

Class Notes

Published Date: November, 2022

PPS: UNIT-1

- ❖ **Introduction to Components of a Computer System**
- ❖ **Idea of Algorithm**
- ❖ **Programming Basics**

FALL SEMESTER, YEAR (I/II sem, 1st yr)

*FALL SESSION (2022-23)
(PPS)*

MS. SHWETA TIWARI

Published Date: November, 2022

shwetatiwario8@recabn.ac.in
shwetatiwario8aug@gmail.com

**By SHWETA TIWARI
Under On: UNIT-1**

PREPARED FOR
Engineering Students
All Engineering College

PREPARED BY
SHWETA TIWARI

Unit Classified into Chapters

PPS: UNIT-1:

**Chapter-1 INTRODUCTION TO PROGRAMING
LANGUAGES**

Chapter-2 BASIC OF 'C' LANGUAGE

Chapter 1

INTRODUCTION TO PROGRAMING LANGUAGES

Programming Languages

Programming languages are the languages used to make programs that can be executed by the computer using compiler or interpreter. We can create programs, application software, web sites etc with the help of these programming languages. Programming language can be categorized as below.

- Low Level Language.
- Assembly Language.
- High Level Language.

Low Level Language (Machine Language)

Machine Language is a language that a computer can understand, but it is very difficult for the human beings to understand. Programs written in machine language consist of machine instructions. Machine language consists of strings of binary numbers (0, 1). Each microprocessor can have different machine language.

Advantages:

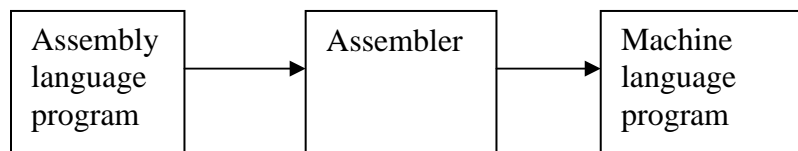
1. Machine language instructions are directly executed, so there is no compilation or translation needed, that's why it is fast in execution.
2. Machine language uses computer memory very well.

Disadvantages:

1. Machine language is a machine dependent.
2. It is very difficult to remember the codes.
3. Modification is difficult in machine language program.
4. High programming skills are required to develop programs in machine language.

Assembly Language

In 1950s operation codes [opcodes] were developed. Assembler is used to convert these codes into machine language. Like it is difficult to remember any code like (01010001 for add) instead of this binary code in assembly language we use **ADD** to add contents of two registers.



Advantages:

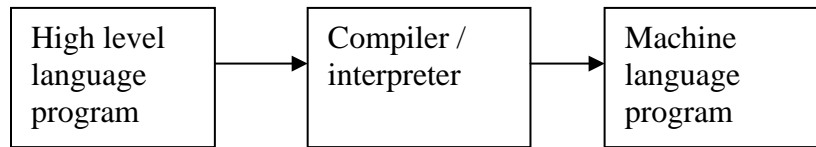
1. Assembly language programs are easier to use than machine language.
2. Error detection is possible and easy.
3. Modification is simple in assembly language program.
4. Less efforts are required as compared to machine language in writing the program.

Disadvantages:

1. These languages are fully machine dependent language.
2. Good programming skills are required to develop a program in assembly language.

HIGH-LEVEL LANGUAGE (HLL)

High level languages are very powerful and these are similar to English language. C, C++, Java, Visual Basic etc are the examples of HLL.

**Advantages:**

1. HLL are easy to learn and understand.
2. It is very easy to write the program.
3. Mostly the syntax of all high level languages are similar.
4. Knowledge of hardware is not essential to write program.
5. Program written in HLL are easy to modify.

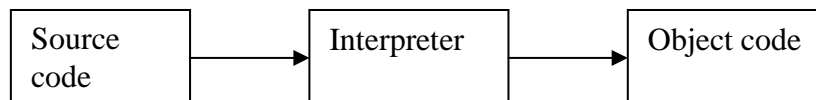
Disadvantages:

1. Compiler or interpreter is required to convert HLL into low level language.
2. Due to conversion these are slower in speed.
3. Specific hardware and software are required for some high level language.

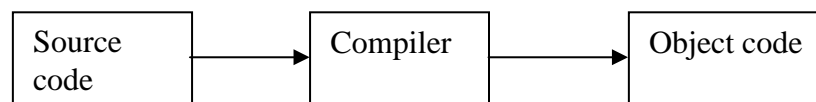
Compiler And Interpreter

Compiler and interpreter both are used to convert the high-level language program into the object code.

