

CHAPTER NO.	CHAPTER NAME	PAGE NO.
1	FUNCTIONS IN PYTHON	2
2	RECURSION	10
3	FILE HANDLING	13
4	CREATE & IMPORT PYTHON LIBRARIES	24
5	PROGRAM EFFICIENCY	28
6	DATA VISUALIZATION USING PYPLOT	32
7	DATA STRUCTURE –I (LINEAR LIST)	45
8	DATA STRUCTURE-II (STACK AND QUEUE)	55

By: Vikash Kumar Yadav

PGT-Computer Science

K.V.No.-IV ONGC Vadodara

## CHAPTER-1

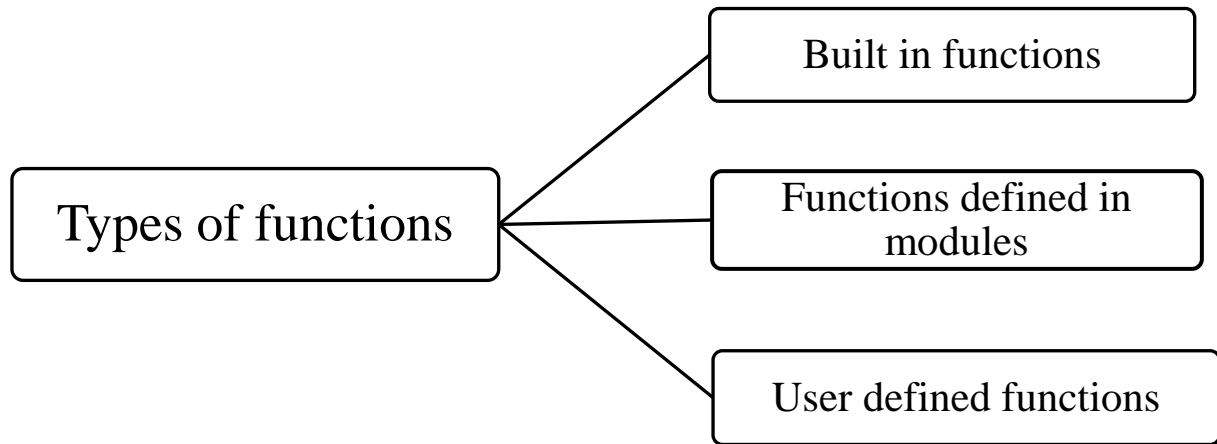
### FUNCTIONS IN PYTHON

**1.1 Definition:** Functions are the subprograms that perform specific task. Functions are the small modules.

#### 1.2 Types of Functions:

There are **two** types of functions in python:

1. Library Functions (Built in functions)
2. Functions defined in modules
3. User Defined Functions



**1. Library Functions:** These functions are already built in the python library.

**2. Functions defined in modules:** These functions defined in particular modules. When you want to use these functions in program, you have to import the corresponding module of that function.

**3. User Defined Functions:** The functions those are defined by the user are called user defined functions.

#### 1. Library Functions in Python:

These functions are already built in the library of python.

For example: `type( )`, `len( )`, `input( )` etc.

## 2. Functions defined in modules:

### a. Functions of math module:

To work with the functions of math module, we must import math module in program.

**import math**

S. No.	Function	Description	Example
1	sqrt( )	Returns the square root of a number	>>>math.sqrt(49) 7.0
2	ceil( )	Returns the upper integer	>>>math.ceil(81.3) 82
3	floor( )	Returns the lower integer	>>>math.floor(81.3) 81
4	pow( )	Calculate the power of a number	>>>math.pow(2,3) 8.0
5	fabs( )	Returns the absolute value of a number	>>>math.fabs(-5.6) 5.6
6	exp( )	Returns the e raised to the power i.e. $e^3$	>>>math.exp(3) 20.085536923187668

### b. Function in random module:

random module has a function randint( ).

- randint( ) function generates the random integer values including start and end values.
- Syntax: randint(start, end)
- It has two parameters. Both parameters must have integer values.

Example:

```
import random  
n=random.randint(3,7)
```

\*The value of n will be 3 to 7.

## 3. USER DEFINED FUNCTIONS:

The syntax to define a function is:

```
def function-name ( parameters) :  
    #statement(s)
```

Where:

- Keyword **def** marks the start of function header.
- A function name to uniquely identify it. Function naming follows the same rules of writing identifiers in Python.
- Parameters (arguments) through which we pass values to a function. They are optional.
- A colon (:) to mark the end of function header.
- One or more valid python statements that make up the function body. Statements must have same indentation level.
- An optional return statement to return a value from the function.

### Example:

```
def display(name):  
    print("Hello " + name + " How are you?")
```

### 1.3 Function Parameters:

A functions has two types of parameters:

1. **Formal Parameter:** Formal parameters are written in the function prototype and function header of the definition. Formal parameters are local variables which are assigned values from the arguments when the function is called.
2. **Actual Parameter:** When a function is *called*, the values that are passed in the call are called *actual parameters*. At the time of the call each actual parameter is assigned to the corresponding formal parameter in the function definition.

### Example :

```
def ADD(x, y):                #Defining a function and x and y are formal parameters  
    z=x+y  
    print("Sum = ", z)  
a=float(input("Enter first number: " ))  
b=float(input("Enter second number: " ))  
ADD(a,b)    #Calling the function by passing actual parameters
```

In the above example, **x** and **y** are formal parameters. **a** and **b** are actual parameters.

## 1.4 Calling the function:

Once we have defined a function, we can call it from another function, program or even the Python prompt. To call a function we simply type the function name with appropriate parameters.

### Syntax:

function-name(parameter)

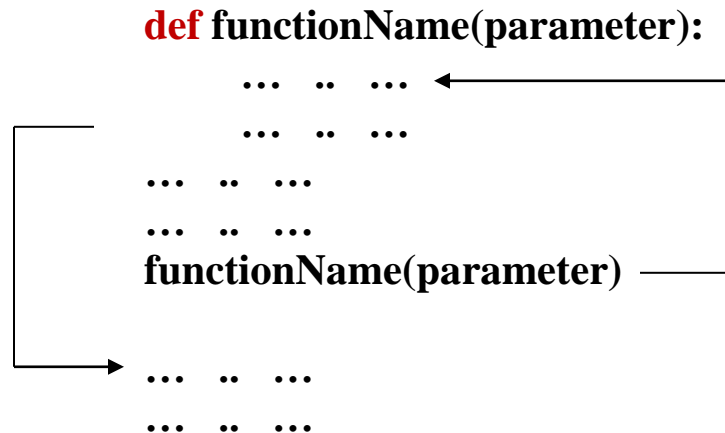
### Example:

ADD(10,20)

### OUTPUT:

Sum = 30.0

### How function works?



### The return statement:

The **return** statement is used to exit a function and go back to the place from where it was called.

There are two types of functions according to return statement:

- Function returning some value (non-void function)
- Function not returning any value (void function)

**a. Function returning some value (non-void function) :**

**Syntax:**

```
return expression/value
```

**Example-1: Function returning one value**

```
def my_function(x):  
    return 5 * x
```

**Example-2 Function returning multiple values:**

```
def sum(a,b,c):  
    return a+5, b+4, c+7  
  
S=sum(2,3,4)      # S will store the returned values as a tuple  
print(S)
```

**OUTPUT:**

```
(7, 7, 11)
```

**Example-3: Storing the returned values separately:**

```
def sum(a,b,c):  
    return a+5, b+4, c+7  
  
s1, s2, s3=sum(2, 3, 4)      # storing the values separately  
print(s1, s2, s3)
```

**OUTPUT:**

```
7 7 11
```

**b. Function not returning any value (void function) :** The function that performs some operations but does not return any value, called void function.

```
def message():  
    print("Hello")  
  
m=message()  
print(m)
```

## OUTPUT:

Hello

None

### 1.5 Scope and Lifetime of variables:

Scope of a variable is the portion of a program where the variable is recognized. Parameters and variables defined inside a function is not visible from outside. Hence, they have a local scope.

There are two types of scope for variables:

1. Local Scope

2. Global Scope

1. **Local Scope:** Variable used inside the function. It can not be accessed outside the function. In this scope, The lifetime of variables inside a function is as long as the function executes. They are destroyed once we return from the function. Hence, a function does not remember the value of a variable from its previous calls.

2. **Global Scope:** Variable can be accessed outside the function. In this scope, Lifetime of a variable is the period throughout which the variable exists in the memory.

#### Example:

```
def my_func():
    x = 10
    print("Value inside function:",x)

x = 20
my_func()
print("Value outside function:",x)
```

## OUTPUT:

Value inside function: 10

Value outside function: 20

Here, we can see that the value of **x** is **20** initially. Even though the function `my_func()` changed the value of **x** to **10**, it did not affect the value outside the function.

This is because the variable **x** inside the function is different (local to the function) from the one outside. Although they have same names, they are two different variables with different scope.

On the other hand, variables outside of the function are visible from inside. They have a **global** scope.

We can read these values from inside the function but cannot change (write) them. In order to modify the value of variables outside the function, they must be declared as global variables using the keyword **global**.

### Programs related to Functions in Python topic:

1. Write a python program to sum the sequence given below. Take the input **n** from the user.

$$1 + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \dots + \frac{1}{n!}$$

#### Solution:

```
def fact(x):
    j=1
    res=1
    while j<=x:
        res=res*j
        j=j+1
    return res
n=int(input("enter the number : "))
i=1
sum=1
while i<=n:
    f=fact(i)
    sum=sum+1/f
    i+=1
print(sum)
```

2. Write a program to compute GCD and LCM of two numbers

```
def gcd(x,y):
    while(y):
        x, y = y, x % y
```



```
    return x

def lcm(x, y):
    lcm = (x*y)//gcd(x,y)
    return lcm

num1 = int(input("Enter first number: "))
num2 = int(input("Enter second number: "))

print("The L.C.M. of", num1,"and", num2,"is", lcm(num1, num2))
print("The G.C.D. of", num1,"and", num2,"is", gcd(num1, num2))
```

## CHAPTER-2

### RECURSION

**2.1 Definition:** A function calls itself, is called recursion.

**2.2 Python program to find the factorial of a number using recursion:**

**Program:**

```
def factorial(n):
    if n == 1:
        return n
    else:
        return n*factorial(n-1)

num=int(input("enter the number: "))
if num < 0:
    print("Sorry, factorial does not exist for negative numbers")
elif num == 0:
    print("The factorial of 0 is 1")
else:
    print("The factorial of ",num," is ", factorial(num))
```

**OUTPUT:**

enter the number: 5

The factorial of 5 is 120

**2.3 Python program to print the Fibonacci series using recursion:**

**Program:**

```
def fibonacci(n):
    if n<=1:
        return n
    else:
```

```
return(fibonacci(n-1)+fibonacci(n-2))
```

```
num=int(input("How many terms you want to display: "))
```

```
for i in range(num):
```

```
    print(fibonacci(i)," ", end=" ")
```

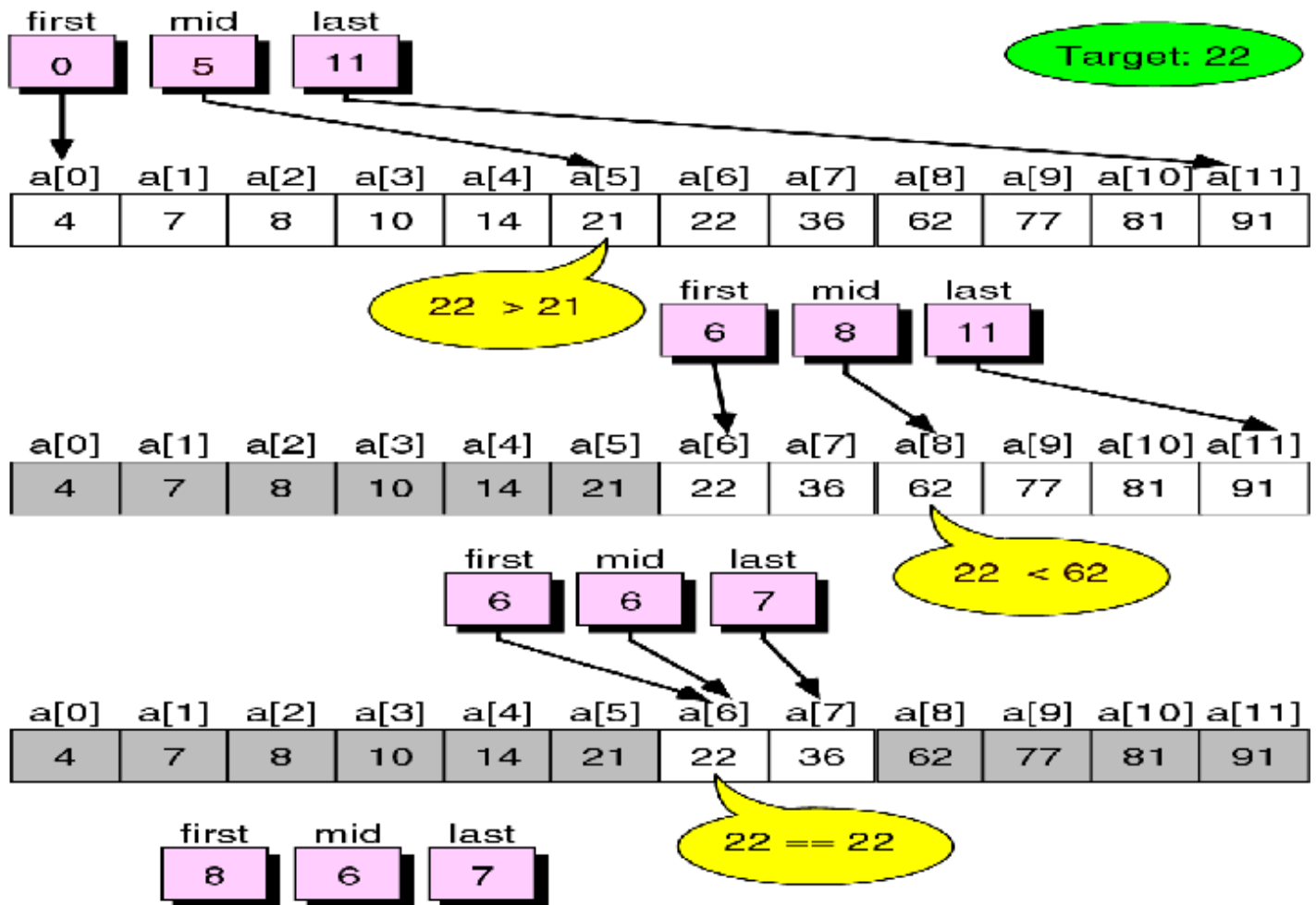
## OUTPUT:

How many terms you want to display: 8

0 1 1 2 3 5 8 13

## 2.4 Binary Search using recursion:

**Note:** The given array or sequence must be sorted to perform binary search.



Function terminates

## Program:

```
def Binary_Search(sequence, item, LB, UB):
    if LB>UB:
        return -5          # return any negative value
    mid=int((LB+UB)/2)
    if item==sequence[mid]:
        return mid
    elif item<sequence[mid]:
        UB=mid-1
        return Binary_Search(sequence, item, LB, UB)
    else:
        LB=mid+1
        return Binary_Search(sequence, item, LB, UB)

L=eval(input("Enter the elements in sorted order: "))
n=len(L)
element=int(input("Enter the element that you want to search :"))
found=Binary_Search(L,element,0,n-1)
if found>=0:
    print(element, "Found at the index : ",found)
else:
    print("Element not present in the list")
```

## CHAPTER-3

### FILE HANDLING

#### 3.1 INTRODUCTION:

**File:-** A file is a collection of related data stored in a particular area on the disk.

**Stream:** - It refers to a sequence of bytes.

File handling is an important part of any web application.

#### 3.2 Data Files:

Data files can be stored in two ways:

1. **Text Files:** Text files are structured as a sequence of lines, where each line includes a sequence of characters.
2. **Binary Files :** A binary file is any type of file that is not a text file.

S. No.	Text Files	Binary Files
1.	Stores information in ASCII characters.	Stores information in the same format which the information is held in memory.
2.	Each line of text is terminated with a special character known as EOL (End of Line)	No delimiters are used for a line.
3.	Some internal translations take place when this EOL character is read or written.	No translation occurs in binary files.
4.	Slower than binary files.	Binary files are faster and easier for a program to read and write the text files.

#### 3.3 Opening and closing a file:

##### 3.3.1 Opening a file:

To work with a file, first of all you have to open the file. To open a file in python, we use `open( )` function.

The `open( )` function takes two parameters; *filename*, and *mode*. `open( )` function returns a file object.

**Syntax:**

```
file_objectname= open(filename, mode)
```

**Example:**

To open a file for reading it is enough to specify the name of the file:

```
f = open("book.txt")
```

The code above is the same as:

```
f = open("book.txt", "rt")
```

Where "r" for read mode, and "t" for text are the default values, you do not need to specify them.

### 3.3.2 Closing a file:

After performing the operations, the file has to be closed. For this, a `close( )` function is used to close a file.

**Syntax:**

```
file-objectname.close( )
```

### 3.4 File Modes:

Text file mode	Binary File Mode	Description
'r'	'rb'	Read - Default value. Opens a file for reading, error if the file does not exist.
'w'	'wb'	Write - Opens a file for writing, creates the file if it does not exist
'a'	'ab'	Append - Opens a file for appending, creates the file if it does not exist
'r+'	'rb+'	Read and Write-File must exist, otherwise error is raised.
'w+'	'wb+'	Read and Write-File is created if does not exist.
'a+'	'ab+'	Read and write-Append new data
'x'	'xb'	Create - Creates the specified file, returns an error if the file exists

In addition you can specify if the file should be handled as binary or text mode

"t" – Text-Default value. Text mode

"b" – Binary- Binary Mode (e.g. images)

### 3.5 WORKING WITH TEXT FILES:

#### 3.5.1 Basic operations with files:

- a. **Read** the data from a file
- b. **Write** the data to a file
- c. **Append** the data to a file
- d. **Delete** a file

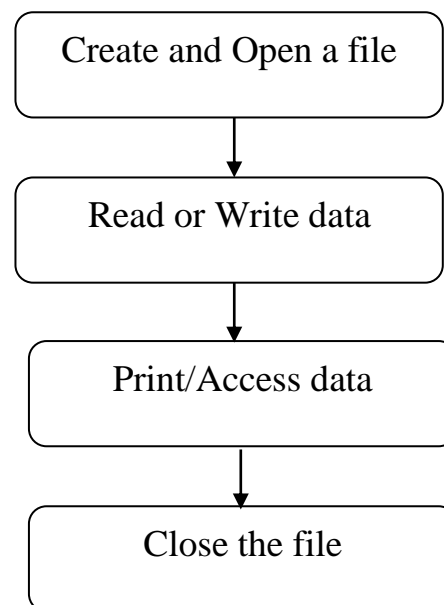
#### a. Read the data from a file:

There are **3 types** of functions to read data from a file.

- **read( )** : reads n bytes. if no n is specified, reads the entire file.
- **readline( )** : Reads a line. if n is specified, reads n bytes.
- **readlines( )**: Reads all lines and returns a list.

#### Steps to read data from a file:

- Create and Open a file using open( ) function
- use the read( ), readline( ) or readlines( ) function
- print/access the data
- Close the file.



