

# Chapter 2

Problem Solving using Computer

# What is a problem?

- Problem is defined as the difference between an existing situation and a desired Situation, that is, in accordance with calculation; a problem is numerical situation and has complex form

# How to solve the problem?

- 1. Problem Analysis.
- 2. Algorithm development.
- 3. Flowchart development
- 4. Coding.
- 5. Compilation and Execution.
- 6. Debugging and Testing.
- 7. Documentation

# Algorithm Development & Flowcharting

# What is an algorithm

- An algorithm is step by step description of activities or methods to be processed for getting desired output from a given input.

OR ,

An algorithm is the step by step description of the procedure written in human understandable language for solving given problems.

# Properties

- **Finiteness:** Program should have finite number of steps to solve a problem.
- **Definiteness:** The action of each step should be defined clearly without any ambiguity.
- **Inputs:** Inputs of the algorithms should be defined precisely, which can be given initially or while the algorithm runs.
- **Outputs:** Each algorithm must result in one or more outputs.
- **Effectiveness:** It should be more effective among the different ways of solving the problems

# Guidelines for writing algorithm

- Use plain language. In any example, english language is used.
- Do not use any programming language specific syntax.
- Every job to be done should be described clearly without any assumptions.
- The developed algorithm must have a single entry and exit point.

Example 1. Write an algorithm to find area of a circle of radius  $r$ .

- Inputs to the algorithm: radius of circle Expected output: area of circle Algorithm
- Step1: Start
- Step2: Declare necessary variables Area,  $r$ ,  $\pi$
- Step3: Read/Input radius  $r$  of circle
- Step4:  $\text{Area} = \pi * r * r$  //Calculation of area
- Step5: print Area
- Step6: End



Example 2. Write an algorithm to find sum of two numbers.

- Step1: Start
- Step2: Declare necessary variables Sum, num1, num2
- Step3: Read/Input num1, num2
- Step4:  $\text{Sum} = \text{num1} + \text{num2}$  //Calculation of Sum
- Step5: print Sum
- Step6: End

# Example 3. Write an algorithm to find whether a number is even or not(Branching)

- Step1: Start
- Step2: Declare necessary variables num
- Step3: Read/Input num1
- Step4:
  - IF (num % 2 == 0 )                      Display num is even
  - Else                                      Display num is odd
- Step5: End

## Example 4. Write an algorithm to print “Hello world” 10 times(Loop)

- Step 1: Start
- Step2: Declare necessary variables count and initialize with value 1  
i.e, count =0
- Step 3: Display “Hello World”
- Step 4: Count =Count +1
- Step 5:  

IF (Count== 10 )	Stop
Else	Go to Step 3


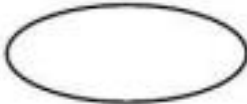






# Flowchart

- A flowchart is a pictorial representation of an algorithm that uses boxes of different shapes to denote different types of instructions

# Advantages of Using Flowcharts

- **Communication:** Flowcharts are better way of communicating the logic of a system to all concerned or involved.
- **Effective analysis:** With the help of flowchart, problem can be analyzed in more effective way therefore reducing cost and wastage of time.
- **Proper documentation:** Program flowcharts serve as a good program documentation, which is needed for various purposes, making things more efficient.
- **Efficient Coding:** The flowcharts act as a guide or blueprint during the systems analysis and program development phase.
- **Proper Debugging:** The flowchart helps in debugging process.
- **Efficient Program Maintenance:** The maintenance of operating program becomes easy with the help of flowchart. It helps the programmer to put efforts more efficiently on that part

## Flowchart Symbols

	Arrow	Used to connect flowchart symbols and direction indicate flow of logic.
	Oval Start/Stop/End	Used to represent beginning and end of task.
	Rectangle	Used for arithmetic and data manipulation operations.
	Input/output	Used for input and output.
	Connectors	Used to connect different flow lines and remote parts of flow charts.
	Decision	Used for decision making and branching operations that have two options.
	Function Call	Used whenever you call the function.
	Loops	Used to indicate for loops