Interpreter:- Interpreter convert the program of High level language into Low level language line by line.



Compiler: - Compiler converts the whole program at a time from High level Language into Low level language.

**Features of a good programming language**

Every programming language has some speciality in it, which make a language suceesful. These specialities help languages to be popular in the computer world. These specialities are also known as features, which are as follows:-

- 1) **Easy to use:-** A good programming language must be simple & easy to learn. The concept should be simple & easy to learn. The concept should be clear & can be easily grasped. The simple languages are easy to develop & to implement.

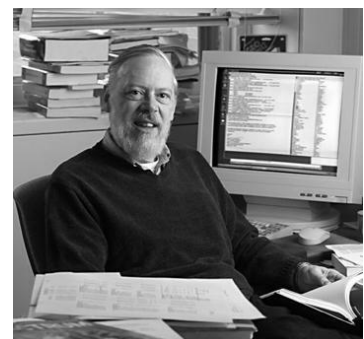
- 2) **Extensibility**:- A good programming language should be extended to sub-program. It increases the readability of the language.
- 3) **Efficiency**:- Programs written in a good language should be efficient. That is, program should take less space & time to run efficiently.
- 4) **Better documentation** :- A good programming language should be easily understood by programmer. Coding should be precise and logics should be upto mark.
- 5) **Flexibility**:- A good programming should have a mechanism to control the computers CPU, memory & registers.
- 6) **Fewer errors**:- A good programming language should have compilers & interpreters which can automatically detect syntax errors. So these can be easily corrected by the programmer.

COMMON LANGUAGES

- 1) **BASIC** (BEGINNERS ALL PURPOSE SYMBOLIC INSTRUCTION CODE):- All purpose symbolic instruction) It is designed in 1964 by John George Kemeny and Thoman kurtic. It is easy for beginners, provide error messages, no need to understand hardware. Its based on fortran-2nd + ALGOL.
- 2) **ALGOL**(ALGORITHM C LANGUAGE) :- Developed in 1950. It is the first language that implement nested functions.
- 3) **FORTRAN**(FORMULA TRANSLATOR):- It is developed for numeric computation and scientific calculation and it is developed by IBM in 1950.
- 4) **COBOL**:- Common Business-oriented language design for business, finance administrative systems. Cobol 2002 also support object-oriented programming.
- 5) **LISP**:- (List processing) It is used to implement lists that is a major data structure developed in 1958.
- 7) **BCPL**:- (Basic combine programming language) It can perform task of many language 'C' is based on this language.
- 7) **LOGO**:- It gives graphics. It is develop in 1968. It is used to create simple programs that can contain simple graphics.
- 8) **SQL** (STRUCTURE QUERY LANGUAGE):- It is used in databases to create database and access information from it.

History Of 'C' language

Many programming languages have been created according to the programmer requirement. In earlier days programmer had to learn many languages to perform different tasks. A language called BCPL(*Basic Combined Programming Language*) designed by **Martin Richards** of the University of Cambridge in 1966 that perform work of many types, this language is also known as 'B' language.C



language is the modified version of basic combined programming language called 'B'. Using this language Dennis Ritchie developed 'C' language at the AT&T bell laboratories in 1972. It is one of the most popular computer language.

This language is also known as High level language as it is closer to human languages and faster from machine language.

NOTE:- C is a case sensitive language . It means uppercase alphabets are different from lowercase alphabets e.g. 'A' is totally different from 'a'

Why we use 'C' language

'C' was developed in 1972 but still programmers use this language due to following reasons.

- (1) **C is portable** means program written for one computer may run successfully on other computers.
- (2) **C is fast** means the executable program runs very fast.
- (3) **C is compact** i.e. the statements written in 'C' language are generally short and very powerful. Several operators can be used in one statement.
- (4) C language has both the simplicity of high level language and speed of low level language, so it is also known as **middle level language**.
- (5) 'C' Compiler is **easily available**.
- (6) It **supports pointers** that can be used to create system programming.
- (7) It can be **run on UNIX** operating system also.
- (8) Large number **of inbuilt** functions are available.
- (9) C has ability to **extend** itself. Users can add their own functions to the c library.
- (10) Program of c can be **divided into small modules** called functions. These function can be reused any time by the user.

