# **Characteristics Of C**

* It is a Procedural Oriented Programming Language. And also a middle level language.
* C has a rich set of built-in functions and operators, these can be used to write any complex code.
* The C compiler combines the capabilities of an assembly language with the features of a high level language.
* C language is well structured programming language, this modular structure makes program debugging, testing and maintenance easier. A proper collection of these modules would make a complete program.
* A C program is basically a collection of functions that are supported by C library.
* Programs written in C are efficient and fast. This is due to its variety of data types and powerful operators.
* C is a portable language I.e., programs written in C in one machine can be easily run on some other machine without any modifications.
* Dynamic storage allocations is possible.

# **Use of Decision Box in Flow Chart**

* A Decision Box is a diamond shaped box in a flow chart.
* It is used for decision making between two or more alternatives.
* Each box has either Yes or No, it indicates the direction that the user should be follow on flow chart.
* It means it is used for testing the condition and it gives two output, if the respected condition is true then it executes the statement in the true block, otherwise statement will executes in the false block.
* Symbolic form of a decision box is:
* For Example:

Assign r=34

print “Not Eligible”

Print “Eligible”

if( r>=18)

True False

# **Purpose of \n And \t Characters**

* An escape sequence is a sequence of characters used in formatting the output and are not displayed while printing text on the screen.
* Each having their own specific function.
* All escape sequences in C are represented by two or more characters.
* First one compulsory being a **backslash,** it's denoted by **\**.
* And the other one is any character in the C character set.
* \n : It is called as **New Line**. And we use it to shift the cursor control to the new line.
* \t : It is called as **Horizontal Tap**. And we use it to shift the cursor to a couple of spaces to the right in the same line.

# C program to check Whether a number is Even or Odd

## Source Code

#include<stdio.h>

#include<conio.h>

Main()

{

int num;

printf(“Enter a number:\n”);

scanf(“%d”,&num);

if(num%2==0)

printf(“\n Number is Even”);

if(num%2!==0)

printf(“\n Number is Odd”);

}

# **Dynamic Memory Allocation**

Whenever define a variable in C whether it is a simple data type, an array or a structure, it’s effectively reserving one or more locations in the computer memory to contain the values that will be stored in that variable. The C compiler automatically allocates the correct amount of storage for you.

Allocating and freeing storage during the execution of the program is called **dynamic memory allocation**.

Suppose you have a program that is designed to read in a set of data from a file into an array in the memory. However, that you don’t know how much data is in the file until the program starts execution.

You have three choices:

1. Define the array to contain the max number of possible elements at compile time.
2. Use a variable length array to dimension the size of the array at runtime.
3. Allocate the array dynamically using one of C's memory allocation routines.

For Example,

Consider a program which tries to use an array of structure with 150 elements

struct temp

{

char name[50];

int num[300];

} big[150];

## Advantages of DMA

* The length of dynamically allocated memory either can be increased or decreased.
* In dynamic memory allocation, we use the **realloc()** function we can change the memory block size as per the user requirement.
* Dynamically created list is called as linked list.
* In case dynamically created lists insertion and deletion can be done very easily and flexible also.
* The previously used memory can also reused in dynamic memory allocation.
* The variables are get allocated until the program unit gets inactive. Memory allocation occurs during runtime.

# **Algorithm for checking the largest number from two entered numbers**

**Algorithm**

Step1: Start

Step2: Declare variables a, b.

Step3: if a>b then /\*Checking\*/

Display “a is the Largest Number”.

Otherwise

Display “b is the Largest Number”.

Step4: Stop.

# **C program to Multiply two numbers given by user at runtime**

**Source Code**

#include<stdio.h>

#include<conio.h>

#include<math.h>

int main()

{

float n1, n2, product;

clrscr();

printf(“Enter the value: \n”);

scanf(“%f”, &n1);

printf(“Enter the value: \n”);

scanf(“%f”, &n2);

product = n1 \* n2;

printf(“Product of %f and %f is %f.”, n1,n2,product);

getch();

return 0;

}

# **Difference between = and ==**

## = Operator

* It is an assignment operator.
* It is used in C, C++ and other programming languages.
* It is used for assigning the value to a variable.
* In this operator, constant term cannot be placed on left hand side.
* Example :

x = (a+b);

y = x;

Here, when first expression evaluates value of (a+b) will be assigned in to x and second expression y=x; value of variable x will be assigned in to y.

## == Operator

* It is a equal operator.
* It is used in C and C++ only.
* It is used for comparing two values. If both values are equal it returns 1 otherwise it returns 0.
* In this operator, constant term can be placed in the left hand side.
* Example :

int x,y;

x = 10;

y = 10;

if(x==y)

printf(“True”);

else

printf(“False”);

When expression x==y evaluates, it will return 1 i.e., condition is true, and True will be print. Otherwise condition is false, and False will be print.

# **Difference between strcat() and strcmp()**

## strcat()

* Which allows use to combine two strings.
* This function takes two arguments :

destination – destination string source – source string

* Syntax :

char \*strcat(char \*destination, const char \*source)

* Example :

#include<string.h>

char \*strcat (s1, s2)

Concatenates the character string s2 to the end of s1, placing a null character at the end of the final string. The function returns s1.

* When we use strcat(), the size of the destination string should be large enough to store the resultant string. If not, we will get the segmentation fault error.

## Strcmp()

* Which allows use to compare one string with another one.
* This function takes two parameters :

str1 – a string str2 – a string

* Syntax :

int strcmp(const char\* str1, const char\* str2);

* Example :

#include<string.h>

int strcmp (s1, s2)

Compare strings s1, s2 and returns a value less than zero if s1 is less than s2, equal to zero if s1 is equal to s2, and greater than zero if s1 us greater than s2.