C LANGUAGE

Kalpesh Chauhan

Introduction

- The C Language is developed for creating system applications that direct interacts to 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.

- It can be defined by following ways:
  - Mother language
  - System programming language
  - Procedure-oriented programming language
  - Structured programming language
  - Mid level programming language

C as a mother language

- C language is considered as the mother language of all the modern languages because **most of the compilers, JVMs, Kernels etc. are written in C language** and most of languages follows c syntax e.g. C++, Java etc.
- It provides the core concepts like array, functions, file handling etc. that is being used in many languages like C++, java, C# etc.

C as a system programming language

- A system programming language is used to create system software. C language is a system programming language because it **can be used to do low level programming (e.g. driver and kernel)**. It is generally used to create hardware devices, OS, drivers, kernels etc. For example, linux kernel is written in C.
- It cant be used in internet programming like java, .net, php etc.

C as a procedural language

- A procedure is known as function, method, routine, subroutine etc. A procedural language **specifies a series of steps or procedures for the program to solve the problem.**
- A procedural language breaks the program into functions, data structures etc.
- C is a procedural language. In C, variables and function prototypes must be declared before being used.

C as a structured programming language

- A structured programming language is a subset of procedural language. **Structure means to break a program into parts or blocks** so that it may be easy to understand.
- In C language, we break the program into parts using functions. It makes the program easier to understand and modify.

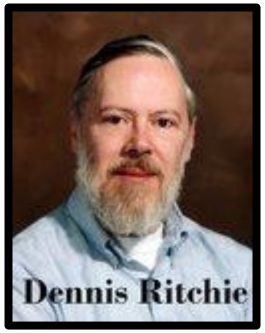
C as a mid-level programming language

- C is considered as a middle level language because it **supports the feature of both low-level and high level language.** C language program is converted into assembly code, supports pointer arithmetic (low level), but it is machine independent (feature of high level).

- **Low level language** is specific to one machine i.e. machine dependent. It is machine dependent, fast to run. But it is not easy to understand.
- **High Level language** is not specific to one machine i.e. machine independent. It is easy to understand.

History of C Language

- **History of C language** is interesting to know. Here we are going to discuss brief history of c language.



- **C programming language** was developed in 1972 by Dennis Ritchie at bell laboratories of AT&T (American Telephone & Telegraph), located in U.S.A.
- **Dennis Ritchie** is known as the **founder of c language.**

- It was developed to overcome the problems of previous languages such as B, BCPL etc.
- Initially, C language was developed to be used in **UNIX operating system.** It inherits many features of previous languages such as B and BCPL.

- Let's see the programming languages that were developed before C language.

Sr.	Language	Year	Developed By
1	Algol	1960	International Group
2	BCPL	1967	Martin Richard
3	B	1970	Ken Thompson
4	Traditional C	1972	Dennis Ritchie
5	K & R C	1978	Kernighan & Dennis Ritchie
6	ANSI C	1989	ANSI Committee

Features of C Language

- C is the widely used language. It provides a lot of **features** that are given below.
  - Simple
  - Machine Independent or Portable
  - Mid-level programming language
  - structured programming language
  - Rich Library
  - Memory Management
  - Fast Speed
  - Pointers
  - Recursion
  - Extensible

Simple

- C is a simple language in the sense that it provides **structured approach** (to break the problem into parts), **rich set of library functions, data types** etc.
-

Machine Independent or Portable

- Unlike assembly language, c programs **can be executed in many machines** with little bit or no change. But it is not platform-independent.

Mid-level programming language

- C is **also used to do low level programming.** It is used to develop system applications such as kernel, driver etc. It **also supports the feature of high level language.** That is why it is known as mid-level language.

Structured programming language

- C is a structured programming language in the sense that **we can break the program into parts using functions.** So, it is easy to understand and modify.

Rich Library

- C **provides a lot of inbuilt functions** that makes the development fast.

Memory Management

- It supports the feature of **dynamic memory allocation.** In C language, we can free the allocated memory at any time by calling the **free**() function.

Speed

- The compilation and execution time of C language is fast.

