Definition of Digital computer:

Digital Computer is a electronic machine or a device that helps to process any kind of information. These are the devices through which users provide some input and get the output within a fraction of seconds.

There are mainly three parts in a digital computer and it consists of –

Input: The user normally provides the data to the device that is known as input.

Processing: The input that is provided by the user is processed internally using some defined sequence.

Output: Once the processing is completed, based on the input, the output is displayed to the user.

A digital computer computer is a combination of Hardware and Software.

Hardware:- Hardware refers to the physical components of a computer. Computer Hardware is any part of the computer that we can touch these parts. Examples of hardware in a computer are the Processor, Memory Devices, Monitor, Printer, Keyboard, Mouse.

Software:- Software is a collection of instructions, procedures, documentation that performs different tasks on a computer system. Examples of software are Ms Word, Excel, Power Point, Google Chrome, Photoshop, MySQL etc.

Computer software is classified into two types

1. **Application Software**:- Application software are used to perform a particular task. Examples of application software are Ms Word, Excel, Power Point, Google Chrome, Photoshop, MySQL etc.

2. **System Software**:- System software provides an environment on which application software can execute and also provides an interface between user and computer hardware. Examples of system software are operating system(OS), compiler, linker etc.

Among all system software OS is the primary one because without it user's are unable to use any computer hardware and perform any task. Examples of operating systems are different version of Microsoft Windows like XP, Vista,7, 8,10 ,Unix, different distribution of Linux like mint, fedora, Red Hat , MAC etc

Types of computer on the basis of capacity

a) Super Computer

It is the fastest type of computer. Supercomputers are very expensive and are employed for specialized applications that require immense amounts of mathematical calculations. For example, weather forecasting requires a supercomputer. Other uses of supercomputers include animated graphics, fluid dynamic calculations, nuclear energy research, and petroleum exploration.

b) Mainframe Computer

A very large and expensive computer capable of supporting hundreds or even thousands of users simultaneously. In the hierarchy computers mainframes are just below supercomputers.

Note: The chief difference between a supercomputer and a mainframe is that mainframes are more powerful than supercomputers because they support more simultaneous programs. But supercomputers can execute a single program faster than a mainframe.

c) Mini Computer

A minicomputer is a multiprocessing system capable of supporting from 4 to about 200 users simultaneously.

d) Micro Computer

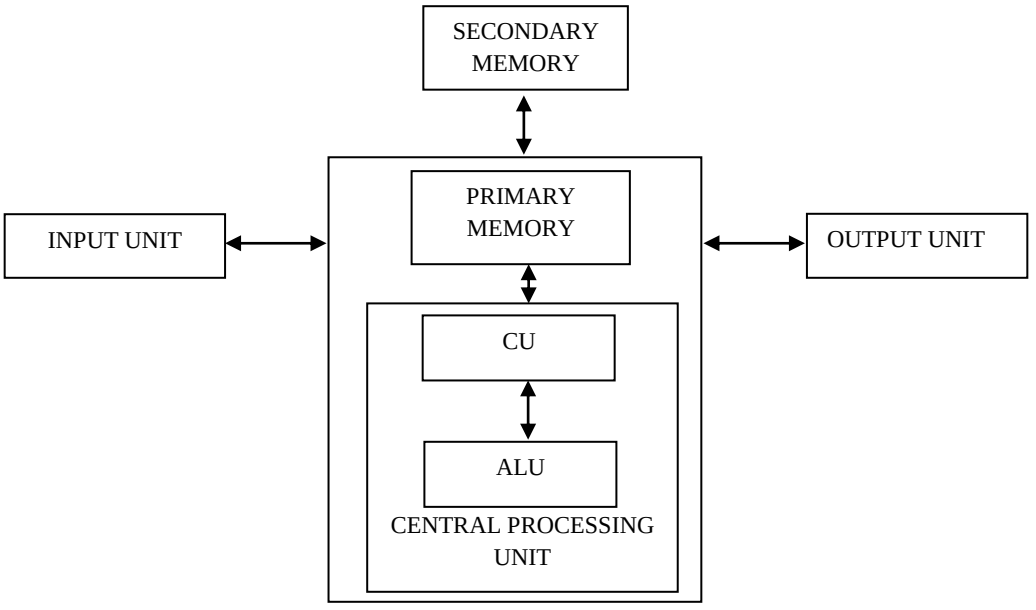
i. **Desktop Computer:** a personal or micro-mini computer sufficient to fit on a desk.