Let a text file “**Book.txt**” has the following text:

*“Python is interactive language. It is case sensitive language.  
It makes the difference between uppercase and lowercase letters.  
It is official language of google.”*

**Example-1: Program using read( ) function:**

<pre>fin=open("D:\\python programs\\Book.txt",'r') str=fin.read( ) print(str) fin.close( )</pre>	<pre>fin=open("D:\\python programs\\Book.txt",'r') str=fin.read(10) print(str) fin.close( )</pre>
<p><b>OUTPUT:</b> Python is interactive language. It is case sensitive language. It makes the difference between uppercase and lowercase letters. It is official language of google.</p>	<p><b>OUTPUT:</b> Python is</p>

**Example-2: using readline( ) function:**

```
fin=open("D:\\python programs\\Book.txt",'r')
str=fin.readline( )
print(str)
fin.close( )
```

**OUTPUT:**

Python is interactive language. It is case sensitive language.

**Example-3: using readlines( ) function:**

```
fin=open("D:\\python programs\\Book.txt",'r')
str=fin.readlines( )
print(str)
fin.close( )
```

**OUTPUT:**

['Python is interactive language. It is case sensitive language.\n', 'It makes the difference between uppercase and lowercase letters.\n', 'It is official language of google.']



❖ **Some important programs related to read data from text files:**

**Program-a: Count the number of characters from a file. (Don't count white spaces)**

```
fin=open("Book.txt",'r')
str=fin.read()
L=str.split()
count_char=0
for i in L:
    count_char=count_char+len(i)
print(count_char)
fin.close()
```

**Program-b: Count the number of words in a file.**

```
fin=open("Book.txt",'r')
str=fin.read()
L=str.split()
count_words=0
for i in L:
    count_words=count_words+1
print(count_words)
fin.close()
```

**Program-c: Count number of lines in a text file.**

```
fin=open("Book.txt",'r')
str=fin.readlines()
count_line=0
for i in str:
    count_line=count_line+1
print(count_line)
fin.close()
```

**Program-d: Count number of vowels in a text file.**

```
fin=open("D:\\python programs\\Book.txt",'r')
str=fin.read()
```

```

count=0
for i in str:
    if i=='a' or i=='e' or i=='i' or i=='o' or i=='u':
        count=count+1
print(count)
fin.close( )

```

**Program-e : Count the number of 'is' word in a text file.**

```

fin=open("D:\\python programs\\Book.txt",'r')

str=fin.read( )

L=str.split( )

count=0

for i in L:

    if i=='is':

        count=count+1

print(count)

fin.close( )

```

**b. Write data to a file:**

There are **2 types** of functions to write the data to a file.

- **write( ):** Write the data to a file. Syntax:

write(string)                    (for text files)

write(byte\_string)            (for binary files)

- **writelines( ):** Write all strings in a list L as lines to file.

To write the data to an existing file, you have to use the following mode:

**"w"** - Write - will overwrite any existing content

**Example: Open the file "Book.txt" and append content to the file:**

```
fout= open("Book.txt", "a")  
fout.write("Welcome to the world of programmers")
```

**Example: Open the file "Book.txt" and overwrite the content:**

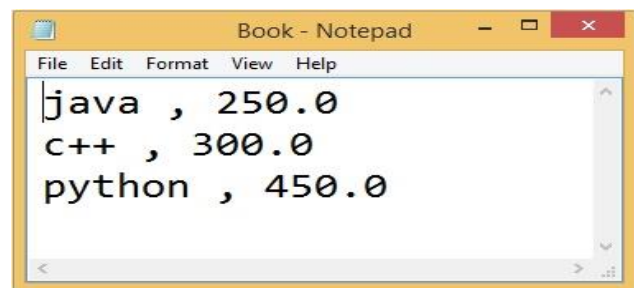
```
fout = open("Book.txt", "w")  
fout.write("It is creative and innovative")
```

**Program: Write a program to take the details of book from the user and write the record in text file.**

```
fout=open("D:\\python programs\\Book.txt",'w')  
n=int(input("How many records you want to write in a file ? :"))  
for i in range(n):  
    print("Enter details of record :", i+1)  
    title=input("Enter the title of book : ")  
    price=float(input("Enter price of the book: "))  
    record=title+" , "+str(price)+"\n"  
    fout.write(record)  
fout.close( )
```

**OUTPUT:**

```
How many records you want to write in a file ? :3  
Enter details of record : 1  
Enter the title of book : java  
Enter price of the book: 250  
Enter details of record : 2  
Enter the title of book : c++  
Enter price of the book: 300  
Enter details of record : 3  
Enter the title of book : python  
Enter price of the book: 450
```



**c. Append the data to a file:**

This operation is used to add the data in the end of the file. It doesn't overwrite the existing data in a file. To write the data in the end of the file, you have to use the following mode:

**"a"** - Append - will append to the end of the file.

**Program: Write a program to take the details of book from the user and write the record in the end of the text file.**

```
fout=open("D:\\python programs\\Book.txt",'a')
n=int(input("How many records you want to write in a file ? :"))
for i in range(n):
    print("Enter details of record :", i+1)
    title=input("Enter the title of book : ")
    price=float(input("Enter price of the book: "))
    record=title+" , "+str(price)+'\n'
    fout.write(record)
fout.close( )
```

### OUTPUT:

How many records you want to write in a file ? :2

Enter details of record : 1

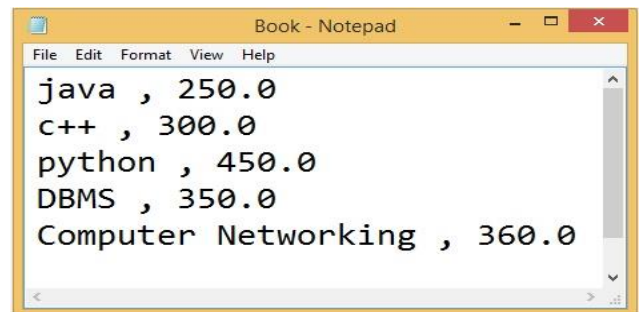
Enter the title of book : DBMS

Enter price of the book: 350

Enter details of record : 2

Enter the title of book : Computer  
Networking

Enter price of the book: 360



**d. Delete a file:** To delete a file, you have to import the **os** module, and use **remove( )** function.

```
import os
os.remove("Book.txt")
```

### Check if File exist:

To avoid getting an error, you might want to check if the file exists before you try to delete it:

Example

Check if file exists, *then* delete it:

```
import os
if os.path.exists("Book.txt"):
    os.remove("Book.txt")
else:
    print("The file does not exist")
```

### 3.6 WORKING WITH BINARY FILES:

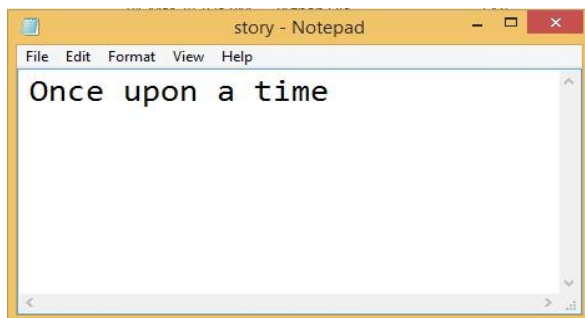
Binary files store data in the binary format (0's and 1's) which is understandable by the machine. So when we open the binary file in our machine, it decodes the data and displays in a human-readable format.

#### (a) Write data to a Binary File:

##### Example:

```
fout=open("story.dat","wb")           # open file in write and binary mode
Norml_str= "Once upon a time"
encoded_str= Norml_str.encode ("ASCII") # encoding normal text
fout.write(encoded_str)                # Write data in file
fout.close( )
```

##### Output:



#### (b) Read the data from Binary File:

##### Example:

```
fin=open("story.dat","rb")           # open file in read and binary mode
str1= fin.read()                      # read the data from file
decoded_str=str1.decode("ASCII")      # decode the binary data
print(decoded_str)
fin.close( )
```

## Output:

Once upon a time

### 3.7 tell() and seek() methods:

- **tell() :** It returns the current position of cursor in file.

#### *Example:*

```
fout=open("story.txt","w")
fout.write("Welcome Python")
print(fout.tell( ))
fout.close( )
```

## Output:

14

- **seek(offset) :** Change the cursor position by bytes as specified by the offset.

#### **Example:**

```
fout=open("story.txt","w")
fout.write("Welcome Python")
fout.seek(5)
print(fout.tell( ))
fout.close( )
```

## Output:

5

### 3.8 File I/O Attributes

Attribute	Description
name	Returns the name of the file (Including path)
mode	Returns mode of the file. (r or w etc.)
encoding	Returns the encoding format of the file
closed	Returns True if the file closed else returns False

***Example:***

```
f = open("D:\\story.txt", "r")
print("Name of the File: ", f.name)
print("File-Mode : ", f.mode)
print("File encoding format : ", f.encoding)
print("Is File closed? ", f.closed)
f.close()
print("Is File closed? ", f.closed)
```

**OUTPUT:**

```
Name of the File: D:\\story.txt
File-Mode : r
File encoding format : cp1252
Is File closed? False
Is File closed? True
```

## CHAPTER-4

### CREATE & IMPORT PYTHON LIBRARIES

#### 4.1 INTRODUCTION:

Python has its in-built libraries and packages. A user can directly import the libraries and its modules using *import* keyword. If a user wants to create libraries in python, he/she can create and import libraries in python.

#### 4.2 How to create Libraries/Package in Python?

To create a package/Library in Python, we need to follow these three simple steps:

1. First, we create a directory and give it a package name. Name should be meaningful.
2. Then create the python files (Modules) which have classes and functions and put these **.py** files in above created directory.
3. Finally we create an `__init__.py` file inside the directory, to let Python know that the directory is a package.

#### Example:

Let's create a package named **Shape** and build three modules in it namely **Rect**, **Sq** and **Tri** to calculate the area for the shapes rectangle, square and triangle.

**Step-1** First create a directory and name it **Shape**. (In this case it is created under `C:\Users\ViNi\AppData\Local\Programs\Python\Python37-32\` directory path.)

**Step-2** Create the modules in **Shape** directory.

To create **Module-1(Rect.py)**, a file with the name `Rect.py` and write the code in it.

```
class Rectangle:
    def __init__(self):
        print("Rectangle")

    def Area(self, length, width):
        self.l=length
        self.w=width
        print("Area of Rectangle is : ", self.l*self.w)
```



To create **Module-2(Sq.py)**, a file with the name Sq.py and write the code in it.

```
class Square:
    def __init__(self):
        print("Square")
    def Area(self, side):
        self.a=side
        print("Area of square is : ", self.a*self.a)
```

To create **Module-3 (Tri.py)**, a file with the name Tri.py and write the code in it.

```
class Triangle:
    def __init__(self):
        print("Trinagle")

    def Area(self, base, height):
        self.b=base
        self.h=height
        ar= (1/2)*self.b*self.h
        print("Area of Triangle is : ", ar )
```

**Step-3 Create the \_\_init\_\_.py file.** This file will be placed inside **Shape** directory and can be left blank or we can put this initialisation code into it.

```
from Shape import Rect
from Shape import Sq
from Shape import Tri
```

That's all. Package created. Name of the package is Shape and it has three modules namely Rect, Sq and Tri.

### 4.3 How to use the package in a program:

Following steps should be followed to use the created package.

**Step-1** Create a file main.py in the same directory where **Shape** package is located.

**Step-2** Write the following code in main.py file

```
from Shape import Rect
from Shape import Sq
from Shape import Tri

r=Rect.Rectangle() #Create an object r for Rectangle class
r.Area(10,20)      # Call the module Area() of Rectangle class by passing argument

s=Sq.Square()     #Create an object s for Square class
s.Area(10)        # Call the module Area() of Square class by passing argument

t=Tri.Triangle()  #Create an object t for Triangle class
t.Area(6,8)       # Call the module Area() of Triangle class by passing argument
```

## OUTPUT:

Rectangle

Area of Rectangle is : 200

Square

Area of square is : 100

Trinagle

Area of Triangle is : 24.0

## 4.4 Import the package in program:

### Method-1

To use the package/library in a program, we have to import the package in the program. For this we use **import** keyword.

If we simply import the Shape package, then to access the modules and functions of **Shape** package, we have to write the following code:

### Syntax:

Package-name.Module-name.function-name(parameter)

### Example:

```
import Shape
```

```
r=Shape.Rect.Rectangle( )
```

```
s=Shape.Sq.Square( )
```

```
t=Shape.Tri.Triangle( )
```

### **Method-2:**

If we want to access a specific module of a package then we can use *from* and *import* keywords.

#### **Syntax:**

```
from package-name import module-name
```

#### **Example:**

```
from Shape import Rect
```

```
r=Rect.Rectangle( )
```

```
s=Sq.Square( )
```

```
t=Tri.Triangle( )
```

### **Method-3:**

If a user wants to import all the modules of Shape package then he/she can use \* (asterisk).

#### **Example:**

```
from Shape import *
```

```
r=Rect.Rectangle( )
```

```
s=Sq.Square( )
```

```
t=Tri.Triangle( )
```

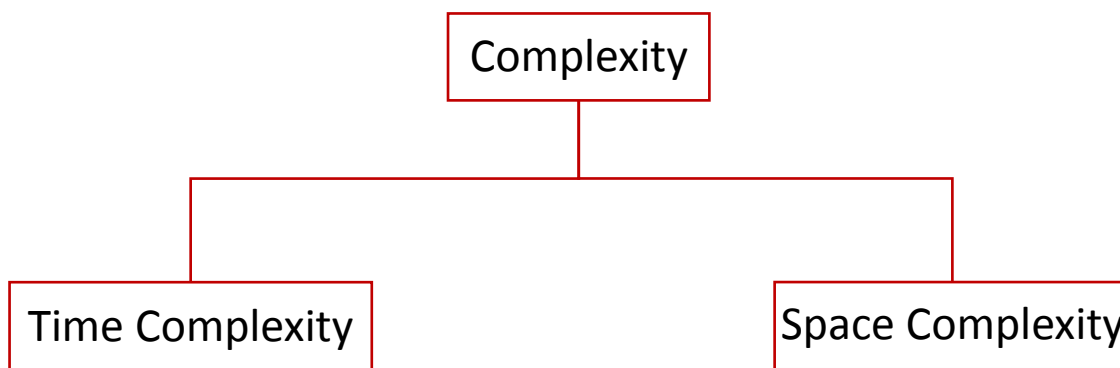
## CHAPTER-5

### PROGRAM EFFICIENCY

#### 5.1 INTRODUCTION:

Program efficiency is a property of a program related to time taken by a program for execution and space used by the program. It is also related to number of inputs taken by a program.

**Complexity:** To measure the efficiency of an algorithm or program in terms of space and time. The program which uses minimum number of resources, less space and minimum time, is known as good program.



Time and space complexity depends on lots of things like hardware, operating system, processors, size of inputs etc.

Time Complexity of an algorithm:

- Best case
  - Average case
  - Worst case
- 
- **Best case:** The minimum number of steps that an algorithm can take for any input data values.
  - **Average case:** The efficiency averaged on all possible inputs. We normally assume the uniform distribution.
  - **Worst case:** The maximum number of steps that an algorithm can take for any input data values.

## 5.2 Estimation the complexity of a program:

Suppose you are given a list **L** and a value **x** and you have to find if **x** exists in List **L**.

Simple solution to this problem is traverse the whole List **L** and check if the any element is equal to **x**.

```
def found(x, n, L):
    for i in range(n):
        if L[i]==x:
            return True
    else:
        return False
```

Each of the operation in computer take approximately constant time. Let each operation takes **t** time. The number of lines of code executed is actually depends on the value of **x**. During analysis of algorithm, mostly we will consider worst case scenario, i.e., when **x** is not present in the list **L**. In the worst case, the **if** condition will run **n** times where **n** is the length of the list **L**. So in the worst case, total execution time will be  $(n*t+t)$ .  $n*t$  for the **if** condition and **t** for the **return** statement.

As we can see that the total time depends on the length of the list **L**. If the length of the list will increase the time of execution will also increase.

### 5.2.1 Big O Notation:

**Order of growth** is how the time of execution depends on the length of the input.

In the above example, we can clearly see that the time of execution is linearly depends on the length of the list.

Order of growth in algorithm means how the time for computation increases when you increase the input size. It really matters when your input size is very large.

Order of growth provide only a **crude description of the behaviour of a process**.

For example, a process requiring  $n^2$  steps and another process requiring  $1000n^2$  steps and another process requiring  $3n^2+10n+17$  steps. All have  $O(n^2)$  order of growth. Order of growth provides a useful indication of how we may expect the behavior of the process to change as we change the size of the problem. Depending on the algorithm, the behaviour changes. So, this is one of the most important things that we have to care when we design an algorithm for some given problem.

There are different notations to measure it and the most important one is Big O notation which gives you the worst case time complexity.

Complexity of certain well known algorithms:

- (a) Linear Search :  $O(n)$
- (b) Binary Search :  $O(\log n)$
- (c) Bubble Sort :  $O(n^2)$

### 5.2.2 Wall Clock Time :

The wall-clock time is not the number of seconds that the process has spent on the CPU; it is the elapsed time, including time spent waiting for its turn on the CPU (while other processes get to run).

Performance of algorithm is inversely proportional to the wall clock time it records for a given input size.

Normal Program (Without Recursion)	Recursive Program
<pre>import time start=time.time( ) num = int(input("Enter a number: ")) factorial = 1 if num &lt; 0:     print("factorial does not exist for negative numbers") elif num == 0:     print("The factorial of 0 is 1") else:     for i in range(1, num + 1):         factorial = factorial*i     print("The factorial of",num,"is",factorial)</pre>	<pre>import time def factorial(n):     if n == 1:         return n     else:         return n*factorial(n-1) start=time.time() num=int(input("enter the number: ")) if num &lt; 0:     print("factorial does not exist for negative numbers") elif num==0:     print("The factorial of 0 is 1")</pre>

<pre>end=time.time() t=end-start print("Total time taken: ", t)</pre>	<pre>else:     print("The factorial of ",num," is ", factorial(num)) end=time.time( ) t=end-start print("Total time taken :", t)</pre>
<p><b>OUTPUT:</b></p> <p>Enter a number: 5</p> <p>The factorial of 5 is 120</p> <p>Total time taken: 1.0168840885162354</p>	<p><b>OUTPUT:</b></p> <p>enter the number: 5</p> <p>The factorial of 5 is 120</p> <p>Total time taken : 0.9867522716522217</p>

Recursive program is taking less time than without recursion program. So, Program with recursion is efficient. Therefore, we can conclude that Performance of algorithm is inversely proportional to the wall clock time.

## CHAPTER-6

### DATA VISUALIZATION USING PYPLOT

#### 6.1 INTRODUCTION:

Data visualization basically refers to the graphical or visual representation of information and data using charts, graphs, maps etc.

- pyplot is an interface, which exists in matplotlib library.
- To draw the lines, charts etc. first of all we have to install matplotlib library in our computer and import it in the python program.
- matplotlib is a package which is used to draw 2-D graphics.

#### 6.1.1 Installing the matplotlib :

**Step-1:** Download the wheel package of matplotlib. To download go to the following url:

*<https://pypi.org/project/matplotlib/#files>*

**Step-2:** Now install it using following commands on command prompt:

```
python -m pip install -U pip
```

```
python -m pip install -U matplotlib
```

#### 6.2 COMMONLY USED CHART TYPES:

1. Line Chart
2. Bar Chart
3. Pie Chart

#### 6.2.1 Line Chart:

- A line chart displays information as a series of data points called “markers”.
- import the matplotlib library and pyplot interface.
- pyplot interface has **plot( )** function to draw a line.
- To set the labels for x-axis and y-axis, **xlabel( )** and **ylabel( )** functions are used.
- Use **show( )** function to display the line.

