PRESENTATION OF ALGORITHMS

- There are two commonly used tools to help to document program logic (the algorithm).

- These are;
  - flowcharts and
  - Pseudocode

- Generally, ***flowcharts*** work well for ***small*** problems but,
- ***Pseudocode*** is used for ***larger*** problems

PRESENTATION OF ALGORITHMS

- **FLOWCHARTS**
- **Flowcharting** is a tool developed in the computer industry, for showing the steps involved in a process.

- A flowchart is a diagram made up of *boxes*, *diamonds* and other shapes, *connected by arrows* –

- each shape represents a step in the process, and the arrows show the order in which they occur.

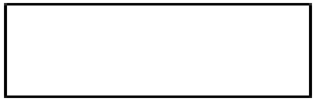
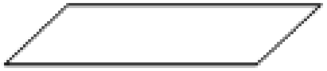
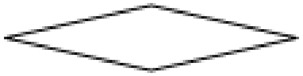
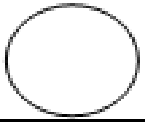
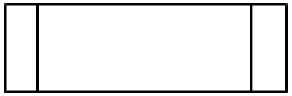
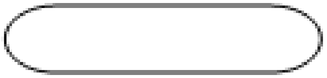
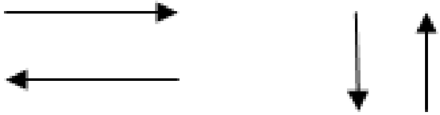
PRESENTATION OF ALGORITHMS

- Flowcharting combines symbols and flow lines, to show figuratively the operation of an algorithm.

- ***Flowcharting Symbols***
- There are ***6 basic symbols commonly*** used in flowcharting of assembly language programs:

- ***Terminal, Process, and input/output, Decision, Connector and Predefined Process.***

PRESENTATION OF ALGORITHMS

Symbol	Name	Function
	Process	Indicates any type of internal operation inside the Processor or Memory
	input/output	Used for any Input / Output (I/O) operation. Indicates that the computer is to obtain data or output results.
	Decision	Used to ask a question that can be answered in a binary format (Yes/No, True/False)
	Connector	Allows the flowchart to be drawn without intersecting lines or without a reverse flow.
	Predefined Process	Used to invoke a subroutine or an interrupt program.
	Terminal	Indicates the starting or ending of the program, process, or interrupt program.
	Flow Lines	Shows direction of flow.

PRESENTATION OF ALGORITHMS

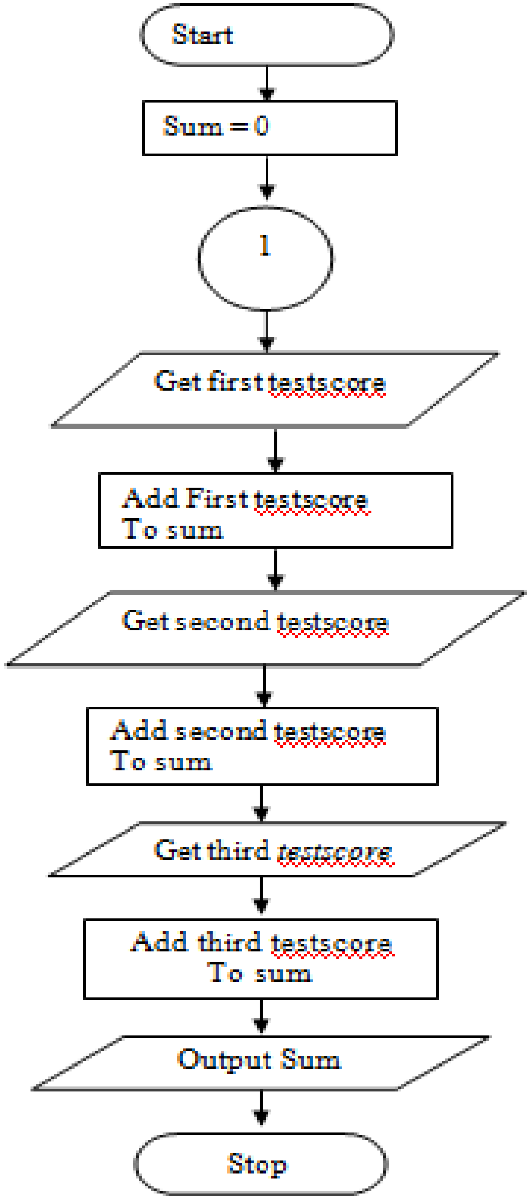
- The flow of data between steps is indicated by arrows, or flow lines.

- ***Examples of Algorithms and Flowcharts***

- ***Example 1.*** Design an algorithm and the corresponding flowchart for adding the test scores as given below:
- 26, 49, 98

- ***Algorithm*** & its corresponding ***flowchart***

1. Start
2. Sum = 0
3. Get the first testscore
4. Add first testscore to sum
5. Get the second testscore
6. Add to sum
7. Get the third testscore
8. Add to sum
9. Output the sum
10. Stop



PRESENTATION OF ALGORITHMS

- **PSEUDOCODE**
- Is one of the tools that can be used to write a preliminary plan that can be developed into
- a ***computer program***.
- **Pseudocode** is a generic way of describing an algorithm without use of any specific
- programming language ***syntax***.

