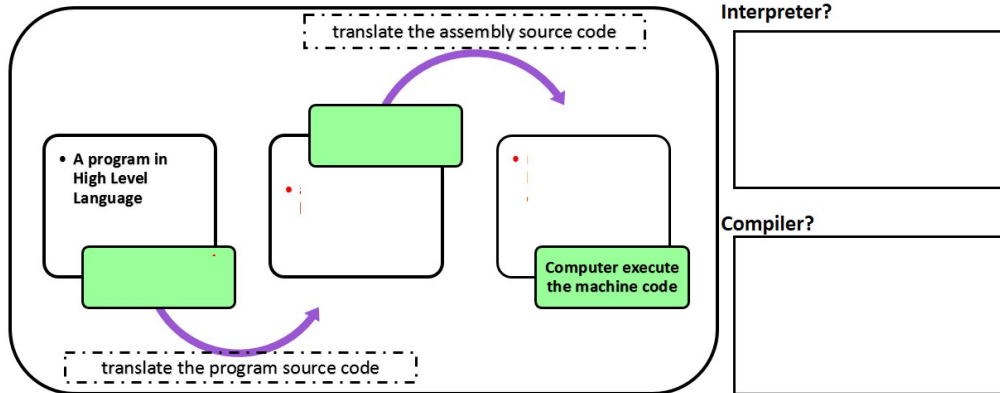


# History Of Programming Language

Topic 1&2

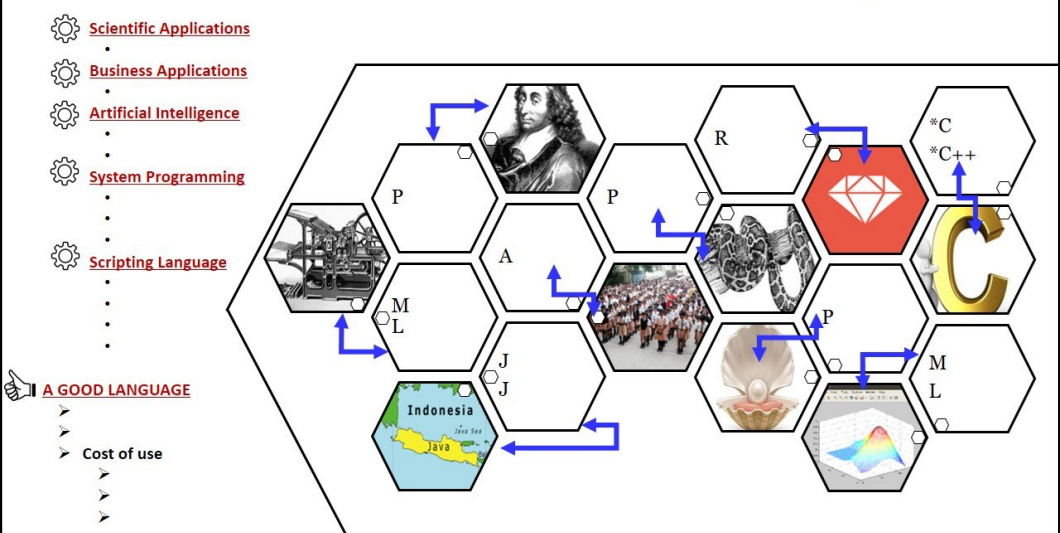
<http://www.codeconquest.com/what-is-coding/how-does-coding-work/>

## How Coding Works?



# Programming Language Names:

Topic 1&2



## PROGRAMMING LIFE CYCLE (PLC): Introduction

Topic 3

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

1

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).

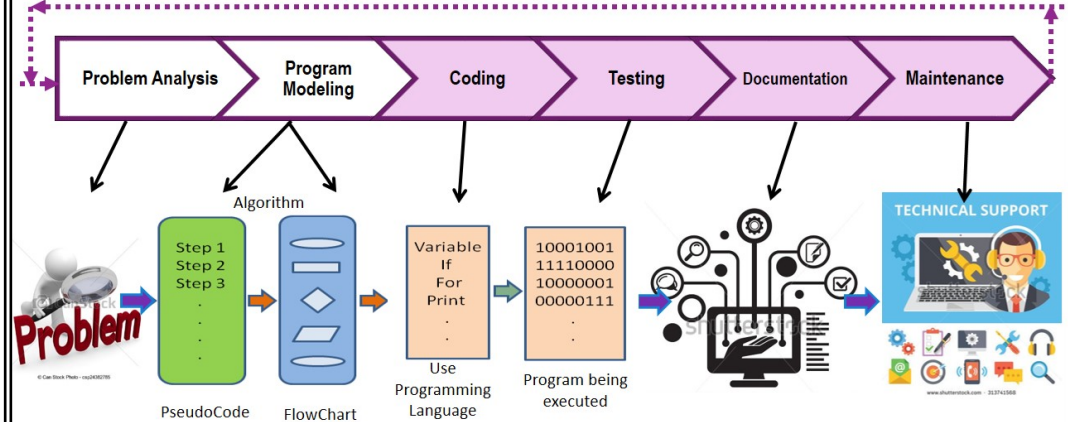
## PROGRAMMING LIFE CYCLE (PLC): Introduction