Pointer

- C provides the feature of pointers. We can directly interact with the memory by using the pointers. We **can use pointers for memory, structures, functions, array** etc.

Recursion

- In c, we **can call the function within the function.** It provides code reusability for every function.

Extensible

- C language is extensible because it **can easily adopt new features.**

How to Install

- There are many compilers available for c and c++. You need to download any one. Here, we are going to use **Turbo C++**. It will work for both C and C++. To install the Turbo C software, you need to follow following steps.

- Download Turbo C++
- Create turboc directory inside c drive and extract the tc3.zip inside c:\turboc
- Double click on install.exe file
- Click on the tc application file located inside c:\TC\BIN to write the c program

C Programs

- A C program can vary from 3 lines to millions of lines and it should be written into one or more text files with extension **".c"**.

Text Editor

- This will be used to type your program. Examples of few a editors include Windows Notepad, OS Edit command, Brief, Epsilon, EMACS, and vim or vi.
- The name and version of text editors can vary on different operating systems. For example, Notepad will be used on Windows, and vim or vi can be used on windows as well as on Linux or UNIX.

The C Compiler

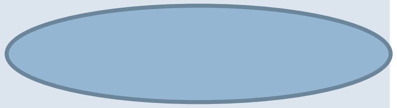
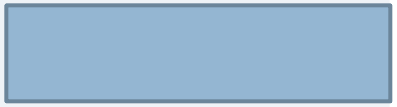
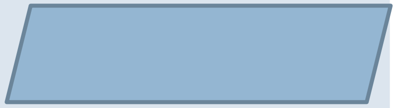
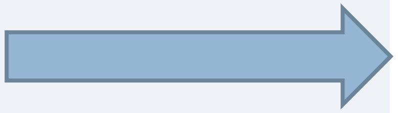
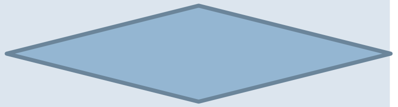
- The source code written in source file is the human readable source for your program. It needs to be "compiled", into machine language so that your CPU can actually execute the program as per the instructions given.
- The compiler compiles the source codes into final executable programs. The most frequently used and free available compiler is the GNU C/C++ compiler.