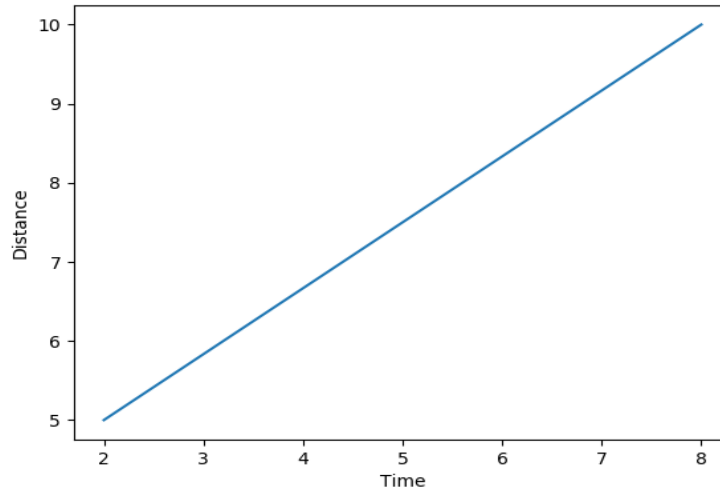
#### Program:

```
import matplotlib.pyplot as pl          #pl is alias for matplotlib.pyplot
x=[2, 8]                                # values of x1 and x2
y=[5, 10]                                #values of y1 and y2
pl.plot(x,y)                             # to create a line
```



```
pl.xlabel("Time")      #label name for x-axis
pl.ylabel("Distance")  #label name for y-axis
pl.show()              #to show the line
```

## Output:



## Formatting the line chart:

- Line : Change the line color, width and style
- Marker : Change the marker Type, size and color

### a. Change the line color, width and style:

#### *Line color:*

#### *Syntax:*

```
matplotlib.pyplot.plot(data1, data2, color-code)
```

#### *Example:*

```
matplotlib.pyplot.plot(x, y, 'r')  #'r' is color code for red colour
```

#### *Different color code:*

color	code
Red	'r'
Green	'g'

Blue	'b'
Yellow	'y'
Magenta	'm'
Black	'k'
Cyan	'c'
White	'w'

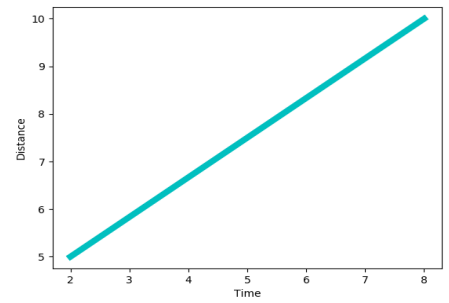
### ***Line width:***

*Syntax:*

```
matplotlib.pyplot.plot(data1, data2, linewidth=value)
```

*Example:*

```
matplotlib.pyplot.plot(x, y, 'c', linewidth=6)
```



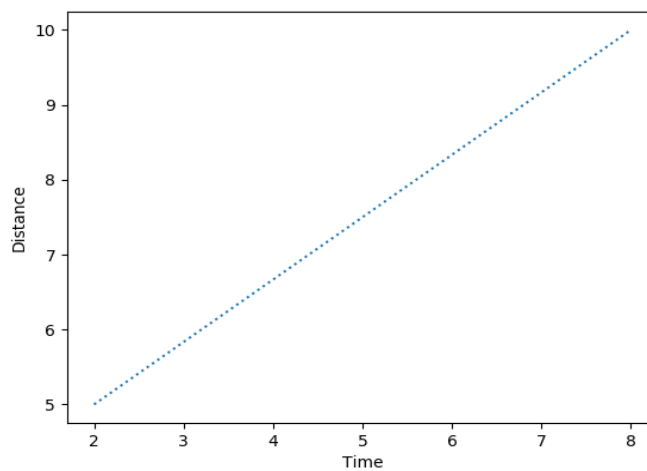
### ***Line style:***

*Syntax:*

```
matplotlib.pyplot.plot(data1, data2, linestyle = value)
```

*Example:*

```
matplotlib.pyplot.plot(x, y, ':')
```



Different types of line styles:

Line Style	Style Name	Lines
-	Solid line (default)	_____
--	Dashed line	-----
:	Dotted line	.....
-.	Dash-dot line	- . - . - . - .

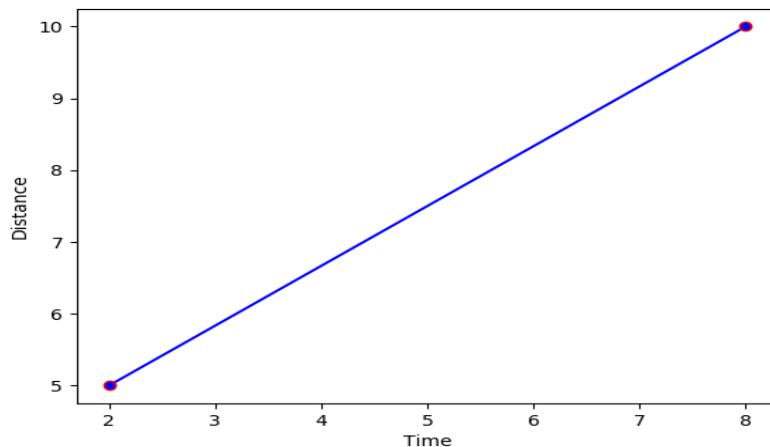
**b. Change the marker type, size and color:**

To change marker type, its size and color, you can give following additional optional arguments in plot( ) function.

**marker=marker-type, markersize=value, markeredgecolor=color**

*Example:*

```
import matplotlib.pyplot as plt
x = [2, 8]
y = [5, 10]
plt.plot(x, y, 'b', marker='o', markersize=6, markeredgecolor='red')
plt.xlabel("Time")
plt.ylabel("Distance")
plt.show( )
```

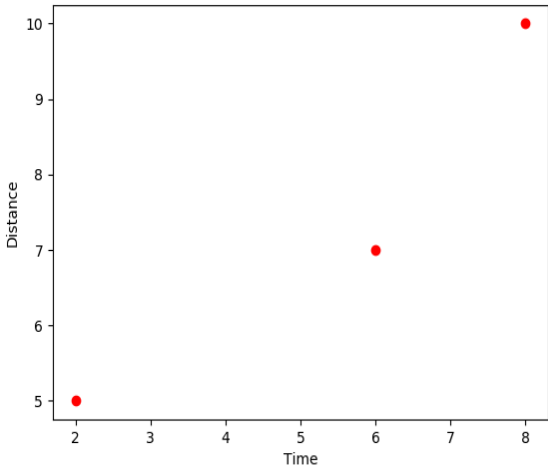
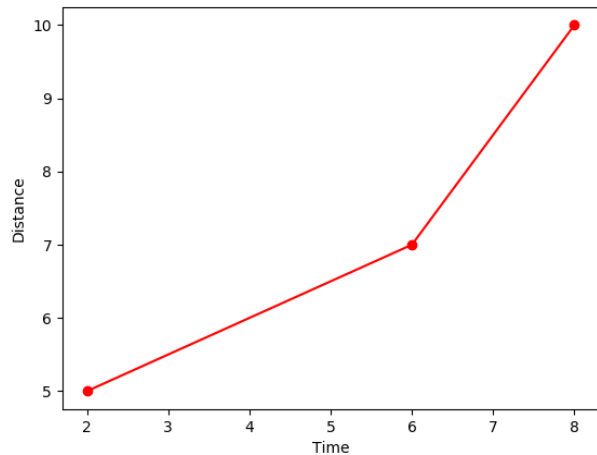


Some marker types for plotting:

marker	description	marker	description
'o'	circle marker	's'	square marker
'x'	cross marker	'*'	star marker
'D'	diamond marker	'p'	pentagon marker

The **markersize** is to be specified in points and **markeredgecolor** must be a valid color.

**Note:** color of the line and marker type can be written combined e.g. 'ro' means red color of the line with circle marker. If you don't specify the **linestyle** separately along with linecolor and marker type combination. Example:

with combination of line color and marker type	with combination of line color, marker type along with <i>linestyle</i>
<pre>import matplotlib.pyplot as pl x=[2,6,8] y=[5,7,10] pl.plot(x,y,'ro') pl.xlabel("Time") pl.ylabel("Distance") pl.show( )</pre>	<pre>import matplotlib.pyplot as pl x=[2,6,8] y=[5,7,10] pl.plot(x,y,'ro', linestyle='-') pl.xlabel("Time") pl.ylabel("Distance") pl.show( )</pre>
	

## 6.2.2 Bar Chart:

- A bar chart is a graphical display of data using bars of different heights.
- import the matplotlib library and pyplot interface.
- pyplot interface has **bar()** function to create a bar chart.
- To set the labels for x-axis and y-axis, **xlabel()** and **ylabel()** functions are used.
- Use **show()** function to display the bar chart.

There are two types of bar charts:

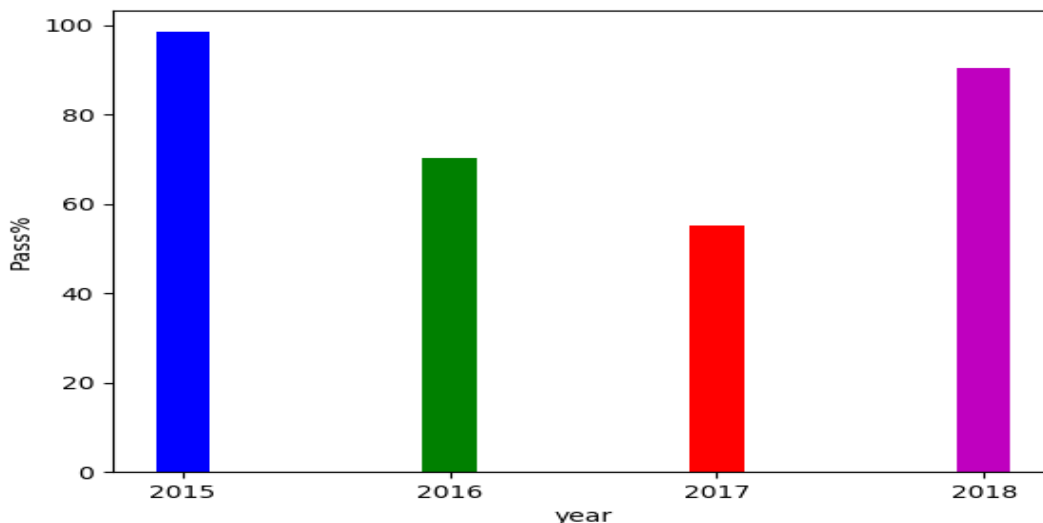
1. Single bar chart
2. Multiple bar chart

### 1. Single bar chart:

**Program: To display the year-wise result of a school using bar chart.**

```
import matplotlib.pyplot as pl
year=['2015','2016','2017','2018'] # list of years
p=[98.50,70.25,55.20,90.5] #list of pass percentage
c=['b','g','r','m'] # color code of bar charts
pl.bar(year, p, width=0.2, color=c) # bar() function to create the bar chart
pl.xlabel("year") # label for x-axis
pl.ylabel("Pass%") # label for y-axis
pl.show() # function to display bar chart
```

### OUTPUT:



The `bar()` function has *width* argument to set the width of all bars. It has *color* argument also, to change the color of bars.

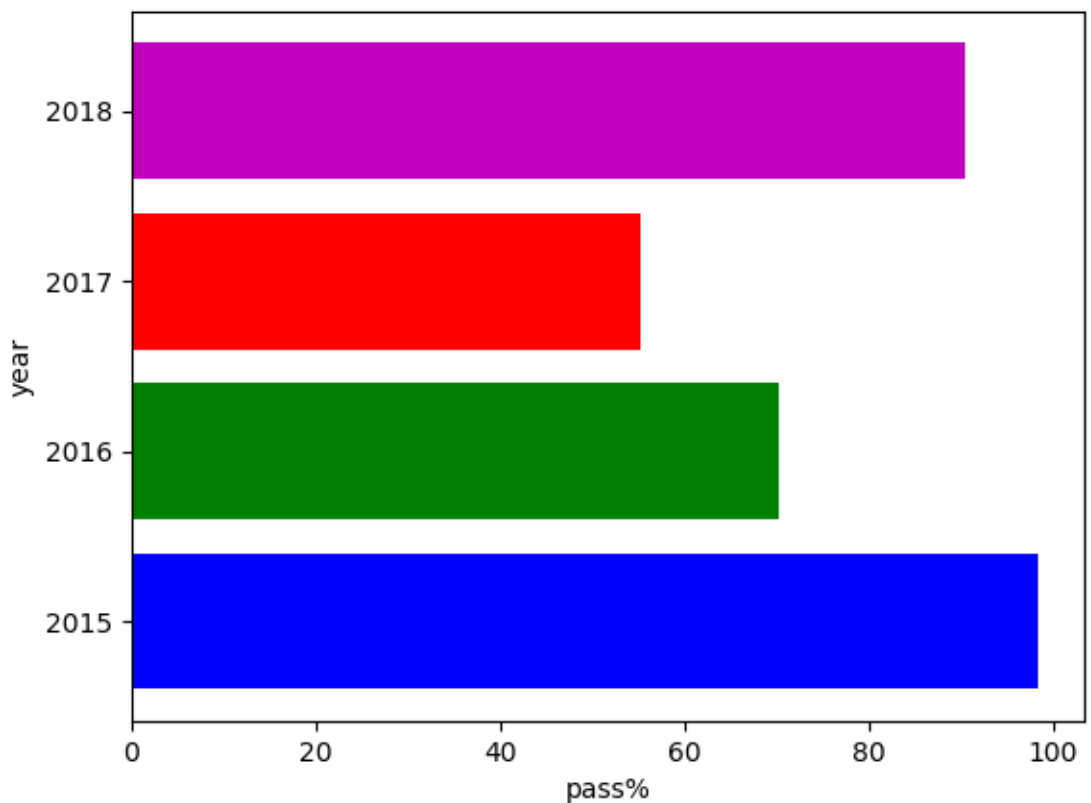
### Creating a horizontal bar chart:

For this we can use `barh()` function (bar horizontal). Swap the names of axis labels.

#### Example:

```
import matplotlib.pyplot as plt
year=['2015','2016','2017','2018']
p=[98.50,70.25,55.20,90.5]
c=['b','g','r','m']
plt.barh(year, p, color = c)
plt.xlabel("pass%")
plt.ylabel("year")
plt.show()
```

#### OUTPUT:

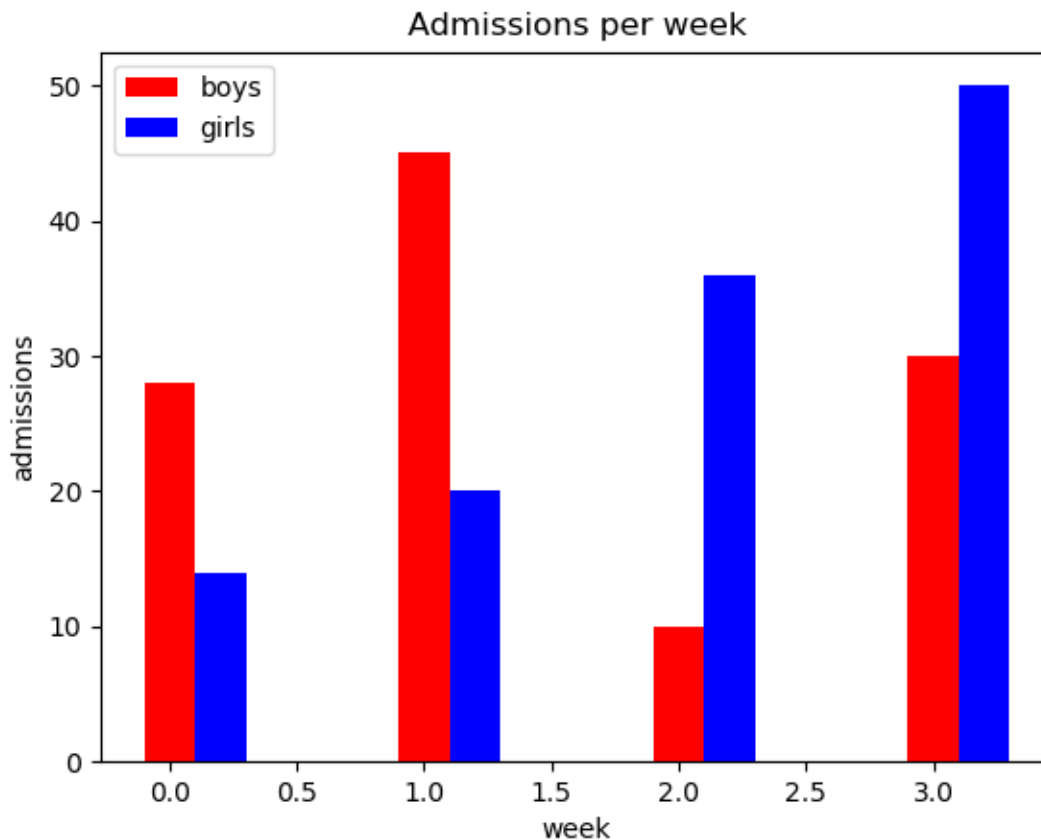


## 2. Multiple Bar Charts:

### *Program:*

```
import matplotlib.pyplot as plt
import numpy as np      # importing numeric python for arange( )
boy=[28,45,10,30]
girl=[14,20,36,50]
X=np.arange(4)         # creates a list of 4 values [0,1,2,3]
plt.bar(X, boy, width=0.2, color='r', label="boys")
plt.bar(X+0.2, girl, width=0.2,color='b',label="girls")
plt.legend(loc="upper left") # color or mark linked to specific data range plotted at location
plt.title("Admissions per week") # title of the chart
plt.xlabel("week")
plt.ylabel("admissions")
plt.show( )
```

### **OUTPUT:**



### Syntax for legend( ):

`matplotlib.pyplot.legend(loc= 'postion-number or string')`

### Example:

`matplotlib.pyplot.legend(loc= 'upper left')`

### OR

`matplotlib.pyplot.legend(loc= 2)`

### Values for loc argument in legend( ):

location string	location code
upper right	1
upper left	2
lower left	3
lower right	4
center	10

### 6.2.3 Pie Chart:

- A pie chart is a circular representation of data in the form of sectors.
- import the matplotlib library and pyplot interface.
- pyplot interface has **pie( )** function to create a pie chart.
- Use **show( )** function to display the bar chart.

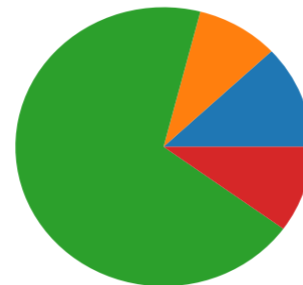
### Example:

```
import matplotlib.pyplot as pl
```

```
part=[12,9,69,10]
```

```
pl.pie(part)
```

```
pl.show( )
```





### Add Labels to slices of pie chart:

To display labels of pie chart, **pie()** function has **labels** argument.

