

Problem Solving Through programming in C

Course Code:ONL1001

Ms. Shubhra dwivedi

Assistant Professor

School - SCOPE

VIT-AP Amaravati

Shubhra.d@vitap.ac.in

Introduction

- ▶ The computer is a machine that follows the set of instructions called a **program**. Any computing has to be performed independently without depending on the programming language and the computer.
- ▶ The problem solving techniques involves the following steps
 1. Define the problem.
 2. Formulate the mathematical model.
 3. Develop an algorithm.
 4. Write the code for the problem.
 5. Test the program.

Problem solving approach

► 1. Define the Problem

- A clear and concise problem statement is provided.
- The problem definition should specify the input and output.
- Full knowledge about the problem is needed.

Example: TO FIND THE AVERAGE OF TWO NUMBERS.

► 2. Formulate the Mathematical Problem

- Any technical problem provided can be solved mathematically.
- Full knowledge about the problem should be provided along with the underlying mathematical concept.

Example: $(data1 + data2) / 2$

Continue....

► 3. Develop an Algorithm

- An algorithm is the sequence of operations to be performed.
- It gives the precise plan of the problem.
- An algorithm can be of flowchart or pseudo code.

Example: Problem Definition: TO FIND THE AVERAGE OF TWO NUMBERS.

Algorithm:

1. Set the sum of the data values to 0.
2. Set the count of the data values to zero.
3. As long as the data values exist, add the next data value to the sum and add 1 to the count.

Continue.....

4. To compute the average, divide the sum by the count.

5. Print the average.

Task: To find the average of 20 and 30 manually.

$20 + 30 = 50$; $50/2 = 25$.

► 4. Write the Code for the Problem

- The algorithm developed must be converted to any programming language.
- The compiler will convert the program code to the machine language which the computer can understand.

► 5. Test the Program

- Testing involves checking errors both syntactically and semantically.

Continue.....

- The errors are called as “bugs”.
- When the compiler finds the bugs, it prevents compiling the code from programming language to machine language.
- Check the program by providing a set of data for testing.

Program

A **program** consists of a series of instructions that a computer processes to perform, the required operation. In addition, it also contains some fixed data, required to perform the instructions, and the process of defining those instructions and data.

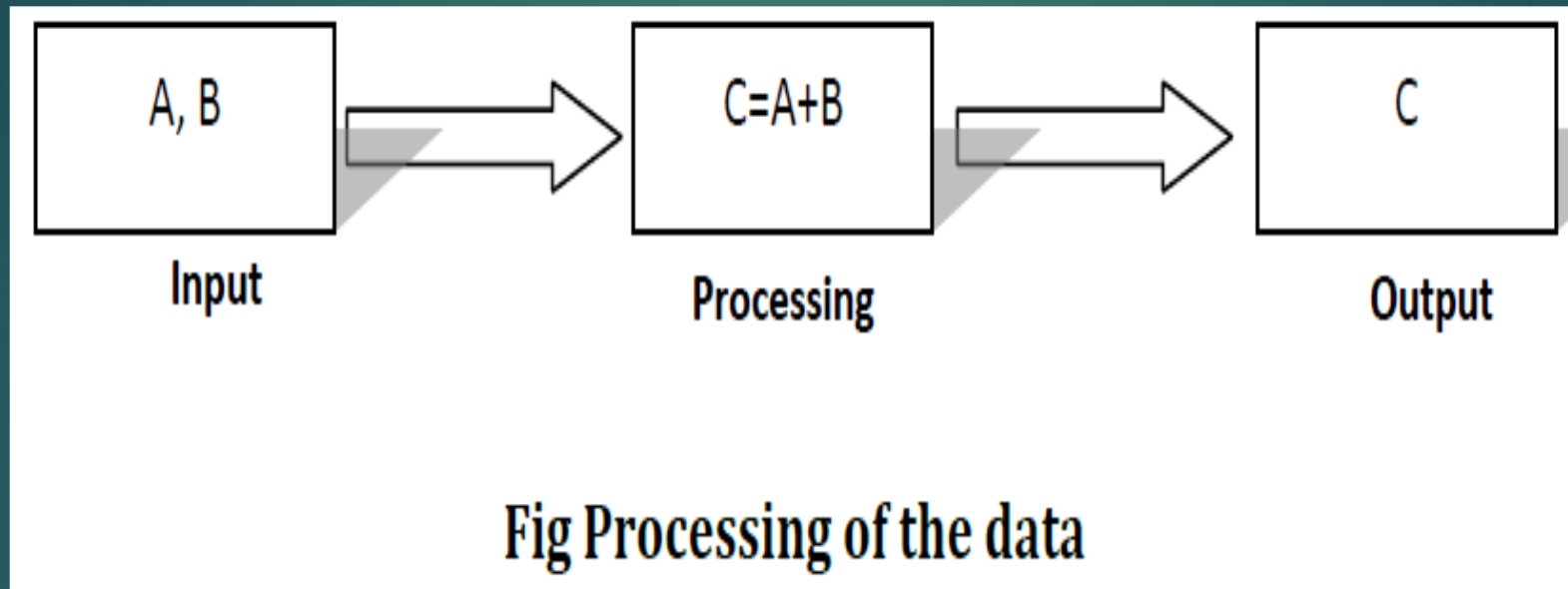
► In order to design a program, a programmer must determine three basic rudiments:

- The instructions to be performed.
- The order in which those instructions are to be performed.
- The data required to perform those instructions.