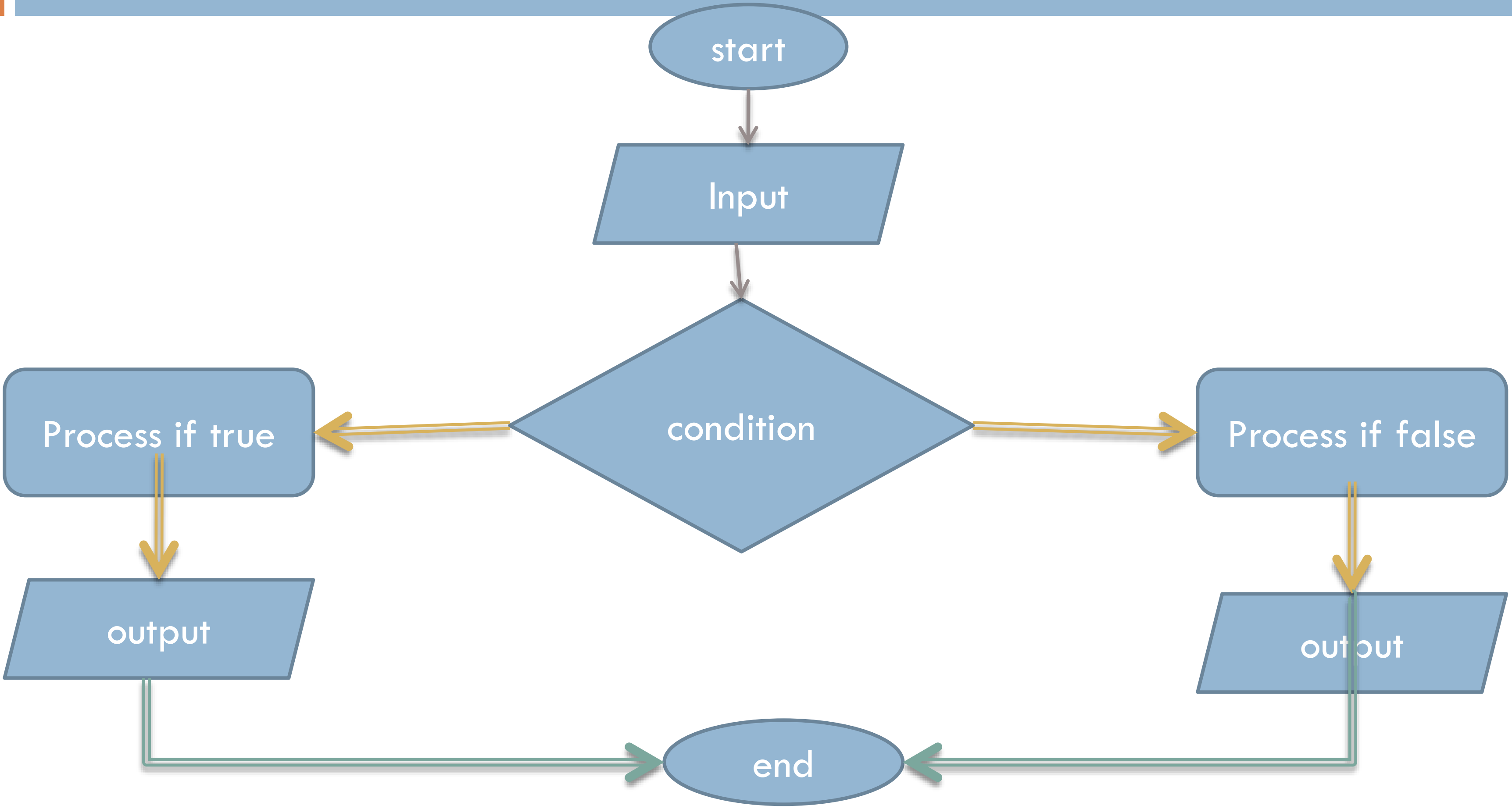
Basic programming techniques

- A **flowchart** is a type of diagram that represents an algorithm, workflow or process, showing the steps as boxes of various kinds, and their order by connecting them with arrows. This diagrammatic representation illustrates a solution model to a given problem.

- Flow chart use various symbols to define state of program.

Symbols

Sr.	Name	Use	Symbol
1	Oval	Start / Stop	
2	Rectangle	Process	
3	Parallelogram	Input / Output	
4	Lines	Flow of Program	
5	Diamond	Condition	


start
Input
condition
Process if true
Process if false
output
output
end

Structure of C Programs

Documentation section

Link Section

Definition Section

Global Declaration section

Main function()

{

    Declaration / executable part

}

User defined functions

First Program in C

```
#include <stdio.h>
#include <conio.h>
void  main()
{
   clrscr();
   printf("Hello C Language");
   getch();
}
```

Explanation

- The #include is pre process directives to define or add files before going to actual compilation.
- The clrscr() use to clean the previous output form screen
- The void main is main function of every program in c. execution always start by the main function.

- The printf() is use to print output on screen / terminal.

- The getch() use to get single character from keyboard (to view constant output)

How to run

- Press **altr + f9** to Compile the program.
- Press **crtl + f9** to run the program.

Must Remember

- C language is totally case sensitive language. Must type the all the code in proper case only.