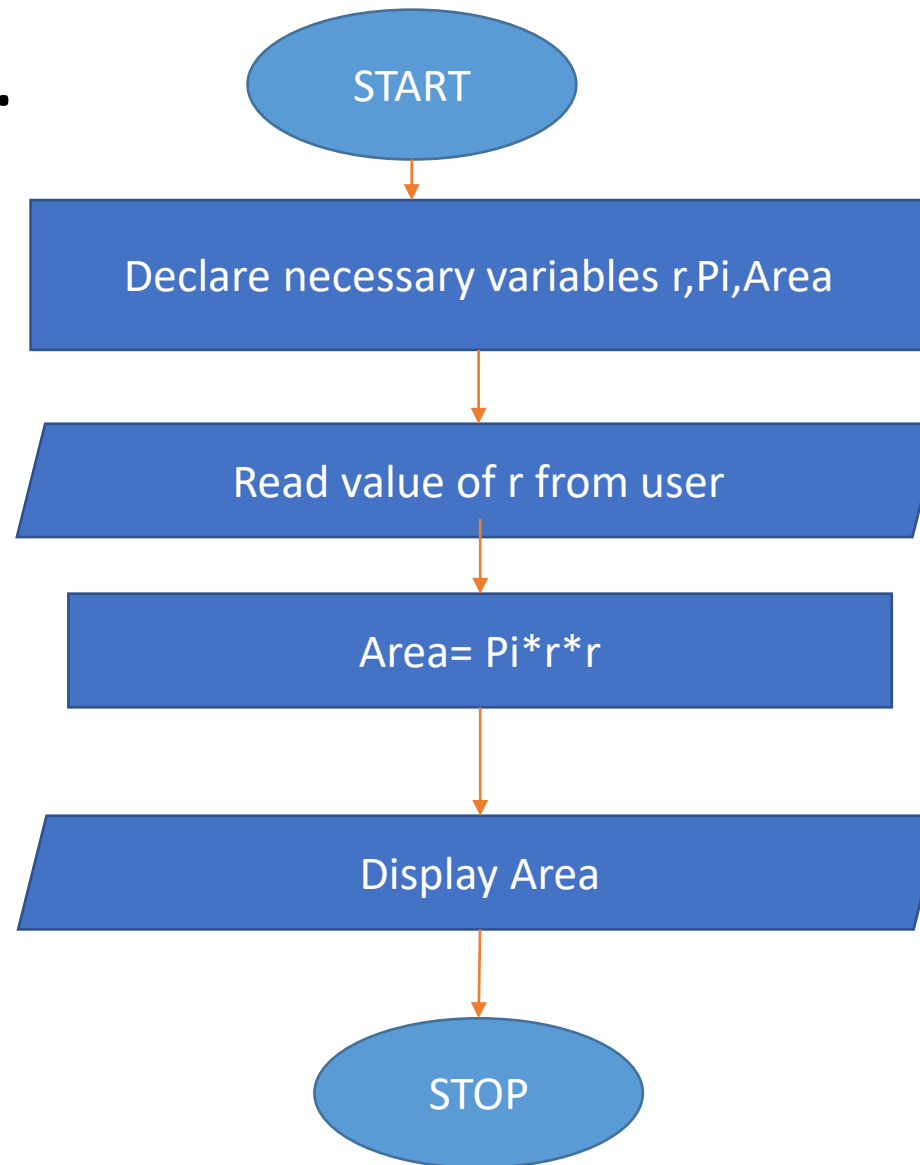
# Guidelines to be followed to draw a flowchart

- [?] In drawing a proper flow chart, all necessary requirements should be listed out in logical order.
- [?] The flowchart should be clear, neat and easy to follow. There should not be any room for ambiguity in understanding the flow chart.
- [?] The usual direction of the flow of a procedure Only one flow line should come out from a process symbol.
- [?] Only one flow line should enter a decision symbol, but two or three flow lines, one for each possible answer, should leave the decision symbol.
- [?] Only one flow line is used in conjunction with terminal symbol.

