Unit II
Computational Thinking and Programming - I

Problem and Problem Solving

In computer science, "problem" refers to a task or challenge that requires a solution. The process of identifying a problem, developing an algorithm, and implementing an algorithm to develop a computer program is called Problem Solving. Computers may be used to solve various daily life problems such as Train Ticket Booking, Online Shopping and Net-Banking etc.

Steps required for solving a problem

- Analyzing the problem
- Developing an Algorithm
- Coding
- Testing and Debugging

Analyzing the Problem

This stage focuses on understanding the problem. If we do not have a clear understanding of the problem, we may develop a computer program that cannot solve the problem correctly. In this stage, we figure out the inputs, the outputs and the processing required to convert the input into the output.

Developing Algorithm

This stage focuses on creating a logical sequence of instructions, called an Algorithm. The algorithm can be executed by a computer to generate the desired output. An algorithm has a distinct start and end point, as well as a defined number of steps. For a given problem, more than one algorithm may be possible and the most suitable algorithm may be chosen.

Algorithm for finding whether a number is Even or Odd

```
START
Step 1 → Take an integer number A
Step 2 → Divide A by 2, and store the remainder as r
Step 3 → If r is equal to 0, A is an Even Number
Step 4 → Else it is an Odd Number
STOP
```

Algorithm for finding whether a number is a Prime number or Not

```
START
Step 1 → Take an integer number A
Step 2 → Set divisor as 2
Step 3 → Set flag_value as True
Step 4 → Repeat from divisor  to A-1
            4a. divide A by divisor and store the remainder as r
            4b. If r is zero, set flag_value to False
            4c. Increment divisor by 1
Step 3 → If flag_value is False, A is not prime
Step 4 → Else A is prime
STOP
```

Coding

Coding is the process of creating computer programs.

Testing

Testing is a process to check if an application is working as expected (and not working abnormally). The main objective of Testing is to find errors.

Debugging

Debugging is the activity to fix the errors found in the application during the testing phase.

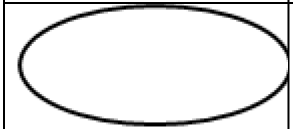
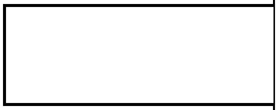
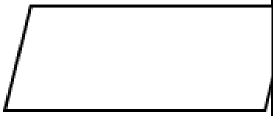
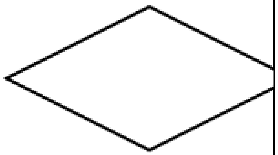
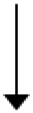
Representation of Algorithms

There are two common methods of representing an algorithm —flowchart and pseudocode.

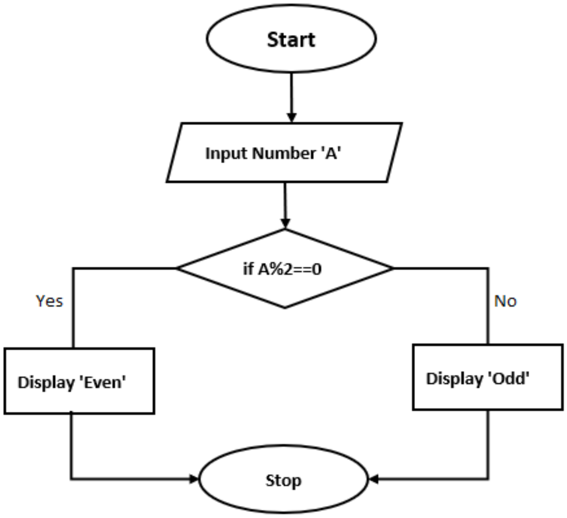
Flowchart

- A flowchart is a graphical representation of an algorithm.
- A flowchart contains various shapes which are connected by arrows, which shows the flow of control.

Shapes used in Flowchart

Shape	Shape Name	Usage
	Oval	Start/Stop
	Rectangle	Process
	Parallelogram	Input/Output
	Diamond	Decision/Condition
	Arrow	Flow of Control / Connections

Draw a flow-chart to identify whether a number taken as the input from the user is an even number or an odd number?

Algorithm	**Flow Chart**
START Step 1 → Take an integer number A as input Step 2 → Divide A by 2, and store the remainder as r Step 3 → If r is zero, Display 'Even' Step 4 → Else Display 'Odd' STOP	 Start Input Number 'A' if A%2==0 Yes No Display 'Even' Display 'Odd' Stop

Pseudocode

- Pseudocode is a way of representing an algorithm in readable and easy language.
- Pseudocode is not an actual program. So, it cannot be executed.
- Some of the frequently used keywords while writing pseudocode are INPUT, COMPUTE, PRINT IF/ELSE, START, STOP

Advantages of Pseudo-Code:

1. Easily convertible to a Programming Language
2. Easy to understand and read

Write a pseudocode for identifying if a number is even or odd?

```
INPUT number A
COMPUTE remainder as r = A%2
IF r ==0 PRINT 'Even'
ELSE PRINT 'Odd'
```

Decomposition

Decomposition is the process of breaking a complex computer problem into smaller parts that are easily manageable and solvable.