

Computer Basics and C –programming language

~Shinjit Kamal Borah

Definition:

Computer is an electronic device that processes data according to a set of instructions to produce desired results. In this definition, we have seen some basic terminologies that define a computer system. A system must possess these characteristics or parts to be called as computer or computer like device. These are

1. Electronic device: This part is termed as **hardware**, in the study of computer. This part is visible and tangible. When we say the term computer it simply does not imply the physical part only, it also includes the characteristics of the system. Like when we say human it does not imply the body of a human only. Here, the physical part is known as Hardware which consists of electronic device. *Try to learn the difference between electrical device and electronic device.*
2. Data: The next important term found in the definition. Data is simply a value which is insufficient to give some knowledge, but collection of data becomes information, from which we can draw knowledge. (Think about a number 25, does it mean anything to you? Now consider Roll No 25, is it meaningful now? This is the difference between data and information. 25 is data but Roll No 25 is a piece of information.) If we do not provide the data computer can do nothing. We term data as an **input** in terms of computer. For example to print an application we must provide the alphabets using keyboard.
3. Set of instructions: It is known as the **software**. Computer can perform any task only if sufficient instructions are provided. Therefore, we need different software to perform different task in computer. For instance, to listen to music we need media player.
4. Desired Result: Computer is used as a tool to perform tasks to get some specific results. This result is known as **output**. Computer must be able to provide us the correct result.

Mathematically computer can be defined as a function. $y = f(x)$, where x is domain and y is co-domain Similarly,

$$\text{Output} = \text{Computer}(\text{Input})$$

From this we can say, computer performs only one task it converts the input to the output.

Components of Computer:

- C.P.U.: A central processing unit (CPU), also called a central processor or main processor, is the electronic circuitry within a computer that executes instructions that make up a computer program (set of instruction). The CPU performs basic arithmetic,

logic, controlling, and input/output (I/O) operations specified by the instructions in the program. It is also known as the brain of a computer.

- **Memory:** Computer memory is any physical device capable of storing information temporarily or permanently. We can store and retrieve data and information to and from computer memory. To process data and information, we must place these data and information in memory.
 - **Control Unit (CU):** The control unit (CU) is a component of the CPU that directs the operation of the processor. It tells the computer's memory, arithmetic and logic unit and input and output devices how to respond to the instructions that have been sent to the processor.
 - **ALU(Arithmetic Logic Unit):** The arithmetic logic unit (ALU) is a digital circuit within the processor that performs integer arithmetic and bitwise logic operations. The inputs to the ALU are the data words to be operated on (called operands), status information from previous operations, and a code from the control unit indicating which operation to perform. Depending on the instruction being executed, the operands may come from internal CPU registers or external memory, or they may be constants generated by the ALU itself.
- **Input Unit:** This unit is responsible for providing data and control signals to a computer system. As computer works only on binary data i.e. 0 and 1. This unit converts the input signal to binary or digital signal. The devices that take user input and provide to the CPU are known as Input Devices. Examples of input devices are keyboard, scanner, mouse etc.
 - **Output Unit:** This unit is responsible for providing information to an user from a computer system. As computer works only on binary data i.e. 0 and 1. This unit converts binary or digital signal to human readable format. The devices that take user output from the CPU and provide to the users are known as Output Devices. Examples of output devices are monitor, projector, printer etc.

