UNIT-1

Module – 1: (Introduction to Programming)

Introduction to components of a computer system

- Memory, processor, I/O Devices
- Storage, Operating system, Concept of assembler, Compiler, Interpreter, Loader and Linker.

Idea of Algorithm:

- Representation of Algorithm
- Flowchart, Pseudo code with examples
- From algorithms to programs, Source code.

Programming Basics:

- Structure of C program
- Writing and executing the first C program.
- Syntax and logical errors in compilation, Object and executable code.
- Components of C language. Standard I/O in C.
- Fundamental data types, Variables and Memory locations.
- Storage classes.

Short Questions & Answers

Q**1. What are the various functions of an Operating System?**

Ans: (i) Process Management (ii) Memory Management (iii) CPU Scheduling
(iv) I/O Management (v) Network Management. (vi) Security management.

Q2.What is the working of Loader System Utility?

Ans: This is an OS utility that copies programs from a storage device to RAM(Main Memory) , where they can be executed. Types of loaders are like Absolute loader, bootstrap loader

Q3.What is the difference between low level language and high level language?

Ans: Low level language is machine language in form of 0 and 1 or Assembly language in Mnemonics (e.g : ADD , SUB ,DIV, STR, LOAD)which can be understand by computer hardware. It is very difficult to understand by the user. Each type of CPU has its own machine language depending on the hardware.

High level language is those which are used by programmers to create computer instructions, coding the programs and to develop the software's. These are closer to human languages and are combination of natural language like English & some mathematical expressions and Language Syntax and rules. These are easier to understand, write and maintain as compared to low level languages. **Example: Ada, Pascal, FORTRAN, C, C++, Java, C #, COBOL.**

Q4. What do you mean by internal and external commands of MS-DOS?

Ans: **Internal commands** are stored in the system memory. These are the command frequently used by user. ***Example: cd, copy, md , rd, Time, Ver. , Del , Type, Echo, Rename, cls, Path, Erase etc.***

External commands are those that require large requirements and/or not commonly used. Many of the external commands are located in the Windows\System 32 directories.
Example: Append, Attrib, Chkdsk, Deltree, DOSshell, MSbackup, Ping, Net, Telnet, Xcopy, Defrag, Doskey, Format, Fast help etc.

Q5.What is Arithmetic Logic Unit in CPU?

Ans: ALU performs two types of operations—Arithmetic & logic. Arithmetic operations are like addition, subtraction, division trigonometrically functions (Sine, Cosine, etc.), square roots.

Logical operations involve Boolean Logic like: AND, OR, NOT.

Q6. What is ann Operating System(OS) in computer terminology ? Mention any five types of Operating System.

Ans: (i) An operating system is a system software which works as an interface between the user and hardware of computer system.

(ii) OS is responsible for managing and coordinating various activities and resource sharing within the given limit of computer. Example : Microsoft windows, Linux, Android, Mac OS, Symbian etc.

Various types of OS are as follows :

(i) Real time OS (ii) Multi user and single user OS (iii) Multitasking OS (iv) Multi Programming OS (v) Distributed OS (vi) Network OS

(vii) Embedded OS.

Q7. What are language translators? Give examples.

Ans: Translator is a program that converts a code written in one language to another language.

Example : Compiler , Assembler , Interpreter. It also checks and points the errors in a program. Compiler resides in the main memory during compilation of a code. It has a separate editor in which user types the program.

Ques 8. What are the differences b/w compiler and interpreter?

Ans :

S.NO	COMPILER	INTERPRETER
1.	Entire program is checked for errors and statement in a one go, lists all the error statements, converts into machine code.	Errors are checked statement by statement and interpreter converts into machine code and executes it.
2.	Does not stop when an erroneous statements is encountered.	Stops when an erroneous statements is encountered
3.	Separate command is issued to execute Statement.	Executes the statement after converting a non erroneous data of program.
4.	Generally a separate editor is used to enter the program.	A built in editor is available.
5.	Compiler resides in the main memory during the process of compilation only. Hence storage is not wasted.	Interpreter should be raised in main memory throughout the process of checking for errors.