Limitations Of 'C' language or Constraints in C

- (1) 'C' language does not support exception handling.
- (2) Bound checking is not performed by the 'C' compiler.
- (3) 'C' language does not support concepts of OOP.
- (4) 'C' language does not support Unicode characters.
- (5) Pointers are difficult to understand for the beginners.

C/C++ compilers

- Turbo C++
- Borland C++
- Pelles C
- Intel C++
- VisualAge C++
- Tiny 'C' Compiler

Phases Of Software Development:

A software development process consists of various phases, each phase ends with some defined output that become the input of next phase. The phases are performed in a specific order. The main reason to develop software in phases is to ensure that software should be developed in given time and cost. Various phases to develop a program are:-

1. Requirement analysis phase: The Requirement analysis identifies what is needed from the software. Requirements analysis is done in order to understand the problem that will be solved by the software. This phase bridges the gap between client and the programmer. The outcome of this phase is the software requirement specification document (**SRS**). The person responsible for the requirements analysis is often called the **analyst**.

2. Software Design: The purpose of this phase is to plan a solution of the problem specified by the **SRS**.

It is of two types:

- **System Design or Top level Design:** It identifies the various modules that should be in the system, the specifications of the modules and interconnections between the various modules. DFDs[Data Flow Diagram] are the part of system design.
- **Detailed Design:** It identifies the internal logic of the various modules. Data structures, algorithms, flow charts are developed in this part of the designing phase.

3. Coding: This phase convert the design of the software into code using some programming language. Well written code can reduce the testing and maintenance cost and efforts.

4. Testing:- Its basic function is to detect errors in the software. After the coding phase, programs are executed for testing purpose.

The different types of testing are:

- a. Unit testing:-It tests each module separately and detects coding errors in the module.
- b. Integration testing:- When we test all modules individually, now it is time to combine all modules with each other. Integration testing is performed when we combine the modules with each other.
- c. System testing:- In integration testing we combine all the modules now we will test whole of the project it is known as system testing.
- d. User Acceptance Testing: - This testing is performed by the user to check whether system is created according to the needs of the user or not.

5. Implementation: After testing, software is implemented on the client computer to perform its actual work.

Client –server approach

Frontend:- Frontend is used by the user. The user doesn't know anything about the back process e.g. ATM is a machine where user is working as frontend user. Softwares used for frontend are VB,ASP.

Backend:- Softwares used in backend are foxpro,oracle,sql server etc. Backend is handled & maintained by the database administrator(DBA).e.g. ATM's back process is controlled by the programmer.

Errors

Errors are the bugs that can produce unexpected results, mainly errors are divided into two types.

1. Syntax errors: These errors occur due the incorrect method to write the code. It may be due the missing comma, semi colon etc. Like in following line, error will be displayed because string is not enclosed in " ".

```
printf(hello);
```

2. Logical errors: These errors are not due to any syntax mistake, but due to incorrect logic. The compiler does not show the logical errors. Like to find average of 4 numbers if we forget to write brackets then the result will not be according to our expectation $avg = 10 + 20 + 30 + 40 / 4$; in this the value of $avg = 70$ instead of 25.

Programming techniques:-Various Programming techniques are used to make the program simple, understandable and easy to modify. Some of the commonly used programming techniques are:-

1. Top down design: - Using this design, program is divided into the smaller blocks, which can be linked and called whenever needed. When it is called it begins from the bigger block to the smaller till the program ends.

2. Bottom up design: - It is also designed in the same manner and divides the whole program into the smaller blocks, but when it is called it begins from the smaller block to the bigger till the program end.

3. Modular design: - A level consists of one or more modules. Each module performs its own task and each module has minimal contact with other modules, so that each module can be implemented separately. Execution is controlled by the main program

Generations Of Computer Languages

First-generation Language

First-generation language is the lowest level computer language. Information is given to the computer by the programmer in binary form. Binary instructions are the same as on/off signals. The language consists of zeros and ones.

Second-generation Language

Assembly language was the second generation of computer language. Assembly language consists of letters of the alphabet. This makes programming much easier as compared to binary form.

Third-generation Language

These languages are similar to spoken English. Third-generation computer languages are much easier than first two generations.

Fourth-generation Language

Fourth-generation languages typically consist of English-like words and phrases. These languages can include graphic such as icons and buttons etc

Important Questions [Short Questions]

1. What do you mean by Programming language?
2. Write History of 'C' language.
3. What are ASCII Codes?
4. Write difference between high level language and low level language.
5. Define compiler.
6. Define interpreter.
7. Define assembler.
8. Name various high level language.