ii. **Laptop Computer:** a portable computer complete with an integrated screen and keyboard. It is generally smaller in size than a desktop computer and larger than a notebook computer.

iii. **Palmtop Computer/Digital Diary /Notebook /PDAs:** a hand-sized computer. Palmtops have no keyboard but the screen serves both as an input and output device.

iv. **Workstation:** A workstation is simply a desktop computer that has a more powerful processor, additional memory and enhanced capabilities for performing a special group of task, such as 3D Graphics or game development.

Introduction to components of a computer system



A computer system consists of mainly four basic units

- **input unit**
- **storage unit**
- **central processing unit**
- **output unit.**

Central Processing unit further includes **Arithmetic logic unit and control unit.** Storage unit is divided into two parts **Primary** and **Secondary memory.** These components of a computer performs four major operations or functions

• it accepts data or instructions as input,
• it stores data and instruction
• it processes data as per the instructions,
• it gives results in the form of output.

Input Unit: - An input unit is device through which data and programs from user are entered into the computer. The input unit links the external environment with the computer system. Input unit accepts data and instruction in human readable form, but it transforms data and instructions into binary codes before further processing. This transformation is accomplished by units that called input interfaces.
In short, an input unit performs the following functions.

1. It accepts (or reads) the list of instructions and data from the outside world.
2. It converts these instructions and data in computer acceptable format.
3. It supplies the converted instructions and data to the computer system for further processing.

Example : - Keyboard, mouse, scanner and bar-code reader.

Output Unit: - An output unit is a device through which results stored in the computer memory are made available to the user. It links the computer with the external environment. As computers work with binary code, the results produced are also in the binary form. Hence, before supplying the results to the outside world, it must be converted to human acceptable (readable) form. This task is accomplished by units called output interfaces.

In short, the following functions are performed by an output unit.

1. It accepts the results produced by the computer which are in coded form and hence cannot be easily understood by us.
2. It converts these coded results to human acceptable (readable) form.
3. It supplied the converted results to the outside world.

Example: - Monitor, printer and speaker.

Storage Unit: - The data and instructions that are entered into the computer system through input units have to be stored inside the computer before the actual processing starts. Similarly, the results produced by the computer after processing must also be kept somewhere inside the computer system temporarily or permanently before being passed on to the output units. Storage unit is designed for this perpose only.
the following functions are performed by an storage unit

1. All the data to be processed and the instruction required for processing (received from input devices).
2. Intermediate results of processing.
3. Final results of processing before these results are released to an output device.

There are two types of storage unit available in all modern computers

- Primary Memory
- Secondary Memory

Primary Memory: - This type of memory is usually small in size but very fast. Primary Memory is internal memory of the computer. RAM AND ROM both form part of primary memory. The primary memory provides main working space to the computer.

- **Ramdom Access Memory(RAM)/Main Memory**: - the data and instructions needs to be stored in the main memory before execution. The primary storage is referred to as random access memory (RAM) because it is possible to randomly select and use any location of the memory directly store and retrieve data. The storage of data and instructions inside the primary storage is disappeared from RAM as soon as the power to the computer is switched off. That's why main memory is also known as volatile memory. There are two types os RAM
- **Read Only Memory (ROM)**: - here is another memory in computer, which is called Read Only Memory (ROM). The storage of program and data in the ROM is permanent. The ROM stores some standard processing programs supplied by the manufacturers to operate the

personal computer. The ROM can only be read by the CPU but it cannot be changed. ROM does not lose their content on failure of power suppl. That's why, it is known as non-volatile memory. There are different types of ROM available

- **Programmable Read Only Memory(PROM)**: - Data is written into a ROM at the time of manufacture. However the content can be programmed by user with a special PROM programmer. PROM provides flexible and economical fixed programs and data.
- **Erasable Programmable Read Only Memory(EPROM)**: - this allow the programmer to erase the contents of the ROM and reprogram it. The contents of the EPROM cells can be erased using ultra violet light using EPROM programmer.
- **Electrically Erasable Programmable Read Only Memory(EEPROM)**: - In this type of ROM, the contents of the cells can be erased electrically, by applying a high voltage. The EEPROM need not to be removed physically for reprogramming.