#### Example:

```
import matplotlib.pyplot as plt
part=[12,9,69,10]
plt.pie(part, labels=['abc','pqr','xyz','def'])
plt.show()
```



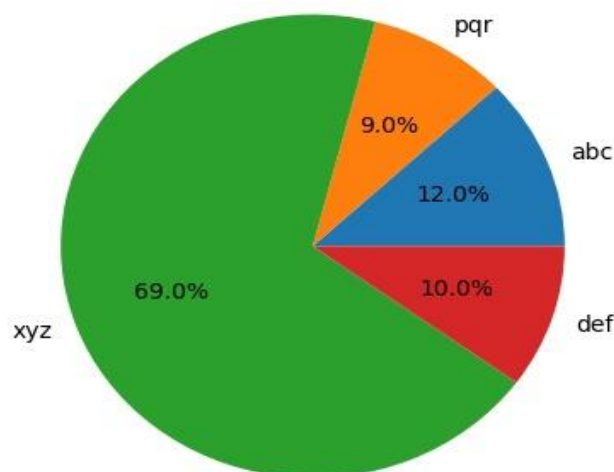
### Add the percentage value to pie chart:

To view percentage of share in a pie chart, you need to add **autopct** argument in **pie()** function.

#### Example:

```
import matplotlib.pyplot as plt
part=[12,9,69,10]
plt.pie(part, labels=['abc','pqr','xyz','def'], autopct= '% 1.1f%% ')
plt.show()
```

### OUTPUT:



- **autopct** argument calculates automatic percentage for each share.
- **autopct** argument has a formatted string as a value. Example : %1.1f%%
- Syntax for formatted string is:

**%<flag><width> . <precision><type>% %**

- **% symbol:** In the starting and ending, the % symbol specifies that it is a special formatted string. One more % after the type is to print a percentage sign after value.
- **flag:** if digits of value are less than width, then flag is preceded to value.
- **width:** Total number of digits to be displayed (including the digits after decimal point). If the value has more digits than the width specifies, then the full value is displayed.
- **precision:** Number of digits to be displayed after decimal point.
- **type:** Specifies the type of value. f or F means float type value, d or i means integer type value.

### Example:

**%07.2f%%**

### Where

% = starting and ending % specifies that it is a special formatted string

0= flag

7=width

2= digits after decimal point

f=float type

%= percentage sign to be displayed after value

### Calculation of percentage for each share:

= (value \* 100)/sum of all values

### Example:

if a list being plotted contains values as: [12,25,10,32]

then percentage of first share:

$$= (12*100)/(12+25+10+32)$$

$$= 1200/79$$

$$=15.19$$

### Change the color of slices:

Add *colors* argument to pie( ) function.

#### Example:

```
import matplotlib.pyplot as plt
```

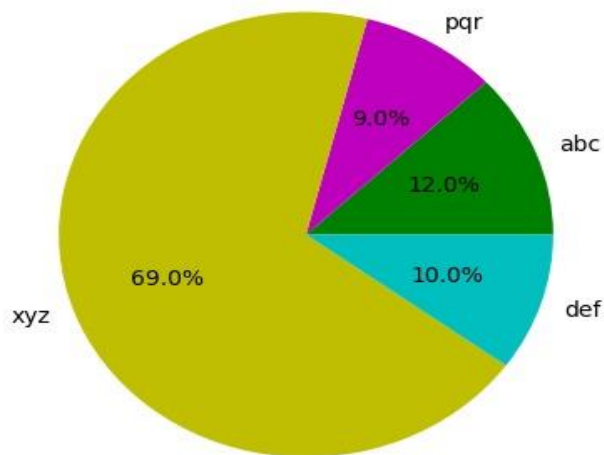
```
part=[12,9,69,10]
```

```
clr=['g','m','y','c']      # list of colors
```

```
plt.pie(part, colors=clr, labels=['abc','pqr','xyz','def'], autopct='%1.1f%%') # adding colors
```

```
plt.show()
```

### OUTPUT:



### Exploding a slice:

Add *explode* argument to pie( ) function.

#### Example:

```
import matplotlib.pyplot as plt
```

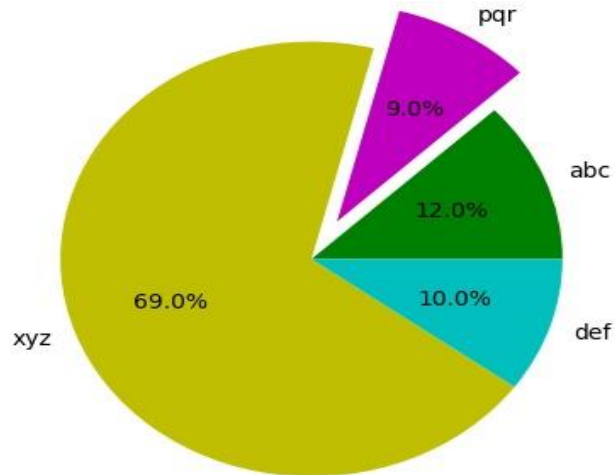
```
part=[12,9,69,10]
```

```
clr=['g','m','y','c']
```

```
ex=[0, 0.2, 0, 0]      # exploding 2nd slice with the distance 0.2 units
```

```
pl.pie(part, colors=clr, labels=['abc','pqr','xyz','def'], autopct='%1.1f%%', explode=ex)  
pl.show( )
```

**OUTPUT:**



## CHAPTER-7

### DATA STRUCTURE-I (LINEAR LIST)

#### 7.1 INTRODUCTION:

**Definition:** The logical or mathematical model of a particular organization of data is called data structure. It is a way of storing, accessing, manipulating data.

#### 7.2 TYPES OF DATA STRUCTURE:

There are two types of data structure:

1. Linear data structure
2. Non-Linear data structure

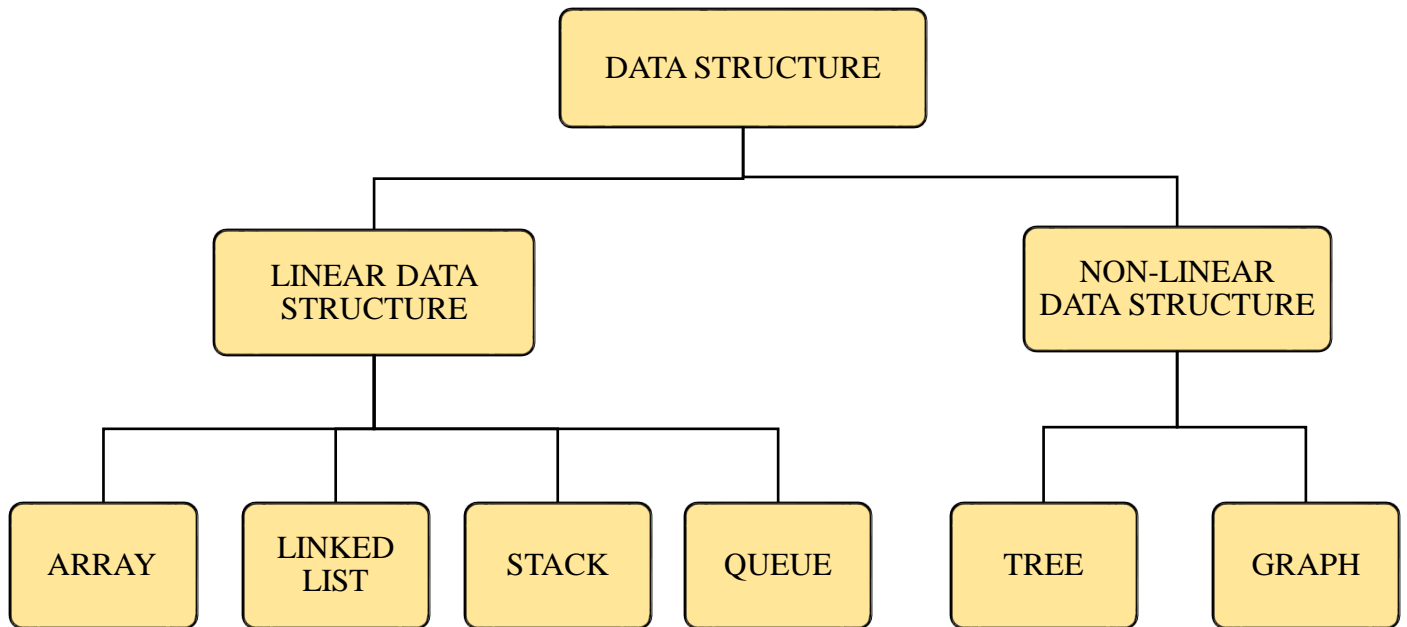


Fig: Types of data structure

**1. Linear data structure:** It is simple data structure. The elements in this data structure creates a sequence. Example: Array, linked list, stack, queue.

**2. Non-Linear data structure:** The data is not in sequential form. These are multilevel data structures. Example: Tree, graph.

### 7.3 OPERATION ON DATA STRUCTURE:

There are various types of operations can be performed with data structure:

1. **Traversing:** Accessing each record exactly once.
2. **Insertion:** Adding a new element to the structure.
3. **Deletion:** Removing element from the structure.
4. **Searching:** Search the element in a structure.
5. **Sorting:** Arrange the elements in ascending and descending order.
6. **Merging:** Joining two data structures of same type. (not covered in syllabus)

### 7.4 ARRAY (LISTS) IN DATA STRUCTURE:

An array or list is the collection of elements in ordered way. There are two types of arrays:

1. One dimensional array (1-D Lists)
2. Multi-dimensional array (Nested Lists)

**7.4.1. One Dimensional array:** It is the collection of homogeneous elements in an order.

**a. Traversing 1-D array (List):**

```
L=[10,20,30,40,50]
n=len(L)
for i in range(n):
    print(L[i])
```

**Output:**

```
10
20
30
40
50
```

**b. Inserting Element in a list:** There are two ways to insert an element in a list:

- (i) If the array is not sorted
- (ii) If the array is sorted

**(i) If the array is not sorted:** In this case, enter the element at any position using insert( ) function or add the element in the last of the array using append( ) function.

**Example:**

```
L=[15,8,25,45,13,19]
```

```
L.insert(3, 88)           # insert element at the index 3
```

```
print(L)
```

**Output:**

```
[15, 8, 25, 88, 45, 13, 19]
```

**(ii) If the array is sorted:** In this case, import bisect module and use the functions bisect( ) and insert( ).

bisect( ) : identifies the correct index for the element and returns the index.

insert( ) : Inserts the element in the list in its correct order.

**Example:**

```
import bisect
```

```
L=[10,20,30,40,50]
```

```
print("Array before insertion the element:", L)
```

```
item=int(input("Enter the element to insert in array: "))
```

```
pos=bisect.bisect(L,item)           #will return the correct index for item
```

```
bisect.insort(L,item)               #will insert the element
```

```
print("Element inserted at index: ", pos)
```

```
print("Array after insertion the element : ", L)
```

**OUTPUT:**

```
Array before insertion the value: [10, 20, 30, 40, 50]
```

```
Enter the element to insert in array: 35
```

```
Element inserted at index : 3
```

```
Array after insertion the element : [10, 20, 30, 35, 40, 50]
```

**Note: bisect( ) works only with that lists which are arranged in ascending order.**

**c. Deletion of an element from a List:** To delete an element from a list we can use `remove()` or `pop()` method.

**Example:**

```
L=[10,15,35,12,38,74,12]
print("List Before deletion of element: ", L)
val=int(input("Enter the element that you want to delete: "))
L.remove(val)
print("After deletion the element", val,"the list is: ", L)
```

**OUTPUT:**

```
List Before deletion of element: [10, 15, 35, 12, 38, 74, 12]
Enter the element that you want to delete: 12
After deletion the element 12 the list is: [10, 15, 35, 38, 74, 12]
```

**d. Searching in a List:**

There are two types of searching techniques we can use to search an element in a list. These are:

- (i) Linear Search
- (ii) Binary Search

**(i) Linear Search:** It is a simple searching technique.

**Program:**

```
L=eval(input("Enter the elements: "))
n=len(L)
item=eval(input("Enter the element that you want to search : "))
for i in range(n):
    if L[i]==item:
        print("Element found at the position :", i+1)
```



```
        break
    else:
        print("Element not Found")
```

**Output:**

Enter the elements: 56,78,98,23,11,77,44,23,65

Enter the element that you want to search : 23

Element found at the position : 4

**(ii) Binary Search:** (Theory already discussed in the chapter recursion).

**Program:**

```
def BinarySearch(LIST,n,item):
    LB=0
    UB=n-1
    while LB<=UB:
        mid=int((LB+UB)/2)
        if item<LIST[mid]:
            UB=mid-1
        elif item>LIST[mid]:
            LB=mid+1
        else:
            return mid
    else:
        return -1

L=eval(input("Enter the elements in sorted order: "))
size=len(L)
element=int(input("Enter the element that you want to search :"))
found=BinarySearch(L,size,element)
```

```

if found >= 0:
    print(element, "Found at the position : ", found+1)
else:
    print("Element not present in the list")

```

**OUTPUT:**

Enter the elements in sorted order: [12,23,31,48,51,61,74,85]

Enter the element that you want to search : 61

61 Found at the position : 6

***Linear Search Vs Binary Search:***

Linear Search	Binary Search
Access of elements sequentially.	Access of elements randomly.
Elements may or may not be in sorted order.	Elements must be in sorted order i.e. ascending or descending order
Takes more time to search an element.	Takes less time to search an element.
easy	tricky
Efficient for small array.	Efficient for larger array

**e. Sorting:** To arrange the elements in ascending or descending order. There are many sorting techniques. Here we shall discuss two sorting techniques:

- (i) Bubble sort
- (ii) Insertion sort

**(i) BUBBLE SORT:** Bubble sort is a simple sorting algorithm. It is based on comparisons, in which each element is compared to its adjacent element and the elements are swapped if they are not in proper order.

### First Pass

4	13	1	7
---	----	---	---

4	1	13	7
---	---	----	---

4	1	13	7
---	---	----	---

4	1	7	13
---	---	---	----

### Second Pass

4	1	7	13
---	---	---	----

1	4	7	13
---	---	---	----

1	4	7	13
---	---	---	----

1	4	7	13
---	---	---	----

### Third Pass

1	4	7	13
---	---	---	----

1	4	7	13
---	---	---	----

1	4	7	13
---	---	---	----

Finish

### PROGRAM:

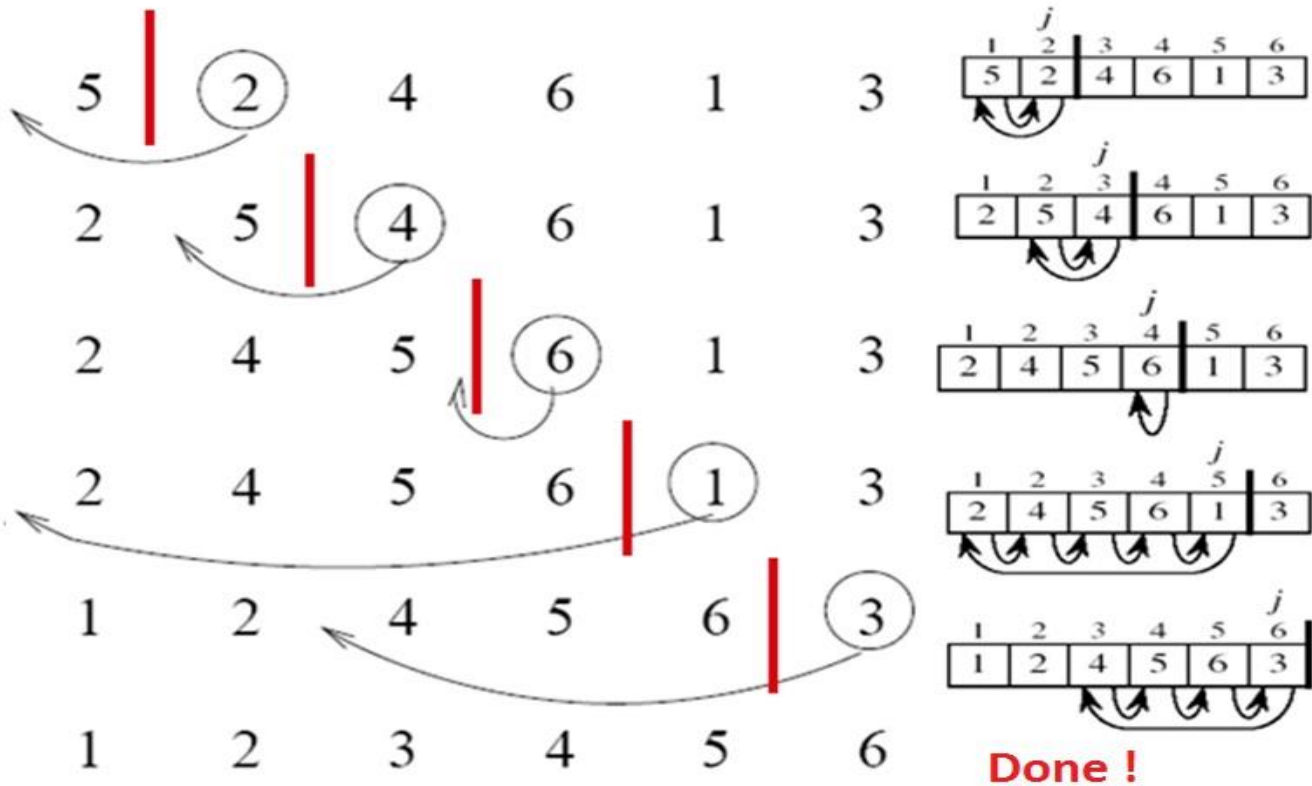
```
L=eval(input("Enter the elements:"))
n=len(L)
for p in range(0,n-1):
    for i in range(0,n-1):
        if L[i]>L[i+1]:
            L[i], L[i+1] = L[i+1],L[i]
print("The sorted list is : ", L)
```

### OUTPUT:

Enter the elements:[60, 24, 8, 90, 45, 87, 12, 77]

The sorted list is : [8, 12, 24, 45, 60, 77, 87, 90]

(ii) **INSERTION SORT:** Sorts the elements by shifting them one by one and inserting the element at right position.



**PROGRAM:**

```
L=eval(input("Enter the elements: "))
```

```
n=len(L)
```

```
for j in range(1,n):
```

```
    temp=L[j]
```

```
    prev=j-1
```

```
    while prev>=0 and L[prev]>temp:      # comparison the elements
```

```
        L[prev+1]=L[prev]              # shift the element forward
```

```
        prev=prev-1
```

```
    L[prev+1]=temp                      #inserting the element at proper position
```

```
print("The sorted list is :",L)
```

## OUTPUT:

Enter the elements: [45, 11, 78, 2, 56, 34, 90, 19]

The sorted list is : [2, 11, 19, 34, 45, 56, 78, 90]

**7.4.2. Multi-Dimensional array (Nested Lists):** A list can also hold another list as its element. It is known as multi-dimensional or nested list.

A list in another list considered as an element.