Program Flow

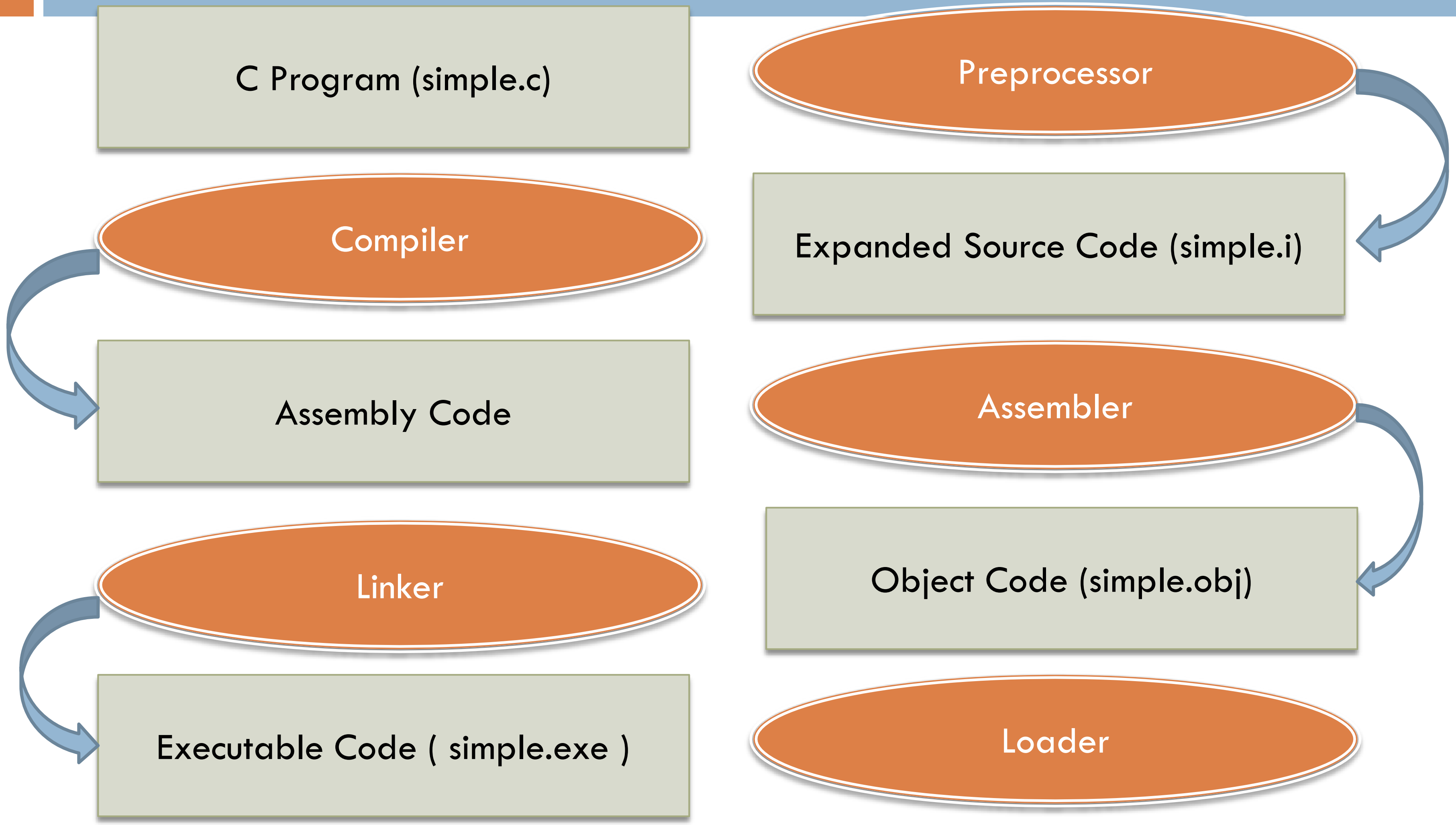
- The C program follows many steps in execution.

- C program (source code) is sent to preprocessor first. The preprocessor is responsible to convert preprocessor directives into their respective values. The preprocessor generates an expanded source code.

- Expanded source code is sent to compiler which compiles the code and converts it into assembly code.
- The assembly code is sent to assembler which assembles the code and converts it into object code. Now a simple.obj file is generated.
- The object code is sent to linker which links it to the library such as header files. Then it is converted into executable code. A simple.exe file is generated.

- The executable code is sent to loader which loads it into memory and then it is executed. After execution, output is sent to console.

Diagram of Program Flow



printf() and scanf() functions in C

- The printf() and scanf() functions are used for input and output in C language. Both functions are inbuilt library functions, defined in stdio.h (header file).

printf()

- The **printf() function** is used for output. It prints the given statement to the console.
- The syntax of printf() function is given below:
- printf("format string",argument_list);

Format String

%d	integer
%c	character
%s	String
%f	Float
%u	unsigned

scanf()

- The **scanf() function** is used for input. It reads the input data from the console.