Topic 3

2

### The development of the system/program

The general steps programmers go through to develop the system

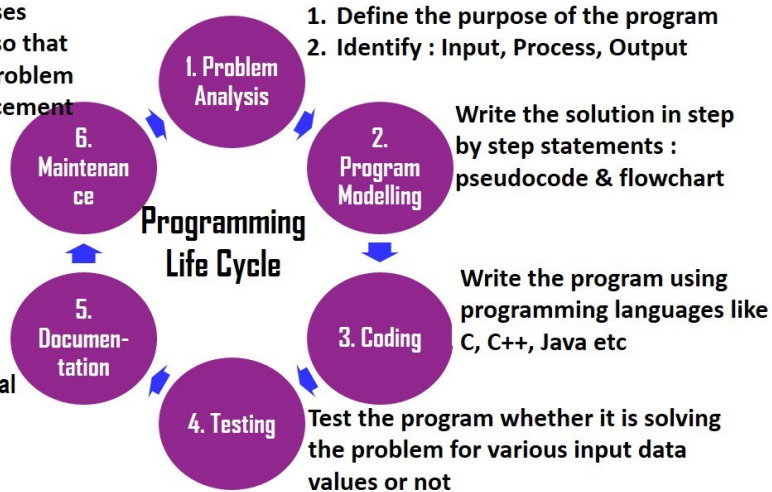


## PROGRAMMING LIFE CYCLE (PLC): Introduction

Topic 3

3

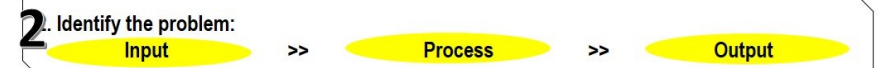
Repeat all the phases from the starting, so that the encountered problem is solved or enhancement is added.



## PROGRAMMING LIFE CYCLE (PLC): Problem Analysis

Topic 3

4



1. Define the purpose of the program

2. Identify the problem:

**Input :** a list of the data provided to solve the problem.

Define:

- Variable for each of the data (being input).
- Data type for variable (being input).
  - i. String or Character
  - ii. Integer or Float

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

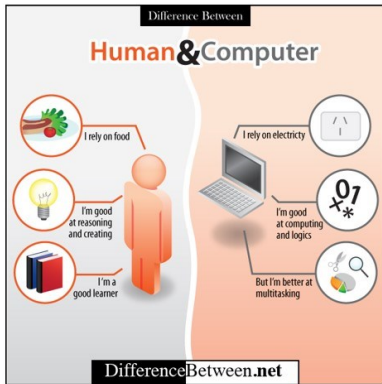
Actions:

- Process the input data.
- Manipulate data into information

**Output:** a list of the outputs required.

Display the results as output:

- Send it to the screen
- Write to a file



- 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.
- Computers** consist of **2 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;**
- 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**

Problem Analysis



**Problem 1:**

- *Ali need to create a program to allow a bookstore clerks to enter quantity of books sold in 3 days of The Big Bad Wolf Sale and display the total of book sold.*



■ **Problem Analysis:**

1. **INPUT** : BookDay1, BookDay2, BookDay3

Variables for 3 days sales

2. **PROCESS:** Calculate  
 $TotalBooksSold = BookDay1 + BookDay2 + BookDay3$

Variables for total book sold

3. **OUTPUT** : TotalBooksSold

Problem Analysis



**Problem:**

- Create a program to read 5 workers salary from keyboard. Program will calculate the average of that salaries.
- Print the average.

■ **Problem Analysis:**

1. **INPUT** : salary1, salary2, salary3, salary4, salary5

Variables for 5 workers salary

2. **PROCESS** : Calculate  
 $AverageSalary = (salary1 + salary2 + salary3 + salary4 + salary5) / 5$

Variables for average salary

3. **OUTPUT** : AverageSalary

Do not use ÷

**Program Modelling=ALGORITHM**

- ✓ **creating model of solution.**
- ✓ **algorithm**, the **step-by-step instruction to solve a problem**, then the program will be written based on the algorithm using a programming language.
- ✓ There are many ways to represent an algorithm; however **2 techniques** commonly used are **pseudo code** and **flowchart**.

while a number is greater than zero,  
do  
Add the number to the running total.  
end if the number is greater than zero

# pseudocode

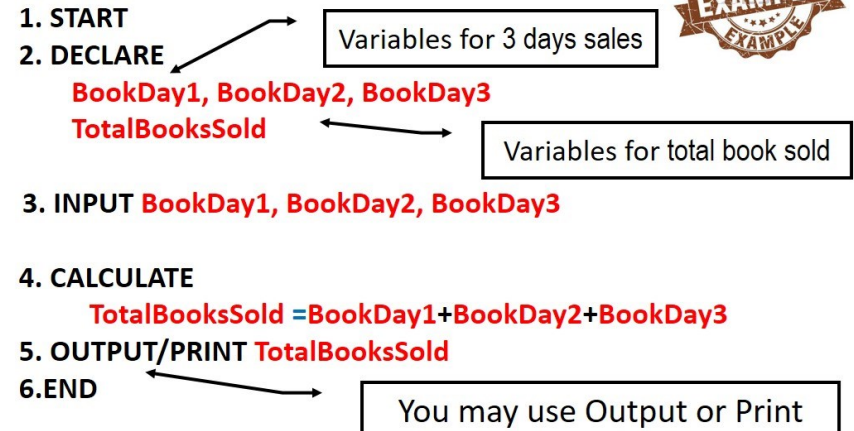
...is a Structured English used to represent an algorithm. It is easy for both users and programmers to read and write algorithms regardless of the programming experience. There is no standard pseudo code but we should follow some conventional rules:

- 1) Statements are written in simple English.
- 2) Each statement is written on a separate line.
- 3) Statements are starting with the text START.
- 4) Statements are ending with the text END.

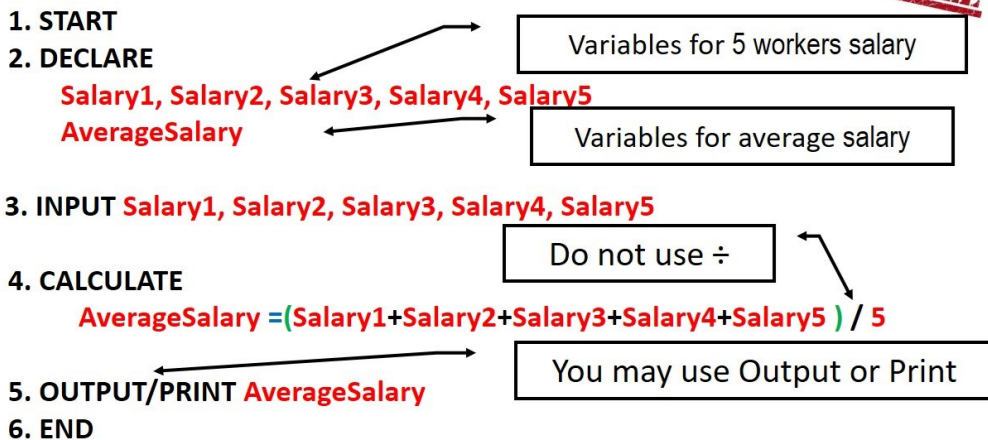
## Pseudo Code Structure:

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

## Problem Analysis > Pseudo Code



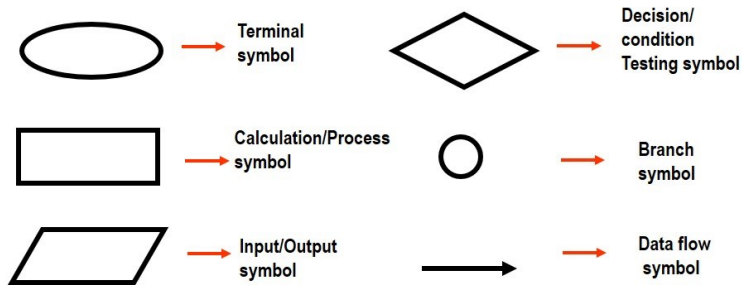
## Problem Analysis > Pseudo Code



## Flow Charts

- A graphic help us describe the logic of our algorithm
- Each type of operation, e.g. input, output, calculation, test a condition, etc.. is represented by a **symbol**
- Symbols are connected by **arrows** to represent the order of the operations
- A flowchart can be relatively easily **translated** to a programming language

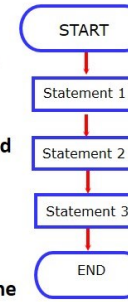
A flowchart is a graphical diagram that define the “flow” of a program using a series of standard geometric symbols and lines, which are connected according to the logic of the algorithm. There are a lot of symbols used in flowcharting but the common 6 are:



### Flowcharts Symbols

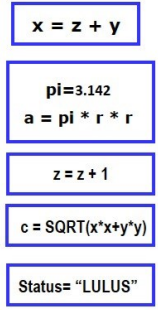
#### Terminal: Start/End

- ✓ 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.



#### Calculation (rectangle)

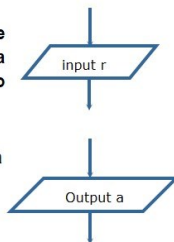
- ✓ You use it to specify a calculation, e.g. arithmetic expression
- ✓ Examples:
  - $x = z + y$
  - $pi=3.142$
  - $a = pi * r * r$
  - $z = z + 1$  (add one to the current value of z and make that the new value of z)
  - $c = \text{SQRT}(x*x+y*y)$
  - $\text{Status} = \text{"LULUS"}$
- ✓ x,z,y,a,pi,r, c and Status are variables



### Flowcharts Symbols

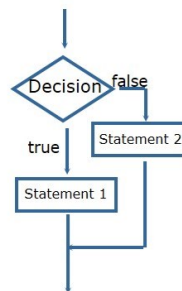
#### Input/Output

- ✓ Use this 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
- ✓ By an output operation we mean a **write** operation of data to a peripheral device from Main Memory
- ✓ **Examples:**
  - Read a value for the radius, r, from the keyboard;
  - print value of the area, a, on the screen



#### Decision or condition Testing (Diamond)

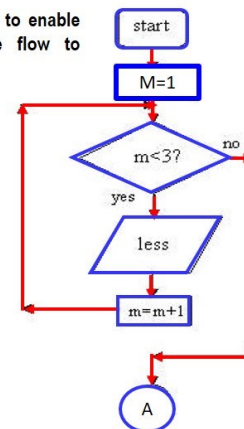
- ✓ Use it to specify a condition to be tested. Based on the condition being true or false the next operation will be determined
- ✓ A decision is composed of :
  1. A condition
  2. An operation to be done if condition is true
  3. An operation to be done if condition is false
- ✓ We use decisions to enable repeating a set of operations multiple times; we call this construct a loop.



### Flowcharts Symbols

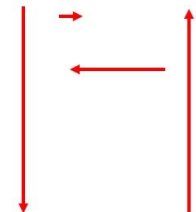
#### Branch

- ✓ Use branch symbol to enable connection to one flow to another flow.

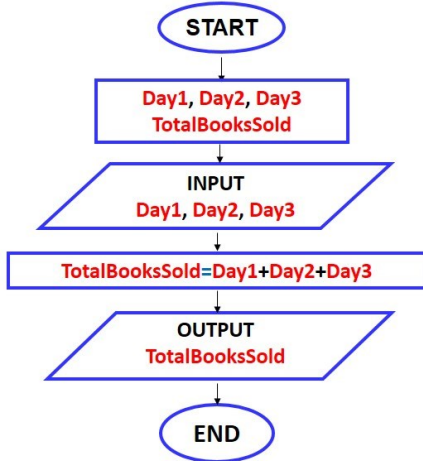


#### Arrow

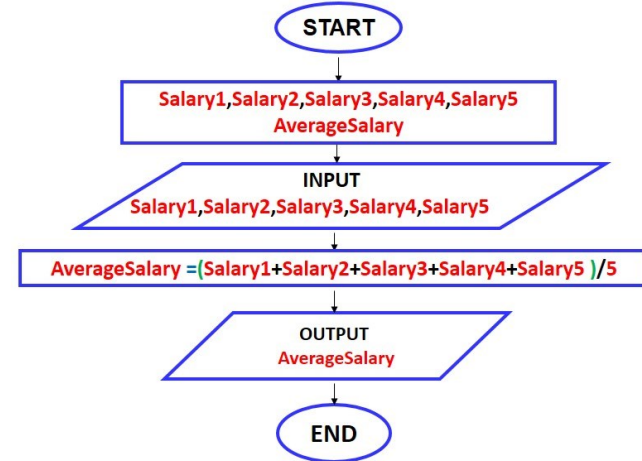
- ✓ Symbols are connected by arrows to represent the sequence (flow) of operation.