### Example:

```
>>>NL=[10, 20, [30,40], [50,60,70], 80]
```

```
>>> len(NL)
```

```
5
```

```
>>>secondlist=[1,2,3,[4,5,[6,7,8],9],10,11]
```

```
>>> len(secondlist)
```

```
6
```

### Accessing the elements from nested list:

#### Example-1:

```
>>> L=[1, 2, 3, [4, 5, [ 6, 7, 8 ], 9 ],10, 11]
```

```
>>> L[1]
```

```
2
```

```
>>> L[3]
```

```
[4, 5, [6, 7, 8], 9]
```

```
>>> L[3][1]
```

```
5
```

```
>>> L[3][2][0]
```

```
6
```

```
>>> L[3][2]
```

```
[6, 7, 8]
```

```
>>> L[3][2][1]
```

7

```
>>> L[3][3]
```

9

### Example-2:

```
>>> L=["Python", "is", "a", ["modern", "programming"], "language", "that", "we", "use"]
```

```
>>> L[0][0]
```

```
'P'
```

```
>>> L[3][0][2]
```

```
'd'
```

```
>>> L[3:4][0]
```

```
['modern', 'programming']
```

```
>>> L[3:4][0][1]
```

```
'programming'
```

```
>>> L[3:4][0][1][3]
```

```
'g'
```

```
>>> L[0:9][0]
```

```
'Python'
```

```
>>> L[0:9][0][3]
```

```
'h'
```

```
>>> L[3:4][1]
```

```
IndexError: list index out of range
```

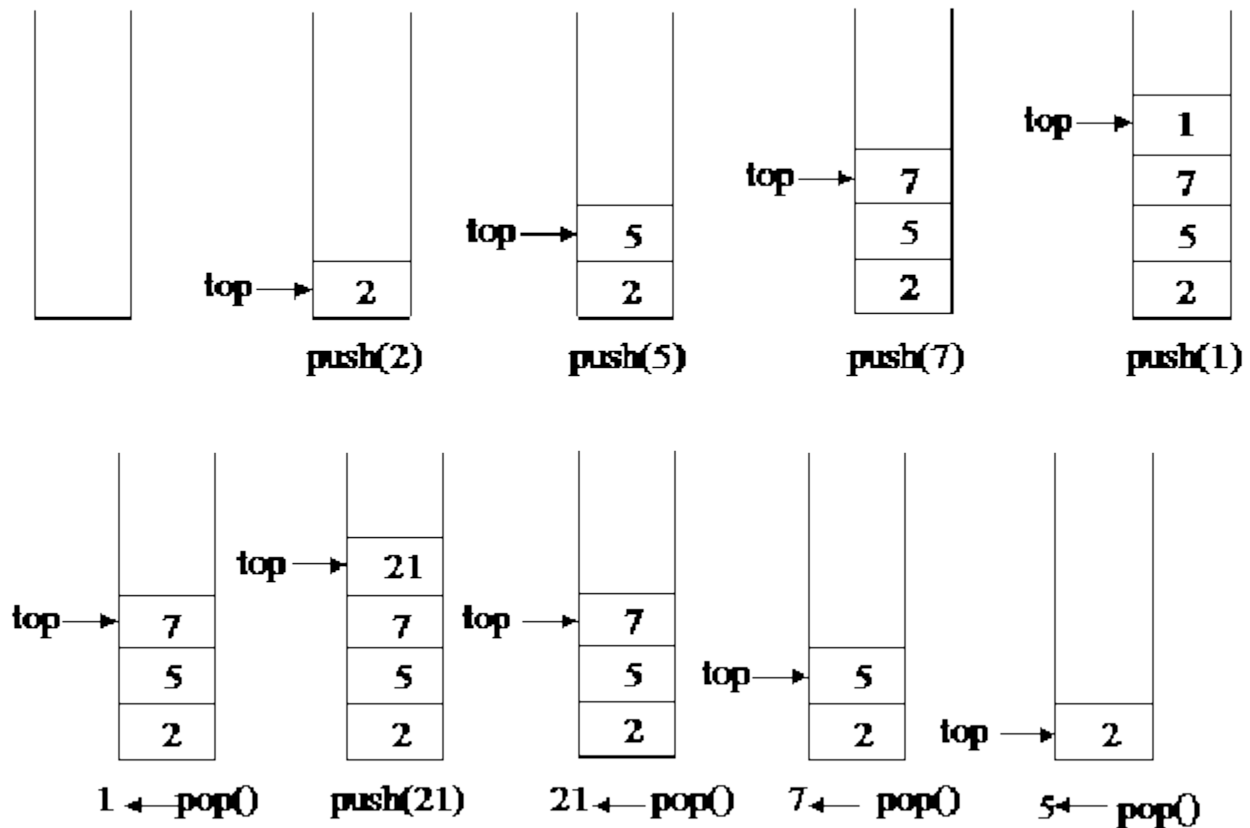
## CHAPTER-8

### DATA STRUCTURE-II (STACK AND QUEUE)

#### 8.1 STACK IN PYTHON:

##### 8.1.1 INTRODUCTION:

- Stack is a linear data structure.
- Stack is a list of elements in which an element may be inserted or deleted only at one end, called the **TOP** of the stack.
- It follows the principle **Last In First Out (LIFO)**.
- There are two basic operations associated with stack:
  - **Push** : Insert the element in stack
  - **Pop** : Delete the element from stack



*Example: Stack Push and Pop operations*

##### 8.1.2 MENU BASED PROGRAMME FOR STACK:

```
class Stack:  
    def __init__(self):  
        self.items = [ ]  
  
    def isEmpty(self):        # Checks whether the stack is empty or not
```

```

    return self.items == [ ]

def push(self, item):          #Insert an element
    self.items.append(item)

def pop(self):                 # Delete an element
    return self.items.pop( )

def peek(self):                #Check the value of top
    return self.items[len(self.items)-1]

def size(self):                # Size of the stack i.e. total no. of elements in stack
    return len(self.items)

s = Stack( )
print("MENU BASED STACK")
cd=True
while cd:
    print(" 1. Push ")
    print(" 2. Pop ")
    print(" 3. Display ")
    print(" 4. Size of Stack ")
    print(" 5. Value at Top ")

    choice=int(input("Enter your choice (1-5) : "))

    if choice==1:
        val=input("Enter the element: ")
        s.push(val)
    elif choice==2:
        if s.items==[ ]:
            print("Stack is empty")
        else:
            print("Deleted element is :", s.pop( ))
    elif choice==3:
        print(s.items)
    elif choice==4:
        print("Size of the stack is :", s.size( ))
    elif choice==5:
        print("Value of top element is :", s.peek( ))
    else:
        print("You entered wrong choice ")

```



```

print("Do you want to continue? Y/N")
option=input( )
if option=='y' or option=='Y':
    cd=True
else:
    cd=False

```

### 8.1.3 STACK POLISH NOTATION:

To evaluate an expression, we use three types of polish notations:

1. Prefix notation : +AB
2. Infix notation : A+B
3. Postfix notation : AB+

The evaluation of expression depends upon order of the element and precedence of an operator.

#### Precedence of Basic Arithmetic operators:

Name of operator	Symbol	Precedence (Highest to Lowest)
Exponentiation	↑	1
Multiplication, division, division floor and modulus	*, /, //, %	2
Addition and Subtraction	+, -	3

### 8.1.4 Evaluation of an expression written in postfix notation:

**Example-1: Evaluate the following expression:**

**P: 5, 6, 2, +, \*, 12, 4, /, -**

**Solution:**

Element	Operation	Stack	Result
5	Push	5	
6	Push	5, 6	
2	Push	5, 6, 2	
+	Pop	5, 8	6+2=8
*	Pop	40	5*8=40
12	Push	40, 12	
4	Push	40, 12, 4	
/	Pop	40, 3	12/4=3
-	Pop	37	40-3=37

**Example-2: Evaluate the following expression:**

**P: 4 , 5, +, 6, 11, -, \*,15, /**

**Solution:**

Element	Operation	Stack	Result
4	Push	4	
5	Push	4, 5	
+	Pop	9	4+5=9
6	Push	9, 6	
11	Push	9, 6, 11	
-	Pop	9, -5	6-11= -5
*	Pop	-45	9* -5 = -45
15	Push	-45, 15	
/	Pop	-45/15	-45/15 = -3

**Example-3: Evaluate the following expression:**

**P: True, False, True, OR, False, AND, NOT, OR**

**Solution:**

Element	Operation	Stack	Result
True	Push	True	
False	Push	True, False	
True	Push	True, False, True	
OR	Pop	True, True	False OR True= True
False	Push	True, True, False	
AND	Pop	True, False	True AND False= False
NOT		True, True	NOT False = True
OR	Pop	True	True OR True = <b>True</b>

### 8.1.5 Infix to Postfix Conversion:

**Example-1 : Convert the following infix expression into its equivalent postfix expression.**

$$Y = (A + B * (C - D) / E)$$

Symbol	Stack	Expression	Description
(	(		Push the symbol in stack
A	(	A	Move the element in expression
+	( +	A	Operator, so push into stack
B	( +	AB	Move the element in expression
*	( + *	AB	Operator, push into stack, * has higher precedence than +, so can come after +
(	( + * (	AB	Push the open parentheses symbol into stack
C	( + * (	ABC	Move the element in expression
-	( + * (-	ABC	Operator, push into stack
D	( + * (-	ABCD	Move the element in expression
)	( + *	ABCD -	Closing parentheses, so pop the elements of the stack up to opening parentheses
/	( + /	ABCD - *	Operator, / and * operators have the same precedence, so associativity from left to right will take place and * will pop from stack and / will push into stack
E	( + /	ABCD - *E	Move the element in expression
)		ABCD - *E / +	Closing parentheses, so pop the elements of the stack up to opening parentheses

**Example-2 : Convert the following infix expression into its equivalent postfix expression.**

$$Y = (A / B - (C * D \uparrow E) + F)$$

Symbol	Stack	Expression	Description
(	(		Push the symbol in stack
A	(	A	Move the element into expression
/	( /	A	Operator, so push into stack
B	( /	AB	Move the element into expression
-	( -	AB/	Operator, - has lower precedence than /. so, / will pop from stack and - will push into stack.
(	( - (	AB/	Push the symbol into stack
C	( - (	AB/C	Move the element into expression
*	( - (*	AB/C	Operator, push into stack
D	( - (*	AB/CD	Move the element into expression
↑	( - (*↑	AB/CD	Operator, ↑ has higher precedence than *, so can come after *.
E	( - (*↑	AB/CDE	Move the element into expression



```
Q.rear=Q.items[len(Q.items)-1]
print("Now Front is: ", Q.front)
print("Now Rear is: ", Q.rear)
```

```
def Dequeue(Q):                                # Delete an element
    d=Q.items.pop(0)
    Q.rear=Q.items[len(Q.items)-1]
    Q.front=Q.items[0]
    print("Now Front is: ", Q.front)
    print("Now Rear: is: ", Q.rear)
    return d
```

```
def size(Q):                                    # Size of the queue i.e. total no. of elements in queue
    return len(Q.items)
```

```
#main method
```

```
q = Queue( )
```

```
print("MENU BASED QUEUE")
```

```
cd=True
```

```
while cd:
```

```
    print(" 1. ENQUEUE ")
```

```
    print(" 2. DEQUEUE ")
```

```
    print(" 3. Display ")
```

```
    print(" 4. Size of Queue ")
```

```
choice=int(input("Enter your choice (1-5) : "))
```

```
if choice==1:
```

```
    val=input("Enter the element: ")
```

```

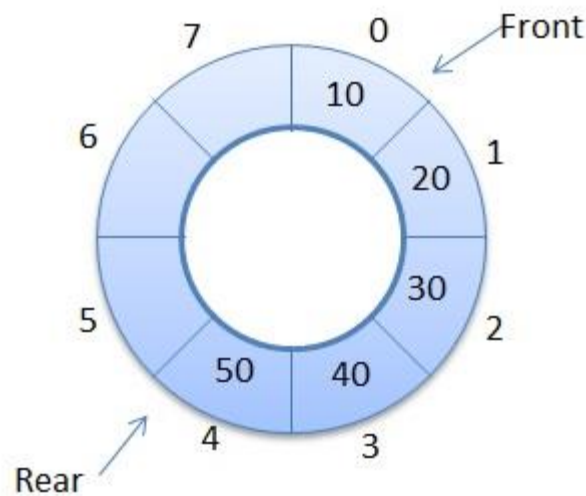
    q.Enqueue(val)
elif choice==2:
    if q.items==[ ]:
        print("Queue is empty")
    else:
        print("Deleted element is :", q.Dequeue( ))
elif choice==3:
    print(q.items)
elif choice==4:
    print("Size of the queue is :", q.size( ))
else:
    print("You entered wrong choice ")

print("Do you want to continue? Y/N")
option=input( )
if option=='y' or option=='Y':
    cd=True
else:
    cd=False

```

### **8.3 CIRCULAR QUEUE:**

A Circular Queue is a queue but circular in shape, therefore after the last position, the next place in the queue is the first position.



A Circular Queue can be seen as an improvement over the Linear Queue because:

1. There is no need to reset **front** and **rear**, since they reset themselves. This means that once the **front** or **rear** reaches the end of the Queue, it resets itself to 0.
2. The rear and front can point to the same location - this means the Queue is empty.

Applications of Circular Queue:

- Job Scheduling
- Traffic Signals

### 7.3.1 MENU BASED PROGRAM FOR CIRCULAR QUEUE:

class CircularQueue:

```
def __init__(CQ):          #Constructor
    CQ.queue = [None]*7    # Create a list with None values with the size 7
    CQ.front = -1
    CQ.rear = -1
    CQ.maxSize = 7
```

```

def C_enqueue(CQ,data):          #Adding elements to the queue
    if (CQ.rear+1) % CQ.maxSize==CQ.front:
        CQ.rear = -1
        CQ.rear = (CQ.rear + 1) % CQ.maxSize
        CQ.queue[CQ.rear]=data

    elif CQ.front== -1:
        CQ.front=0
        CQ.rear=0
        CQ.queue[CQ.rear]=data
    else:
        CQ.rear = (CQ.rear + 1) % CQ.maxSize
        CQ.queue[CQ.rear]=data

```

```

def C_dequeue(CQ):              #Removing elements from the queue

    if CQ.front == -1:
        print("Queue is empty")
    elif CQ.front == CQ.rear:
        CQ.queue.pop(CQ.front)
        CQ.front=CQ.front+1
    else:
        CQ.queue.pop(CQ.front)
        CQ.front = (CQ.front + 1) % CQ.maxSize

```



```
#main
q = CircularQueue()
print("MENU BASED CIRCULAR QUEUE")
cd=True
while cd:
    print("1. ENQUEUE")
    print("2. DEQUEUE")
    print("3. DISPLAY ")
    print("4. Front Position ")
    print("5. Rear Position ")

    choice=int(input("Enter your choice (1-5) : "))

    if choice==1:
        val=input("Enter the element: ")
        q.C_enqueue(val)

    elif choice==2:
        q.C_dequeue()

    elif choice==3:
        print(q.queue)

    elif choice==4:
        print("Front element position :", q.front)

    elif choice==5:
        print("Rear element position : ", q.rear)
```

else:

```
    print("You entered invalid choice: ")
```

```
print("Do you want to continue? Y/N")
```

```
option=input( )
```

```
if option=='y' or option=='Y':
```

```
    cd=True
```

```
else:
```

```
    cd=False
```

## **2.1 INTRODUCTION:**

**Network:** - To connect more than one devices via a medium, is called network.



### ***Why do we need network?***

1. Communication
2. Resource sharing
3. Reduce Cost

## **2.2 TYPES OF NETWORKS:**

1. Local Area Network (LAN)
2. Metropolitan Area Network (MAN)
3. Wide Area Network (WAN)
4. Personal Area Network (PAN)

### **1. LAN:**

\*Use in small local area, like in an institute or an organization.

- \* Devices are connected via physical medium.
- \* Limited distance, up to 150 Meter.
- \* Example - **Intranet**

## 2. MAN:

- \* Larger than LAN.
- \* Used in Metropolitan cities.
- \* Range up to 50 KM.

## 3. WAN:

- \* Large network
- \* Public
- \* Example – **Internet**

## 4. PAN:

- \* For very small distance
- \* Private Communication
- \* Example: **Bluetooth**

## 2.3 WEB VS INTERNET:

### WEB:

- World Wide Web started in 1989, launched by ‘ *Tim Berners Lee*’ .
- The web is a service that run on the internet.
- It is collection of web pages.

### INTERNET:

- It is a network of networks.
- Web is a part of internet and runs on internet.
- It is a public network.

## 2.4 CLOUD COMPUTING:

- A kind of internet based computing where resources, storage, data and information computing services are provided on-demand.
- One can use remote servers using cloud computing instead of configure or install its personal server.
- It is a way to deploy and manage the services over internet.
- Example:  
**Storage and Backup:** Google Drive, Dropbox, Onedrive

**Chatbots:** Google assistant, Siri, Alexa

**Scalable uses (Subscription models):** Netflix

**Communication :** Skype, WhatsApp

**Productivity:** Microsoft Office 365, Google docs

**Business:** Marketo, Salesforce

**Social Networking:** Facebook, Twitter, Myspace

## 2.4.1 TYPES OF CLOUD COMPUTING:

There are 4 types of cloud computing:

- a. **Public cloud:** Public accessibility, anyone can use.
- b. **Private cloud:** The cloud infrastructure works for one organization. It can be managed by the company itself. Privately accessible.
- c. **Hybrid cloud:** Infrastructure consists of two or more clouds (private, public or community)
- d. **The cloud community:** The infrastructure is shared by several organizations that have common interests or requirements.

### Public Cloud Vs Private Cloud:

Points	Public Cloud	Private Cloud
Resources	Owned and operated by cloud service provider. Can be used by publicly.	exclusively used by an organization
Services	Services always maintained on a public network i.e. over internet	Services always maintained on a private network.
Infrastructure	Owned and managed by cloud service provider.	Infrastructure dedicated to a business or organization.
Maintenance	Depends upon cloud service provider.	Easier to use and customize the resources.
Cost	Lower	Expensive
Data security	Less	High & improved security
Flexibility	Less	More
Example	Microsoft Azure	Often used by govt. agencies and financial institutes.