Important Questions [Long Questions]

1. Explain phases of software development.
2. Write Applications of 'C' language .
3. Write limitations of 'C' language .
4. Advantages and limitations of High level languages.
5. Advantages and limitations of Low level language.
6. Explain errors and various types of errors.
7. Write Note on:
 - a. Top-Down Approach
 - b. Bottom-up Approach
8. Explain Various Generations of Computer language .

Chapter 2

BASIC OF 'C' LANGUAGE

Before Writing 'C' Program

Before starting programming in 'C' Language, it is better to learn some basic concepts like:-

1. Important logics that will be used in common 'C' programs.
2. Character set of 'C' language.
3. Constants
4. Variables
5. Keywords
6. Data types
7. Operands
8. Operators
9. How to open 'C' Compiler.
10. Compile and run 'C' program
11. Open and save 'C' program.
12. Algorithms and Flow Charts.

1. Basic Logics [Formulas]: Some of the basic formulas that we will use in 'C' are:-

Area of Rectangle	$A = L * B$
Area Of Circle	$A = 3.14 * R * R$
Volume Of Box	$V = L * B * H$
Average Of 3 Numbers	$AVG = (A + B + C) / 3$
Convert Fahrenheit Temperature To Centigrade	$^{\circ}C = 0.56(^{\circ}F - 32)$ or $5/9(^{\circ}F - 32)$
Convert Centigrade Temperature To Fahrenheit	$^{\circ}F = 1.8(^{\circ}C) + 32$ or $(9/5 \times ^{\circ}C) + 32$
Area Of Square	$A = L * L$
Area Of Triangle	$A = (L * B) / 2$
Swap A and B	$T = A$ $A = B$ $B = T$
Discriminant of Quadratic equation	$D = B * B - 4 * A * C$

2. Character set: A character represent alphabet, digit or special symbol that we can use in 'C' language. In 'C' we can type mainly three types of characters.

- 1) Alphabets :A to Z and a to z
- 2) Numbers: 0,1,2,3,4,5,6,7,8,9
- 3) Special Symbols : { } " : ; [] () + - * & ^ % < . ? , = !
etc

3. Constant: The value of a constant is fixed and we cannot change the value of the constant. We use const keyword to declare constant in 'C' language.

Types of c constants:-

'C' constants can be divided into two major categories.

a) *Primary constants*

Integer constants, real constants, character constants.

b) *Secondary constants*

Array, pointers, structures, union, enum.

4. Variables: The value of variable can be changed during the program. Variables can be local or global.

Local variables: Local variables are defined with in the pair of braces or between the body of the function. The scope of the variable is with in the function.

Global variables: These variables are defined out side of all the functions. And these variables can be used in any function with in the file because the scope of these variables are global to the file.

Difference between constant & variable:-

Constant	Variable
1) Constant value doesn't change during execution of the program.	1) Variable vary during execution of the program.
2) It is fixed quantity.	2) It's value can be changed.
3)e.g. $z=2x+5y$ here 2,5,are constants.	3)e.g. $z=2x+5y$ here x,y are variables.

5. Keywords: Keywords are the reserved words with some specified function. The meanings of keywords are already defined in 'C' compiler. We cannot use keywords as variable names. In C there are 32 keywords like **if, for, while, void, int, float, do, break, continue, goto, struct, enum, double etc.**

6. Data types: The value stored in variable or constant depends upon data type, or we can say Data types represent what type of data will be stored in the variable . In 'C' language we have three basic datatypes **int**(used to store numbers without decimal), **float**(used to store numbers with or without decimal), **char**(used to store alphanumeric characters). Data types available in C are :-

TYPE	SIZE (Bits)	Format Specifier	Range
char	8	%c	-128 to 127
int or signed int	16	%d	-32768 to 32767
unsigned int	16	%u	0 to 65535
long int or signed long int	32	%ld	-2147483648 to 2147483647
unsigned long int	32	%lu	0 to 4294967295
float	32	%f	3.4 e-38 to 3.4 e+38
double	64	%lf	1.7e-308 to 1.7e+308
long double	80	%Lf	3.4 e-4932 to 3.4 e+4932

7. Operands: $c=a+b$, In this expression a,b,c are operands. These are variables that store any value.

8. Operator: operators refer to symbols that represent a particular operation to be performed on data. Types of operators are:-

Assignment operator: Assignment operators are used to assign the values to a variable. Assignment operators are ($=$, $-=$, $/=$, $*=$ etc).