Problem Analysis > Pseudo Code > Flow Chart



Problem Analysis > Pseudo Code > Flow Chart



Variables?

- are symbolic names that represent a value within a computer program
- Variables are "containers" for storing information.

Valid example?

Variable	Explanation
NAME	Capital Letter and Small Letters are Allowed
name	name
name_1	Digits and Underscore is allowed along with alphabets
_SUM	Underscore at the first position is allowed.
Total_Sales	We can concatenate multiple words with underscore
firstName	Best Style to concatenate multiple words (Changing case of First Letter of Successive Word)

Rules?

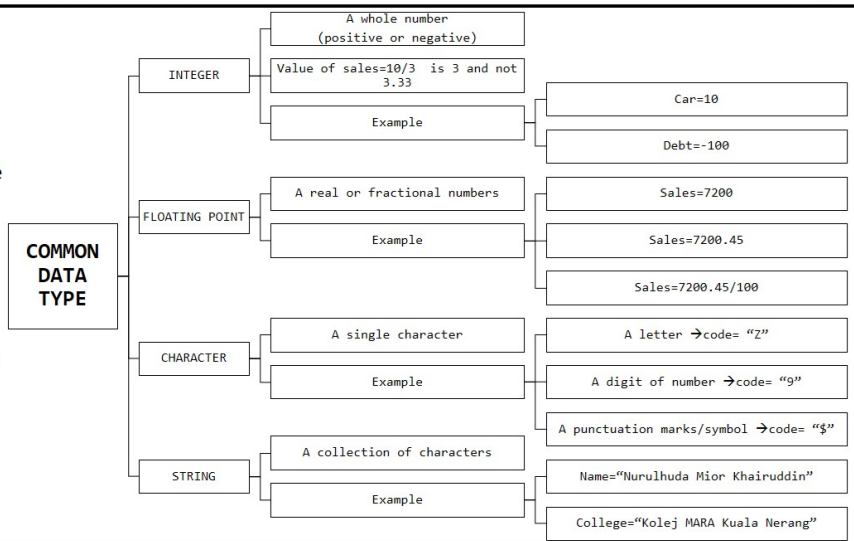
- A variable name must start with a alphabet or underscore (\_).
- A variable name can only contain alphabets and underscores (A-z, 0-9, and \_).
- A variable cannot start with a number.
- Blank space is not allowed in variables

Invalid example?

Variable	Explanation
int	Keyword name cannot be given to variable.
num^2	Special characters are not allowed.
num 1	Spaces are not allowed.
2num	Digits are allowed but NOT as first Character

Data Type?

- is a classification that specifies which **type** of value a variable has and what **type** of mathematical, relational or logical operations can be applied to it without causing an error



## EXPRESSION & OPERATION : Introduction

 Topic 7 **24**

### ❑ Expression?

In **programming**, an **expression** is any legal combination of symbols that represents a value. Each **programming** language and application has its own rules for what is legal and illegal. For example, in the C language  $x+5$  is an **expression**, as is the character string "MONKEYS."

What is expression? Webopedia Definition  
[www.webopedia.com/TERM/E/expression.html](http://www.webopedia.com/TERM/E/expression.html)

### ❑ Example?

- $x = 2$
- $y = x + 5$
- animal = "monkey"
- Every expression consists of at least one **operand** and one or more **operators**. Operands are values, whereas operators are symbols that represent particular actions.

## EXPRESSION & OPERATION : Introduction

 Topic 7 **25**

### ❖ Operations?

A specific action that represents by operator

### ❑ 3 types of Operation in Programming?

- ❖ Mathematical
- ❖ Relational
- ❖ Logical
- ❖ Increment / Decrement

## EXPRESSION & OPERATION :Mathematical Operation

 Topic 7 **26**

### ❑ Mathematical operation

#### ❑ Addition (+)

- ❑  $A=A+1$
- ❑  $X=10/100$
- ❑  $COMMISSION=SALES +(SALES* X)$

#### ❑ Multiplication (\*)

- ❑  $Y=X*2$
- ❑  $A=81$
- ❑  $TOTAL=A*9$

#### ❑ Subtraction (-)

- ❑  $Y=X-2$
- ❑  $A=A-1$

#### ❑ Division (/)

- ❑  $Y=X/2$
- ❑  $A=81$
- ❑  $TOTAL=A/9$

#### ❑ Modulus (%)

- ❑  $REMINDER =VALUE \% DIVISOR$

### ❑ PRECEDENCE

- ❑ First : ( ) - bracket
- ❑ Second : Multiplication (\*), Division (/), Modulus (%)
- ❑ Third : Addition (+) Subtraction (-)

## Expression & Operation :Relational Operation

 Topic 7 **27**

### ❑ Relational operation

#### ❑ Equal To (==)

#### ❑ Less Than (<)

#### ❑ Greater Than (>)

#### ❑ Not Equal To (!=)

#### ❑ Less Than Or Equal To (<=)

#### ❑ Greater Than Or Equal To (>=)

- Used to form expression that can be evaluated as **TRUE** or **FALSE**.
- **Example** : we might ask if there are more than 30 students in a class using the following expression:- `NumStudent>30`
- If there are more than 30 students in the class then the expression output is **TRUE**
- If there 30 students or less then the expression output is **FALSE**.

SITUATION?	EXPRESSION
Does Abu make more than RM35 000.00 per year?	<code>AbuSalary &gt; 35000</code>
Did Kedah defeat the PKNS in football match?	<code>KedahGoals &gt; PKNSGoals</code>
Did anyone get a perfect score on the Mid Sem Test	<code>MidSemScore == 100</code>



## Expression & Operation :Logical Operation

Topic 7 **28**

### Logical operation

- AND(&&)
- OR (||)
- NOT(!)

The logical operators for AND (&&) and OR ( ) are used to combine simple relational statements into more complex expressions.

