Pre-Code Planning

Pre-code planning is a crucial step in the software development process that helps developers conceptualize and outline the logic of the software before actual coding begins. This phase includes the creation of flowcharts and pseudocode to visualize and verify the algorithm.

1. **Pseudocode**

- **Structure of Pseudocode**:
    - Use plain language to describe steps.
    - Indentation to show the structure of control statements (if-else, loops).
    - Clear and concise statements to represent actions and decisions.

Pseudocode is a high-level description of an algorithm that uses the structure of programming languages but is written in plain English. It outlines the logic of the program without focusing on the syntax of any particular programming language.

Pseudo code is a term which is often used in programming and algorithm based fields. It is a methodology that allows the programmer to represent the implementation of an algorithm. Simply, we can say that it's the cooked up representation of an algorithm. Often at times, algorithms are represented with the help of pseudo codes as they can be interpreted by programmers no matter what their programming background or knowledge is. Pseudo code, as the name suggests, is a false code or a representation of code which can be understood by even a layman with some school level programming knowledge.

Algorithm: It’s an organized logical sequence of the actions or the approach towards a particular problem. A programmer implements an algorithm to solve a problem. Algorithms are expressed using natural verbal but somewhat technical annotations.

Pseudo code: It’s simply an implementation of an algorithm in the form of annotations and informative text written in plain English. It has no syntax like any of the programming language and thus can’t be compiled or interpreted by the computer.

Advantages of Pseudocode

- Improves the readability of any approach. It’s one of the best approaches to start implementation of an algorithm.
- Acts as a bridge between the program and the algorithm or flowchart. Also works as a rough documentation, so the program of one developer can be understood easily when a pseudo code is written out. In industries, the approach of documentation is essential. And that’s where a pseudo-code proves vital.
- The main goal of a pseudo code is to explain what exactly each line of a program should do, hence making the code construction phase easier for the programmer.

How to write a Pseudo-code?

1. Arrange the sequence of tasks and write the pseudocode accordingly.
2. Start with the statement of a pseudo code which establishes the main goal or the aim. **Example:**

```
This program will allow the user to check
the number whether it's even or odd.
```

1. The way the if-else, for, while loops are indented in a program, indent the statements likewise, as it helps to comprehend the decision control and execution mechanism. They also improve the readability to a great extent.

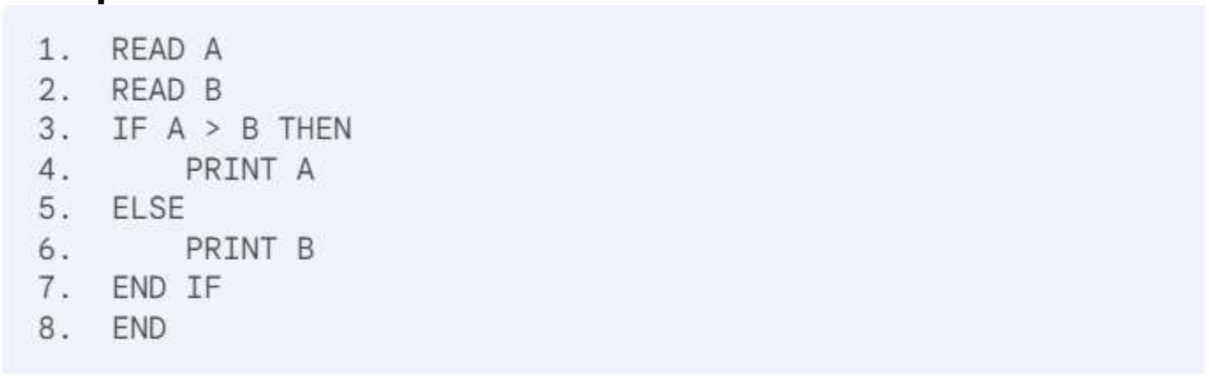
```
Example:
if "1"
    print response
        "I am case 1"
if "2"
    print response
        "I am case 2"
```

1. Use appropriate naming conventions. The human tendency follows the approach to follow what we see. If a programmer goes through a

pseudo code, his approach will be the same as per it, so the naming must be simple and distinct.