Secondary Memory: - Secondary memory is external and permanent in nature. The secondary memory is concerned with magnetic memory. Secondary memory can be stored on storage media like floppy disks, magnetic disks, magnetic tapes. This memory can also be stored optically on Optical disks - CD-ROM. The following are the popular secondary memory:

- **Magnetic Tape:** Magnetic tapes are used for large computers like mainframe computers where large volume of data is stored for a longer time. In PC also you can use tapes in the form of cassettes. The cost of storing data in tapes is inexpensive. Tapes consist of magnetic materials that store data permanently.
- **Magnetic Disk:** You might have seen the gramophone record, which is circular like a disk and coated with magnetic material. Magnetic disks used in computer are made on the same principle. It rotates with very high speed inside the computer drive. Data is stored on both the surface of the disk. For Example-Floppy Disk.
- **Optical Disk:** With every new application and software there is greater demand for memory capacity. It is the necessity to store large volume of data that has led to the development of optical disk storage medium. Example Compact Disk/ Read Only Memory (CD-ROM)

Central Processing Unit (CPU): - The main unit inside the computer is the **CPU**. This unit is responsible for all events inside the computer. It controls all internal and external devices. The control Unit and the Arithmetic and Logic unit of a computer system are jointly known as the Central Processing Unit (CPU). The CPU is the brain of any computer system. In a human body, all major decisions are taken by the brain and the other parts of the body function as directed by the brain. Similarly, in a computer system, all major calculations and comparisons are made inside the CPU and the CPU is also responsible for activating and controlling the operations of other units of a computer system.

Arithmetic Logical Unit (ALU): - **(ALU)** of a computer system is the place where the actual execution of the instructions take place during the processing operations. All arithmetic operation such as addition, substation, multiplication and division are performed and all logical operation like AND, OR, NOT (decisions) are made in the **ALU**. ALU has small amount of special memory unit called registers.

Algorithm: -

An algorithm is a finite set of well-defined instructions or step by step procedure to solve a given problem. In other word, a sequential solution of any program that written in human language, called algorithm. Algorithm is first step of the solution process, after the analysis of problem, programmer writes the algorithm of that problem.

According to Knuth an algorithm must possess the following properties:

Finiteness: The algorithm must always terminate after a finite number of steps.

Definiteness: Each step must be precisely defined; the actions to be carried out must be rigorously and unambiguously specified for each case.

Input: An algorithm has zero or more inputs, taken from a specified set of objects.

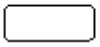
Output: An algorithm has one or more outputs, which have a specified relation to the inputs.

Effectiveness: All operations to be performed must be sufficiently basic that they can be done exactly and in finite length**.**

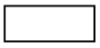
Introduction to flowcharts

A flowchart is a graphical representation of an algorithm. These flowcharts play a vital role in the programming of a problem and are quite helpful in understanding the logic of complicated and lengthy problems. Once the flowchart is drawn, it becomes easy to write the program in any high-level language. Often, we see how flowcharts are helpful in explaining the program to others. Hence, it is correct to say that a flowchart is a must for the better documentation of a complex program.

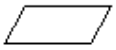
Flowcharts are usually drawn using some standard symbols; however,



Start or end of the program



Computational steps or processing function of a program



Input or output operation

Decision making and branching

Connector or joining of two parts of program

Problem 1: Write an algorithm and draw the flowchart for finding the average of two numbers

Algorithm:

Input: two numbers x and y

Output: the average of x and y

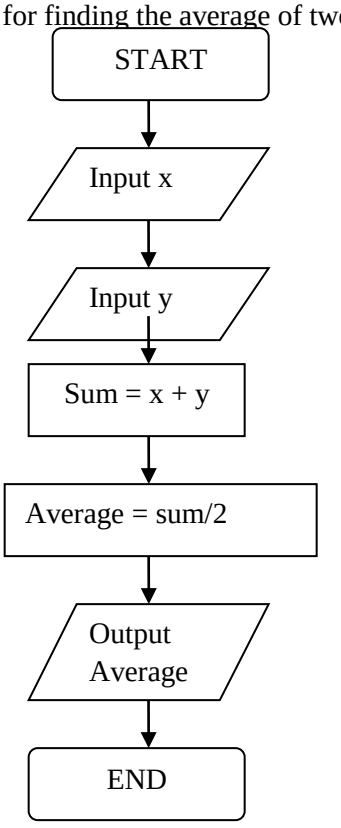
Step 1: input x

Step 2:input y

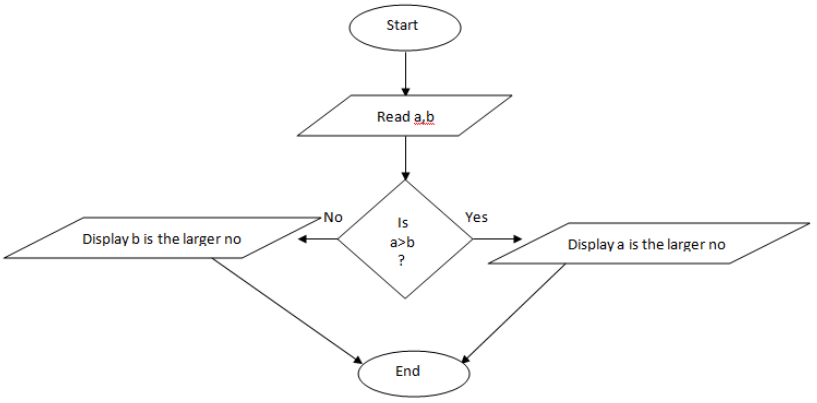
Step 3:sum = x + y

Step 4:average = sum /2

Step 5:output average



Problem 2: Write an algorithm and draw the flowchart for finding largest among two numbers



Step 1: Start

Step 2: Read a, b .

Step 3: If a>b then

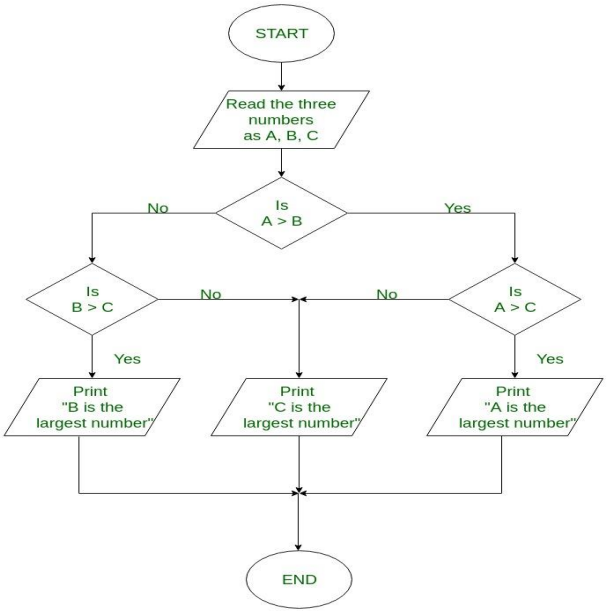
Display “a is the largest number”.

Step 4: Otherwise

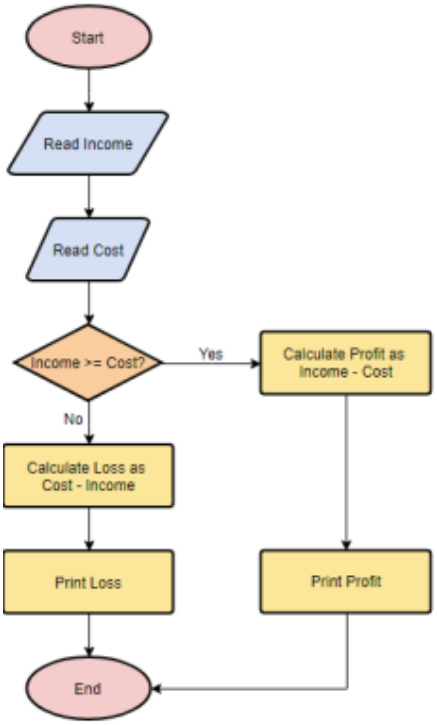
Display “b is the largest number”.