The NOT (!) operator is used to negate a boolean statement.

P	Q	P && Q	P    Q	!P
false	false	false	false	true
false	true	false	true	true
true	false	false	true	false
true	true	true	true	false

## Expression & Operation :Logical Operation

Topic 7 **29**

### Logical operation : EXAMPLE (situation)

- All employee belong to a department.(dept)
- All employee working in dept. A have salaries greater than RM25 000.00.
- All employee working in dept. B make exactly RM40 000.00 per year.
- Abu is an employee and he works at dept A.
- Bazli is an employee and he works at dept B.

Given the left situation are facts, consider the truth values of the following expression:

EXPRESSION	Abu_Salary < 25000 AND Abu_Dept == A
Evaluation:	(False) && (True) =False
EXPRESSION	NOT Abu_Salary < 25000
Evaluation:	! (False) =TRUE
EXPRESSION	Abu_Salary < 25000 OR Bazli_Salary == 40000
Evaluation:	(False)    (True) =TRUE
EXPRESSION	Abu_Salary >25000 AND NOT Bazli_Salary < 40000
Evaluation:	(True) && !(False) (True) && (True) =TRUE

## Expression & Operation :Logical Operation

Topic 7 **30**

### Logical operation : EXERCISE (Situation)

- All employee belong to a department.(dept)
- All employee working in dept. A have salaries greater than RM25 000.00.
- All employee working in dept. B make exactly RM40 000.00 per year.
- Abu is an employee and he works at dept A.
- Bazli is an employee and he works at dept B.

Given the left situation are facts, consider the truth values of the following expression:

EXPRESSION	Abu_Salary < 25000 OR Abu_Dept == B
Evaluation:	(False)    (False) =False
EXPRESSION	Abu_Salary >= 25000 AND Bazli_Dept == B
Evaluation:	(False) && (True) =False
EXPRESSION	Abu_Salary < 25000 OR Bazli_Salary < 40000 OR Abu_Dept == B
Evaluation:	(False)    (False)    (False) =FALSE
EXPRESSION	NOT (Abu_Salary >25000 AND Bazli_Salary < 40000)
Evaluation:	! ( (True) && (False) ) ! ( (False) && (True) ) =TRUE

## Expression & Operation :In / De - crement Operation

Topic 7 **31**

### 3 types of Operation in Programming?

- Mathematical operation
- Relational
- Logical
- Increment/ Decrement

#### Increment operation :

- Use to add value 1 to the variable
- Format:
  - Variable++ OR
  - Variable=variable+1
- Example:
  - A++ OR
  - A=A+1

#### Decrement operation :

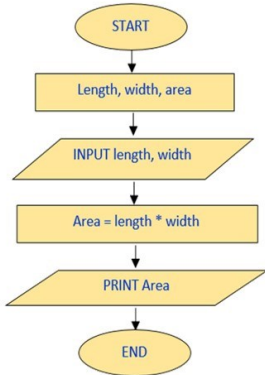
- Use to minus value 1 from the variable
- Format:
  - Variable-- OR
  - Variable=variable-1
- Example:
  - A-- OR
  - A=A-1

## PROGRAM CONTROL STRUCTURE: Introduction

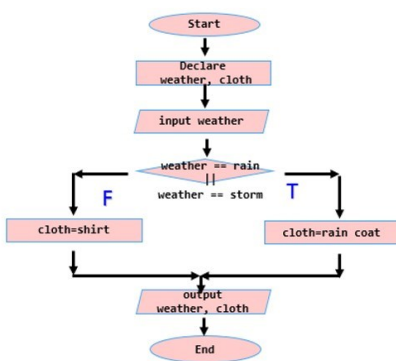
Topic 7 **32**

□ PCS controls how each step of the algorithm executes? There are 3 structures:

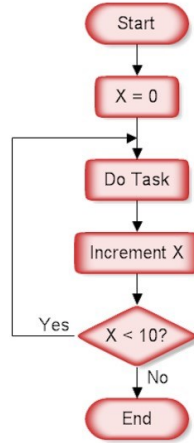
### ➤ Sequence



### ➤ Selection



### ➤ Iteration/ looping



## PROGRAM CONTROL STRUCTURE: Sequence

Topic 7 **33**

The sequence control structure is the straightforward execution of one processing step after another.

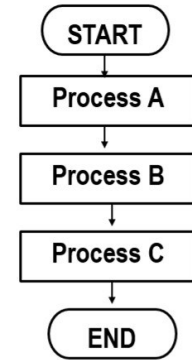
### Sequence's pseudo code:

1. Start
2. Process A
3. Process B
4. Process C
5. End

PSEUDOCODE

FLOWCHART

### Sequence's flowchart:



## PROGRAM CONTROL STRUCTURE: Sequence

Topic 7 **34**

### Problem 1:

All need to create a program to allow a bookstore clerks to enter quantity of books sold in 3 days of The Big Bad Wolf Sale and display the total of book sold.

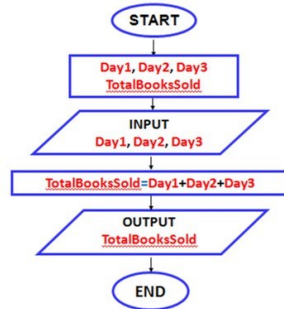
PROB.ANALYSIS

•INPUT BookDay1, BookDay2, BookDay3  
 •PROCESS:  
 Calculate  
 $TotalBooksSold = BookDay1 + BookDay2 + BookDay3$   
 OUTPUT : TotalBooksSold

PSEUDOCODE

1. START
2. DECLARE  
 Int BookDay1, BookDay2, BookDay3  
 Int TotalBooksSold
3. INPUT BookDay1, BookDay2, BookDay3
4. CALCULATE  
 $TotalBooksSold = BookDay1 + BookDay2 + BookDay3$
5. OUTPUT/PRINT TotalBooksSold
6. END

FLOWCHART



## PROGRAM CONTROL STRUCTURE: Sequence

Topic 7 **35**

### Problem 1:

Create a program to read 5 workers salary from keyboard. Program will calculate the average of that salaries. Print the average.