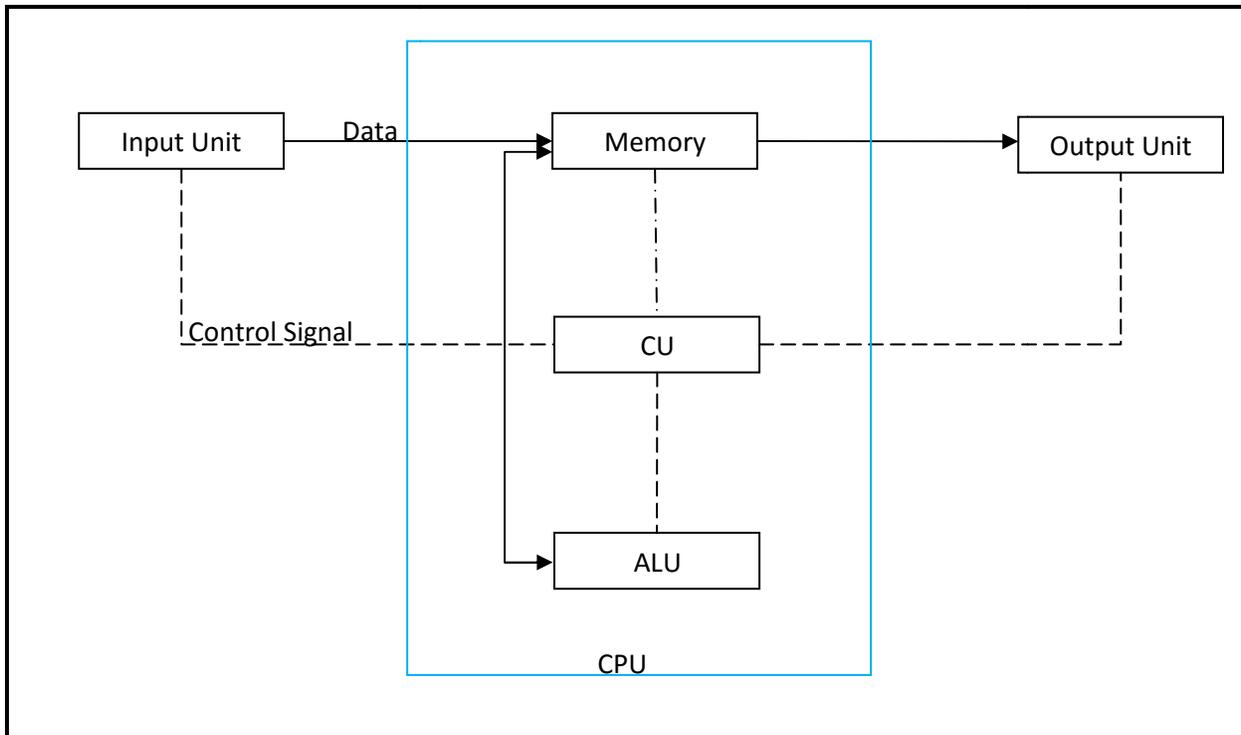
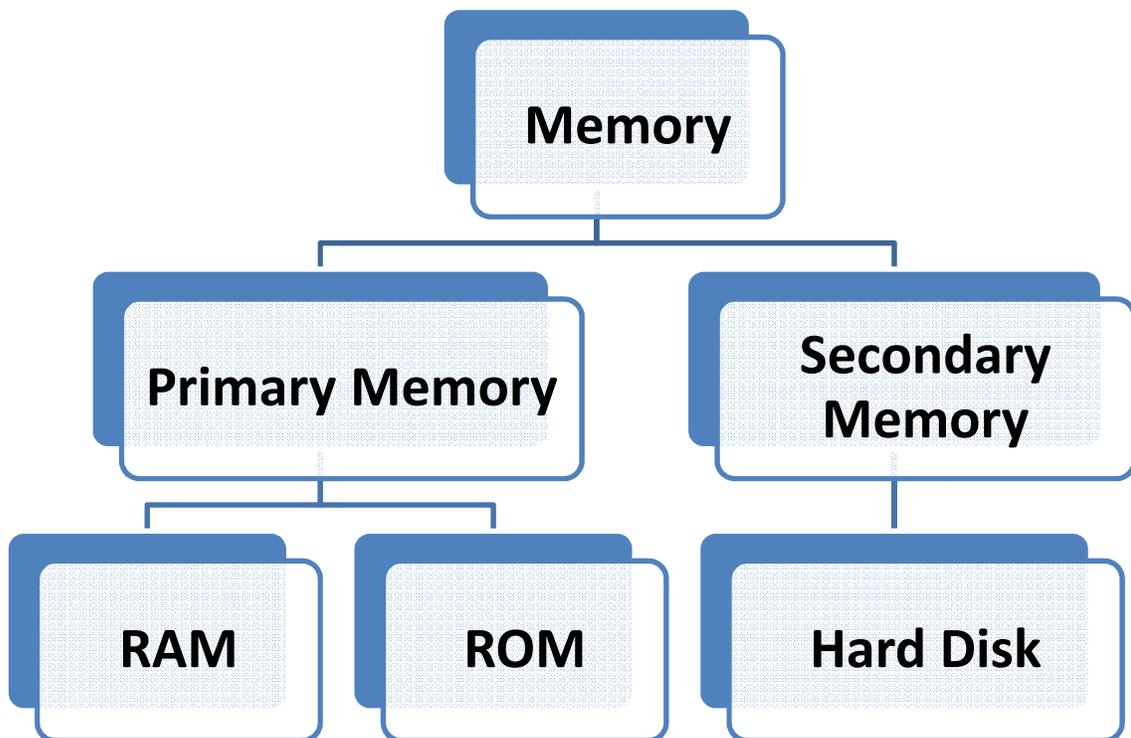