Step 4: End.

Problem 3: Flowchart to find the greater of 3 numbers.



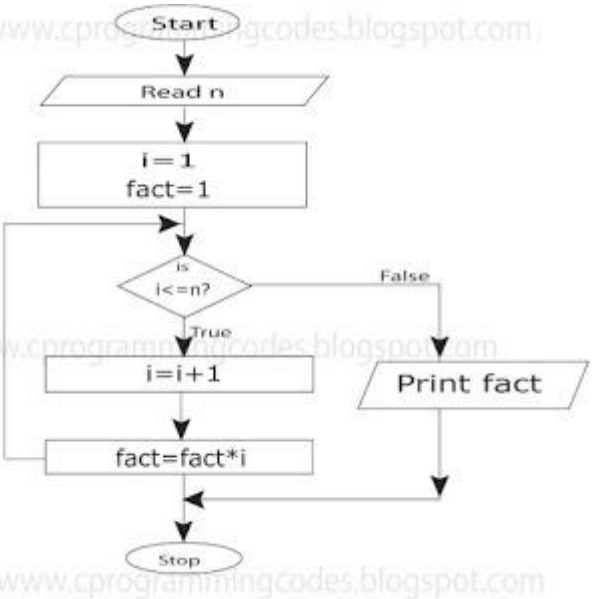
Problem 4 Flowchart to find the profit and loss.



Problem 5: Write an algorithm and draw the flowchart for Factorial of a given number.

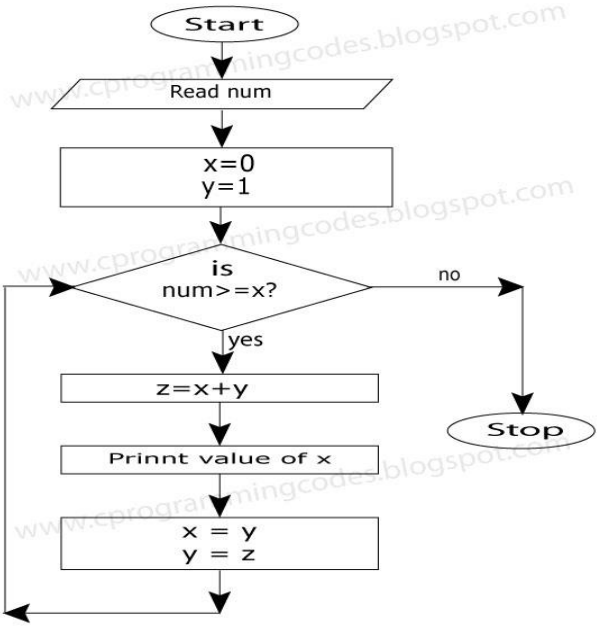
Algorithm to calculate the factorial of a number

step 1. Start
step 2. Read the number n
step 3. [Initialize]
i=1, fact=1
step 4. Repeat step 4 through 6 until i=n
step 5. fact=fact*i
step 6. i=i+1