Ques 9. Define Software. Differentiate between system software and application software.

Ans : Software is basically a collection of programs , where as programs are set of instructions which are stored electronically in computer system. Example : Adobe photo shop, railway reservation software, anti virus software, gaming softwares, operating system.

S.No	System Softwares	Application Softwares
1.	These are low level programs that interact with computer at very basic level.	These are high end softwares used to solve specific problems
2.	Mostly written in assembly language, C	These are written in C , C++, Java, Dot Net, Android
3.	Some utility programs like device drivers are also system softwares. These are generally hardware specific.	These are portable to any type of platform/Operating system
4.	Example : Operating system, Device drivers, compilers, linker , loader.	MS Office, Adobe photo shop, Gaming softwares, Android apps, reservation sites.

Ques 10. What are the different data types?

Ans: In computer programming, a **data type** is a classification identifying one of various types of data, such as floating-point, integer, or Boolean, that determines the possible values for that type; the operations that can be done on that type; and the way the values of that type are stored.

(a) **Integer :** In more common parlance, whole number**;** a number that has no fractional part.

e.g : 23, 678, -18 etc.

(b) Floating-point : A number with a decimal point. For example, 3 is an integer, but 3.5 is a floating-point number. Another representation of floating point constant is **3.5×10^{-3} . Here 3.5 is a mantissa and 10^{-3} is exponent.**

(c) **Character (text):** Readable text . E.g : A, b , Z, c , d.

Ques 11. Mention some Characteristics of computer system.

Ans : (i) Speed : Computers can perform millions of operations per second. Speed is given in nanoseconds and pico seconds.

(ii)Accuracy: A computer is very fast , reliable, and robust electronic device. It always gives accurate results, provided set of instructions are correct data and set of instructions to it.

(ii) **Automation** : These are used to automate any type of mechanical devices , or day to day problems.

(iii) **Versatile** : Computer systems are flexible towards the changing technology, hardware and softwares. They also can be used as personal computers, for scientific purpose, fr weather forecasting , for military operations and business analysis too.

(iv) **Memory** : Like human beings computers can also store large set of information and data Fetched from real world . Memory is used to store information. Primary and secondary memory are designed inside the system.

Long Questions and Answers

Ques 12. What are tokens in C programming language? Describe with examples. Ans: In C programming tokens are defined as the smallest unit of a program which is indivisible. Program is combination of these C tokens. There are following six types of token categories :

(i) **Keywords**: These are set of fixed or reserved words that can not be used as an identifier or variable. E.g: printf, scanf, gets, stdio, clrscr, int , float , char, signed, break, for , while etc.

(ii) **Identifiers:** These are used to identify the data and other objects in a program. These are names given to variables, array, functions consisting of alphabets, numerals and underscore.
Identifiers never start with a number. E.g : a , ab, akash_cse, ab123.

(iii) **Constants :** Those data objects which are fixed and values can't be changed.
E.g : Integer constant : 1 , 34, 257, -675 etc.
Floating constant : 0.02, 3.145159, 2.14E-3 etc.
Char constant : A single character inclosed in a single quotes. E.g : 'a', '&', 'x' etc. In computer characters are stored using machine's character sets in ASCII form.

(iv) **String constants** : Sequence of characters stored within double quotes. E.g : "abc", " xy67", "computer" etc.

(v) **Special characters** : @ , $, &, !, }, % etc.

(vi) **Operators** : this is a symbol that specifies the mathematical, logical, or relational operation to be performed .e.g : = , -, &&, /, =, = = etc.

Ques 13. What is an algorithm? What is its purpose? Mention the characteristics of algorithm. Write an algorithm to print the first 10 natural numbers. Also draw its flow chart.

Ans : Algorithm is a formally defined procedure for performing calculation. If a procedure is formally defined then it must be represented in a formal way i.e programming language. The algorithm gives step by step solution to reach some goal of a problem or to find a solution of existing problem. Purpose of algorithms is to implement re-usability in software applications. Once we have designed the overview of solution to any problem in algorithmic form it can be coded in any language like C , C++,Java etc.