PRESENTATION OF ALGORITHMS

- It is, as the name suggests, *pseudo* code —***it cannot be executed on a real computer***,

- but it models and resembles real programming code, and is written at roughly the same level of detail.
-
- Pseudocode, by nature, exists in various forms, although most borrow syntax from

- popular programming languages (like **C**, **Lisp**, or **FORTRAN**).

PRESENTATION OF ALGORITHMS

- In the algorithm design, the steps of the algorithm are written in free English text and,

- although brevity is desired, they may be as long as needed to describe the particular operation.

PRESENTATION OF ALGORITHMS

- **CONTROL STRUCTURES OR LOGICAL STRUCTURES**
- ***The sequence structure***
- It is a case where the steps in an algorithm are constructed in such a way that,
- ***no condition step is required.***
- The sequence structure is the logical ***equivalent of a straight line***.

PRESENTATION OF ALGORITHMS

- ***For example***, suppose you are required to design an ***algorithm*** for finding the ***average of six numbers***,

- and the ***sum*** of the numbers ***is given***. The ***pseudocode*** will be as follows:

```
 Start
Get the sum
Average = sum / 6
Output the average
Stop
```

PRESENTATION OF ALGORITHMS

- **Example 1:** This is the pseudo-code required to ***input 3 numbers*** from the keyboard and ***output the result***.

```
Use variables: sum, num1, num2, num3 of type integer
start
Accept num1, num2, num3
Sum = num1 + num2 + num3
Print sum
End program
```

PRESENTATION OF ALGORITHMS

- **Example 2:** The following pseudo-code describes an algorithm which will accept two numbers from

- the keyboard and calculate the sum and product displaying the answer on the monitor screen.

```
Use variables sum, product, num1, num2 of type real
start
display “Input two numbers”
accept num1, num2
```

PRESENTATION OF ALGORITHMS

```
sum = num1 + num2
print “The sum is “, sum
product = num1 * num2
print “The Product is “, product
end program
```

PRESENTATION OF ALGORITHMS

- ***Decision Structure or Selection Structure***
- This is the structure where in the algorithm, one has to make a choice of two alternatives
- by making ***decision depending on a given condition***.
- In pseudo-code as in any other programming language **IF** and **ELSE** and **CASE** are the mostly
- ***key words*** used to express ***decision structure***.

PRESENTATION OF ALGORITHMS

- **Example1:** The following pseudo-code algorithm will display the greatest number among the 2 entered numbers.

```
Use variables num1, num2 of type integer
 Display “Enter two numbers”
Accept num1,num2
IF num1>num2 Then
 Display “The greatest is ”num1
Else
Display “The greatest is” num2
```

PRESENTATION OF ALGORITHMS

- The above algorithm will display a wrong message if num1 = num2. The right solution is presented is:

```
Use variables num1, num2 of type integer
 Display “Enter two numbers”
Accept num1,num2
IF num1>num2 Then
 Display “The greatest is ”num1
IF num1<num2 Then
  Display “The greatest is” num2
IF num1=num2 Then
Display “num1 is equal to  num2”
```

PRESENTATION OF ALGORITHMS

- **Example 2:** The following program segment outputs a message to the monitor screen describing

- the insurance available according to a category input by the user.

```
Use variables: category of type character
Display "input category"
Accept category
If category = "U"
    Display "insurance is not available"
```

PRESENTATION OF ALGORITHMS

```
Else
If category =” A” then
Display “insurance is double”
Else
If category = “B” then
   Display “insurance is normal”
Else
If category = “M” then
   Display “insurance is medically dependent”
Else
   Display “entry invalid”
```

PRESENTATION OF ALGORITHMS

- This can be expressed in a ***case statement*** as follows:

```
Use variables: category of type character
Display “input category”
Accept category
DO case of category
  CASE category = U
     Display “insurance not available”
  CASE category = A
     Display “insurance is double”
  CASE category = B
```

PRESENTATION OF ALGORITHMS

```
Display "insurance is normal"
  CASE category = M
      Display "insurance is medically dependent"
  OTHERWISE
    Display "entry is invalid"
ENDCASE
```

- Instead of using the word ***otherwise***, one can use ***else***.

PRESENTATION OF ALGORITHMS

- ***Repetition or Iteration Structure***
- A third structure causes the certain steps to be repeated.

- The ***Repetition*** structure can be ***implemented using***;
  - Repeat Until Loop
  - The While Loop
  - The For Loop

PRESENTATION OF ALGORITHMS

- Any program instruction that repeats some statement or sequence of statements a number of times is called

- iteration or a **loop**.

- The ***commands*** used to create ***iterations or loops*** are all based on ***logical tests***.