► Example: WRITE A PROGRAM TO ADD TWO NUMBERS

1. Input two numbers.
2. Add these two numbers.
3. Display the output.

Continue..



Flowchart

What is an Flowchart ?








- ▶ **Flowchart** is a diagrammatic representation of sequence of logical steps of a program.

OR

- ▶ A **flowchart** is simply a graphical representation of steps. It shows steps in a sequential order, and is widely used in presenting **flow** of algorithms, workflow or processes. Typically, **flowchart** shows the steps as boxes of various kinds, and their order by connecting them with arrows

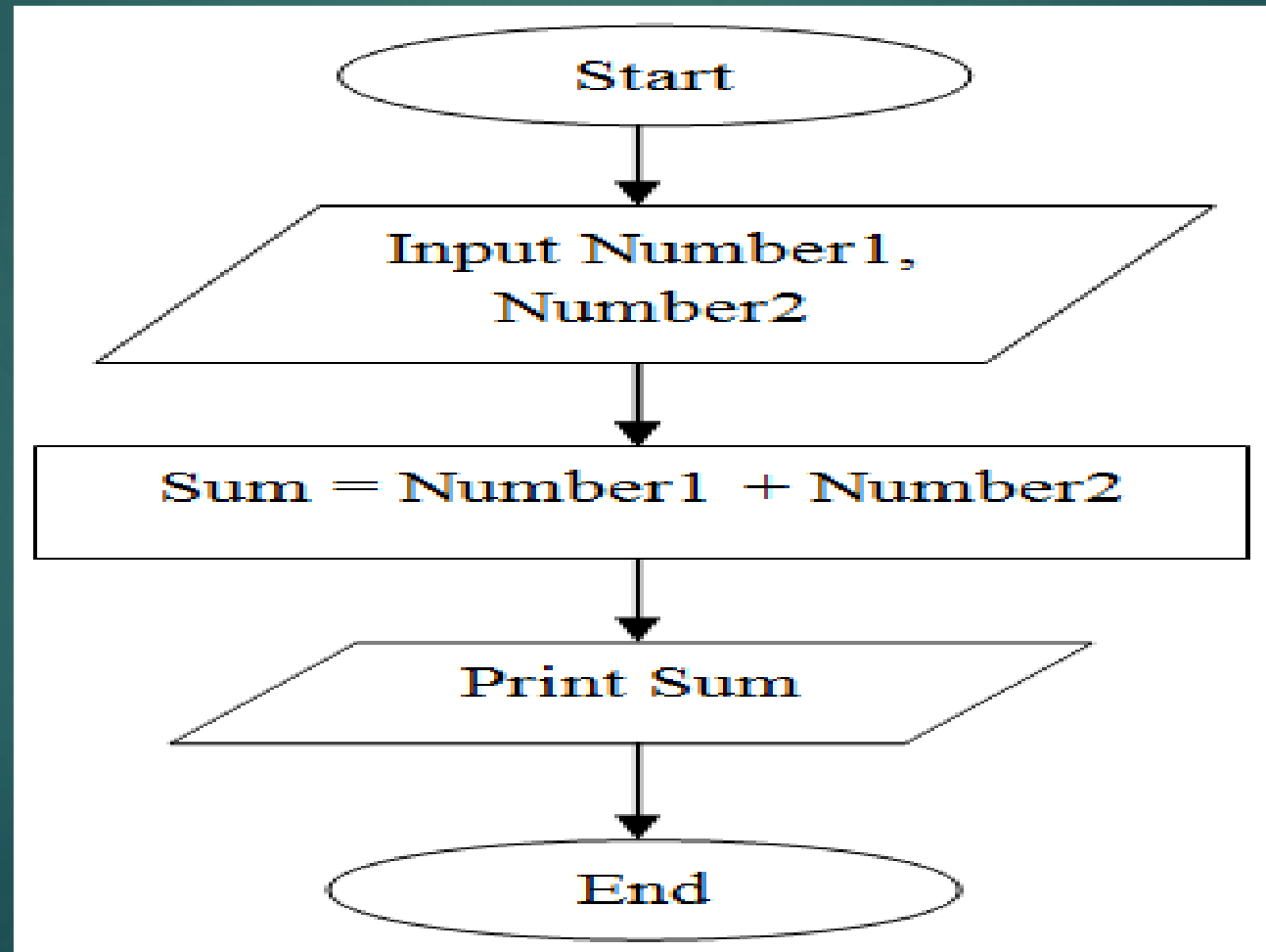
Continue....