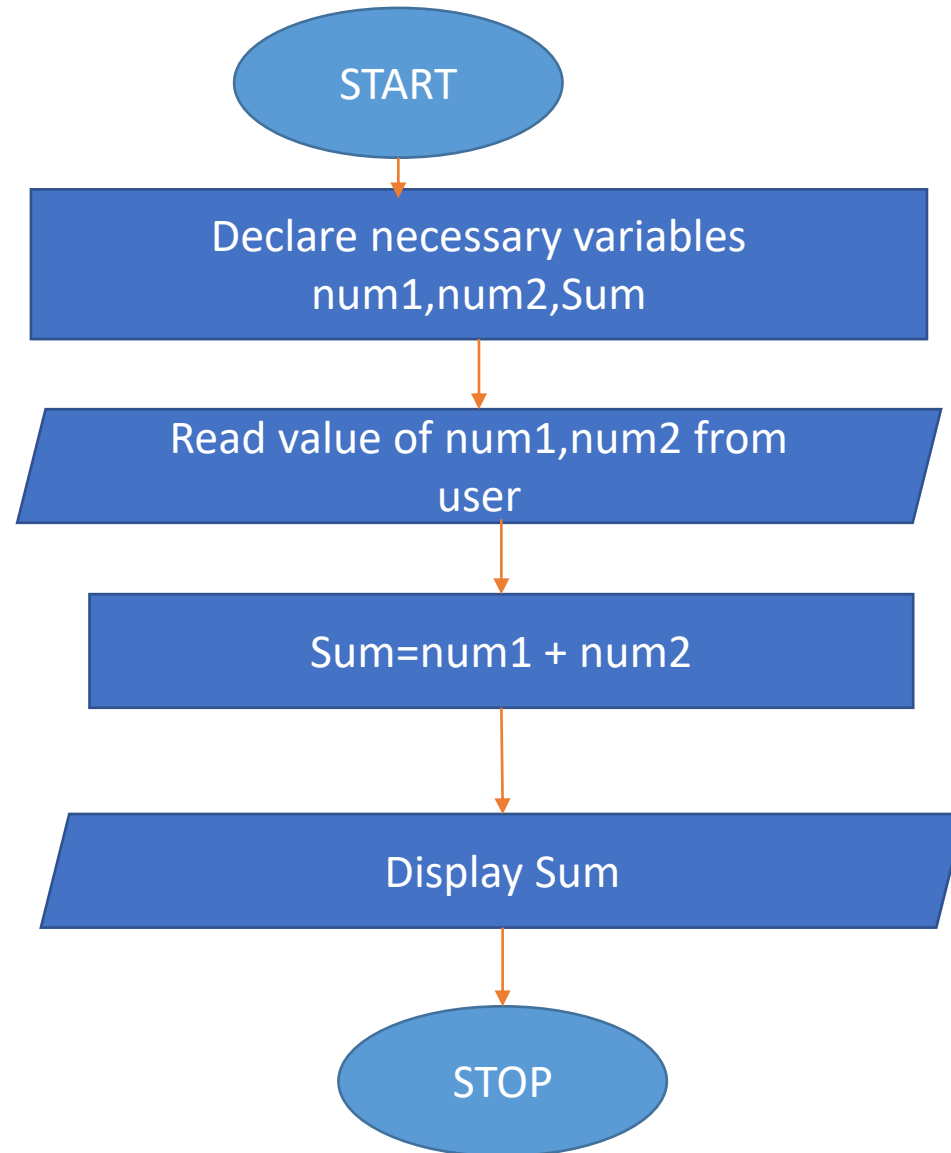
- [?] Write within standard flow chart symbols briefly. As necessary, you can use the annotation symbol to describe data or computational steps more clearly.
- [?] If the flowchart becomes complex, it is better to use connector symbols to reduce the number of flow lines. Avoid the intersection of flow lines if you want to make it more effective and better way of communication.
- [?] Ensure that the flowchart has a logical start and finish.
- [?] It is useful to test the validity of the flowchart by passing through it with a simple test data or system is from left to right or top to bottom.