PRESENTATION OF ALGORITHMS

- **Example 1:** A program segment ***repeatedly*** asks for ***entry*** of a number in the ***range*** 1 to 100

- ***until*** a valid number is entered.

```
REPEAT
DISPLAY “Enter a number between 1 and 100”
ACCEPT number
UNTIL number < 1 OR number > 100
```

PRESENTATION OF ALGORITHMS

- **Example 2:** A survey has been carried out to discover the most popular sport. The results will be typed into the computer for analysis. Write a program to accomplish this.

```
REPEAT
DISPLAY “Type in the letter chosen or Q to finish”
DISPLAY “A: Athletics”
DISPLAY “S: Swimming”
DISPLAY “F: Football”
```

PRESENTATION OF ALGORITHMS

```
DISPLAY “B: Badminton”
DISPLAY “Enter data”
ACCEPT letter
If letter = ‘A’ then
  Athletics = athletics + 1
If letter = ‘S’ then
  Swimming = Swimming + 1
If letter = ‘F’ then
  Football = Football + 1
If letter = ‘B’ then
```

PRESENTATION OF ALGORITHMS

```
Badminton = Badminton + 1
UNTIL letter = ‘Q’
DISLAY “Athletics scored”, athletics, “votes”
DISLAY “Swimming scored”, swimming, “votes”
DISLAY “Football scored”, football, “votes”
DISLAY “Badminton scored”, Badminton, “votes”
```

PRESENTATION OF ALGORITHMS

- **The WHILE loop**
- This type of conditional loop tests for terminating condition at the beginning of the loop.

- In this case no action is performed at all if the first test causes the terminating condition to ***evaluate as false***.
- ***The syntax is***

  WHILE (a condition is true)

  A statement or block of statements

  ENDWHILE

PRESENTATION OF ALGORITHMS

- **Example 1:** A program segment to print out each character typed at a keyboard until the

- ***character 'q' is entered***.

```
WHILE letter <> 'q'
    ACCEPT letter
    DISPLAY "The character you typed is", letter
ENDWHILE
```

PRESENTATION OF ALGORITHMS

- **Example 2:** Write a program that will output the square root of any number input

- ***until the number input is zero***.

- In some cases, a variable has to be initialized before execution of the loop as

- shown in the following example.

PRESENTATION OF ALGORITHMS

```
Use variable: number of type real
DISPLAY “Type in a number or zero to stop”
ACCEPT number
WHILE number <> 0
   Square = number * number
   DISPLAY “The square of the number is”, square
   DISPLAY “Type in a number or zero to stop”
   ACCEPT number
ENDWHILE
```

PRESENTATION OF ALGORITHMS

- ***The FOR Loop***
- This, in its simplest form, uses an initialization of the variable as a starting point,
- a stop condition depending on the value of the variable.
- The variable is incremented after each iteration until it reaches the required value.

PRESENTATION OF ALGORITHMS

- The pseudo-code syntax will be:

```
FOR (starting state, stopping condition, increment)
    Statements
ENDFOR
```

PRESENTATION OF ALGORITHMS

- **Example 1.**

```
FOR (n = 1, n <= 4, n + 1)
    DISPLAY “loop”, n
ENDFOR
```

- The fragment of code will produce the ***output***

```
loop 1
loop 2
loop 3
loop 4
```

PRESENTATION OF ALGORITHMS

- **Example 2:** Write a program to calculate the sum and average of a series of numbers.

- The ***pseudo-code*** solution is:

```
Use variables: n, count of the type integer
Sum, number, average of the type real
DISPLAY “How many numbers do you want to input”
ACCEPT count
SUM = 0
```

PRESENTATION OF ALGORITHMS

```
FOR (n = 1, n <= count, n + 1)
DISPLAY “Input the number from your list”
ACCEPT number
SUM = sum + number
ENDFOR
Average = sum / count
DISPLAY “The sum of the numbers is “, sum
DISPLAY “Average of the numbers is “, average
```

Quiz 2

1. Design an algorithm and the corresponding flowchart for finding the sum of the numbers 2, 4, 6, 8,… n.
2. Using flowcharts, write an algorithm to read 100 numbers and then display the sum.
3. Write an algorithm to read two numbers then display the largest.
4. Write an algorithm to read two numbers then display the smallest
5. Write an algorithm to read three numbers then display the largest.

Quiz 2

6. Write an algorithm to read 100 numbers then display the largest.
7. Write an algorithm to display the factorial of a number.
8. Write an algorithm to display the decimal part of a real.
9. Write an algorithm to display the table of multiplication of 7, starting from 1 up to N. N is to be defined by the user and it must be positive.

Quiz 2

The format of output must be like bellow:

```
7 X 1=7
7 X 2=14
7 X 3=21
```

10. Write an algorithm (flowchart and pseudocode) to display the smallest and the largest number among the N numbers that have been entered by the user. N must be positive.

11. Write an algorithm to constraint the user to enter a number varying between 0 and 20 the 0 and 20 included. The program must display the number of times the user entered a wrong number.