- scanf("format string",argument_list);

Data Types in C

- A data type specifies the type of data that a variable can store such as integer, floating, character etc.

Data Types in C

- Basic
- Derived
- Enumeration
- Void

Basic

- int, char, float, double

Derived

- Array, pointer, structure, union

Enumerated

- enum

Void

- Void

Basic Data Types

- The basic data types are integer-based and floating-point based. C language supports both signed and unsigned literals.

- The memory size of basic data types may change according to 32 or 64 bit operating system.

Integer Data Type

- Storage Size : 2 Bytes
- Bits : 16
- Usable Bits : 15
- First Bit used as sign Bit (+ / -)
- Range is : −32,768 to 32,767
- Default integer type.

Unsigned integer Type

- Storage Size : 2 Bytes
- Bits : 16
- Usable Bits : 16
- Store only positive values.
- Range is : 0 to 65,535
- Must use unsigned keyword before integer.

Long Integer Type

- Storage Size : 4 Bytes
- Bits : 32
- Usable Bits : 31
- First Bit used as sign Bit (+ / -)
- Range is : -2,147,483,648 to 2,147,483,647
- Use long before integer keyword.

Unsigned Long Integer Type

- Storage Size : 4 Bytes
- Bits : 32
- Usable Bits : 32
- Store only positive values.
- Range is : 0 to 4,29,49,67,295
- Must use unsigned long keyword before integer.

Floating point Values

Type	**Size**	**Value Range**	**Precision**
Float	4 byte	1.2E-38 to 3.4E+38	6
double	8 byte	2.3E-308 to 1.7E+308	15
Long double	10 byte	3.4E-4932 to 1.1E+4932	19

Character Data Type

- Use to store any single character.
- Storage size : 1 Byte
- Total Bits : 8 Bits
- Range : 0 to 255 (ASCII)

ASCII

- American Standard Code for Information Interchange, is a character encoding standard (the Internet Assigned Numbers Authority (IANA) prefers the name US-**ASCII**). **ASCII** codes represent text in computers.

Identifier

- A C identifier is a name used to identify a variable, function, or any other user-defined item. An identifier starts with a letter A to Z, a to z, or an underscore '_' followed by zero or more letters, underscores, and digits (0 to 9).

Variables in C

- A **variable** is a name of memory location. It is used to store data. Its value can be changed and it can be reused many times.
- It is a way to represent memory location through symbol so that it can be easily identified.

- Let's see the syntax to declare a variable:
- type variable_list;

Rules for defining variables

- A variable can have alphabets, digits and underscore.
- A variable name can start with alphabet and underscore only. It can't start with digit.
- No white space is allowed within variable name.
- A variable name must not be any reserved word or keyword e.g. int, float etc.

Valid Variable Names

- int a;
- int abc;
- float percentage;
- char gender;
- int total_marks;
- float _result;

Invalid Variable Names

- int 1a;
- float char;
- int 123;
- int x*y;

White Space in C

- A line containing only whitespace, possibly with a comment, is known as a blank line, and a C compiler totally ignores it.
- Whitespace is the term used in C to describe blanks, tabs, newline characters and comments. Whitespace separates one part of a statement from another and enables the compiler to identify where one element in a statement, such as int, ends and the next element begins.

Comment Lines in C

- Comments are like helping text in your C program and they are ignored by the compiler. They start with /* and terminate with the characters */ as shown below.
- // for single line comment
- /* for multiline comment */

Keywords in C

- The following list shows the reserved words in C. These reserved words may not be used as constants or variables or any other identifier names. The reserved keyword have some specific meaning in C compiler.

List of Keyword

auto	else	long	switch
break	enum	register	typedef
case	extern	return	union
char	float	short	unsigned
const	for	signed	void
continue	goto	sizeof	volatile
default	if	static	while
do	int	struct	double

Back ground files

- When you save and compile you can find following files in your bin folder of TC.

file type	extension
C- Source file	.c
application file	.exe
object file	.obj
backup file	.bak

Sizeof Operator

- To get the exact size of a type or a variable on a particular platform, you can use the **sizeof** operator. The expressions *sizeof(type)* yields the storage size of the object or type in bytes.