Fig: Block Diagram of a Computer

Types of Memory:



Primary memory: Primary memory is computer memory that is accessed directly by the CPU. This includes several types of memory, such as the processor cache, Random Access Memory (RAM) and Read Only Memory (ROM).

Random-access memory (RAM): It is a form of computer memory that can be read and changed in any order, typically used to store working data and machine code. This type of memory is volatile in nature. Data and information cannot be stored permanently here. As it is volatile when power supply is stopped, all data and information get erased automatically.

Read Only Memory (ROM): Read-only memory is a type of non-volatile memory used in computers and other electronic devices. Computer can only read this type of memory and cannot change the stored information. Read-only memory is useful for storing software that is rarely changed during the life of the system, also known as firmware.

Secondary memory: Secondary memory refers to the external storage device which can be used to store data or information permanently. Example Hard Disk, floppy disks, magnetic tape, paper tape, punched cards etc.

In this course we will actually learn how to give instructions to a computer system. We have already discussed about the fundamentals of a computer system. We already know computer works according to sets of instructions provided by user. The process of writing instruction is known as programming. The language in which computer instructions are written is known as programming language and a set of instructions written in programming language is known as a program.

Types of programming language:

1. Machine Level Language: We also learned computer works only on binary or digital signal. As computer is an electronic device, here data and information are represented as voltage or current. If high voltage represents 1 and low voltage represents 0 we call it positive logic and if high voltage represents 0 and low voltage represents 1 we call it negative logic. To instruct the computer if we write the instructions in the binary format i.e. 0 and 1 we called this programming language as *machine level language*. But these types of instructions are very much difficult for human to understand and remember. At the time, these instructions were machine dependent.
2. Assembly Level Language: Instead of writing whole the instruction in machine code or binary format, in *assembly level language* operators are written as mnemonics and operands are written in binary. Assembler is used to convert the *assembly level language* to *machine level language*.
3. High Level Language: *High level languages* can be easily converted to *machine level language* as well these languages can be easily understood by human also. High-level languages are like English-like language, with fewer words also known as keywords and

fewer ambiguities. Each high level language will have its own syntax and keywords. The meaning of the word syntax is grammar.

There are two types of translators for high level language programs to convert to *machine level language*. They are **interpreter** and **compiler**.

Examples of high level language are C, C++, Java, Pascal, Fortran etc.

Before we write the actual program we try to solve the problem, as a design is made before the actual building is built. In computing the technique which is used to solve a problem before writing the actual program is known as Algorithm.

Algorithm: Algorithm is a step by step method to solve a particular problem. It must be precise, concise and unambiguous.

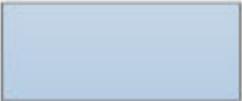
Algorithm can be compared with a recipe. The way a recipe explains how a particular dish can be prepared; algorithm does the same to solve a particular problem.

Documentation of an algorithm: An algorithm is documented in any of the following method.

- Natural Language: We can write *algorithm* using natural language like English. Here, steps are mentioned one by one in listed format.
- Programming Language: An *algorithm* can be written as a program
- Pseudo Code: In pseudo code, *algorithms* are written as a combination of both programming and natural language. In practice, *algorithms* are mainly written in pseudo code.
- Flowchart: Flowchart is the pictorial or diagrammatic representation of an *algorithm*.

Flowchart: Flowchart is the pictorial or diagrammatic representation of an *algorithm*. The flowchart shows the steps as boxes of various kinds, and their order by connecting the boxes with arrows. This diagrammatic representation illustrates a solution model to a given problem. Flowcharts are used in analyzing, designing, documenting or managing a process or program in various fields.