step 7. Print fact
step 8. Stop

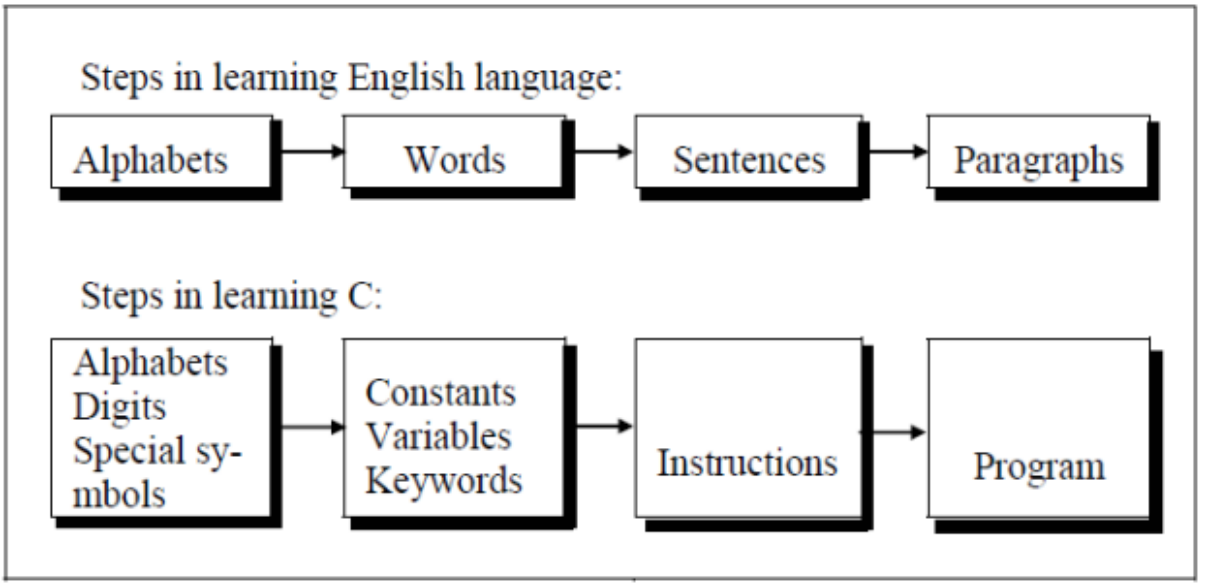
Problem 6: Write an algorithm and draw the flowchart for Fibonacci series up to n terms.



C is a high-level programming language developed at AT & T's Bell Laboratories of USA in 1972. It was designed and written by a man named Dennis Ritchie. It was initially designed for programming UNIX operating system. Now the software tool as well as the C compiler is written in C. Major parts of popular operating systems like Windows, UNIX, Linux is still written in C. This is because even today when it comes to performance (speed of execution) nothing beats C.

There is a close analogy between learning English language and learning C language. The classical method of learning English is to first learn the alphabets used in the language, then learn to combine these alphabets to form words, which in turn are combined to form sentences and sentences are combined to form paragraphs. Learning C is similar and easier. Instead of straight-away learning how to write programs, we must first know what alphabets, numbers and special symbols are used in C, then how

using them constants, variables and keywords are constructed, and finally how are these combined to form an instruction. A group of instructions would be combined later on to form a program. a computer program is just a collection of the instructions necessary to solve a specific problem.



Character set :

A character denotes any alphabet, digit or special symbol used to represent information. Valid alphabets, numbers special symbols allowed in C are

Alphabets	A, B, ….., Y, Z a, b, ……, y, z
Digits	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
Special symbols	~ ‘ ! @ # % ^ & * ( ) _ - + = | \ { } [ ] : ; " ' < > , . ? /

Identifiers :

Identifiers are user defined word used to name of entities like variables, keywords and constants.

1. **Keywords**:- There are certain words reserved for doing specific task, these words are known as reserved word or keywords. These words are predefined and always written in lower case or small

letter. These keywords can't be used as a variable name as it assigned with fixed meaning. Some examples are int, short, signed, unsigned, default, volatile, float, long, double, break, continue, typedef, static, do, for, union, return, while, do, extern, register, enum, case, goto, struct, char, auto, const etc.

2. **Constant: -** Constant is any value that cannot be changed during program execution. In C, any number, single character, or character string is known as a constant. A constant is an entity that doesn't change. There are 4 types of constant supported in c language

- Integer constant: An integer constant are whole number which have no decimal point. Types of integer constants are:

  Decimal constant: 0-------9(base 10)
  Octal constant: 0-------7(base 8)

  Hexadecimal constant: 0----9, A------F(base 16)

- **Real constant**: Real constant is also called floating point constant. To construct real constant, we must follow the rule of, -real constant must have at least one digit. -It must have a decimal point. -It could be either positive or negative. -Default sign is positive. -No commas or blanks are allowed within a real constant. Ex.: +325.34 426.0 -32.76

- **Character constant**: Character constant represented as a single character enclosed within a single quote. These can be single digit, single special symbol or white spaces such as '9','c','$', ' ' etc. Every character constant has a unique integer like value in machine's character code as if machine using ASCII (American standard code for information interchange).

  A---------- Z ASCII value (65-90)

  a----------z ASCII value (97-122)

- **String constant**: Set of characters are called string and when sequence of characters are enclosed within a double quote (it may be combination of all kind of symbols) is a string constant. String constant has zero, one or more than one character and at the end of the string null character(\0) is automatically placed by compiler. Some examples are ",sarathina" , "908", "3"," ", "A" etc.