Arithmetic operator: These are used to perform arithmetic calculation on the operands. Arithmetic operators are $+$, $-$, $*$, $/$, $\%$.

Relational operator: Use to check relation between operands. E g. $>$, $<$, $<=$, $>=$ etc.

Logical operator: These operators are used to attach more than one relational operator in one statement. These are $\&\&$ (AND), $\|\|$ (OR), $!$ (NOT).

Conditional operators: These are used when we want to execute the statements according to the condition.

Bitwise operators: Bitwise operators are special operators designed to work on the bit level.

Bitwise Operator	Meaning
$\&$	Bitwise AND
$\ $	Bitwise OR
\wedge	Bitwise exclusive OR
$<<$	Shift left
$>>$	Shift right
\sim	One's complement

9. How to install c into your computer:-

- 1) First of all , find turboc .exe file.
- 2) Copy this file and paste it on c:drive.
- 3) Now, open DOS prompt.
- 4) Type cd\ and press enter.
- 5) C:\turboc.exe -d. It will extract all directories from turboc c to your c drive.

10. How to open 'C' Compiler.

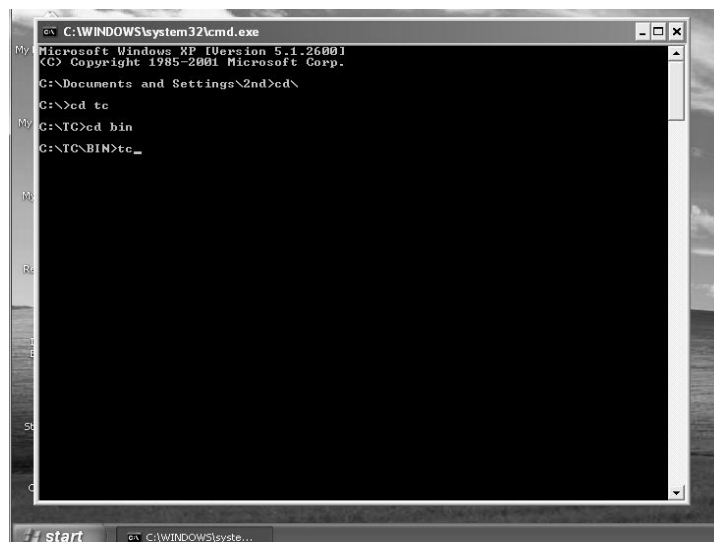
If you are writing program in windows98, 2000,2003,XP then you can use any compiler of 'C'. But if you use windows vista or window 7 then it is better that you use 32-bit or 64-bit Compiler according to the operating system.