Variable Definition in C

- A variable definition tells the compiler where and how much storage to create for the variable. A variable definition specifies a data type and contains a list of one or more variables of that type as follows.
- type variable_list;
- int x,y,z;

Variable initialized

- Variables can be initialized (assigned an initial value) in their declaration. The initializer consists of an equal sign followed by a constant expression as follows.

- type variable_name = value;

- int d = 3, f = 5; // definition and initializing d and f.

- Note : newly created variable have garbage values.

Lvalues in C

- Expressions that refer to a memory location are called "lvalue" expressions. An lvalue may appear as right-hand side of an assignment.

The #define Preprocessor

- Define keyword use to define any name or value at the start of the program.

- #define LENGTH 10

The const Keyword

- The const keyword use to define constant value of any variable this means no one made change value of this variable at run time.

- const int I = 10;

- Note that it is a good programming practice to define constants in CAPITALS

Operators

- An operator is simply a symbol that is used to perform operations. There can be many types of operations like arithmetic, logical, bitwise etc.

- Arithmetic Operators
- Relational Operators
- Shift (Bitwise) Operators
- Logical Operators
- Ternary or Conditional Operators
- Assignment Operator
- Misc Operator

Arithmetic Operators

+	Adds two operands.
−	Subtracts second operand from the first.
*	Multiplies both operands.
/	Divides numerator by de-numerator.
%	Modulus Operator and remainder of after an integer division.
++	Increment operator increases the integer value by one.
--	Decrement operator decreases the integer value by one.

Relational Operators

- ‘==’ operator checks whether the two given operands are equal or not. If so, it returns true. Otherwise it returns false. For example, **5==5** will return true.
- ‘!=’ operator checks whether the two given operands are equal or not. If not, it returns true. Otherwise it returns false. It is the exact boolean complement of the ‘==’ operator. For example, **5!=5** will return false.
- ‘>’ operator checks whether the first operand is greater than the second operand. If so, it returns true. Otherwise it returns false. For example, **6>5** will return true.

- ‘<‘ operator checks whether the first operand is lesser than the second operand. If so, it returns true. Otherwise it returns false. For example, **6<5** will return false.
- ‘>=’ operator checks whether the first operand is greater than or equal to the second operand. If so, it returns true. Otherwise it returns false. For example, **5>=5** will return true.
- ‘<=’ operator checks whether the first operand is lesser than or equal to the second operand. If so, it returns true. Otherwise it returns false. For example, **5<=5** will also return true.

Shift (bitwise) operators

- **& (bitwise AND)** Takes two numbers as operands and does AND on every bit of two numbers. The result of AND is 1 only if both bits are 1.
- **| (bitwise OR)** Takes two numbers as operands and does OR on every bit of two numbers. The result of OR is 1 any of the two bits is 1.
- **^ (bitwise XOR)** Takes two numbers as operands and does XOR on every bit of two numbers. The result of XOR is 1 if the two bits are different.
- **<< (left shift)** Takes two numbers, left shifts the bits of the first operand, the second operand decides the number of places to shift.

- **>> (right shift)** Takes two numbers, right shifts the bits of the first operand, the second operand decides the number of places to shift.
- **~ (bitwise NOT)** Takes one number and inverts all bits of it

```
void main()
{
    unsigned char a = 5, b = 9; // a = 5(00000101), b =
    9(00001001)
    printf("a = %d, b = %d\n", a, b);
    printf("a&b = %d\n", a&b); // The result is 00000001
    printf("a|b = %d\n", a|b);  // The result is 00001101
    printf("a^b = %d\n", a^b); // The result is 00001100
    printf("~a = %d\n", a = ~a);   // The result is 11111010
    printf("b<<1 = %d\n", b<<1);  // The result is 00010010
    printf("b>>1 = %d\n", b>>1);  // The result is 00000100
}
```

Logical Operators in C

&&	Called Logical AND operator. If both the operands are non-zero, then the condition becomes true.
||	Called Logical OR Operator. If any of the two operands is non-zero, then the condition becomes true.
!	Called Logical NOT Operator. It is used to reverse the logical state of its operand. If a condition is true, then Logical NOT operator will make it false.