Psuedo Code : Algorithms can be represented in the fofm of pseudo code and flow chart both. Psuedo code is the way to write any algorithm using some formal notations of mathematics, programming syntax and English words step by step.

Characteristics of algorithm are as follows:

(i) Algorithm must be precise and simple.

(ii) Algorithm must be reusable.

(iii) Algorithm must be written in finite number of steps.

(iv) Algorithm must be unambiguous. It should not represent more than one meaning or for same input there should not be different output.

(v) Algorithm must be effective and should be terminated easily in minimum time .

Control Structures used in algorithms are :

(i) Sequence : Each step of algorithm is executes in specific order one after another.

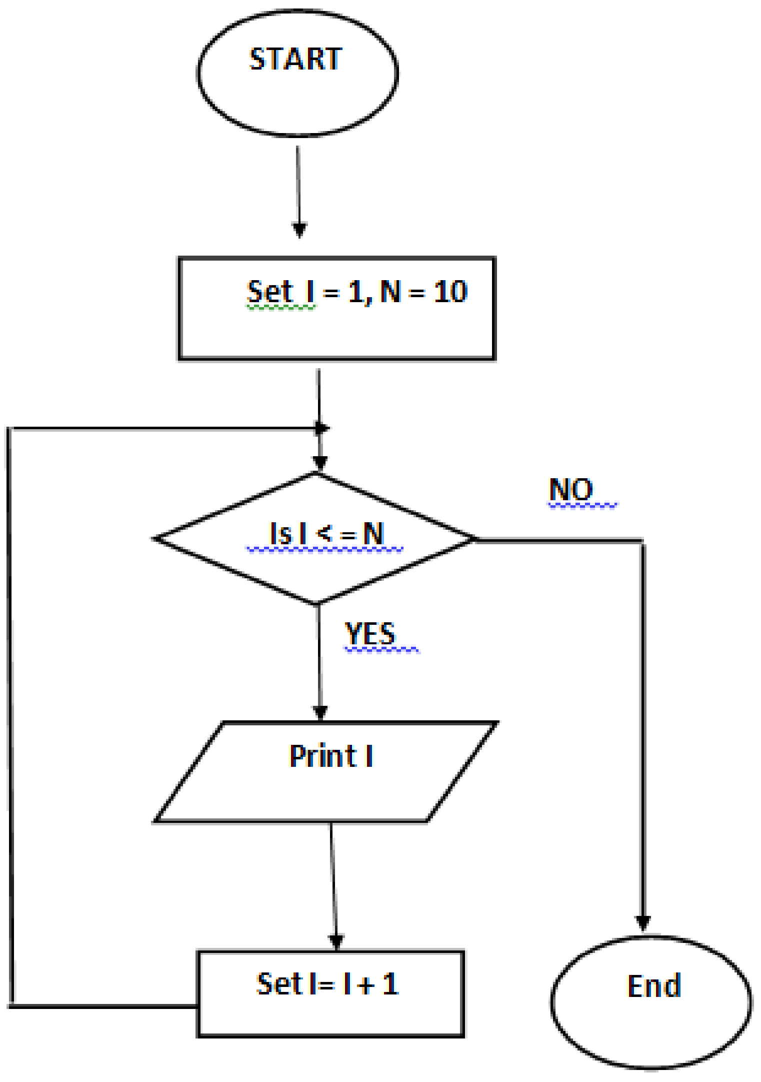
(ii) Decision : When the output of a process depends on certain decision criterion, then Decision statements are used. E.g : if x = y, then print " EQUAL". Decision statements may either evaluate to TRUE or FALSE.

(iii) Repetition: This involves execution of one or more steps for a number of times repeatedly. Programming constructs such as while, do-while and for loops are used.