## 2.4.2 Advantages and disadvantages of cloud computing:

### *Advantages:*

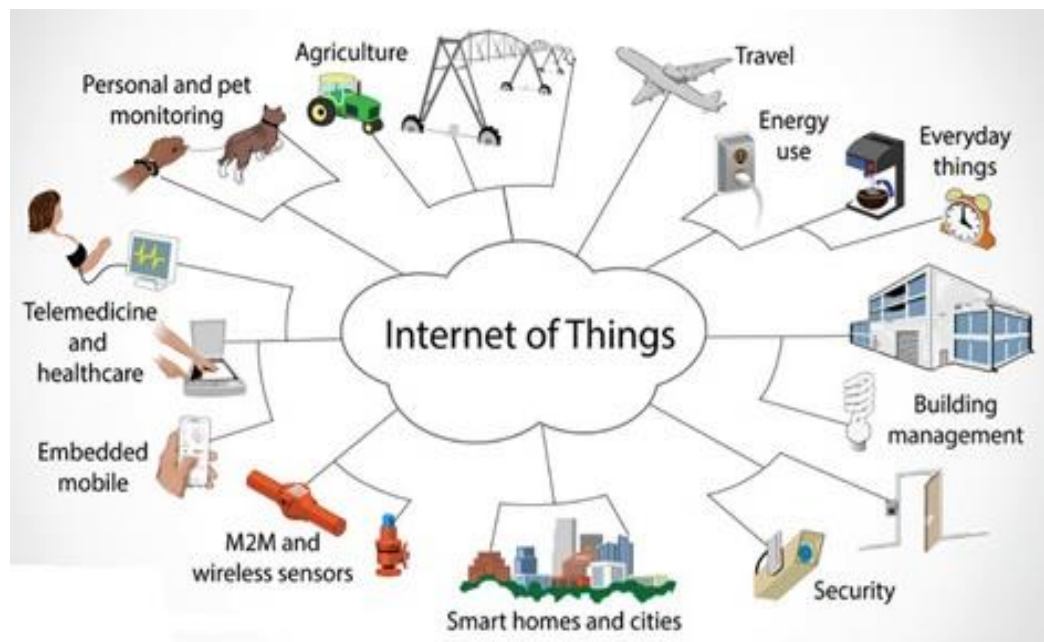
- **Cost reduction** : Companies can save money by using shared resources.
- **Increased storage** : Instead of purchasing large amount of storage, companies can increase the storage as per requirement on nominal charges.
- **Highly Automated** : As per software and hardware requirement, service providers keeps the things up to date and available.
- **Greater mobility** : Once the information is stored in the cloud, one can access the information anywhere, anytime.
- **Keeps things updated**

### *Disadvantages:*

- Security
- Data privacy
- Where the data stored.
- Always need of internet to access data
- Service unavailability

## 2.5 IoT (Internet of Things) :

- Connect the physical objects through internet for communication and sense interaction.
- **Example:**
  - **Smart Home:** To control the home remotely, like start the A.C., lock or unlock the door, etc.
  - **Wearables:** They collect the data and information about the user and processed it for the user. Fitness and health devices, smart watches etc.
  - **Cars:** To control the car or other vehicles.
  - **Smart Cities:** Smart surveillance, automated transportation, smarter energy management systems, water distribution, urban security and environmental monitoring all are examples of internet of things applications for smart cities.
  - **Agriculture:** Sensing for soil moisture and nutrients, controlling water usage for plant growth and determining custom fertilizer are some simple uses of IoT.
  - **Smartphone detection.**



### 2.5.1 COMPONENTS OF IoT:

- a. Low power embedded system
- b. Cloud Computing
- c. Availability of big data
- d. Network connection

### 2.5.2 Advantages of IoT:

1. **Data:** The more the information, the easier it is to make the right decision. Knowing what to get from the grocery while you are out, without having to check on your own, not only saves time but is convenient as well.
2. **Tracking:** The computers keep a track both on the quality and the viability of things at home. Knowing the expiration date of products before one consumes them improves safety and quality of life. Also, you will never run out of anything when you need it at the last moment.
3. **Time:** The amount of time saved in monitoring and the number of trips done otherwise would be tremendous.
4. **Money:** The financial aspect is the best advantage. This technology could replace humans who are in charge of monitoring and maintaining supplies.

### 2.5.3 Disadvantages of IoT:

1. **Compatibility:** As of now, there is no standard for tagging and monitoring with sensors. A uniform concept like the USB or Bluetooth is required which should not be that difficult to do.
2. **Complexity:** There are several opportunities for failure with complex systems. For example, both you and your spouse may receive messages that the milk is over and both of

you may end up buying the same. That leaves you with double the quantity required. Or there is a software bug causing the printer to order ink multiple times when it requires a single cartridge.

3. **Privacy/Security:** Privacy is a big issue with IoT. All the data must be encrypted so that data about your financial status or how much milk you consume isn't common knowledge at the work place or with your friends.

4. **Safety:** There is a chance that the software can be hacked and your personal information misused. The possibilities are endless. Your prescription being changed or your account details being hacked could put you at risk. Hence, all the safety risks become the consumer's responsibility.

### 2.6 Transmission Medium:

A medium which is used to connect the devices and transfers the data from one device to another device.

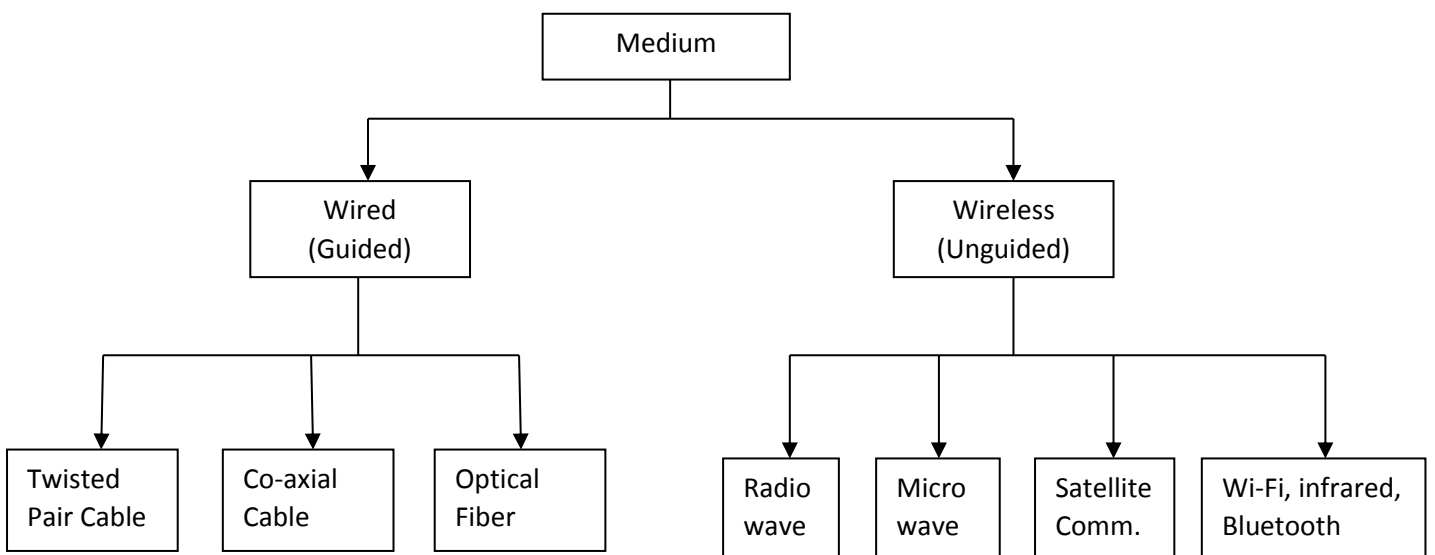


fig: Twisted Pair Cable

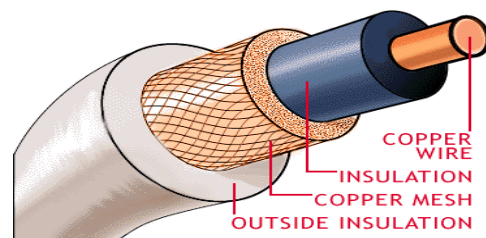


fig: Co-axial Cable



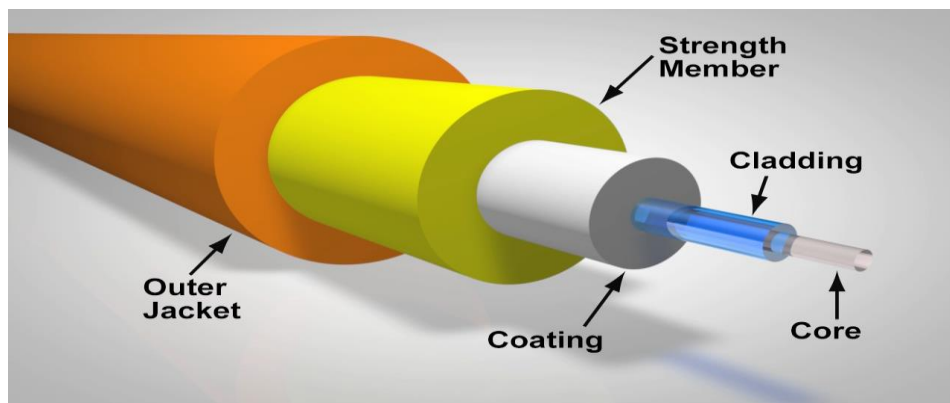
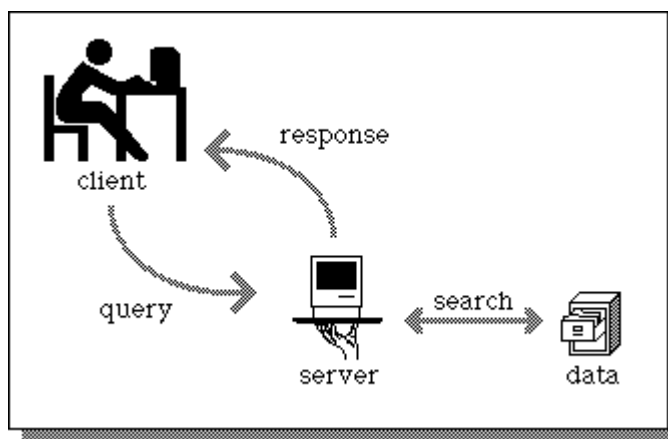


fig: Optical Fiber Cable (Principle: Total Internal Reflection)

## 2.7 Client-Server Architecture:

**Client:** A client which sends the request or query to the server.

**Server:** Server is a machine that takes the query from client, process the query and sends the result back to client.



*Fig.: Client-Server Interaction*

*Types of server:*

1. Dedicated Sever
2. Non-Dedicated Server

**Dedicated Server:** A dedicated server is a server whose only job is to help workstations access data, software and hardware. It does not double up as a workstation.

**Non-dedicated Server:** A non-dedicated server acts as a server as well as a workstation.

## 2.8 NETWORK DEVICES:

1. Modem
2. Hub
3. Switch
4. Gateway
5. Bridge
6. Router
7. Repeater
8. NIC (Network Interface Card)

### 1. Modem:

- The full form of modem is Modulator and demodulator.
- A modem is a device or program that enables a computer to transmit data over telephone or cable lines.
- A modem converts analog signal to digital signal and vice-versa.
- Modem connects computer to internet.
- There are two types of modem:
  - a. Internal Modem
  - b. External Modem

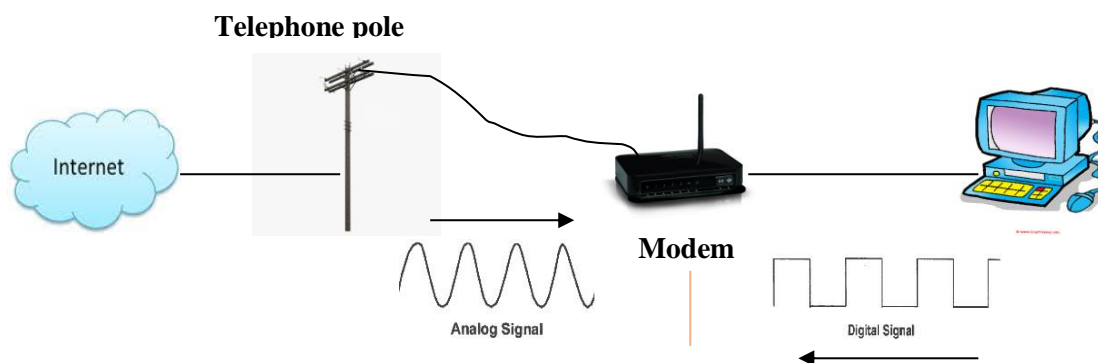


Fig. : Working of Modem

### 2. Hub:

- A network device that contains multiple ports.
- Provides multiple connections.
- When a packet arrives at one port, it is copied to the other ports so that all segments of the LAN can see all packets.
- Two types of hub :
  - a. Active Hub
  - b. Passive Hub

# Hub

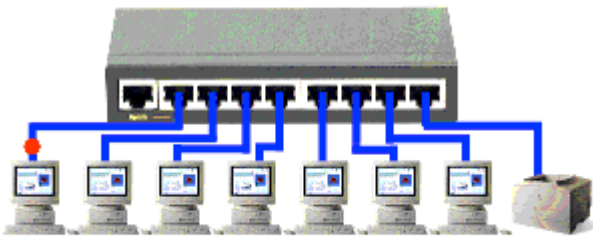


Fig: Hub

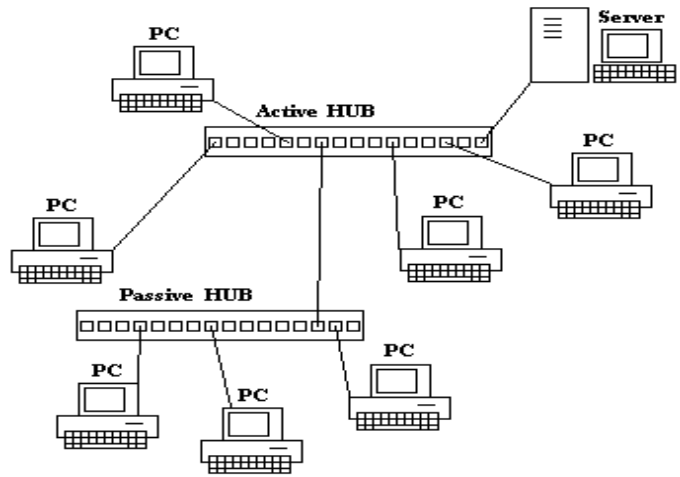


Fig. : Active and Passive Hub

### 3. Switch:

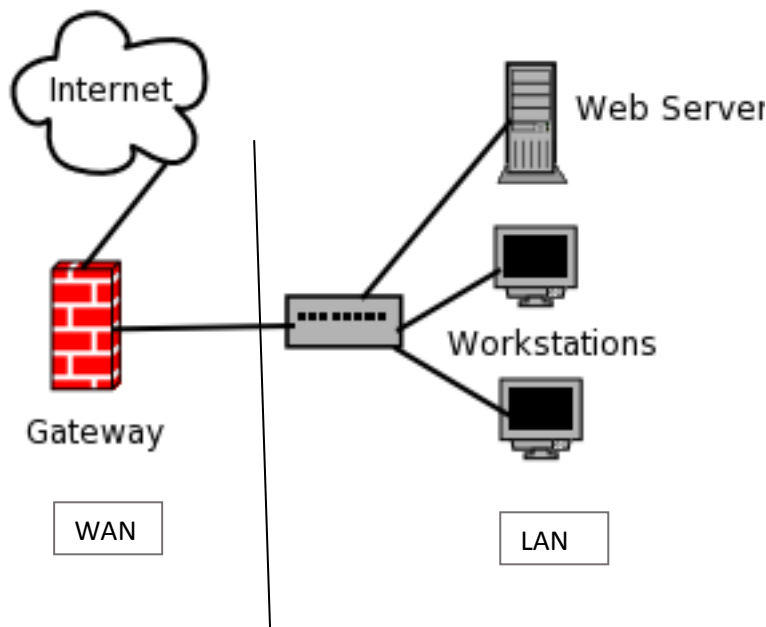
- A switch is called *smart hub*.
- Provides multiple connections
- A device that filters and forwards packets between LAN segments.



HUB	SWITCH
Hub passes the frame to every port.	Passes the frame to a specific port, because it keeps a record of MAC address.
Creates lot of traffic on network	Less traffic
Hub shares its bandwidth with each and every port, so bandwidth divided among all the nodes, which will degrade performance.	Switch allocates full bandwidth to each of its port. So user always access maximum amount of bandwidth.
Slow speed	Fast speed

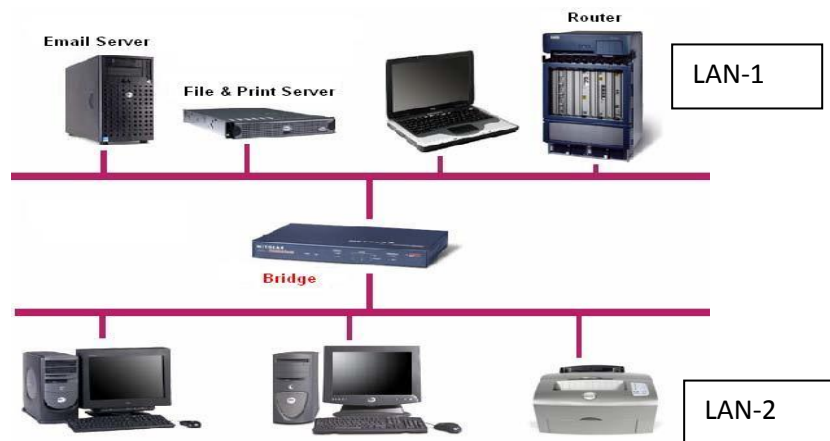
### 4. Gateway:

- A gateway is a network point that acts as an entrance to another network.
- Used to connect two dissimilar networks.



### 5. Bridge:

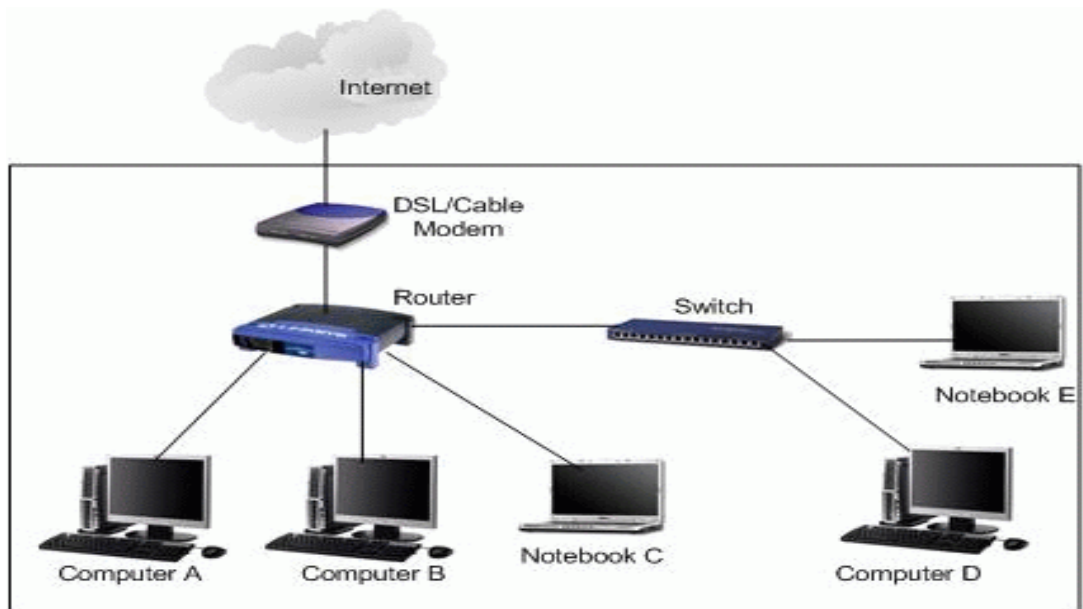
A device that connects two local-area networks (LANs), or two segments of the same LAN that use the same protocol, such as Ethernet.



### 6. Router:

A router is a device that forwards data packets along networks. A router is connected to at least two networks, commonly two LANs or WANs. Routers are located at gateways, the places where two or more networks connect.

A router acts as a dispatcher, choosing the best path for information to travel so it's received quickly.



## 7. Repeater:

Network repeaters regenerate and amplify the weak signals to transmit the information for long distance.



**8. NIC (Network Interface Card):** NIC card has a physical address of a system; this physical address known as **MAC** address.

A **MAC** address is a 6- byte address with each byte separated by a colon. First 3-bytes have Manufacturer id and last 3-bytes represent Card id.