Symbols of Flowcharts:

Symbol	Name	Function
	Start/end	An oval represents a start or end point
	Arrows	A line is a connector that shows relationships between the representative shapes
	Input/Output	A parallelogram represents input or output
	Process	A rectangle represents a process
	Decision	A diamond indicates a decision

Q. Write the algorithm to find the largest of N numbers.

Ans:

Algorithm to find the largest of N numbers

1. Take the list of N numbers
2. Assign the first number of the list as the largest i.e. largest=list \rightarrow first
3. Compare the largest with the next number of list; continue the process till the end of the list.
4. If the next number of list is larger than the largest, assign this number as the largest.
5. Display the largest as the largest number of N numbers.

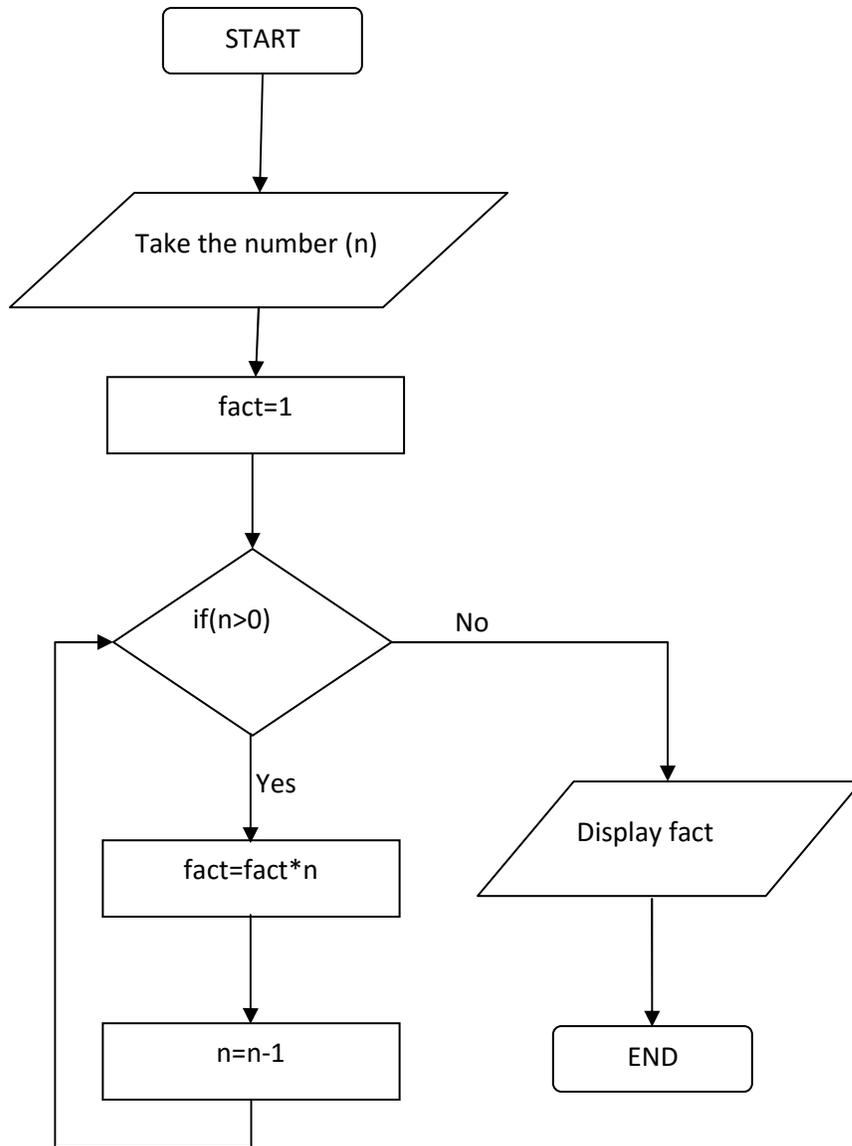
Q. Write the algorithm to calculate the simple interest.