3. **Variable**: Variable is a data name which is used to store some data value or symbolic names for storing program computations and results. The value of the variable can be change during the execution of the program. We have to declare the variable before using the variable.

General syntax for declaring a variable

Datatype Variable_name;

The variable names are separated by commas and the declaration is terminated by semicolon.

Example

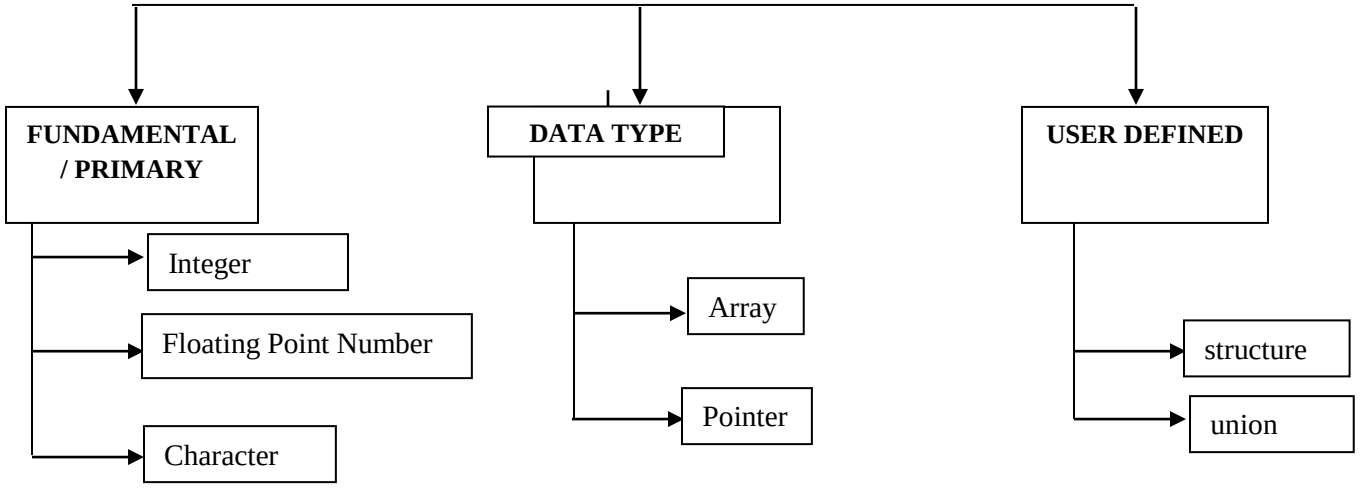
int a,b;

float c,d,e;

C is rich in datatype. It tells the compiler that the variable is going to store which type of data(integer, floating or character) and how much amount of memory is going to allocate for the variable.

Declaring variable: - There are two purposes of declaring a variable

- It tells the compiler about the name of the variable.
- It specifies what type of data the variable will hold.



Integer:- Integer is again divided into 3 types short, int and long int. The following table describes the memory requirement, format specifier and range of number stored by that datatype .

	short	**int**	**long int**
Memory requirement	2 =2*8=16 bits	4 =4*8=32 bits	8 =8*8=64 bits
Range of number	$-(2^{16-1})to+((2^{16-1})-1)$	$-(2^{32-1})to+((2^{32-1})-1)$	$-(2^{64-1})to+((2^{64-1})-1)$

Format specifier	%d	%d	%ld

Floating Point Number: Floating point numbers is again divided into 3 types float, double and long double. The following table describes the memory requirement, format specifier and range of number stored by that datatype .

	float(precision upto 6 decimal points)	**double (precision upto 15 decimal points)**	**long double (precision upto 19 decimal points)**
Memory requirement (in bytes)	4 =4*8=32 bits	8 =8*8=64 bits	10 =10*8=80 bits
Range of number	3.4E-38 to 3.4E+38	1.7E-308 to 1.7E+308	3.4E-4932 to 1.1E+4932
Format specifier	%f	%lf	%Lf

Character:- This type of variable can store a single character.