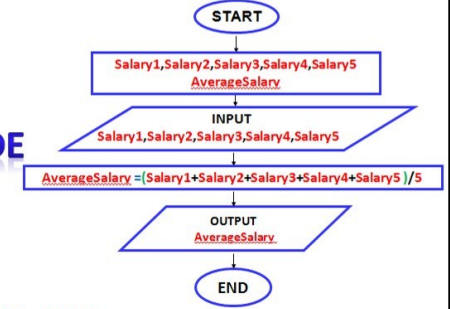
### PROB.ANALYSIS

1.INPUT : salary1, salary2, salary3, salary4, salary5  
 2.PROCESS :  
 Calculate  
 $AverageSalary = (salary1 + salary2 + salary3 + salary4 + salary5) / 5$   
 3. OUTPUT : AverageSalary

PSEUDOCODE

1. START
2. DECLARE  
 Salary1, Salary2, Salary3, Salary4, Salary5  
 AverageSalary
3. INPUT Salary1, Salary2, Salary3, Salary4, Salary5
4. CALCULATE  
 $AverageSalary = (Salary1 + Salary2 + Salary3 + Salary4 + Salary5) / 5$
5. OUTPUT/PRINT AverageSalary
6. END

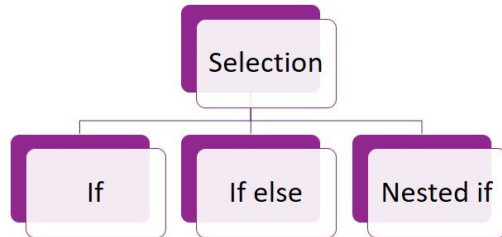
### FLOWCHART



## PROGRAM CONTROL STRUCTURE: Selection

Topic 7 **36**

❖ There are three main types of selection control structure:



❖ This control structure represents the decision-making process.

❖ The choice depending on whether the condition is true or false.

❖ Nested IF structure occurs when there is (are) IF statement(s) within an IF statement.

## PROGRAM CONTROL STRUCTURE: Selection

Topic 7 **37**

❖ FORMAT: PSEUDOCODE

• IF

1. START
2. IF (Condition) Then
3. PROCESS 1
4. PROCESS 2
5. End IF
6. END

• IF -ELSE

1. START
2. IF (Condition) Then
3. PROCESS 1
4. PROCESS 2
5. ELSE
6. PROCESS 4
7. PROCESS 5
8. PROCESS 6
9. End IF
10. END

• NESTED IF

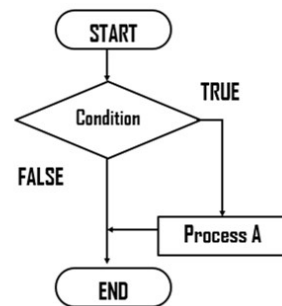
1. START
2. IF (Condition1) Then
3. PROCESS 1
4. IF (CONDITION2) THEN
5. PROCESS 2
6. ENDIF
7. ELSE
8. Process 3
9. END IF
10. END

## PROGRAM CONTROL STRUCTURE: Selection

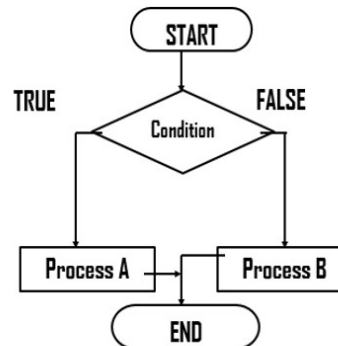
Topic 7 **38**

❖ FORMAT: FLOWCHART

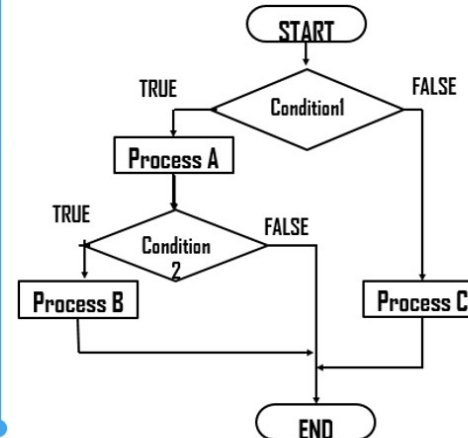
• IF



• IF -ELSE



• NESTED IF



PROGRAM CONTROL STRUCTURE: Selection Topic 7 39

❖ EXAMPLE : SELECTION >> IF

- Create a program that :
  - will display RAIN COAT if user input RAIN or STORM as the weather.
  - Nothing will be displayed for other input.

PROGRAM CONTROL STRUCTURE: Selection IF Topic 7 40

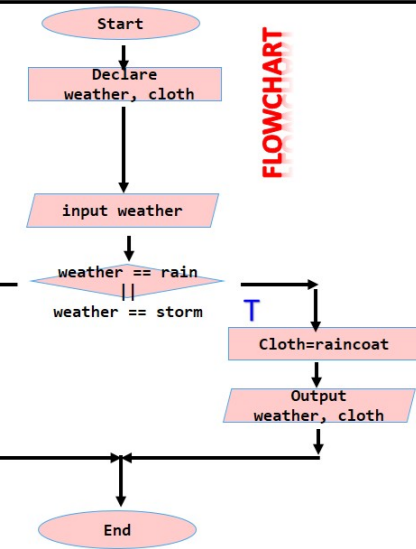
	INPUT	PROCESS	OUTPUT
	weather	if weather == rain or weather == storm assign cloth=raincoat Output weather, cloth end if	weather, cloth
test1	storm		storm, raincoat
test2	rain		rain, raincoat
test3	sunny		(no output)

PROB. ANALYSIS

PSEUDOCODE

```

1. START
2. DECLARE weather, cloth
3. Input weather
4. IF (weather ==rain || weather==storm) Then
5.     cloth=rain coat
6.     OUTPUT weather,cloth
7. End IF
8. END
    
```



PROGRAM CONTROL STRUCTURE: Selection Topic 7 41

❖ EXAMPLE : SELECTION >> IF.. Else

- Create a program that :
  - will display RAIN COAT if user input RAIN or STORM as the weather.
  - will display SHIRT for other input.