2. Use appropriate sentence casings, such as CamelCase for methods, upper case for constants and lower case for variables.
3. Elaborate everything which is going to happen in the actual code. Don’t make the pseudo code abstract.
4. Use standard programming structures such as ‘if-then’, ‘for’, ‘while’, ‘cases’ the way we use it in programming.
5. Check whether all the sections of a pseudo code is complete, finite and clear to understand and comprehend.
6. Don’t write the pseudo code in a complete programmatic manner. It is necessary to be simple to understand even for a layman or client, hence don’t incorporate too many technical terms.

- **Example of Pseudocode**:



```
READ A
READ B
IF A > B THEN
    PRINT A
ELSE
    PRINT B
END IF
END
```

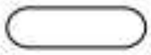
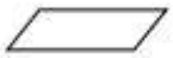
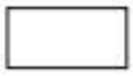
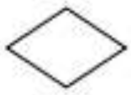
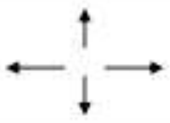
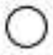
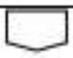
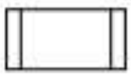
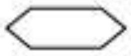
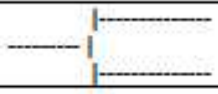
- **Benefits of Pseudocode**:
    - Bridges the gap between the algorithm and actual code.
    - Facilitates communication among developers and stakeholders.
    - Helps identify logical errors before coding begins.
    - Provides a clear plan to follow during coding.

2. Flow Chart

Flowcharts are nothing but the graphical representation of the data or the algorithm for a better understanding of the code visually. It displays step-by-step solutions to a problem, algorithm, or process. It is a pictorial way of representing steps that are preferred by most beginner-level programmers to understand algorithms of computer science, thus it contributes to troubleshooting the issues in the algorithm. A flowchart is a picture of boxes that indicates the process flow sequentially. Since a flowchart is a pictorial representation of a process or algorithm, it's easy to interpret and understand the process. To draw a flowchart, certain rules need to be followed which are followed by all professionals to draw a flowchart and are widely accepted all over the countries

Symbols Used in Flowcharts: Different types of boxes are used to make flowcharts flowchart Symbols. All the different kinds of boxes are connected by arrow lines. Arrow lines are used to display the flow of control. Let's learn about each box in detail.

- **Oval**: Represents the start and end points.
- **Rectangle**: Represents a process or action.
- **Diamond**: Represents a decision point.
- **Parallelogram**: Represents input or output.
- **Arrows**: Indicate the direction of flow.

Symbol Name	Symbol	function
Oval		Used to represent start and end of flowchart
Parallelogram		Used for input and output operation
Rectangle		Processing: Used for arithmetic operations and data-manipulations
Diamond		Decision making. Used to represent the operation in which there are two/three alternatives, true and false etc
Arrows		Flow line Used to indicate the flow of logic by connecting symbols
Circle		Page Connector
		Off Page Connector
		Predefined Process /Function Used to represent a group of statements performing one processing task.
		Preprocessor
		Comments

Uses of Flowcharts in Computer Programming/Algorithms

The following are the uses of a flowchart:

- It is a pictorial representation of an algorithm that increases the readability of the program.
- Complex programs can be drawn in a simple way using a flowchart.

- It helps team members get an insight into the process and use this knowledge to collect data, detect problems, develop software, etc.
- A flowchart is a basic step for designing a new process or adding extra features.
- Communication with other people becomes easy by drawing flowcharts and sharing them.

When to Use Flowchart?

