Chapter 1 : Introduction To C

Introduction

The C Language is developed by Dennis Ritchie for creating system applications that directly interact with the hardware devices such as drivers, kernels, etc.

C programming is considered as the base for other programming languages, that is why it is known as mother language.

In this chapter, we will learn problem solving and steps in problem solving, basic tools for designing solution as algorithms, flowcharts, pseudo code, compilation process, c tokens etc.

Algorithms in C Language

- Algorithm is a step-by-step procedure, which defines a set of instructions to be executed in a certain order to get the desired output.
- Algorithms are generally created independent of underlying languages, i.e. an algorithm can be implemented in more than one programming language.

Characteristics of an Algorithm

- **Input** − An algorithm should have 0 or more well-defined inputs.
- **Output** − An algorithm should have 1 or more well-defined outputs, and should match the desired output.
- **Finiteness** − Algorithms must terminate after a finite number of steps.
- **Feasibility** − Should be feasible with the available resources.
- **Independent** − An algorithm should have step-by-step directions, which should be independent of any programming code.

How to Write an Algorithm

Problem − Design an algorithm to add two numbers and display the result.

```
Step 1 − START
Step 2 − declare three integers a, b & c
Step 3 − define values of a & b
Step 4 − add values of a & b
Step 5 − store output of step 4 to c
Step 6 − print c
Step 7 – STOP
```

Flowcharts

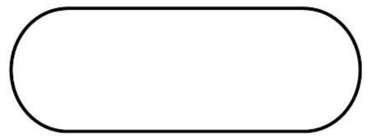
Flowchart is a diagrammatic representation of sequence of logical steps of a program.

Flowcharts use simple geometric shapes to depict processes and arrows to show relationships and process/data flow.

Simple Flowcharting Symbols

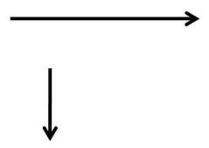
1. Terminal

The rounded rectangles, or terminal points, indicate the flowchart's starting and ending points.



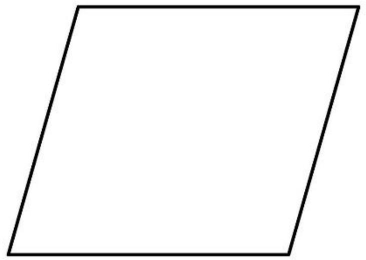
2. Flow Lines

Note: The default flow is left to right and top to bottom (the same way you read English). To save time arrowheads are often only drawn when the flow lines go contrary the normal.



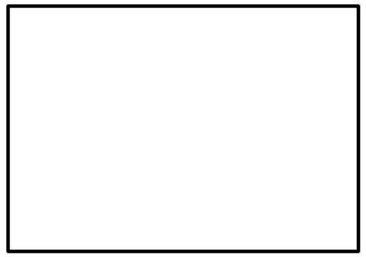
3. Input/Output

The parallelograms designate input or output operations.



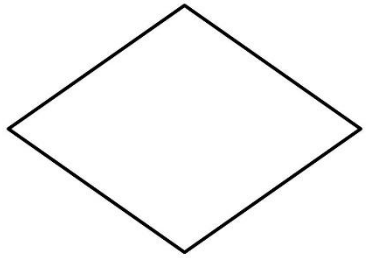
4. Process

The rectangle depicts a process such as a mathematical computation, or a variable assignment.



5. Decision

The diamond is used to represent the true/false statement being tested in a decision symbol.

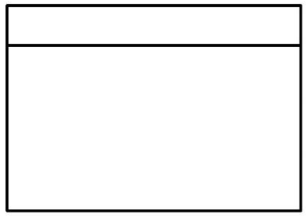


Advanced Flowcharting Symbols

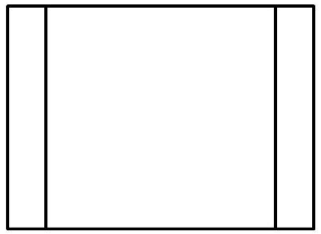
- **Module Call**

A program module is represented in a flowchart by rectangle with some lines to distinguish it from process symbol. Often programmers will make a distinction between program control and specific task modules as shown below.

- **Local module**: usually a program control function.



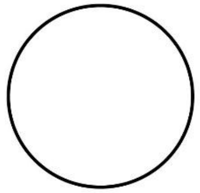
- **Library module**: usually a specific task function.



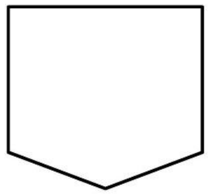
• Connectors

Sometimes a flowchart is broken into two or more smaller flowcharts. This is usually done when a flowchart does not fit on a single page, or must be divided into sections. A connector symbol, which is a small circle with a letter or number inside it, allows you to connect two flowcharts on the same page. A connector symbol that looks like a pocket on a shirt, allows you to connect to a flowchart on a different page.