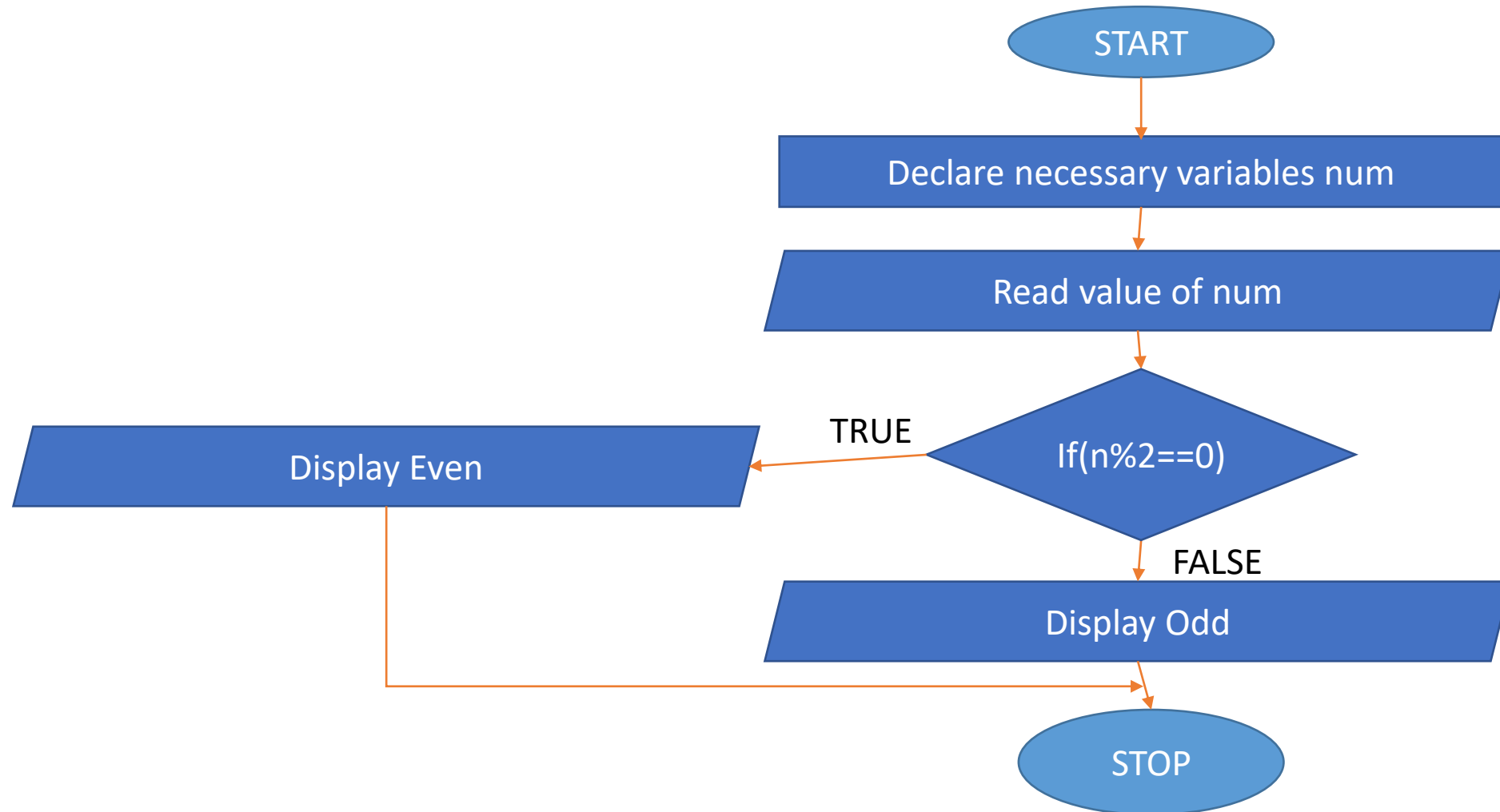
Example 1. Draw a flowchart to find area of a circle of radius  $r$ .