Flowcharts are mainly used in the below scenarios:

- It is most importantly used when programmers make projects. As a flowchart is a basic step to make the design of projects pictorially, it is preferred by many.
- When the flowcharts of a process are drawn, the programmer understands the non-useful parts of the process. So flowcharts are used to separate sound logic from the unwanted parts.
- Since the rules and procedures of drawing a flowchart are universal, a flowchart serves as a communication channel to the people who are working on the same project for better understanding.
- Optimizing a process becomes easier with flowcharts. The efficiency of the code is improved with the flowchart drawing.

Types of Flowcharts

Three types of flowcharts are listed below:

1. **Process flowchart:** This type of flowchart shows all the activities that are involved in making a product. It provides a pathway to analyze

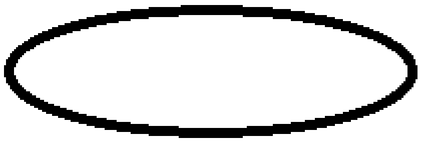
the product to be built. A process flowchart is most commonly used in process engineering to illustrate the relation between the major as well as minor components present in the product. It is used in business product modeling to help understand employees about the project requirements and gain some insight into the project.

2. **Data flowchart:** As the name suggests, the data flowchart is used to analyze the data, specifically it helps in analyzing the structural details related to the project. Using this flowchart, one can easily understand the data inflow and outflow from the system. It is most commonly used to manage data or to analyze information to and fro from the system.
3. **Business Process Modeling Diagram:** Using this flowchart or diagram, one can analytically represent the business process and help simplify the concepts needed to understand business activities and the flow of information. This flowchart illustrates the business process and models graphically which paves the way for process improvement.

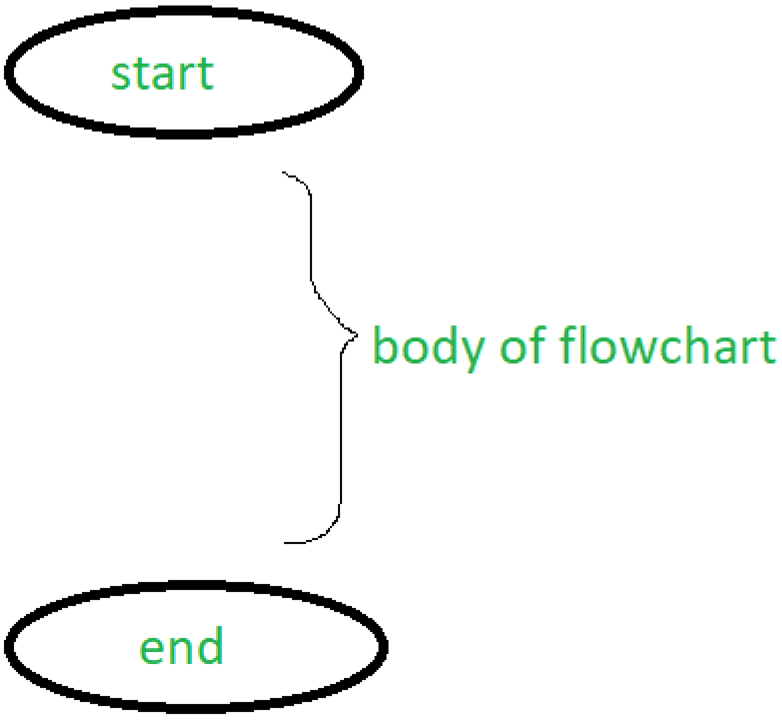
Types of boxes used to make a flowchart

There are different types of boxes that are used to make flowcharts. All the different kinds of boxes are connected to one another by arrow lines. Arrow lines are used to display the flow of control. Let's learn about each box in detail.