	short
Memory requirement	1 =1*8=8 bits
Range of number	$0 to + ((2^8) - 1)$
Format specifier	%c

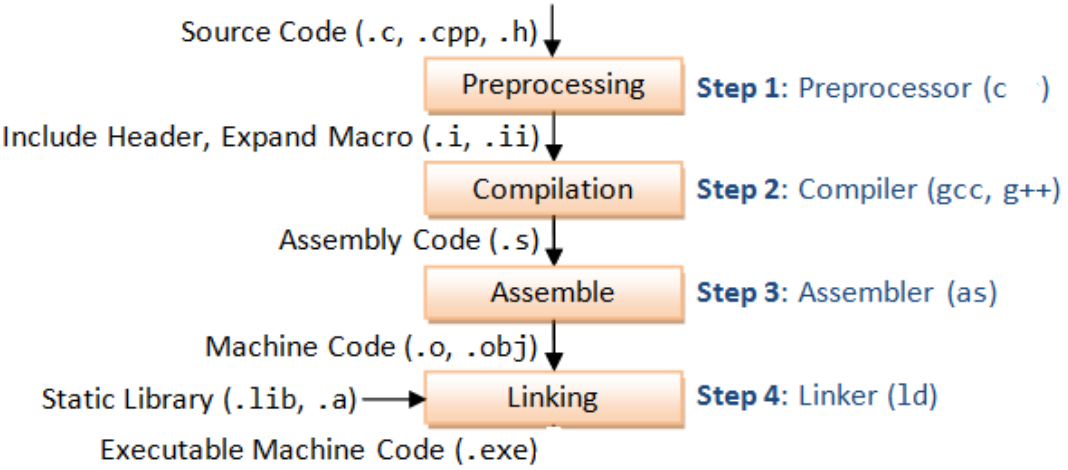
Source code, Object code and executable code: -

Source code: - Source code contains statements that are written by a human known as a programmer in any high-level languages such as Java, C, C++, Python, PHP, JavaScript. It is written by developers in any scripting or high-level language which also contains some lines of comments for a better understanding of the logic behind the code and for easy modification.

We use any text editor like vi, gedit or notepad to write the source code and save the file in the computer memory. A file containing C source code known as C source file. We can identify different types of file by their extension, like any text file has .txt extension. Similarly, any C source file is distinguished from other file by its extension as .c. For example, if we are writing a program for addition of two numbers let call it as add then we have to save it as **add.c**.

Now we want to execute the source code which is written in high level language, but we cannot directly execute the source code because system only understand machine code. Before execution the source code must be converted into executable code (machine code) which is understandable by the system. The process of translating a source code to a executable code is known as **build process**.

Build process contains many sequential steps.



1. **Preprocessing**:- The first step in the build process is preprocessing. For preprocessing we use a program called pre-processor. This program takes our source code .c file as input, do some processing and produces an intermediate code as output. In our case input is add.c and the intermediate code is add.i.

2. **Compilation** :- The next step is compilation where a software called compiler translate the intermediate into lower level code. There are many compilers available for compiling c file. Here we are using GCC compiler. GCC stands for GNU Compiler Collection. Some compilers takes the intermediate code and directly produces machine level code. But GCC compilers produces Assembly code after compilation instead machine level code. Compilers deals with all type of syntax errors present in the source file.

3. **Assemble:-** the extension of the assemble code is .s. The program called assembler takes the assembly code as input and produces object code. **Object code** has .o or .obj extension. The file containing the object code is known as **Object file**.

4. **Linking**:- The linker lings all object code together and links them with standard libraries to produce executable code. The executable code has .exe extension. Then we can able to run the executable code when it it loaded in the main memory by the loader.

Syntax Error :- Every programming language provide some rules for writing programs. The programmers have follows those rules while writing the program. Syntax Errors occur when programmers violate the rules of writing the statements of the programming language. During compilation, if the source code contains syntax error, then the location of the error and the type of the error is displayed by the compiler. Then we have to remove the error and recompile the file to get the object file. Example, misspelling of 'int' is a syntax error.

Logical Error: - A program with logical error will not cause the program to terminate the execution but the generated output is wrong. A logical error occurs due to a fault in the algorithm. The programmer have to find out the logical error because the compiler will not generate any error message for logical error.