Steps to open 'C' compiler

1. Select Run from start and type command or cmd in run box.



2. At DOS Prompt type [if 'C' is installed in c drive]
cd\
cd tc
cd bin
tc



3. With these commands 'C' editor will open that look like

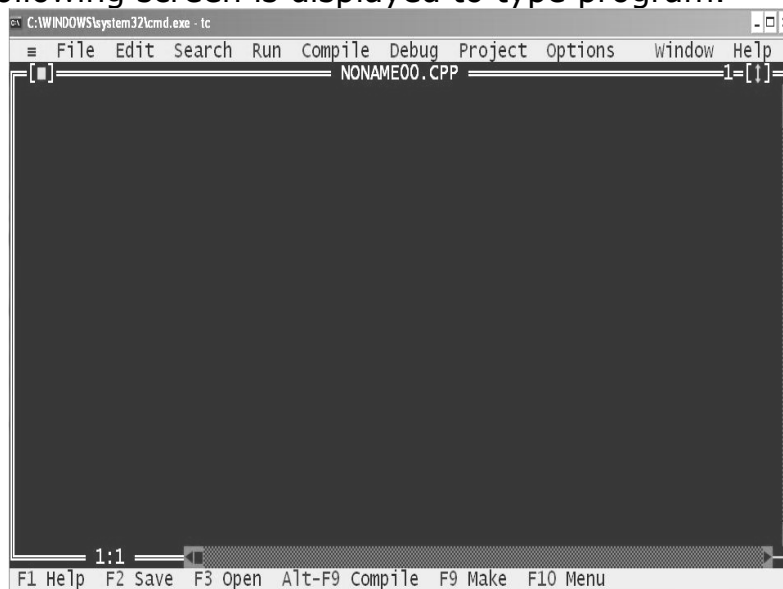


To maximize use alt+enter

4. To start new program select new from file menu

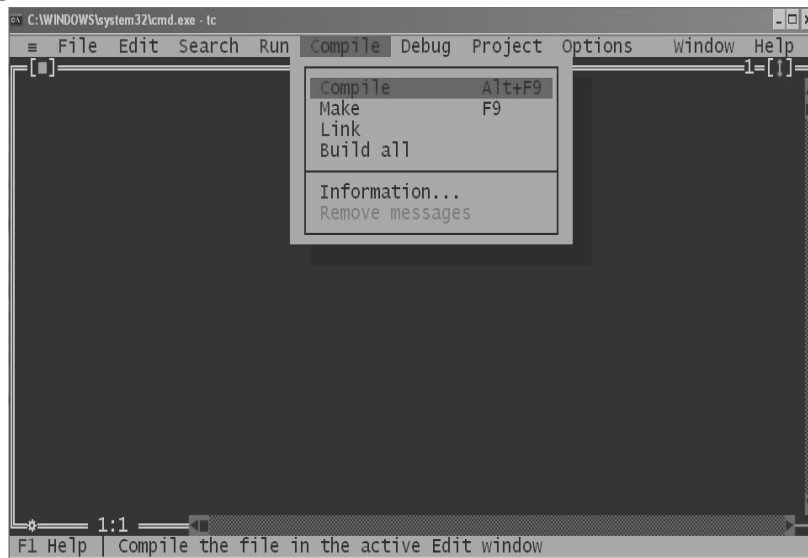


5. Following screen is displayed to type program.



11- Compile and run 'C' program

To compile the program select compile option from the Compile menu or Press Alt + F9



To run the program select run option from the run menu or Press Ctrl + F9

12. Open and save 'C' program.

To open a file select open from file menu or press F3, and to save select save from file menu or press F2, the extension of 'C' program is .c

Steps to take C programs in printed form:

- 1) Go to c:\tc\bin and press enter. Right click on the file having extension .c or .cpp and open it with notepad.
- 2) Now go to file menu of notepad & click on print option.
- 3) The printed form of required C program will be in your hands.

Algorithms

Algorithms are the English like steps to solve a problem. It is the part of software designing. Before creating a program or software we develop its design using some tools like algorithms, DFDs, Flowcharts etc. Mainly we follow following steps to design an algorithm.

Step 1. START //it represents beginning of the algorithm

Step 2. DECLARE //the variables used in algorithm are declared in this step.

Step 3. INPUT // here we input the values

Step 4. FORMULA// the required result is generated in this step.

Step 5. OUTPUT // display the result.


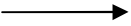




Step 6. STOP // end of algorithm

***We can use if else, goto ,loops etc according to requirement*

Flowchart:-

A flowchart is pictorial representation to solve a problem. It is the graphical representation of an algorithm. Also we can say that these are the symbolic diagrams which tells the user about the data flow, sequence and processing logic in information processing.

We can use following symbols to create flowchart

Symbol	Symbol Name	Symbol Description
	Process	Show a Process or action step. Any kind of processing like mathematical calculations are depicted using processing box. It is nothing but a rectangle which indicates processing.
	Flowline (Arrow)	Flow line shows the direction of flow of control.
	Decision	Decision box is used when there are 2 options (Yes/No). It is used as if-else. It is used to take any decision in the program. It returns the value in true or false.
	Connector	Connector connects the various portion of the flowchart. A connector is used when the flowchart is split between two pages
	Input/output	This is used to input and print data .
	Terminator	It represents starting and ending point of the flowchart. We use start & stop option in it. The shape of terminal symbol is in oval shape.

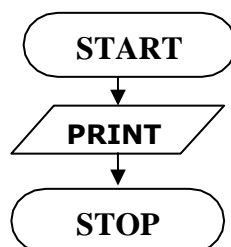
Q1. Write an algorithm and draw flowchart

Algorithm

Step 1: START
Step 2: PRINT "hello"

Step 3: STOP

Flowchart

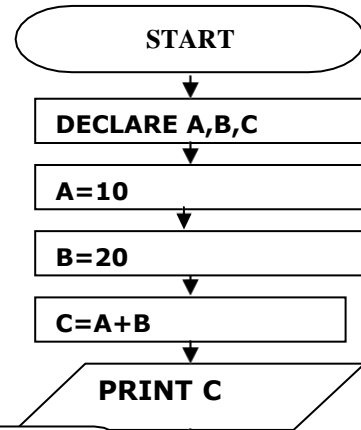


Q2. Write an algorithm and draw flowchart to add 2 numbers.