Ans:

1. Start
2. Read Principal Amount, Rate and Time
3. Calculate Interest using formula $SI = ((\text{amount} * \text{rate} * \text{time}) / 100)$
4. Print Simple Interest
5. Stop

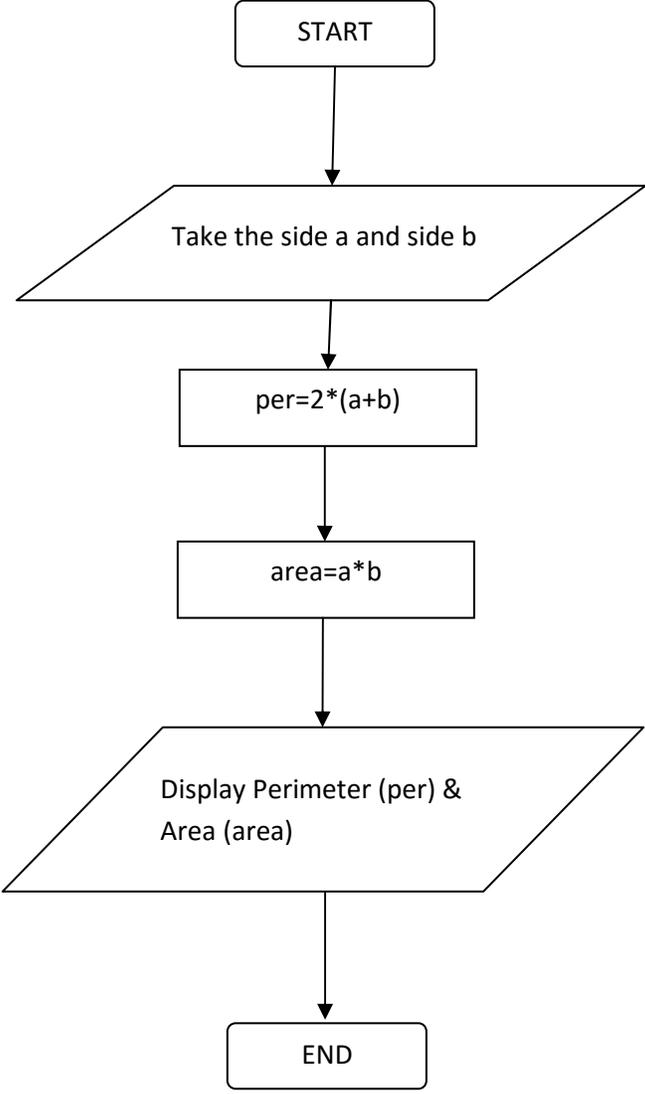
Q. Draw the flow chart to find the factorial of a number.

Ans:



Q. Draw a flowchart to find the perimeter and area of a rectangle.

Ans:



C-programming Language:

C is general purpose high level programming language used to write instructions to perform tasks in a computer system. A set of instructions written any programming language is known a program or source code. A collection of programs with their documentation is known as software. Basically, C language like any other programming language is used to write the codes of software. As we have already studied, a compiler is needed to covert C language to machine level language so that it can be executed by a computer system. For this purpose, there are lots of C- compilers like Turbo C++, Code Block, GCC C etc.

```
/*program to display "Hello C" */
#include<stdio.h>
void main()
{
    printf("Hello C");
}
```

This is an example of a C program to display the message "Hello C" in the computer screen. Before writing a program we must learn the basic concepts of C language, which are discussed below:

C Token: C tokens are the basic buildings blocks in C language which are constructed together to write a C program. Each and every smallest individual unit in a C program is known as C token. C tokens are of six types. They are,

1. **Keywords:** Keywords are pre-defined words in a C compiler. Each keyword is meant to perform a specific function in a C program. There are 32 keywords in C language. e.g.: int, while etc. Here, in the above program **void** is a keyword.
2. **Identifiers:** Identifier refers to a name given to entities such as variables, functions, structures etc. Identifiers must be unique. They are created to give a unique name to an entity to identify it, if it does not belong to any other tokens like keywords, constants etc. e.g.: main, printf in the above program.
3. **Constants:** Constants refer to fixed values. Like 10, 20, 30.5, 'A' etc. that are used in program.

Constant type	data type (Example)
Integer constants	int (53, 762, -478 etc) unsigned int (5000u, 1000U etc) long int, long long int (483,647 2,147,483,680)
Real or Floating point constants	float (10.456789) doule (600.123456789)
Octal constant	int (Example: 013 /*starts with 0 */)
Hexadecimal constant	int (Example: 0x90 /*starts with 0x*/)

character constants	char (Example: 'A', 'B', 'C')
string constants	char (Example: "ABCD", "Hai")

4. **Strings:** Strings are defined as an array of characters. e.g.: "Hello C" in the above program.
5. **Special symbols:** Special symbols have some special meaning in the C programming language for the compiler and it is used to perform some special function/task in C. e.g.: (), {}, etc.
6. **Operators:** An operator is a symbol that tells the compiler to perform specific mathematical or logical functions. e.g.: +, /, -, * etc.