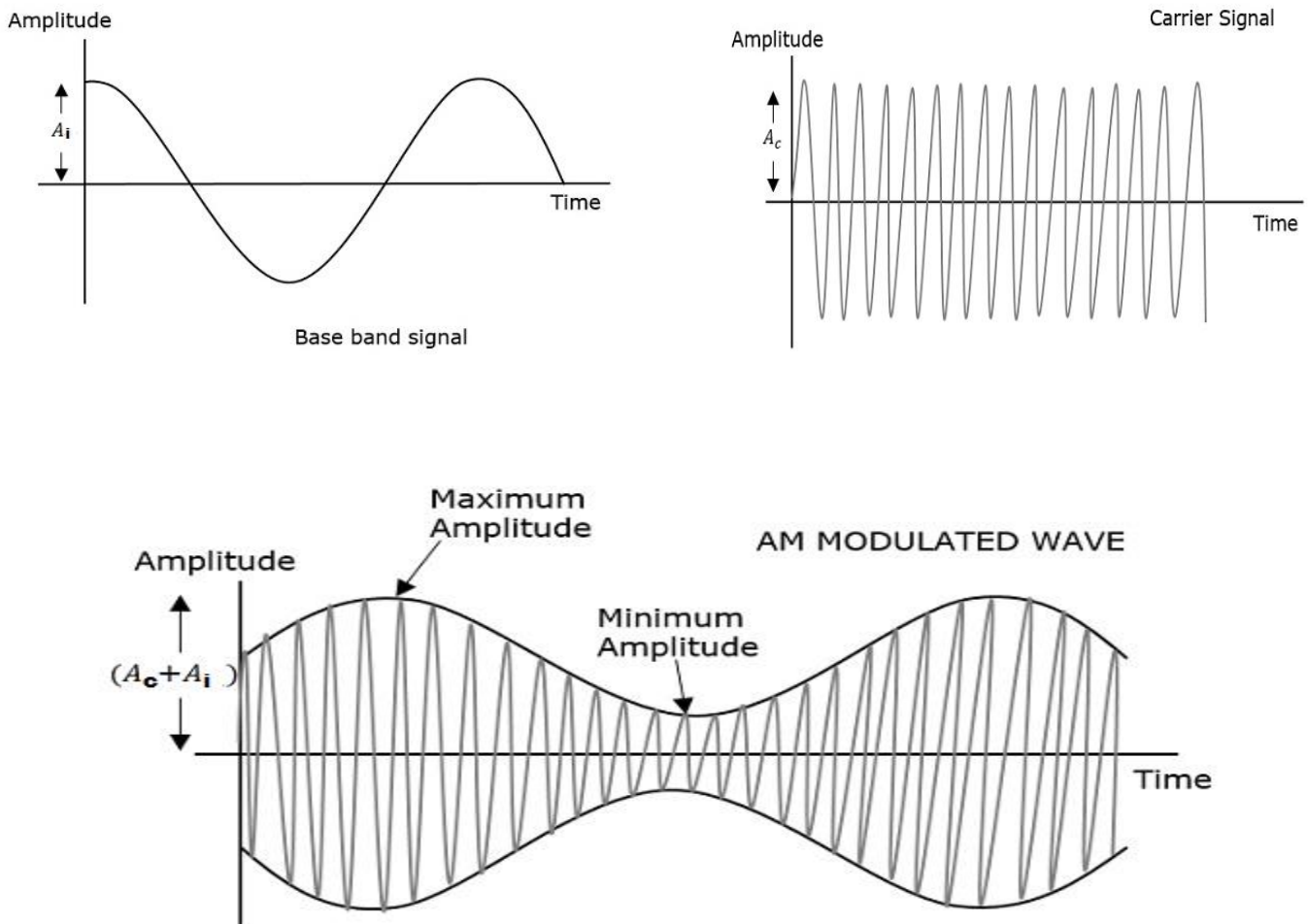
$$\underbrace{10:BE:05}_{\text{Manufacturer id}}:\underbrace{56:3F:CB}_{\text{Card id}}$$

## 2.9 AMPLITUDE MODULATION (AM):

**Modulation:** A process in which the information signal is imposed on carrier wave to transmit the information for long distance, is called modulation.

- Information signal is known as baseband signal or modulating signal. ( $A_i \rightarrow$  Amplitude of information signal)
- Common form of carrier wave is sinusoidal. ( $A_c \rightarrow$  Amplitude of carrier wave)
- Modulated wave  $A_m = A_c + A_i$

**Amplitude Modulation:** A process in which the amplitude of carrier wave is varied according to information (Baseband) signal. The frequency remains constant in Amplitude modulation.



### Applications of Amplitude Modulation (AM):

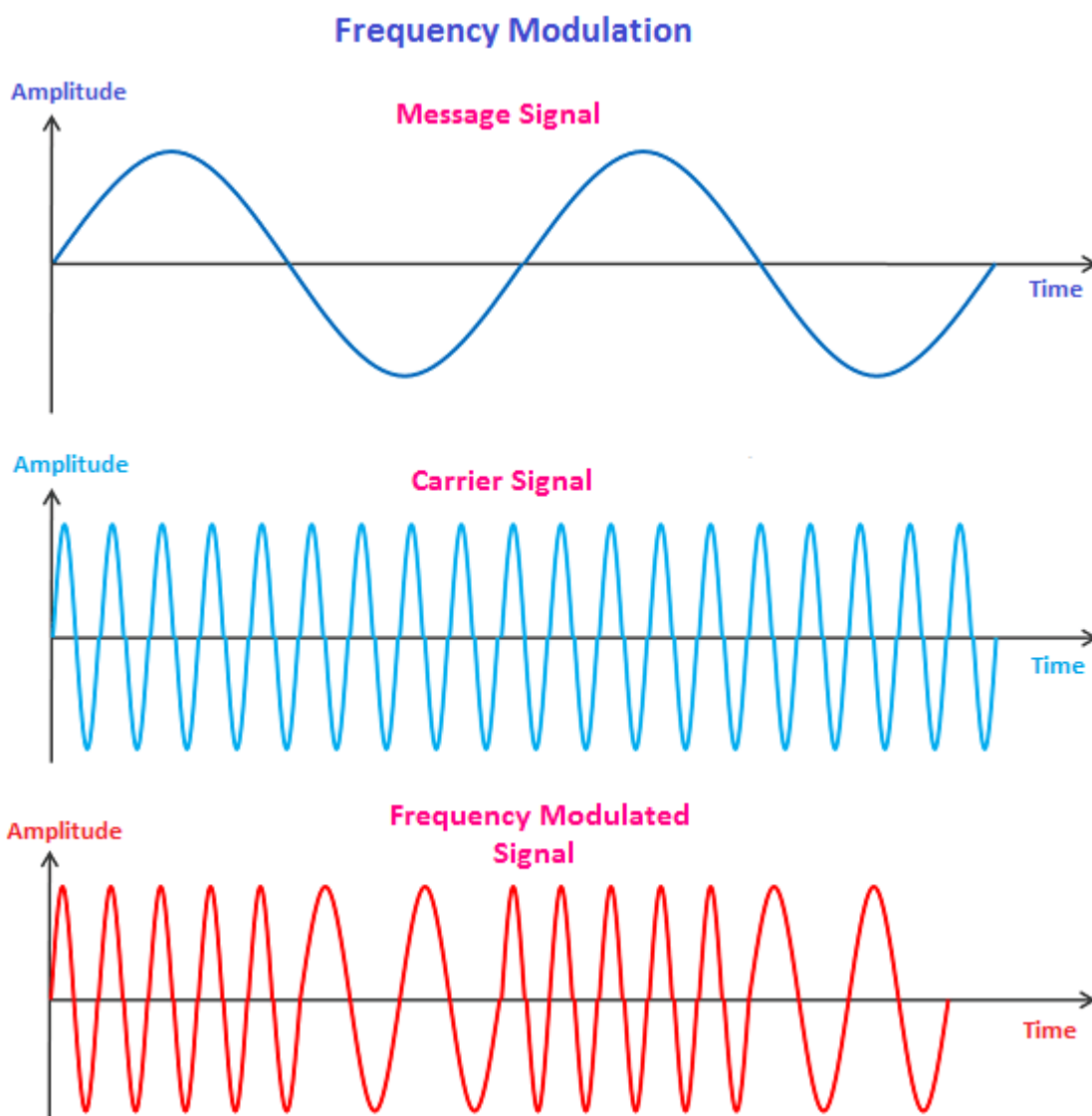
- **Broadcast Transmission:** widely used for broadcasting on the long, medium and short wave bands.
- **Air band Radio:** It is used for ground to air radio communications as well as two way radio links for ground staff as well.
- Short range wireless links

## Advantages and Disadvantages of Amplitude Modulation (AM):

Advantages	Disadvantages
It is simple to implement.	It is not efficient in terms of its power uses.
It can be demodulated using a circuit which has a few components.	It is not efficient in terms of its use of bandwidth.
AM receivers are very cheap as no specialised components are needed. So cost is reduced.	It is prone to high levels of noise

### 2.10 FREQUENCY MODULATION (FM):

**Frequency Modulation:** A process in which the frequency of carrier wave is varied according to information (Baseband) signal. The amplitude remains constant in Frequency modulation.



## **Applications of Frequency Modulation (FM) :**

- Radar
- Seismic prospecting
- EEG ( ElectroEncephaloGraphy ) monitoring of new-born's etc.
- Broadcasting of FM radio.
- It is also used in music synthesis, some systems that use video-transmission and also for magnetic tape-recording systems.

## **2.11 Collision in wireless Networks:**

**Collision:** Collision, in computer networking, is a condition that occurs when two or more computers on a network try to transmit packets at the same time. The network detects the collision of the two transmitted packets and discards them both.

### **2.11.1 Collision Avoidance in Wireless Network:**

#### **CSMA/CA : Carrier Sense Multiple Access with Collision Avoidance:**

- It is a protocol which avoids the packet collision on wireless network.
- The access point in the wireless network uses CSMA/CA protocol.
- CSMA/CA acts to prevent collisions before they happen.
- In CSMA/CA, if a node wants to send packets, it checks to be sure the channel is clear (no other node is transmitting at the time). If the channel is clear, then the packet is sent. If the channel is not clear, the node waits for a randomly chosen period of time, and then checks again to see if the channel is clear. This period of time is called the **backoff factor**, and is counted down by a backoff counter. If the channel is clear when the backoff counter reaches zero, the node transmits the packet. If the channel is not clear when the backoff counter reaches zero, the backoff factor is set again, and the process is repeated.
- RTS (Request to Send) and CTS (Clear to send) are used in CSMA/CA.



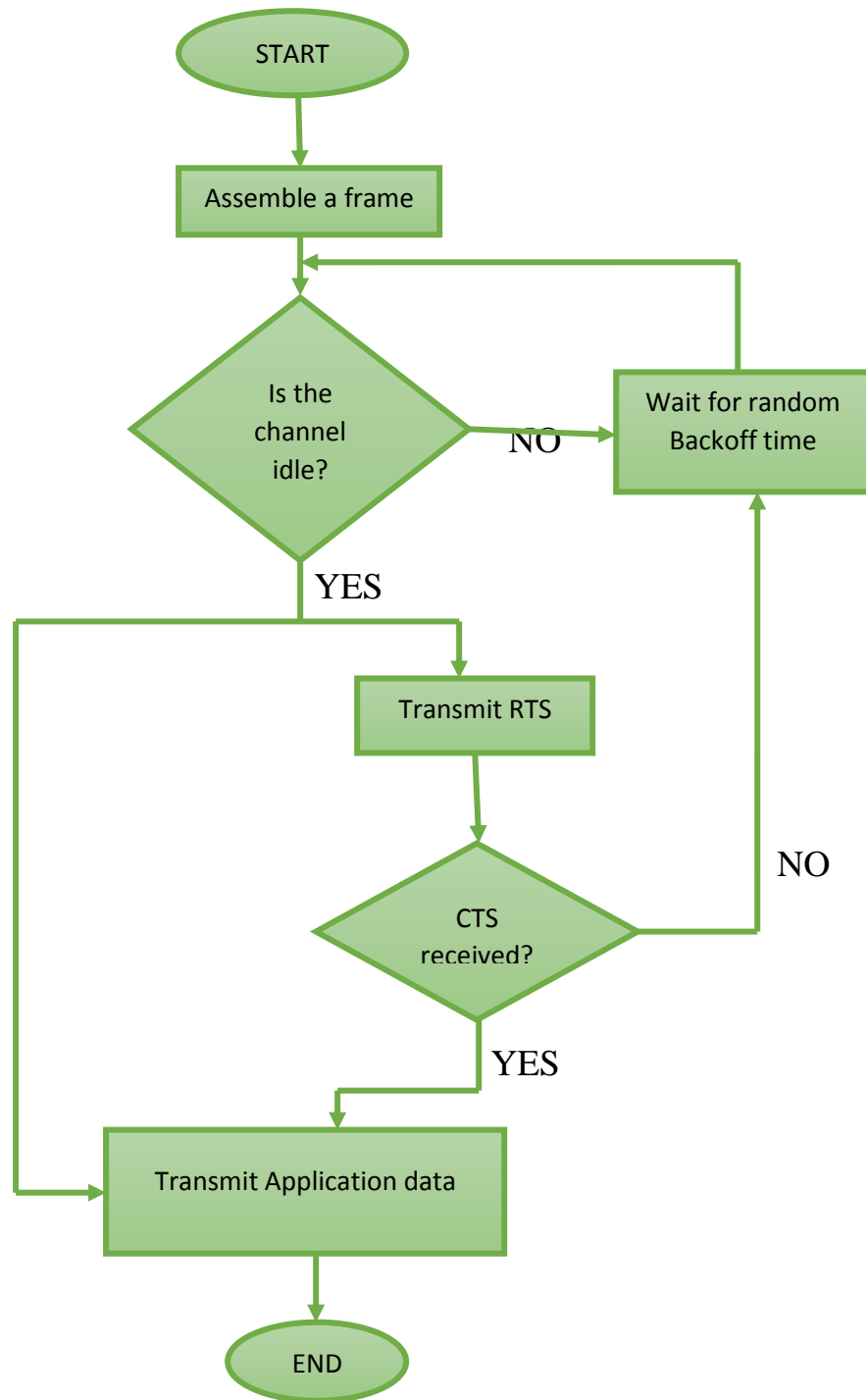


Fig: CSMA/CA Protocol

## 2.12 ERROR CHECKING:

**Error** : A condition when the receiver's information does not matches with the sender's information. During transmission, digital signals suffer from noise that can introduce errors in the binary bits travelling from sender to receiver. That means a 0 bit may change to 1 or a 1 bit may change to 0.

Error Detecting Codes (Implemented either at Data link layer or Transport Layer of OSI Model). Whenever a message is transmitted, it may get scrambled by noise or data may get corrupted. To avoid this, we use error-detecting codes which are additional data added to a given digital message to help us detect if any error has occurred during transmission of the message.

Basic approach used for error detection is the use of redundancy bits, where additional bits are added to facilitate detection of errors.

Some popular techniques for error detection are:

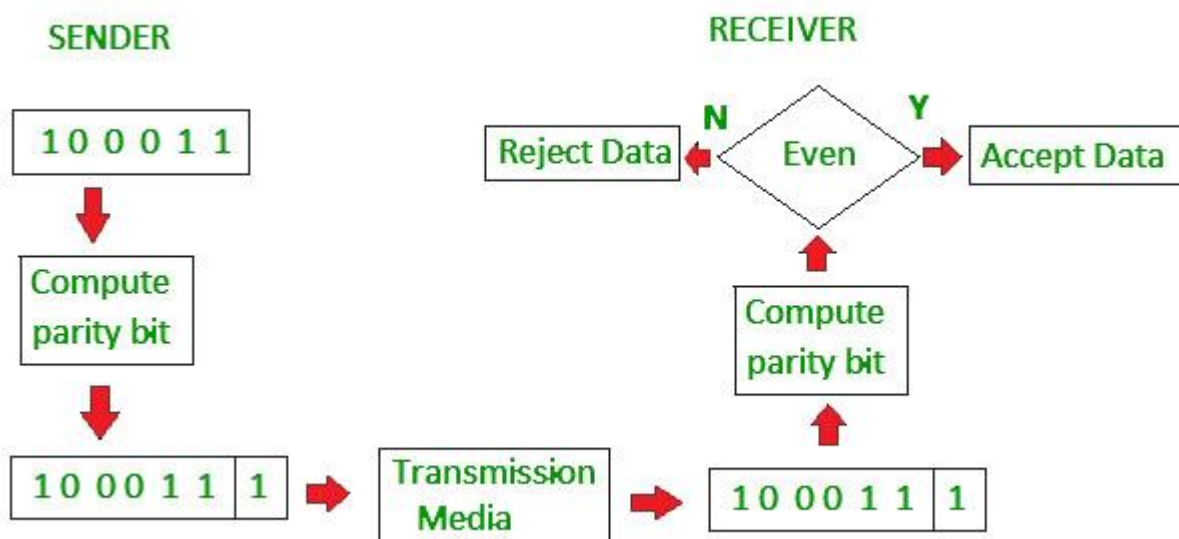
1. Simple Parity check
2. Two-dimensional Parity check
3. Checksum
4. Cyclic redundancy check

### 1. Simple Parity check

Blocks of data from the source are subjected to a check bit or parity bit generator form, where a parity of :

- 1 is added to the block if it contains odd number of 1's, and
- 0 is added if it contains even number of 1's

This scheme makes the total number of 1's even, that is why it is called even parity checking.



### 2. Two-dimensional Parity check:

Parity check bits are calculated for each row, which is equivalent to a simple parity check bit. Parity check bits are also calculated for all columns, then both are sent along with the data. At the receiving end these are compared with the parity bits calculated on the received data.

### Original Data

10011001	11100010	00100100	10000100
----------	----------	----------	----------

### Row parities

10011001	0
11100010	0
00100100	0
10000100	0
11011011	0

Column parities →

100110010	111000100	001001000	100001000	110110110
-----------	-----------	-----------	-----------	-----------

### Data to be sent

### 3. Checksum:

- In checksum error detection scheme, the data is divided into k segments each of m bits.
- In the sender's end the segments are added using 1's complement arithmetic to get the sum. The sum is complemented to get the checksum.
- The checksum segment is sent along with the data segments.
- At the receiver's end, all received segments are added using 1's complement arithmetic to get the sum. The sum is complemented.
- If the result is zero, the received data is accepted; otherwise discarded.