Algorithm :

- Step 1: START
- Step 2: DECLARE A,B,C
- Step 3: A=10
- Step 4: B=20
- Step 5: C=A+B
- Step 6: PRINT C
- Step 7: STOP

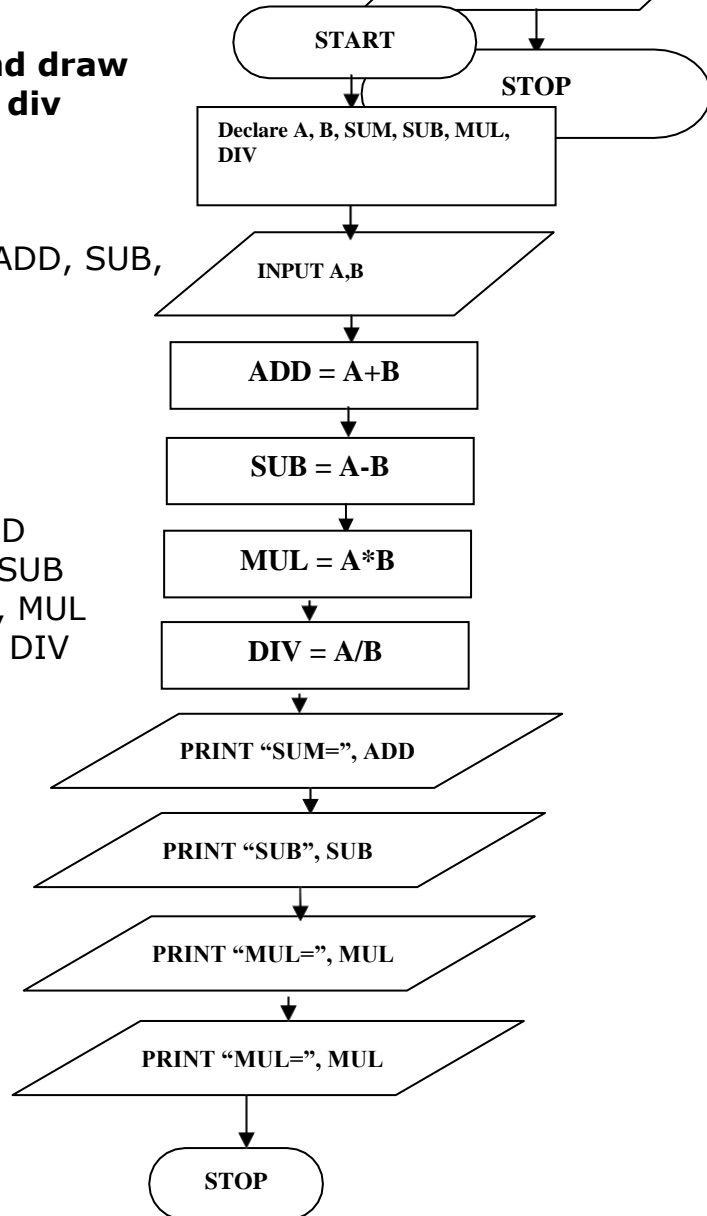
Flowchart:->



Q3. Write an algorithm and draw flowchart to add, sub, mul, div 2 numbers.

Algorithm

- Step 1: START
- Step 2: DECLARE A, B, ADD, SUB, MUL, DIV
- Step 3: INPUT A, B
- Step 4: ADD A+B
- Step 5: SUB A-B
- Step 6: MUL A*B
- Step 7: DIV A/B
- Step 8: PRINT "SUM=", ADD
- Step 9: PRINT "SUB=", SUB
- Step 10: PRINT "MUL=", MUL
- Step 11: PRINT "DIV=", DIV
- Step 12: STOP



Q4. Write an algorithm and draw flowchart to find greatest among two numbers.