Algorithm to Print the First N natural numbers

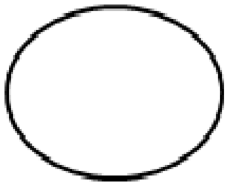
```
Step 1: [initialize] set I= 1, N= 10
Step 2 : Repeat steps 3 and 4 while I <=N
Step 3 : Print I
Step 4 : Set I = I + 1
          [ End of loop]
Step 5 : End
```

Flow Chart

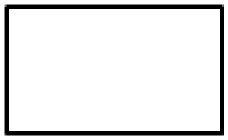


Ques 14. What are the symbols used in flowchart?

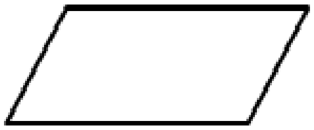
Ans: A flow chart consists of various boxes with different shapes connected by flow lines Flow lines have arrows that tell the direction of command or data flow between the boxes. Symbols used are as follows



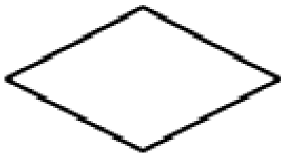
Oval: Represents Start / End point of the



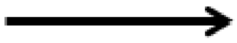
Rectangle: Represents Process of an algorithm



Parallelogram: Represents Input/ Output of Algorithms.