► Flowchart Symbols

Symbol	Symbol Name	Purpose
	Start/Stop	Used at the beginning and end of the algorithm to show start and end of the program.
	Process	Indicates processes like mathematical operations.
	Input/ Output	Used for denoting program inputs and outputs.
	Decision	Stands for decision statements in a program, where answer is usually Yes or No.
	Arrow	Shows relationships between different shapes.
	On-page Connector	Connects two or more parts of a flowchart, which are on the same page.
	Off-page Connector	Connects two parts of a flowchart which are spread over different pages.

Continue....

- ▶ Flowchart for Sum of two numbers



Benefits of Flowcharts

- ▶ A flowchart helps to clarify how things are currently working and how they could-be improved. The reasons for using flowcharts as a problem-solving tool are given below.

i) Makes Logic Clear

ii) Communication

iii) Effective Analysis

iv) Useful in Coding

v) Proper Testing and Debugging

vi) Appropriate Documentation

Limitations of Flowcharts

- ▶ Flowchart can be used for designing the basic concept of the program in pictorial form but cannot be used for programming purposes. Some of the limitations of the flowchart are given as follows:

i) Complex

ii) Costly

iii) Difficult to Modify

iv) No Update

PSEUDO CODE



- ▶ Pseudo code is made up of two words: pseudo and code. Pseudo means imitation and code refers to instructions, written in a programming language. As the name suggests, pseudo code is not a real programming code, but it models and may even look like programming code.
- ▶ Pseudo code uses plain English statements rather than symbols, to represent the processes of a computer program. It is also known as PDL (Program Design Language), as it emphasizes more on the design aspect of a computer program or structured English, because usually pseudo code instructions are written in normal English, but in a structured way.

Continue...

- ▶ If an algorithm is written in English, the description may be at such a high level that it may prove difficult to analyze the algorithm and then to transform it into actual code.
- ▶ If instead, the algorithm is written in code, the programmer has to invest a lot of time in determining the details of an algorithm, which he may choose not to implement (since, typically, algorithms are analyzed before deciding which one to implement).
- ▶ Therefore, the goal of writing pseudo code is to provide a high-level description of an algorithm, which facilitates analysis and eventual coding, but at the same time suppresses many of the details that are insignificant.

Continue....

► **Example: The pseudo code given below calculates the area of a rectangle.**

1. PROMPT the user to enter the height of the rectangle
2. PROMPT the user to enter the width of the rectangle
3. COMPUTE the area by multiplying the height with width
4. DISPLAY the area
5. STOP

Continue...

- ▶ Pseudo code uses some keywords to denote programming processes. Some of them are:
 - **Input:** READ, OBTAIN, GET, and PROMPT
 - **Output:** PRINT, DISPLAY, and SHOW
 - **Compute:** COMPUTE, CALCULATE, and DETERMINE
 - **Initialise:** SET and INITIALISE
 - **Add One:** INCREMENT

Pseudo code Guidelines

Writing pseudo code is not a difficult task. Even if you do not know anything about the computers or computer languages, you can still develop effective and efficient pseudo codes, if you are writing in an organized manner.

Here are a few general guidelines for developing pseudo codes:

- ▶ Statements should be written in simple English and should be programming language independent. Remember that pseudo codes only describe the logic plan to develop a program, it is not programming.
- ▶ Steps must be understandable, and when the steps are followed, they must produce a solution to the specified problem.
- ▶ Pseudo codes should be concise.
- ▶ Each instruction should be written in a separate line and each statement in pseudo code should express just one action for the computer.
- ▶ Capitalize keywords such as READ, PRINT, and so on.
- ▶ Each set of instructions is written from top to bottom, with only one entry and one exit.
- ▶ It should allow for easy transition from design to coding in programming language.

Benefits of Pseudo code

Pseudo code provides a simple method of developing the program logic as it uses everyday language to prepare a brief set of instructions in the order in which they appear in the completed program. It allows the programmer to focus on the steps required to solve a program rather than on how to use the computer language. Some of the most significant benefits of pseudo code are:

- ▶ Since it is language independent, it can be used by most programmers.
- ▶ It is easier to develop a program from a pseudo code than with a flowchart.
- ▶ Often, it is easy to translate pseudo code into a programming language, a step which can be accomplished by less experienced programmers.
- ▶ Unlike flowcharts, pseudo code is compact and does not tend to run over many pages. It's simple structure and readability makes it easier to modify.

Limitations Of Pseudo code

Although pseudo code is a very simple mechanism to simplify problem-solving logic, it has its limitations. Some of the most notable limitations are:

- ▶ It does not provide visual representation of the program's logic.
- ▶ There are no accepted standards for writing pseudo codes.
- ▶ Pseudo code cannot be compiled nor executed, and there are no real formatting or syntax rules.