Algorithm

Step 1: START

Step 2: DECLARE A, B

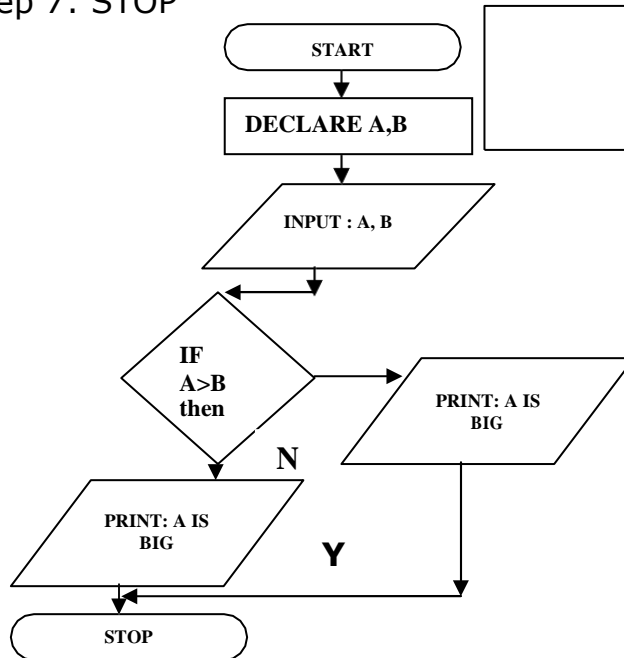
Step 3: INPUT A, B

Step 4: IF A>B THEN GOTO STEP 5 ELSE GOTO STEP 6

Step 5: PRINT "A IS GREATEST" GOTO 7

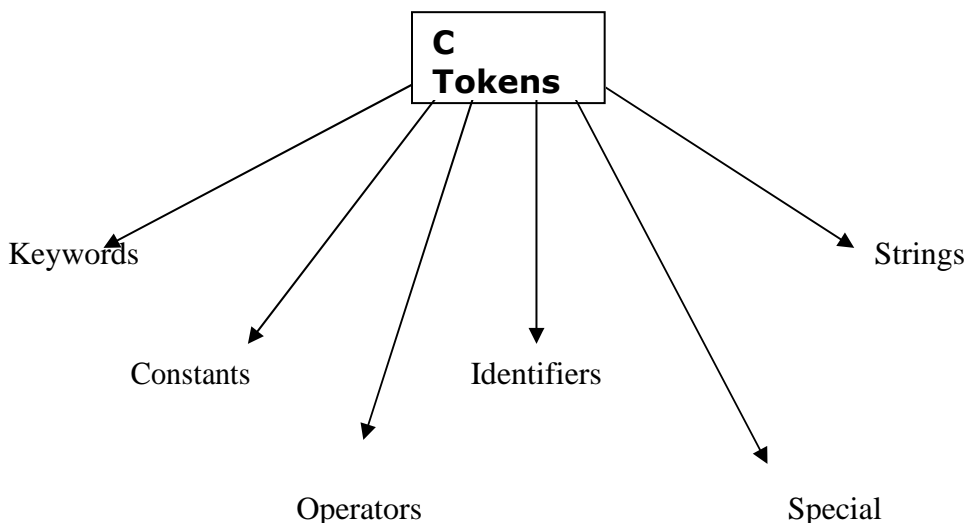
Step 6: PRINT " B IS GREATEST"

Step 7: STOP



Pseudo code:- Pseudo code is like an algorithm but pseudo codes are nearer to the program. It is the outline of the program and it cannot be compiled nor executed. It is easy to convert Pseudo code into program as compare to algorithm.

TOKENS:- The individual words ,special symbols , operators, punctuation marks are called tokens. These are given below:



s
y
m
b
o
l
s

COMMENTS: These are the statements which are not executed by the compiler. It means that these statements are ignored by the compiler.

Comments can be of two types:-

- 1) Single line comments
- 2) Multi line comments

Single line comments:- Single line comments are used to display any statements for understanding of the user. It can be used by using double slash (//) symbols.

e.g. //Single line comments

Multi line comments:- When user wants to write number of statements for knowledge of the user, but ignored by the compilers then we use multi line comments.

e.g. /*-----
-----*/

Important Questions [Short Questions]

1. Define Character set Of 'C' language.
2. Define Constant and Variables.
3. Define Keywords.
4. Define Data types.
5. Define Operand and Operators.
6. How we can open 'C' compiler and run our program?
7. Write Difference between 'a' and "a".
8. What are shorthand or shortcut or short circuit assignment operators?

Important Questions [Long Questions]

1. Define algorithms and basic structure of an algorithm.
2. Define Flowchart and symbols used in it.
3. Explain basic Data types of 'C' language.
4. Write algorithm and draw flowchart to find simple interest.
5. Write an algorithm to find roots of the quadratic equation.
6. Explain operators present in 'C' language.