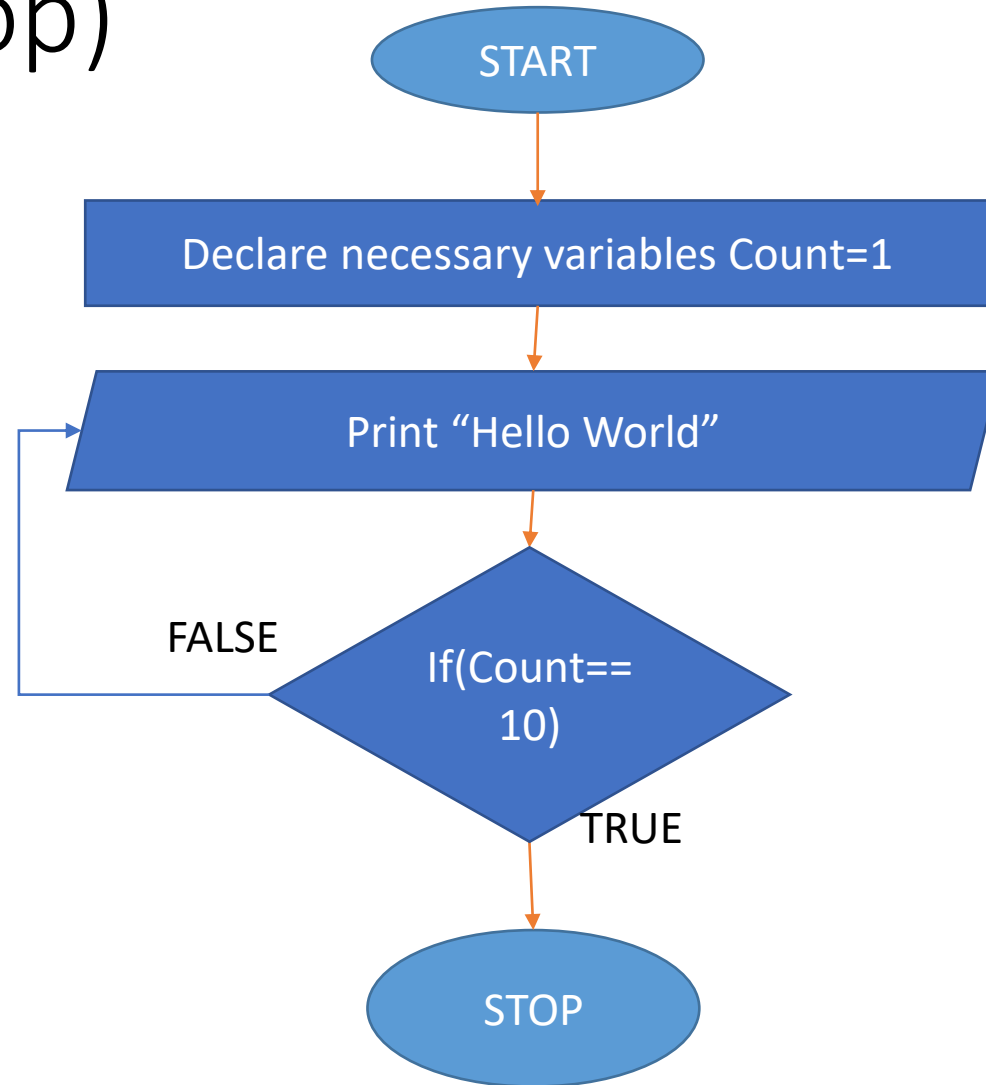
Example 2. Draw a flowchart to find sum of two numbers.



Example 3. Draw a flowchart to find whether a number is even or not(Branching)



Example 4. Draw a Flowchart to print “Hello world” 10 times(Loop)



## **Write an algorithm and draw flow chart to calculate factorial of given Number**

- Step 1: Start
- Step 2: Declare necessary Variable n, fact, i and Initialize Variable fact=1 and i=1
- Step 3: Read number from User
- Step 4: Repeat Until  $i \leq \text{number}$ 
  - 4.1 fact=fact\*i
  - 4.2 i=i+1
- Step 5: Print fact
- Step 6: Stop

**Write an algorithm and draw flow chart to check if given Number is prime or not.**

# Tutorials Sheet I

1. Write an algorithm and draw flowchart for finding sum of any two numbers.
2. Write an algorithm and draw flowchart for calculating Simple Interest.
3. Write an algorithm and draw flow chart to determine whether a number is positive or negative.
4. Write an algorithm and draw flow chart to test if a number is even or odd.
5. Write an algorithm and draw flow chart to find largest among two numbers.
6. Write an algorithm and draw flow chart to find larger number among three numbers.
7. Write an algorithm and draw flow chart to calculate factorial of given Number.
8. Write an algorithm and draw flow chart to check if given Number is prime or not.
9. Write an algorithm and draw flow chart to find square roots of quadratic equation (Both real and Imaginary)