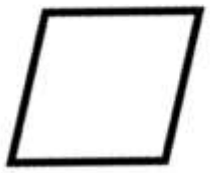
1. Terminal



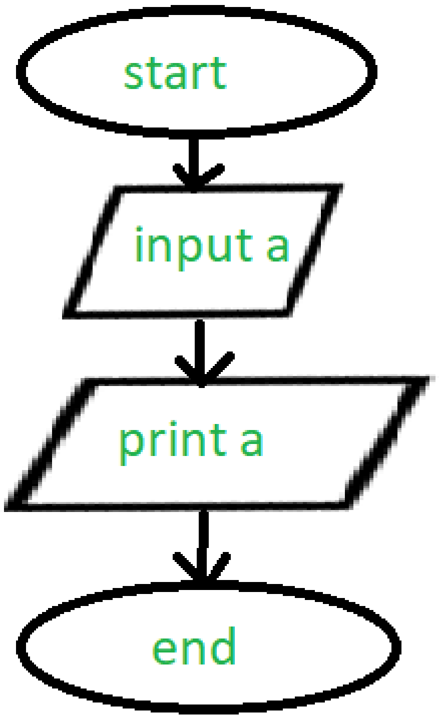
This box is of an oval shape which is used to indicate the start or end of the program. Every flowchart diagram has an oval shape that depicts the start of an algorithm and another oval shape that depicts the end of an algorithm. For example:



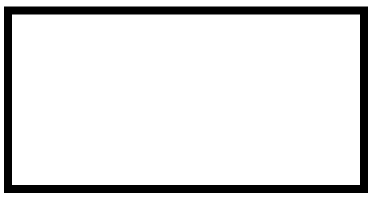
2. Data



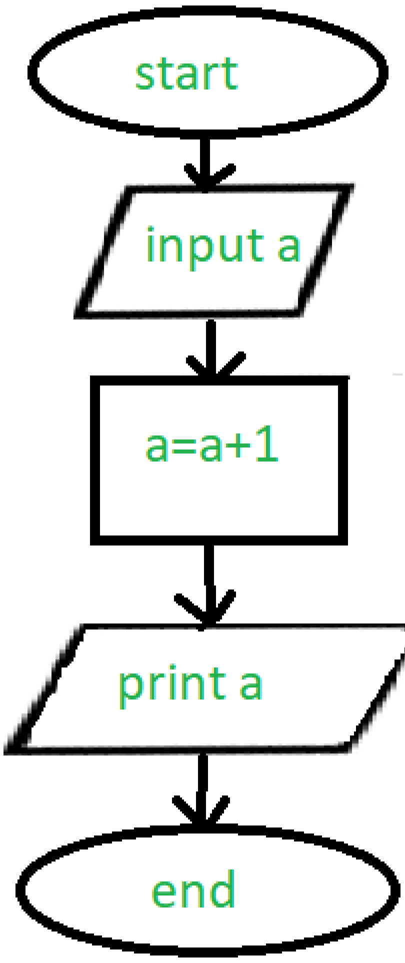
This is a parallelogram-shaped box inside which the inputs or outputs are written. This basically depicts the information that is entering the system or algorithm and the information that is leaving the system or algorithm. For example: if the user wants to input a from the user and display it, the flowchart for this would be:



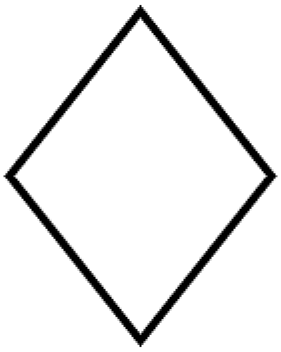
3. Process



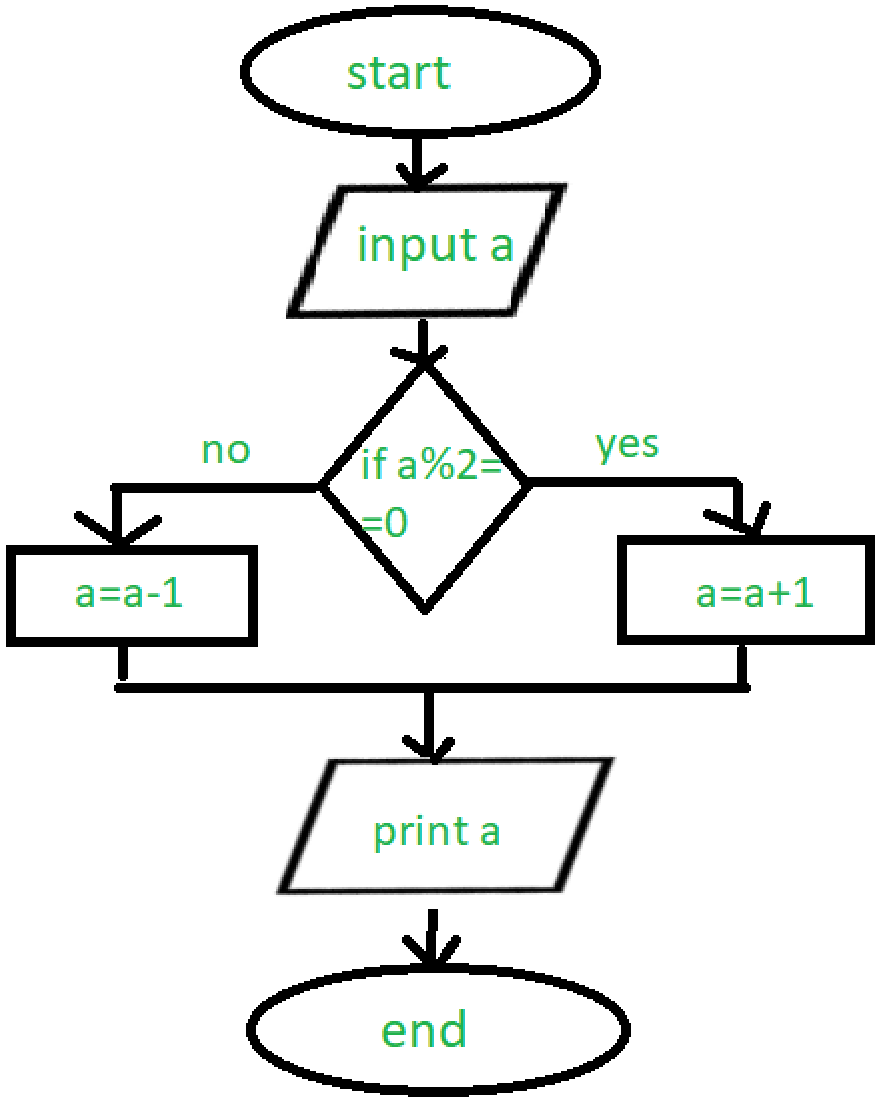
This is a rectangular box inside which a programmer writes the main course of action of the algorithm or the main logic of the program. This is the crux of the flowchart as the main processing codes is written inside this box. For example: if the programmer wants to add 1 to the input given by the user, he/she would make the following flowchart:



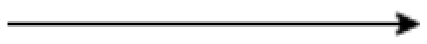
4. Decision



This is a rhombus-shaped box, control statements like if, condition like a > 0, etc are written inside this box. There are 2 paths from this one which is "yes" and the other one is "no". Every decision has either yes or no as an option, similarly, this box has these as options. For example: if the user wants to add 1 to an even number and subtract 1 if the number is odd, the flowchart would be:

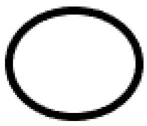


5. Flow

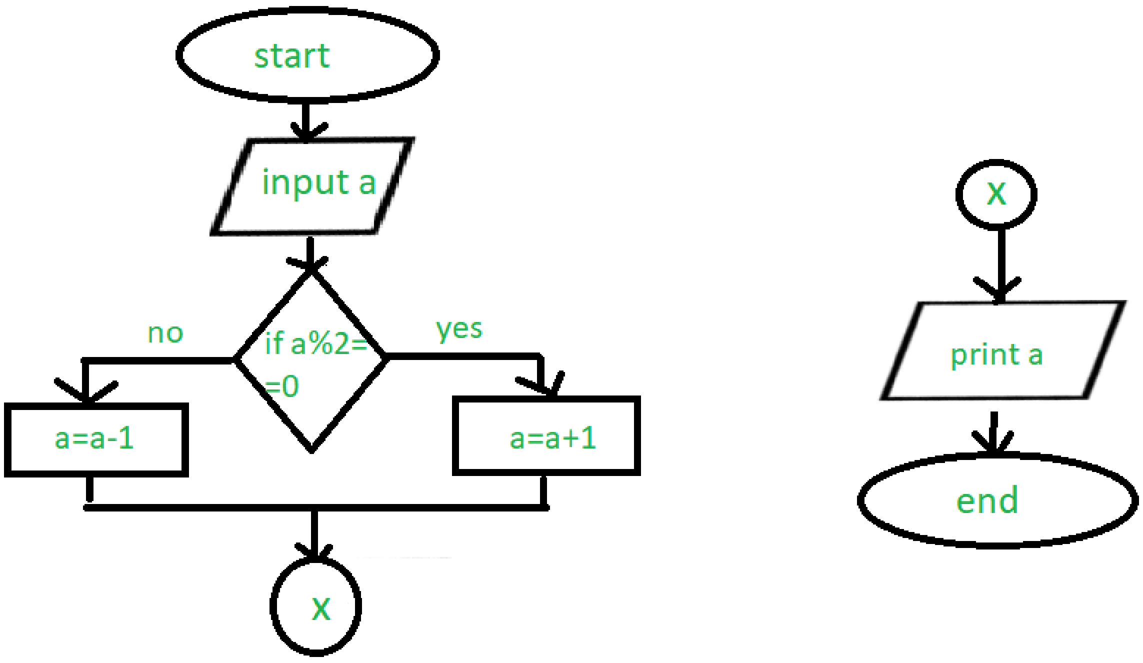


This arrow line represents the flow of the algorithm or process. It represents the direction of the process flow. in all the previous examples, we included arrows in every step to display the flow of the program. arrow increases the readability of the program.

6. On-Page Reference



This circular figure is used to depict that the flowchart is in continuation with the further steps. This figure comes into use when the space is less and the flowchart is long. Any numerical symbol is present inside this circle and that same numerical symbol will be depicted before the continuation to make the user understand the continuation. Below is a simple example depicting the use of On-Page Reference



Advantages of Flowchart

- It is the most efficient way of communicating the logic of the system.

- It acts as a guide for a blueprint during the program design.
- It also helps in the debugging process.
- Using flowcharts we can easily analyze the programs.
- flowcharts are good for documentation.

Disadvantages of Flowchart

- Flowcharts are challenging to draw for large and complex programs.
- It does not contain the proper amount of details.
- Flowcharts are very difficult to reproduce.
- Flowcharts are very difficult to modify.

2. Verify Algorithm

Once you have your flowchart or pseudocode, it's essential to verify its logic. Here are some methods for verification:

- **Walkthrough**: Manually trace the steps using test cases (different inputs) to ensure the process produces the expected output.
- **Peer Review**: Share your flowchart or pseudocode with colleagues or other stakeholders for feedback on clarity and logic.
- **Comparison**: Compare your algorithm with existing solutions to check if your approach is sound and efficient.

Conclusion

Pre-code planning, including the use of flowcharts and pseudocode, is an essential phase in the software development life cycle. It helps developers understand and refine the algorithm before coding, ensuring a smoother development process and reducing the likelihood of errors. Verification of the algorithm through dry runs, peer reviews, and test cases further ensures the logic is sound and ready for implementation.