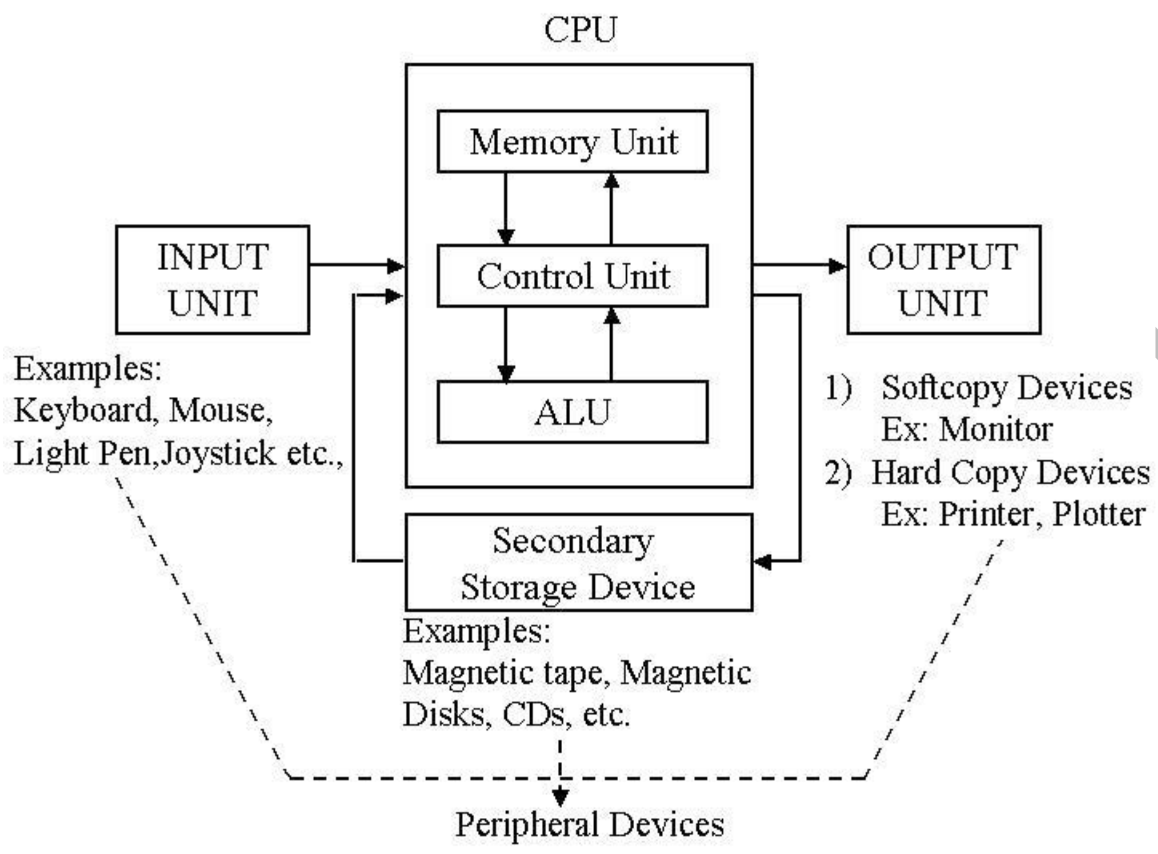


Diamond: Represents Decision Making / Conditions in



Arrow: Connector Between different shapes showing relationships in algorithm

Ques 15.Draw the block diagram of computer systems .What are major functions of a computer. Also explain its components.



Major functions of a computer are as following :

(a) Accepting the data or(input).
(b) Storing the data.
(c) Processing the data.
(d) Display results (output)
(e) Controlling and coordinating all the operations inside the computer.

Components of a computer system

(i) **Input :** This is the process of entering data and instructions /programs into the computer system with the help of input devices (e.g : Keyboard, Mouse, Microphone etc.)Input device converts the input data into binary code (0 or 1) for further processing**.**

(ii) **Storage :** Storage is the process of saving and storing the data and information in computer memory. It also stores intermediate results.

(a) **Primary Storage :** This is aslo called main memory, which is directly accessible by the CPU at very high speeds.It is used to store the data and parts of programs, the intermediate resultsof processing and current job results.
Primary memeory is volatile in nature i.e. as sson as computer is switched off, data is erased/lost from the memory. It is also expensive E.g : RAM(random access Memory)

(b) **Secondary storage :** Also known as auxiliary memory. This is non-volatile memory And data is permanently stored in this. E.g : magnetic Disks, CD ,DVD, Hard Disk, Pen Drives, Memory Cards etc.

Central Processing Unit (CPU) : This is also called processor which is chip inside the Cabinet of computer system . It is referred as the brain of computer. It carries out instructions of a computer program by performing basic arithmetic , logical , control and I/O operations specified by instruction. Speed of CPU is measured in Giga Hertz (Ghz).

(iii) **Control Unit :** A control unit handles all the processor signals. It directs all the input and output flow, fetches instructions from microprograms and directs it to other units of computer by timing and control signals.Control unit takes care about correct execution of instructions.

(iv) **ALU : Arithmetic and logic unit** is used to perform arithmetic opeartions and logical operations. Arithmetic operations include adition , subtraction, division, multiplication , trignometrical operation (like Sin, Cos etc)
Logical operations consists of Boolean operations that predict either TRUE/1 or FALSE/0. Example : logical AND, NOT, XOR, OR etc.

(v) **Output :** This consists of operation that provides the result regarding certain input And sends data from computer to another device provided in the system.
O/P peripheral devices are printers, speakers, monitors, projectors, headphones etc.

Ques 16. Write Short Notes on the following:

(i) Top down programming **(ii) Structured Programming**
(iii) Syntax and logical errors **(iv) Linker & Loader**

Ans : (i) Top down programming : In this approach of programming a system is break down into smaller sub systems , so that we can see more detail concept of modules. In a top down approach an overview of the system is firstly formulated , then specifying but not detailing any first level sub systems.

Each sub system is then refined into more primary and smaller sub systems. This is called modularization. Modularization makes the program to understand , write and debug.

Structured Programming : This implements logical structure on the program to be written. Thus program becomes more efficient, simple and easy. This allows program loaded into memory and to be reused .Modules are coded separately and tested. After that all the modules are combined together to make larger sub **system. Language like C , Pascal support structured programming. Each module has its own local data and** performs specific task.

Syntax and logical errors : When compiler compiles the program then due certain wrong inputs and wrong codes errors occur. Syntax Error occurs when code or program has grammatical mistakes due to mistyping. Like missing parenthesis, not using ; etc.Syntax error leads to no output.

Logical Error are those which give the output which is not required for certain inputs due to wrong interpretation of algorithm and logic in code.

Loader : This is an Operating system utility that copies programs from a storage A storage device to main memory, where they can be executed.

