**DAY-7**

**C Programming:**

**Structure of c:**

Documentation

Link section

Definition section

Global declaration

Main function

{

Variable declaration

Executable part

}

Sub program

**Documentation:**

/\*

Description:

Ex: modules add, sub

Int add(int val1, int val2): add is doing addition of val1 with val2 and return the result to the called(main)

Author: DP

DOC/DOM: 29-10-2024

Version: 0.1v

1.0 stable version => basing of the project

\*/

C Tokens:

1. Keywords – reserved words -32
2. Identifiers –
3. Constants – numeric constants - 10, string constants –‘a’
4. Strings
5. Special symbols
6. Operators

Strings:

Int c “2002”+2 => not possible

Int c ‘2’ +2 => possible

Int a=10;

If(!a) – 0

Data types:

1. Primary data types
2. M
3. P

Modifiers:

1. Signed
2. Unsigned
3. Long
4. Short

Terminary operator:

int a=10,b=20,c=50,res=0;

res = (a>b)?((a>c)?a:c):((b>c)?b:c);

printf("%d",res);

Size of operator:

sizeof(datatype)

sizeof() is a operator not a function because when we write a++ in function sizeof(a++) the expression didn’t execute.

Managing i/p and o/p operations:

i/p stmt – scanf,getchar()

o/p stmt – printf()

There are 2 types of I/O stmts. They are:

1. Formatted – printf(), scanf() – which specify the data type
2. Unformatted – getchar(), gets(). putchar(), puts(), etc – not specify the data type

Getc,putc --🡪 read/write single chat -> stream -file or user

Getchar, putchar 🡪 !! -🡪 std i/o

Gets, puts -🡪 !! -🡪 for string

Getch() => DOS platform => not echo read char to the screen

Scanf():

Scanf(“control string”, address list);

Ex: num=386;

Scanf(“%3d”.&num);

3 refers to width of the number.

Printf(“%05d”,678); --🡪 0 0 6 7 8

5 refres to width

0 refers fill the remaing left space with 0

To clear the buffer we have to use ‘fflush(stdin)’ – it works for only single character.

Assignment:

Scan the employee record in diff formats

Name

Id

Gender

Adress

Ph no

salary

designation

by entering the space for character in scanf it takes end the user enters.

Base address – starting address

Address of i: 3472

Address of f: 3476

Address of ch: 3471

Address of str1: 3488

Address of d: 3480

Printf(“\naddress of i: %u”,&i);

Don’t give the space before and after %d in scanf.

Don’t give ‘\n,\t’ in scanf.

Scanf(“%d %d”,&I,&J); --🡪 it is ok.

* If we want to store the output in buffer we use ‘sprintf()’. – ref p4.c in day07
* Convert the integer to string by using buffer. And also we can use ‘atoi’.