<u>On-Page Connector</u>

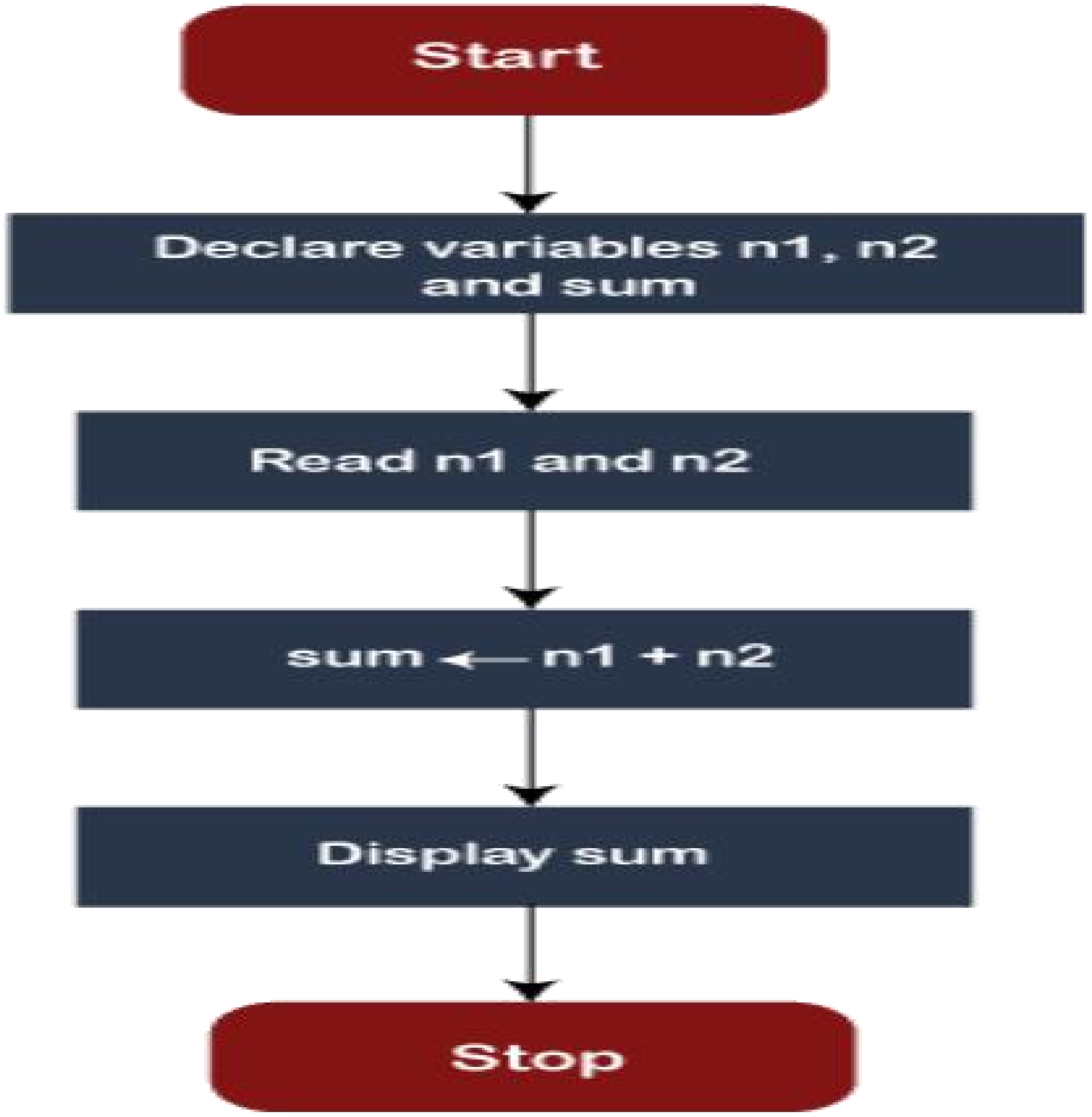


<u>Off-Page Connector</u>



<u>Example</u> :

Design a flowchart for adding two numbers entered by the user.