Absolute Loader : Loads the object program from translation time address and transfers control to it.

Bootstrap loader : This loader is responsible for loading the OS when system is turned ON and transferring control to it. It is present in ROM area of main memory.

Linker : Also called link editor / binder. It is a program that combines object modules to form an executable program. This simplifies the designing of large softwares because they can be divided into smaller modules and afterwards can be combined using linker.

Ques 17. What is the structure of C program ? What are header files and their uses. Explain formatted input output functions.

Ans : <u>Structure of C Language program</u>

1) Comment line

2) Preprocessor directive

3) Global variable declaration

4) main function( )

{

Local variables;

Statements;

}

User defined function

}

}

Comment line

It indicates the purpose of the program. It is represented as

/*……………………………..*/

Comment line is used for increasing the readability of the program. It is useful in explaining the program and generally used for documentation. It is enclosed withi the decimeters. Comment line can be single or multiple line but should not be nested. It can be anywhere in the program except inside string constant & character constant.

Preprocessor Directive:

#include<stdio.h> tells the compiler to include information about the standard input/output library. It is also used in symbolic constant such as #define PI 3.14(value). The stdio.h (standard input output header file) contains definition &declaration of system defined function such as printf( ), scanf( ) etc. Generally printf() function used to display and scanf() function used to read value

Global Declaration:

This is the section where variable are declared globally so that it can be access by all the functions used in the program. And it is generally declared outside the function.

main()

It is the user defined function and every function has one main() function from where actually program is started and it is encloses within the pair of curly braces. The main( ) function can be anywhere in the program but in general practice it isplaced in the first position.

Syntax :

```
main()
{
……..
……..
……..
}
```

The main( ) function return value when it declared by data type as

```
int main( )
{ return 0}
```

The main function does not return any value when void (means null/empty) as

```
void main(void ) or void main()
{
printf (“C language”);
}
```

Output: C language

Example : **/*First c program with return statement*/**

```
#include <stdio.h>
int main (void)
{
printf ("welcome to c Programming language.\n");
return 0;
}
```

Output: welcome to c programming language

Header Files : When we work with large projects, it is desirable to separate out some procedures from main() function of the program. Some procedures need to be used several times in different programs. So one solution is to copy the code of that procedure from one program to another several times , which results in waste of time.

So another solution is to make procedures and store them in a different file called header files. System defined Header files contain the definitions of Library functions like printf(), scanf(), getch() , puts(), pow(), clrscr()

etc.***Conventionally , header files ends with a ' . h' extension and names can use only letters, digits, dashes , and underscores. Header files can be system define in compiler it self or programmers can design their own header files too.***

Examples : **stdio.h** : standard input & output (functions like printf() ,scanf() are defined.)

conio.h : console input & output. (getch(), clrscr () are defined.)

string.h : for string handling functions. ( )

math.h : Math functions like pow(), sin(), log() , sqrt() are defined.

stdlib.h : Library functions like exit (),

Foramtted Input & output functions : C language supports two formatting functions

printf () and scanf (). ***Printf() is used to convert text data stored in the program onto text stream for o/p to the monitor. Scanf is used to convert text stream coming from keyboard to data values and stores them in program variables.*** These are called as ***formatted Input and output functions*** because they have special format to read and write The text from streams and then converting them into binary stream.

<u>Syntax of printf() :</u> printf (" control string" , var1, var2 …..var n).

Printf ("Hello welcome to C program")

Printf (" %d %d %f " a , b , c), where a and b are integer variables and c is float variable.

%d and %f are known as format specifiers.

<u>Syntax of scanf()</u> : scanf (" Control string" , & var1, & var2, ……..,& var n).

