**Algorithm:**

**Definition:**

An algorithm is a step-by-step procedure or set of rules designed to perform a specific task or solve a problem. It is a well-defined, logical sequence of actions.

**Characteristics:**

1. **Finiteness**: An algorithm must always terminate after a finite number of steps.
2. **Definiteness**: Each step must be precisely defined and unambiguous.
3. **Input**: It should accept zero or more inputs.
4. **Output**: It must produce at least one output.
5. **Effectiveness**: Each step must be simple enough to be performed in a finite time.

**Advantages:**

* **Simplifies complex problems**: Breaks down large problems into manageable steps.
* **Efficient solution design**: Helps in planning before coding.
* **Language-independent**: Can be implemented in any programming language.

**Disadvantages:**

* **Time-consuming**: Writing a detailed algorithm takes time.
* **No direct implementation**: Needs to be translated into a programming language before execution.

**Examples:**

1. **Algorithm to find the largest of three numbers:**
   * Start
   * Input three numbers a, b, and c
   * If a > b and a > c, print "a is the largest"
   * Else if b > c, print "b is the largest"
   * Else, print "c is the largest"
   * End
2. **Algorithm to calculate the sum of n numbers:**
   * Start
   * Initialize sum = 0
   * For i = 1 to n, do sum = sum + i
   * Output sum
   * End

**Flowchart:**

**A flowchart** is a diagram that represents the sequence of steps in a process or a program. It uses different shapes to show different types of actions or steps, connected by arrows that indicate the flow of control.

* **Oval**: Represents the start or end of the program. It is used to signify where the flow begins and where it terminates.
* **Rectangle**: Represents a process or an operation, such as a calculation or an assignment in a program.
* **Parallelogram**: Represents input or output operations, like reading data from a user or displaying results on the screen.
* **Diamond**: Represents a decision point that branches the flow based on a condition, such as an if statement in programming.
* **Arrows**: Show the direction of flow, indicating the order in which the steps are executed.

By visualizing these steps, flowcharts help in understanding and planning the logic of a program before writing actual code.

**Advantages:**

* **Easy visualization**: Helps to clearly visualize the flow of a process or algorithm.
* **Simplifies debugging**: Easier to trace errors and inefficiencies in the flow.
* **Communication**: Acts as a tool for communicating processes with others.

**Disadvantages:**

* **Complexity**: Flowcharts can become complex for larger systems.
* **Time-consuming**: Creating detailed flowcharts may require significant time.

**Examples:**

1. **Flowchart to find the largest of three numbers:**