Start
Declare variables n1, n2 and sum
Read n1 and n2
sum ← n1 + n2
Display sum
Stop

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

I. Improves the readability of any approach. It’s one of the best approaches to start implementation of an algorithm.

II. 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.

III. 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.

Disadvantages of Pseudocode

I. Pseudocode does not provide a visual representation of the logic of programming.

II. There are no proper format for writing the for pseudocode.

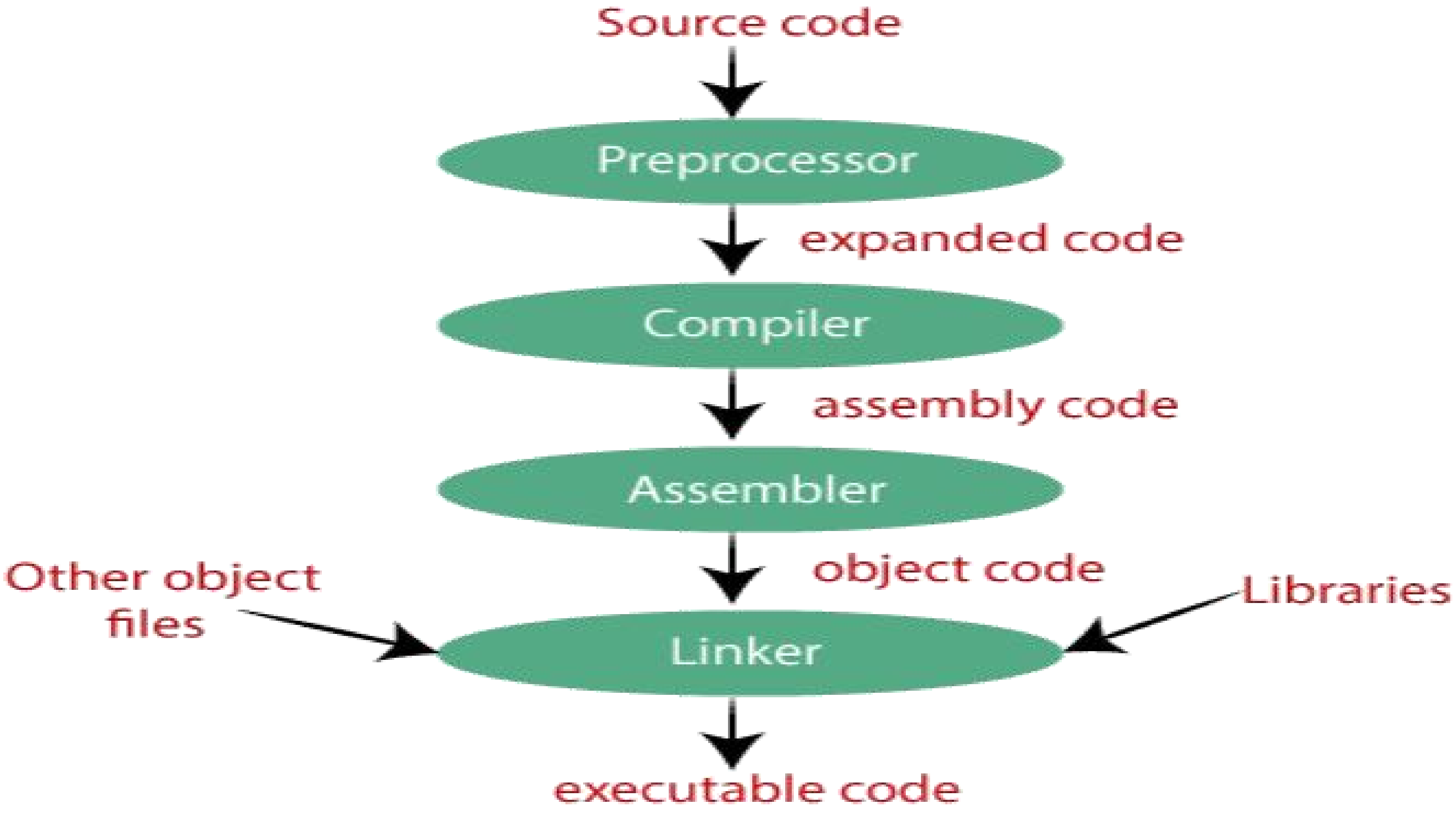
III. In Pseudocode their is extra need of maintain documentation.

IV. In Pseudocode their is no proper standard very company follow their own standard for writing the pseudocode.

Compilation Process

The compilation is a process of converting the source code into object code. It is done with the help of the compiler. The compiler checks the source code for the syntactical or structural errors, and if the source code is error-free, then it generates the object code.

The c compilation process converts the source code taken as input into the object code or machine code. The compilation process can be divided into four steps, i.e., Pre-processor, Compiling, Assembling, and Linking.


Source code
Preprocessor
expanded code
Compiler
assembly code
Assembler
object code
Other object files
Libraries
Linker
executable code

C Tokens

- Tokens are the smallest elements of a program, which are meaningful to the compiler.
- The types of tokens: Keywords, Identifiers, Constant, Strings, Operators, Special Symbols etc.