PROGRAM CONTROL STRUCTURE: Selection IF.. Else Topic 7 42

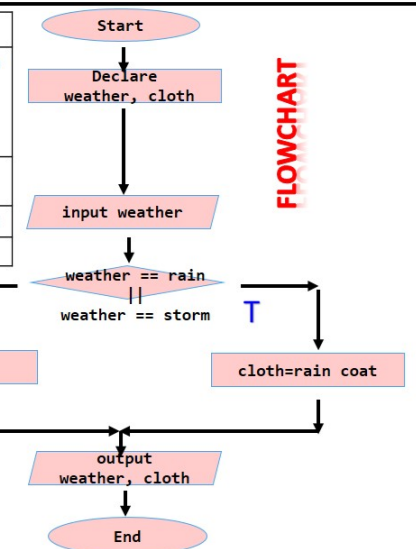
	INPUT	PROCESS	OUTPUT
	weather	if weather == rain or weather == storm assign cloth=raincoat else assign cloth=shirt end if	weather, cloth
test1	storm		storm, raincoat
test2	rain		rain, raincoat
test3	windy		windy, shirt

PROB. ANALYSIS

PSEUDOCODE

```

1. START
2. DECLARE weather, cloth
3. Input weather
4. IF (weather ==rain || weather==storm) Then
5.     cloth=rain coat
6. Else
7.     cloth=shirt
8. End IF
9. OUTPUT weather, cloth
10. END
    
```





## PROGRAM CONTROL STRUCTURE: Selection Topic 7 47

### ❖ EXAMPLE : SELECTION >> Nested If...Else

- A bus company give discount to all their passenger for CNY month to any destination according to this rate:

Age	Discount
12 years and below	50%
Between 13-59	10%
60 years above	20%

- This is the example of the bus ticket printed by the system:



Name: Nurulhuda Mior Khairuddin  
Age: 36                      Seat: 3A  
Fee: RM100.00    After Discount: RM90.00

## PROGRAM CONTROL STRUCTURE: Iteration/Looping Topic 7 49

### ❖ EXAMPLE : SELECTION >> Iteration/Looping : Do While

- There are 8 students in a group. They took a test. Grade are given according this:

Score	Grade
50 and above	Pass
49 and below	Fail

- Print out list name of student, their score and grade a group.
- Summarize the number of student (pass and fail) within the group.

	INPUT	PROCESS	OUTPUT
<b>PROB. ANALYSIS</b>	name	if age<=12 fee=50/100*fee	name
	age	else	age
	fee	if age>12 && age <=59 fee=90/100*fee	fee
	seat	else fee=80/100*fee	new_fee
		end if	seat
	test 1	Ali, 8, 20	Ali, 8, 20, 10, 1A
	test 2	Ain, 20, 100	Ain, 20, 100, 90, 1B
	test 3	Abu, 70, 100	Abu, 70, 100, 90, 12C

**PSEUDOCODE**

```

1. START
2. DECLARE name, age, fee, new_fee, seat
3. Input name, age, fee, new_fee, seat
4. IF Age<=12 Then
5.     New_fee=50/100*fee
6. Else
7.     If Age>12 && age<=59 Then
8.         New_fee=90/100*fee
9.     Else
10.        New_fee=80/100*fee
11.    End IF
12. End If
13. OUTPUT name, age, fee, new_fee, seat
14. END
                    
```

	INPUT	PROCESS	OUTPUT
<b>PROB. ANALYSIS</b>	name	Assign num_pass=0, num_fail=0, counter=1	Within loop:
	score	Do While counter<=8	name
		Input name, score	Score
		IF score<=50 Then	Grade
		grade=fail	
		num_fail=num_fail+1	
		Else	
		grade=pass	
		num_pass=num_pass+1	
		End If	
		OUTPUT name, score, grade	
		Counter=counter+1	
		End Do	
			Outside loop:
			num_fail
			num_pass

**PSEUDOCODE**

```

1. START
2. DECLARE name, score, grade, num_pass, num_fail, counter
3. Assign num_pass=0, num_fail=0, counter=1
4. Do While counter<=8
5. Input name, score
6. IF score<=50 Then
7.     grade=fail
8.     num_fail=num_fail+1
9. Else
10.    grade=pass
11.    num_pass=num_pass+1
12. End If
13. OUTPUT name, score, grade
14. Counter=counter+1
15. End Do
16. OUTPUT num_fail, num_pass
17. END
                    
```

**PROGRAM CONTROL STRUCTURE: Iteration/Looping**  **Topic 7 51**

❖ **EXAMPLE : SELECTION >> Iteration/Looping : Do While**

- There are **n** students in a group. They took a test. Grade are given according to this:

Score	Grade
50 and above	Pass
49 and below	Fail

- Print out list name of student, their score and grade a group.
- Summarize the number of student (pass and fail) within the group.

**PROB. ANALYSIS**

INPUT	PROCESS	OUTPUT
name score	Assign num_pass=0, num_fail=0, counter=1 Do While counter<=n Input name, score IF score<=50 Then grade=fail num_fail=num_fail+1 Else grade=pass num_pass=num_pass+1 End If OUTPUT name, score, grade Counter=counter+1 End Do	Within loop: name Score Grade  Outside loop: num_fail num_pass

**Topic 7 52**

**FLOWCHART**

**PSEUDOCODE**

```

1. START
2. DECLARE name, score, grade, num_pass, num_fail, counter
3. Assign num_pass=0, num_fail=0, counter=1
4. Do While counter<=n
5. Input name, score
6. IF score<=50 Then
7.   grade=fail
8.   num_fail=num_fail+1
9. Else
10.  grade=pass
11.  num_pass=num_pass+1
12. End If
13. OUTPUT name, score, grade
14. Counter=counter+1
15. End Do
16. OUTPUT num_fail, num_pass
17. END
    
```

**PROGRAM CONTROL STRUCTURE: Iteration/Looping**  **Topic 7 53**

❖ **EXAMPLE : SELECTION >> Iteration/Looping : Do While**

- A bus company give discount to all their passenger for CNY month to any destination according to this rate:

Age	Discount
12 years and below	50%
Between 13-59	10%
60 years above	20%

- This is the example of the bus ticket printed by the system:



Name: Nurulhuda Mior Khairuddin  
Age: 36    Seat: 3A  
Fee: RM100.00    After Discount: RM90.00

- If the bus could take only 40 Passenger, count the number of Child, Adult and Old Folks for the a trip.