Data types: Data types specify how we enter data into our programs and what type of data we enter. C language has some predefined set of data types to handle various kinds of data that we can use in our program. These data-types have different storage capacities. C language supports two different types of data types:

1. **Primary data type:** These are fundamental data types in C. Any data used in a program belongs to any of these data-types. These are integer (int) for representing Integer number, floating point (float) for decimal numbers, character (char) for alphabets and void for null.
2. **Derived data types:** Derived data types are derived from primary data-types. Actually, primary data-types are grouped together to form derived data-type. Like array, stucture, union and pointer.

Variable: C variable is a location name in a memory where a program can store, retrieve and manipulate the data. This location is used to hold the value of the variable. The value of the C variable may get changed in the program. C variable can belong to any of the data type like **int, float, char** etc.

Rules for Naming C Variable:

1. Variable name must begin with letter or underscore.
2. Variables are case sensitive
3. They can be constructed with digits, letters.
4. No special symbols are allowed other than underscore.

Example of variable name: a, x, y10, sum, height, _value etc.

Variable declaration Syntax (Syntax is the general structure of statements in a computer language.)

data_type variable_name; Example: int x, y, z; char flat, ch;

Important Operators:

Operator	Description	Example
+	Adds two operands.	$A = 3+5$
-	Subtracts second operand from the first.	$A = X-Y$
*	Multiplies both operands.	$A = B*200$
/	Divides numerator by de-numerator.	$B = A / 2$
%	Modulus Operator gives the remainder after an integer division.	$B = A \% 0$
++	Increment operator increases the integer value by one.	$A++$ Means $A=A+1$
--	Decrement operator decreases the integer value by one.	$A--$ Means $A=A-1$
=	Simple assignment operator. Assigns values from right side operands to left side operand	$C = A + B$ will assign the value of $A + B$ to C
==	Checks if the values of two operands are equal or not. If yes, then the condition becomes true.	$(A == B)$ is not true.
>=	Checks if the value of left operand is greater than or equal to the value of right operand. If yes, then the condition becomes true.	$(A >= B)$ is not true.
See the other operations from books or internet.		

Input/output Command: Input means to provide the program with some data to be used in the program and Output means to display data on screen or write the data to a printer or a file.

Input function: scanf() is used to take the user data. Syntax of scanf() is:

```
scanf("<format specifiers>" , <address of variable>);
```

Example: `scanf("%d%f" , &a,&x);` where a and x are variables.

Output function: printf() is used to display the output. Syntax of printf() is:

```
printf("<string with format specifiers>" ,<variable_names>)
```

Example: `scanf("The result is %f" , &r);` where r is a variable.

Format strings inform the `scanf()` function, the type of input to be expected and in `printf()` it is used to tell the compiler the type of output to print/display.

Format String	Meaning
<code>%d</code>	To scan or print an integer number
<code>%f</code>	To scan or print a decimal number
<code>%c</code>	To scan or print a character
<code>%s</code>	To scan or print a character string. The scanning ends at whitespace.

Similarly, the `getchar()` function reads a character from the terminal and returns it as an integer. This function reads only single character at a time. The `putchar()` function displays the character passed to it on the screen and returns the same character. This function too displays only a single character at a time. The `gets()` function reads a line from stdin(standard input). The `puts()` function is used to provide output.

Conditional Statements: In C language, if, else, switch are known as conditional statements. Conditional statements are used to make a decision based on certain conditions. These conditions are specified by a set of conditional statements having Boolean expressions which are evaluated to a Boolean value true or false.

if-else: The if-else statement in C language is used to execute the code if condition is true or false. It is also called two-way selection statement.

Syntax:

```
if (expression)
{
    //Statements
}
else
{
    //Statements
}
```

Example: A C program to find odd or even number using if-else.