Assignment Operators

=	Simple assignment operator. Assigns values from right side operands to left side operand.
+=	Add AND assignment operator. It adds the right operand to the left operand and assign the result to the left operand.
-=	Subtract AND assignment operator. It subtracts the right operand from the left operand and assigns the result to the left operand.

*=	Multiply AND assignment operator. It multiplies the right operand with the left operand and assigns the result to the left operand.
/=	Divide AND assignment operator. It divides the left operand with the right operand and assigns the result to the left Operand.
%=	Modulus AND assignment operator. It takes modulus using two operands and assigns the result to the left operand.

Ternary operator

- If Condition is true ? then value X : otherwise value Y

- (condition ? True : false)

Comments in C

- Comments in C language are used to provide information about lines of code. It is widely used for documenting code. There are 2 types of comments in C language.

- Single Line Comments
- Multi Line Comments

- Must Provide comments for logic of programs.

Escape Sequence in C

- An escape sequence in C language is a sequence of characters that doesn't represent itself when used inside string literal or character.
- It is composed of two or more characters starting with backslash \. For example: \n represents new line.

List of Character

Sr.	Character	Use
1	\a	Alarm or Beep
2	\b	Backspace
3	\f	Form Feed
4	\n	New Line
5	\r	Carriage Return
6	\t	Tab (Horizontal)
7	\v	Vertical Tab
8	\\\\	Backslash
9	\\'	Single Quote
10	\\"	Double Quote
11	\?	Question Mark
12	\0	Null

Escape Sequence Example

```
void main()
{
     int number=50;
    printf("You\nare\nlearning\n\'c\' language\n\"Do
    you know C language\"");


}
```

Output

```
You
are
learning
'c' language
"Do you know C language"
```

Constants in C

- A constant is a value or variable that can't be changed in the program, for example: 10, 20, 'a', 3.4, "c programming" etc.
- There are different types of constants in C programming.

Sr.	Constant	Example
1	Decimal Constant	10, 20, 450 etc.
2	Real or Floating-point Constant	10.3, 20.2, 450.6 etc.
3	Octal Constant	021, 033, 046 etc.
4	Hexadecimal Constant	0x2a, 0x7b, 0xaa etc.
5	Character Constant	'a', 'b', 'x' etc.
6	String Constant	"c", "c program"

The const Keyword

- The const keyword use to define constant value of any variable this means no one made change value of this variable at run time.
- const int I = 10;
- Note that it is a good programming practice to define constants in CAPITALS

The #define Preprocessor

- Define keyword use to define any name or value at the start of the program.

- #define LENGTH 10

C Token

- In C programs, each individual word and punctuation is referred to as a token. C Tokens are the smallest building block or smallest unit of a C program.

- C Supports Six Types of Tokens:
  - Identifiers
  - Keywords
  - Constants
  - Strings
  - Operators
  - Special Symbols

Identifiers

- Identifiers are names given to different names given to entities such as constants, variables, structures, functions etc.

- int sub1,sub2,total;

- Must use meaning full name for identifiers.

Keywords

- You can't use a keyword as an identifier in your C programs, its reserved words in C library and used to perform an internal operation. The meaning and working of these keywords are already known to the compiler.

Constants

- C Constants is a most fundamental and essential part of C programming language. Constants in C are the fixed values that are used in a program, and its value remains the same during the entire execution of the program.

Operators

- C operators are symbols that are used to perform mathematical or logical manipulations. C programming language is rich with built-in operators. Operators take part in a program for manipulating data and variables and form a part of the mathematical or logical expressions.

Header File

- Header files are helping file of your C program which holds the definitions of various functions and their associated variables that needs to be imported into your C program with the help of pre-processor #include statement. All the header file have a '.h' an extension that contains C function declaration and macro definitions.