**PROB. ANALYSIS**

INPUT	PROCESS	OUTPUT
name age fee seat	Counter=1 num_adult=0, num_child=0, num_old=0 Do while counter<=40 if age<=12 fee=50/100*fee num_child=num_child+1 else if age>12 && age <=59 fee=90/100*fee num_adult=num_adult+1 else fee=80/100*fee num_old=num_old+1 end if (2 times) Counter=counter+1 End Do	Within loop: name age fee new_fee seat  Outside loop: num_child num_adult num_old

**Topic 7 54**

**FLOWCHART**

**PSEUDOCODE**

```

1. START
2. DECLARE name, age, fee, new_fee, seat, num_adult, num_child, num_old
3. Assign num_child=0, num_adult=0, num_old=0, counter=1
4. Do While counter<=40
5. Input name, age, fee, new_fee, seat
6. IF Age<=12 Then
7.   New_fee=50/100*fee
8.   num_child=num_child+1
9. Else
10.  If Age>12 && age<=59 Then
11.    New_fee=90/100*fee
12.    num_adult=num_adult+1
13.  Else
14.    New_fee=80/100*fee
15.    num_old=num_old+1
16.  End IF
17. End IF
18. OUTPUT name, age, fee, new_fee, seat
19. Counter=counter+1
20. End Do
21. OUTPUT num_child, num_adult, num_old
22. END
    
```



## Article 1

Every programmer knows that debugging is a time-consuming nightmare, so it makes a good candidate for programmers to learn how to deal with. There are generally two types of errors: **syntax** errors and **logic** errors.

Syntax errors occur when a program does not conform to the grammar of a programming language, and the compiler cannot compile the source file. Logic errors occur when a program does not do what the programmer expects it to do.

Syntax errors are usually easy to fix because the compiler will tell you where the error occurs and you simply fix the syntax error. For example you may miss a semicolon or a curly bracket where it's supposed to be. Simply locate those errors and fix them.

The real pain in the neck are logic errors. For instance you may be writing a software application that solves a puzzle, but when you run it it tells you that the puzzle cannot be solved. After you dig through the program logic you realize you have done something wrong. After you fix the logic and run the program again it solves the puzzle successfully.





## Article 2

When programmers write code in a high-level language there are two types of errors that they might make: syntax errors and logic errors.

Syntax errors are mistakes such as misspelled keywords, a missing punctuation character, a missing bracket, or a missing closing parenthesis. Nowadays, all famous IDEs such as Eclipse, NetBeans, and Visual Studio (to name a few) detect these errors as you type and underline the erroneous statements with a wavy line. If you try to execute a program that includes syntax errors, you will get error messages on your screen and the program won't be executed. You must correct all the errors and then try to execute the program again.

Logic errors are those errors that prevent your program from doing what you expected it to do. With logic errors you get no warning at all. Your code may compile and run but the result is not the expected one. Logic errors are the most difficult errors to detect. You must revisit your program thoroughly to determine where your error is. For example, consider a program that prompts the user to enter three numbers, and then calculates and displays their average value. The programmer, however, made a typographical error; one of his or her statements divides the sum of the three numbers by 5, and not by 3 as it should. Of course the program is executed as usual, without any error messages, prompting the user to enter three numbers and displaying a result, but obviously not the correct one! It is the programmer who has to find and correct the erroneously written statement, not the computer or the compiler!

# Types of Errors



## Problem 1:

- Program to find Area of rectangle.

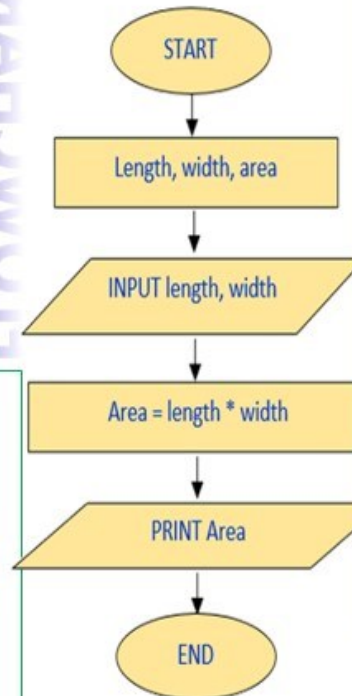
PROB. ANALYSIS

Input	:	length, width
Process	:	Calculate area $area = length * width$
Output	:	area

FLOWCHART

PSEUDOCODE

1. START
2. Declare length, width, area
3. INPUT length, width
4. Calculate area = length \* width
5. PRINT area
6. END



## C++ CODING

```
/* Python Program to find area of rectangle*/  
  
#include <iostream>  
#include <stdlib.h>  
using namespace std;  
int main() {  
    float length, width, area;  
  
    cout<<"\n Enter your length : ";  
    cin>>length;  
  
    cout<<"\n Enter your width : ";  
    cin>>width;  
  
    area = length * width;  
  
    cout<<"\n area="<<area;  
  
    system("PAUSE");  
    return 0; }  

```

## PYHTON CODING

```
# Python Program to find Area Of a rectangle  
  
length = float(input(' Please Enter the length: '))  
width = float(input(' Please Enter the width: '))  
area = length*width  
  
print(" Area= %.2f" %area)}
```