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int num=0;
    printf("enter the number");
    scanf("%d",&num);
    if(n%2==0)
    {
        printf("%d number in even", num);
    }
    else
    {
        printf("%d number in odd", num);
    }
}
```

Switch statement: Switch statement acts as a substitute for a long if-else-if ladder that is used to test a list of cases. A switch statement contains one or more case labels which are tested against the switch expression. If the expression matches to a case, then the associated statements with that case would be executed.

Syntax:

```
switch (expression)
{
    case value1:
        //Statements
        break;
    case value 2:
        //Statements
        break;
    case value 3:
        //Statements
        Break;
    case value n:
        //Statements
        break;
    Default:
        //Statements
}
```

Loop: In programming, a loop is used to repeat a block of code until the specified condition is met. C programming has three types of loops:

1. for loop
2. while loop
3. do...while loop

1. for loop: The syntax of the for loop is:

```
for (initializationStatement; testExpression; updateStatement)
{
    // statements inside the body of loop
}
```

```
/* Program to calculate the sum of first n natural numbers*/
// Positive integers 1, 2, 3...n are known as natural numbers
#include <stdio.h>
int main()
{
    int num, count, sum = 0;
    printf("Enter a positive integer: ");
    scanf("%d", &num);
    // for loop terminates when num is less than count
    for(count = 1; count <= num; ++count)
    {
        sum += count;
    }
    printf("Sum = %d", sum);
    return 0;
}
```

2. while loop: The syntax of the while loop is:

```
while (testExpression)
{
    // statements inside the body of the loop
}
```

3. do...while loop: The do..while loop is similar to the while loop with one important difference. The body of do...while loop is executed at least once. Only then, the test expression is evaluated. The syntax of the do...while loop is:

```
do
{
    // statements inside the body of the loop
}
while (testExpression);
```

C program to find the reverse of a number.

```
#include <stdio.h>
int main() {
    int n, rev = 0, remainder;
    printf("Enter an integer: ");
    scanf("%d", &n);
    while (n != 0) {
        remainder = n % 10;
        rev = rev * 10 + remainder;
        n /= 10;
    }
    printf("Reversed number = %d", rev);
    return 0;
}
```

Array: An array is defined as the collection of similar type of data items stored at contiguous memory locations. Arrays are the derived data type in C programming language which can store the primitive type of data such as int, char, double, float, etc. The Syntax of Array declaration:

data_type array_name[array_size];

Example: int marks [5];

```
/* Program to print the largest and second largest element of the
array. */
#include<stdio.h>
void main ()
{
    int arr[100],i,n,largest,sec_largest;
    printf("Enter the size of the array?");
    scanf("%d",&n);
    printf("Enter the elements of the array?");
    for(i = 0; i<n; i++)
    {
        scanf("%d",&arr[i]);
    }
    largest = arr[0];
    sec_largest = arr[1];
    for(i=0;i<n;i++)
    {
        if(arr[i]>largest)
        {
            sec_largest = largest;
            largest = arr[i];
        }
        else if (arr[i]>sec_largest && arr[i]!=largest)
        {
```

```

        sec_largest=arr[i];
    }
}
printf("largest = %d, second largest = %d",largest,sec_largest);
}

```

C Function: In c, a large program can be divided into the basic building blocks known as function. The function contains the set of programming statements enclosed by {}. A function can be called multiple times to provide reusability and modularity to the C program. In other words, a program is created as the collection of functions. The function is also known as procedure or subroutine in other programming languages.

- Function declaration: A function must be declared globally in a c program to tell the compiler about the function name, function parameters, and return type.
- Function call: Function can be called from anywhere in the program. The parameter list must not differ in function calling and function declaration. We must pass the same number of functions as it is declared in the function declaration.
- Function definition: It contains the actual statements which are to be executed. It is the most important aspect to which the control comes when the function is called. Here, we must notice that only one value can be returned from the function.

C function aspects	Syntax
Function declaration	return_type function_name(argument list);
Function call	function_name(argument_list)
Function definition	return_type function_name(argument list) {function body;}

The syntax of creating function in c language is given below:

```

return_type function_name(data_type parameter...)
{
    //code to be executed
}

```

There are two types of functions in C programming:

Library Functions: Library functions are the functions which are declared in the C header files such as scanf(), printf(), gets(), puts(), ceil(), floor() etc.

User-defined functions: User-defined functions are the functions which are created by the C programmer, so that he/she can use it many times. It reduces the complexity of a big program and optimizes the code.

```
/* Program to calculate the area of the square*/
#include<stdio.h>
int square();
void main()
{
    printf("Going to calculate the area of the square\n");
    float area = square();
    printf("The area of the square: %f\n",area);
}
int square()
{
    float side;
    printf("Enter the length of the side in meters: ");
    scanf("%f",&side);
    return side * side;
}
```