Format specifiers : For integer variables ------ %d

For float variables ------- %f

For character variables ------- %c

Ques 18. Perform the following number system conversions :

(a) $(101101)_2 = (\ ?\ )_{10}$ (b) $(10110101)_2 = (\ ?\ )_{16}$ (C) $(51\ .\ 54)_8 = (\ ?\ )_2$

(d) $(101001\ .\ 1011)_2 = (\ ?\ )_{16}$ (e) $(4307)_8 = (\ ?\ )_{10}$ (f) Convert 174_{10} to binary:

Ans : (a)

$$\begin{aligned} 101101 &= 1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 \\ &= 1 \times 32 + 0 \times 16 + 1 \times 8 + 1 \times 4 + 0 \times 2 + 1 \times 1 \\ &= 32 + 8 + 4 + 1 \end{aligned}$$

(b)

Divide into groups for 4 digits	1011 0101
Convert each group to hex digit	B 5
	$B5_{16}$

(c)

$$51.54_8 = \underline{101}\ \underline{001}.\underline{101}\ \underline{100}_2 = 101001.101100_2$$

(d)

$$101001.1011 = \underline{0010}\ \underline{1001}.\underline{1011}_2 = 29.B_{16}$$

(e)

$$4307_8 = 4\times8^3+3\times8^2+0\times8^1+7\times8^0 = 2247$$

(f)

Convert 174_{10} to binary:

Division by 2	Quotient	Remainder	Bit #
174/2	87	0	0
87/2	43	1	1
43/2	21	1	2
21/2	10	1	3
10/2	5	0	4
5/2	2	1	5
2/2	1	0	6
1/2	0	1	7

So $174_{10} = 10101110_2$

Ques 19. What are storage classes? Explain their types briefly.

Ans : ***Storage class in C*** decides the part of storage to allocate memory for a variable.

- It also determines the scope of a variable. All variables defined in a C program get some physical location in memory where variable's value is stored.
- Memory and CPU registers are types of memory locations where a variable's value can be stored.
- The storage class of a variable in C determines the ***life time*** of the variable if this is ***'global' or 'local'.***

- Along with the life time of a variable, ***storage class*** also determines variable's storage location (memory or registers), the ***scope (visibility level) of the variable***, and ***the initial value of the variable.***
- ***There are four storage classes in C those are automatic, register, static, and external.***

(i) Automatic Storage Class:

- This is the default storage class for all the variables declared inside a function or a block.
- Auto variables can be only accessed within the block/function they have been declared and not outside them (which defines their scope).
- They are assigned a garbage value by default whenever they are declared.
- Variables having automatic storage class are local to the block which they are defined in, and get destroyed on exit from the block.

```
#include <stdio.h>
 int main( )
 {
    auto int i = 1;
    {
        auto int i = 2;
   {
    auto int i = 3;

    printf ( "\n%d ", i);
  }
   printf ( "%d ", i);
  }
   printf( "%d\n", i);
    }
```

```
  OUTPUT
======
                        3 2 1
```

(ii) Register Storage Class:

- The **register specifier** declares a variable of register storage class.
- Variables belonging to register storage class are ***local to the block in which they are defined.***
- Variables are placed in CPU registers, not in memory.
- Register variables are also given no initial value by the compiler.

(iii) Static Storage Class:

- The ***static specifier*** gives the declared variable static storage class.
- Static variables are ***not visible outside their function or file,*** but they maintain their values between calls.
- Static variables can be used within function or file.

- Static variables have ***default initial value zero*** and initialized only once in their lifetime.

#include <stdio.h>
void staticDemo()
{
static int i;
{
static int i = 1;
printf("%d ", i);
i++;
}
printf("%d\n", i);
i++;
}
	int main()
	{
	staticDemo();
	staticDemo();
	}
	OUTPUT
	======
	1 0
	2 1

(iv) External Storage Class:

- The **extern specifier** gives the declared variable external storage class.
- Variable is declared with ***external linkage*** **elsewhere** in the program.
- When extern specifier is used with a variable declaration then ***no storage is allocated to that variable*** .
- Iit is assumed that the variable has already been defined elsewhere in the program.

When we use extern specifier the variable cannot be initialized because with extern specifier variable is declared, not defined.

```
#include <stdio.h>
 extern int x;
 int main()
{
  printf("x: %d\n", x);
}
int x = 10;
```