### Original Data

10011001	11100010	00100100	10000100
----------	----------	----------	----------

k=4, m=8

### Sender

1	10011001	
2	11100010	
	101111011	1
	01111100	
3	00100100	
	10100000	
4	10000100	
	100100100	1
	00100101	
Sum:	00100101	

Checksum: 11011010

### Receiver

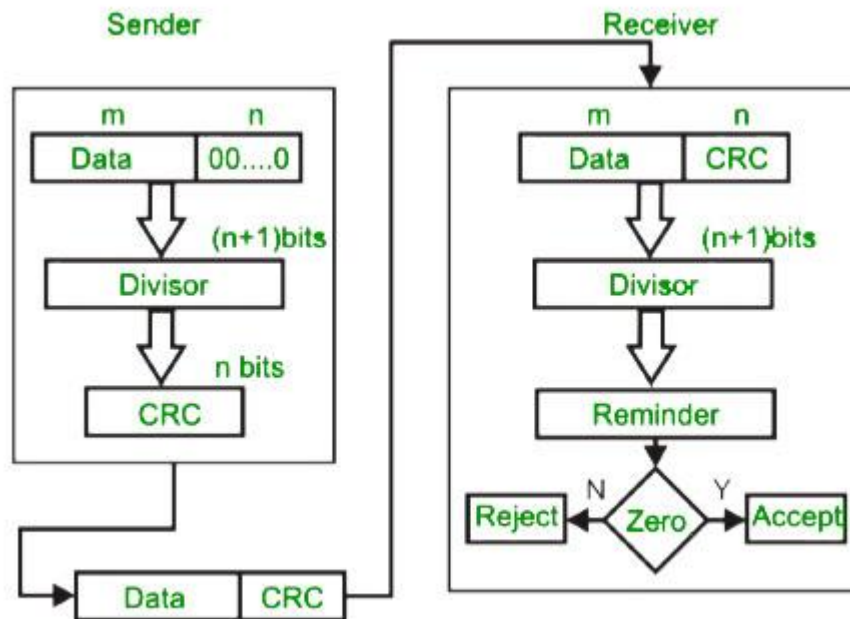
1	10011001	
2	11100010	
	101111011	1
	01111100	
3	00100100	
	10100000	
4	10000100	
	100100100	1
	00100101	
	11011010	
Sum:	11111111	

Complement: 00000000

Conclusion: Accept Data

#### 4. Cyclic redundancy check (CRC)

- Unlike checksum scheme, which is based on addition, CRC is based on binary division.
- In CRC, a sequence of redundant bits, called cyclic redundancy check bits, are appended to the end of data unit so that the resulting data unit becomes exactly divisible by a second, predetermined binary number.
- At the destination, the incoming data unit is divided by the same number. If at this step there is no remainder, the data unit is assumed to be correct and is therefore accepted.
- A remainder indicates that the data unit has been damaged in transit and therefore must be rejected.



#### Example :

original message  
1 0 1 0 0 0 0

@ means X-OR

Sender

```

1001 | 1010000000
@ 1001
-----
0011 000000
@ 1001
-----
0101 0000
@ 1001
-----
0011000
@ 1001
-----
01010
@ 1001
-----
0011

```

Message to be transmitted  
1 0 1 0 0 0 0 0 0 0  
+ 0 1 1  
-----  
1 0 1 0 0 0 0 0 1 1

Generator polynomial  
 $x^3+1$   
 $1 \cdot x^3 + 0 \cdot x^2 + 0 \cdot x^1 + 1 \cdot x^0$   
CRC generator  
1 0 0 1 4-bit

If CRC generator is of n bit then append (n-1) zeros in the end of original message

```

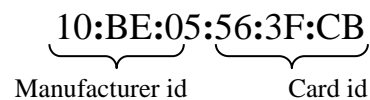
1001 | 1010000011
@ 1001
-----
0011 000011
@ 1001
-----
0101 0011
@ 1001
-----
0011011
@ 1001
-----
01001
@ 1001
-----
0000

```

Zero means data is accepted

Receiver

**2.13 MAC Address :** MAC stands for **Media Access Control**. It is a physical address of a device. A **MAC** address is a 6- byte address with each byte separated by a colon. First 3-bytes have Manufacturer id and last 3-bytes represent Card id.



**2.14 Routing: Routing** is the process of moving packets across a **network** from a source to destination.

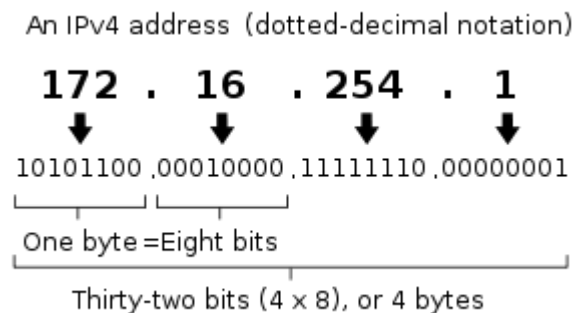
Functions of routing:

- (i) Determine optimal routing path
- (ii) Transfer the packets on a network

Routing protocols use metrics to evaluate what path will be the best for a packet to travel.

### 2.15 IP address:

Each computer has unique address over internet, is called IP address. An **IP address** is an identifier for a computer or device on a TCP/IP network.



#### *Two types:*

- I. IPv4 (32-bits or 4-bytes) : IPv4 provides the host-to-host communication between systems in the internet. IPv4 addresses are canonically represented in dot-decimal notation, which consists of four decimal numbers, each ranging from 0 to 255, separated by dots, e.g., 192.168.1.1.
- II. IPv6 (128-bits or 16-bytes)

### 2.16 Routing Table:

A routing table contains the information to forward a packet along the best path toward its destination. A router uses the routing table.

The routing table contains a list of specific routing destinations, and when the router receives a packet of data, it references the routing table to know where to send that data. The routing table may also contain information on how far each destination is from the router. In essence, a routing table is a map for the router.

There are two types of routing tables:

(i) **Static tables:** These are for static network devices. Static tables do not change unless a network administrator manually changes them.

(ii) **Dynamic tables:** Network devices build and maintain their routing tables automatically by using routing protocols to exchange information about the surrounding network topology.

Write the following command on command prompt to see the routing table:

*route print*

Network Destination	Netmask	Gateway	Interface	Metric
0.0.0.0	0.0.0.0	192.168.0.1	192.168.0.100	10
127.0.0.0	255.0.0.0	127.0.1.1	127.0.1.1	5
192.168.0.0	255.255.255.0	192.168.0.100	192.168.0.100	10
192.168.5.100	255.255.255.255	127.0.5.1	127.0.5.1	20
192.168.55.255	255.255.255.255	192.168.3.100	192.168.3.100	15

**Fig. : Routing Table**

A basic routing table includes the following information:

- **Network Destination:** The IP address of the packet's final destination.
- **Netmask :** It is used to determine the network ID from the IP address.
- **Gateway :** Router to use to reach the specified destination.
- **Interface:** The outgoing network interface (port) of device is used when forwarding the packet to the next hop or final destination
- **Metric:** Assigns a cost to each available route so that the most cost-effective path can be chosen

### 2.17 DNS (Domain Name System):

- DNS is a distributed database used by TCP/IP applications to map between hostname and IP address.
- It translates domain names (hostnames) to IP address.
- If somebody wants to send a message it is necessary to include the destination address. IP address is 32-bit integer address, which is not easy to remember for human being. So, people prefer to assign machine pronounceable, easily remembered names (host name). For this reason Domain Name System is used.



## 2.17.1 Types of DNS servers:

### Three types:

- i. Root Server
- ii. Primary Server
- iii. Secondary Server

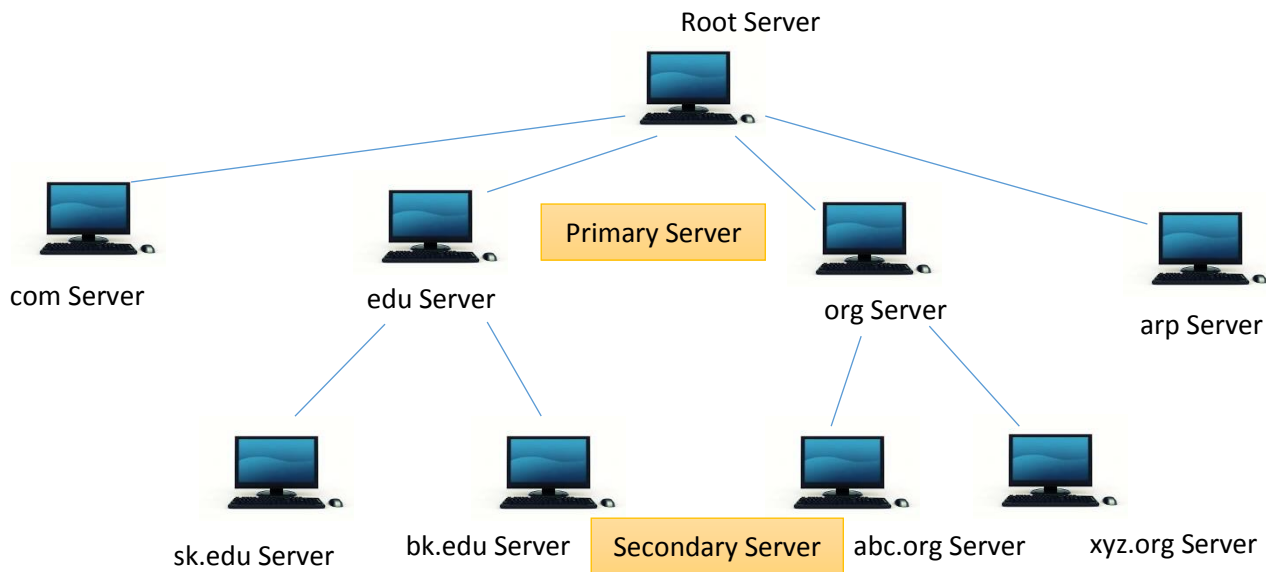


Fig. : Hierarchy of DNS servers

(i) **Root server** : A root server is a server whose zone consists of the whole tree. A root server usually does not store any information about domains but delegates its authority to other servers.

(ii) **Primary Server** : A primary server is a server that stores a file about the zone for which it is an authority. It is responsible for creating, maintaining and updating the zone file. It stores the zone file on local disk. These are also called Top-Level-Domain servers.

(iii) **Secondary Server**: A secondary server is a server that transfers the complete information about a zone from another servers (primary and secondary) and stores the file on local disk. The secondary server neither creates nor update the zone files.

DNS is divided into **three** sections:

- i. Generic Domain
- ii. Country Domain
- iii. Inverse Domain

i. Some popular generic domains:

Domain Name	Meaning
com	Commercial organizations
edu	Educational Institutions
gov	Government Institutions
mil	Military groups
net	Network Support Group
int	International organizations
org	Nonprofit organizations

ii. **Country Domain:** The country domain section uses two-character country abbreviations. Some of country domains are:

**in** (India), **us** (United States), **fr** (France), **uk** (United Kingdom)

iii. **Inverse Domain:** The inverse domain is used to map an address to a name.

**2.18 URL (Uniform Resource Locator):** The URL is a unique identifier of any resource or web page on the internet.

URL has four things:

- i. Protocol
- ii. Host Computer
- iii. Port
- iv. Path

**Syntax:**

protocol: // host : port / path

**Example:**

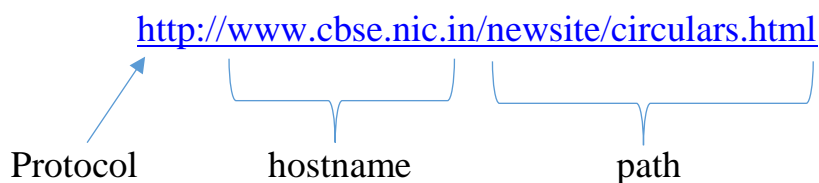


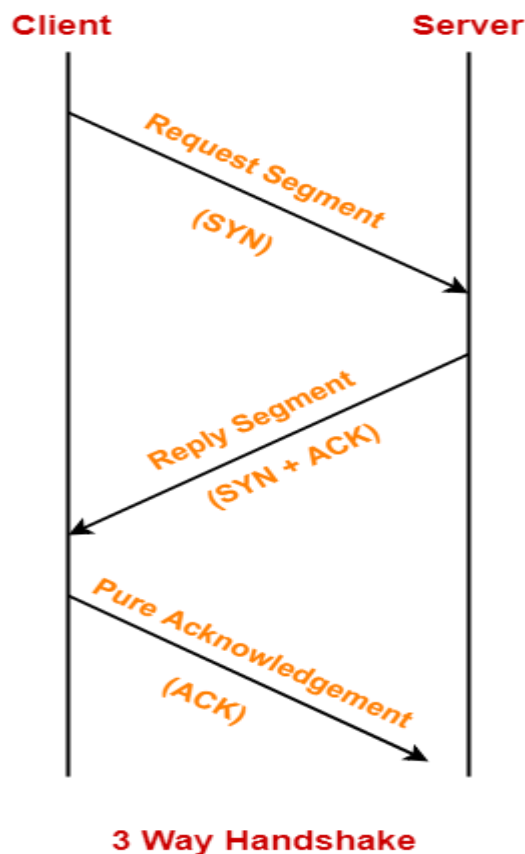
Fig : Example of URL



## 2.19 TCP (Transmission Control Protocol):

- TCP is a connection oriented, reliable data transfer protocol.
- Connection oriented transmission requires three phases:
  - Connection establishment
  - Data transfer
  - Connection termination

**2.19.1 Connection Establishment:** TCP transmits data in full duplex mode. TCP uses **three way handshaking** between sender and receiver for connection establishment. Firstly, each party must initialize communication and get approved from the other party, before any data is transferred.



## 2.19.2 TCP Retransmission:

After establishing the connection,

- Sender starts transmitting TCP segments to the receiver.
- A TCP segment sent by the sender **may get lost or damaged** on the way before reaching the receiver.
- This causes the receiver to send the acknowledgement with same ACK number to the sender or no such acknowledge is forthcoming within a reasonable time means Time-Out.
- As a result, sender retransmits the same segment to the receiver.
- This is called as **TCP retransmission**.

**2.20 Congestion Control in Network:** In this, we try to avoid traffic congestion.

**2.20.1 Congestion:** It may occur, if the load on the network is more than the capacity of the network. It occurs due to queues in the routers and switches.

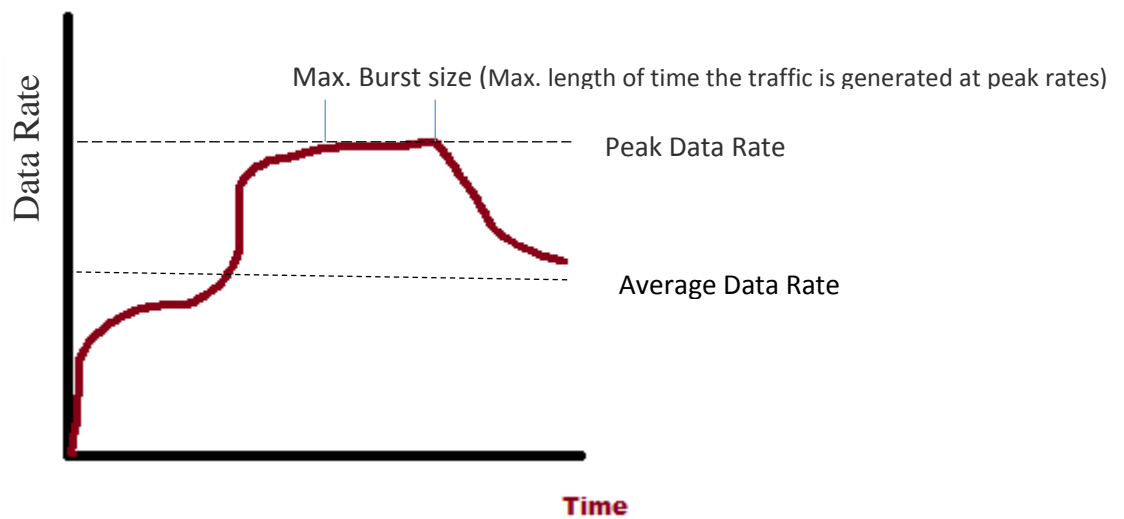
Congestion → load > capacity

Where:

Load means no. of packets sent to network.

Capacity means no. of packets a network can handle.

**2.20.2 Traffic Descriptors:**

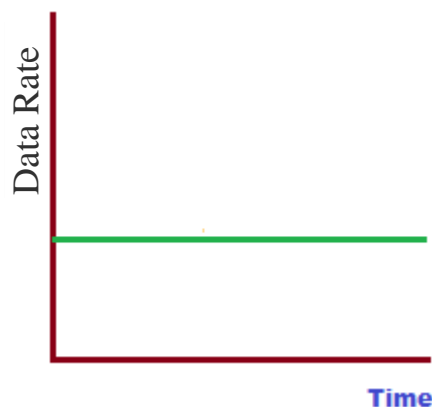


**2.20.3 Traffic profiles:**

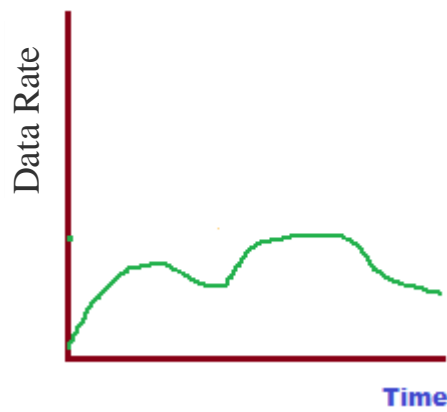
A data flow can have one of the following traffic profiles:

1. Constant Bit Rate (CBR)
2. Variable Bit Rate (VBR)
3. Bursty data

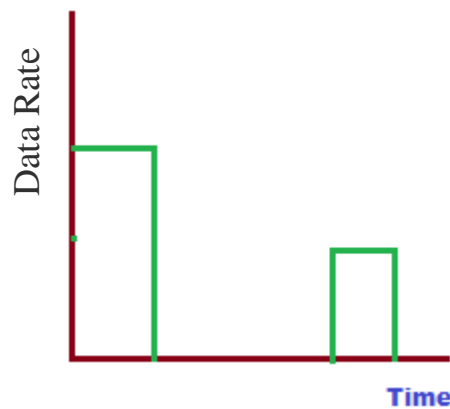
1. **Constant Bit Rate:** Not change in data rate with respect to time.



2. **Variable Bit Rate (VBR)** : Data flow changes with time.

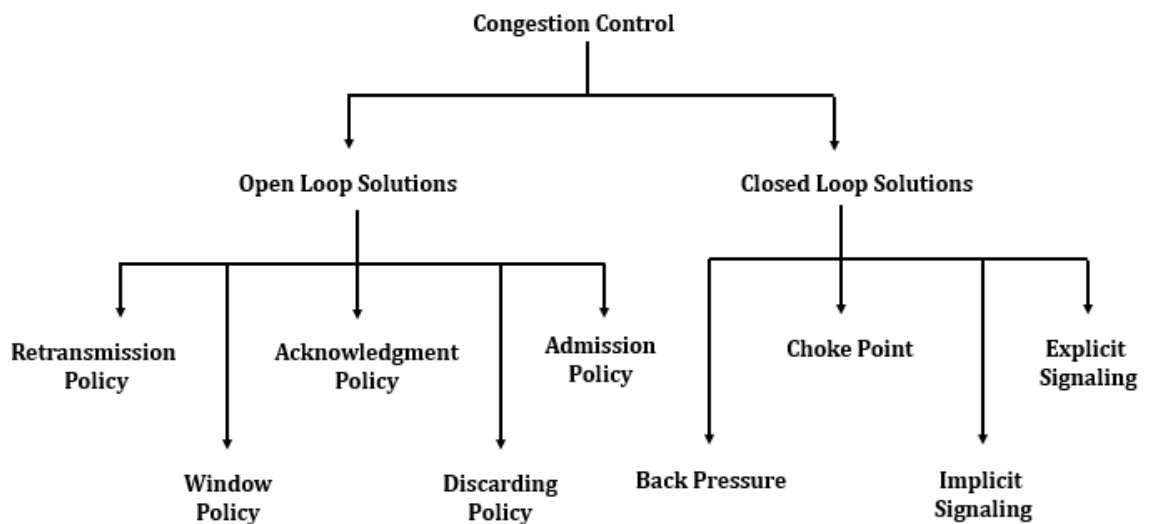


3. **Bursty data** : Data rate changes suddenly.



#### 2.20.4 Congestion Control:

Congestion Control is the techniques and mechanisms which can either prevent congestion from happening or remove congestion after it has taken place.



**Fig: Network congestion control techniques**

**I) Open Loop Congestion Control:** Policies are used to prevent the congestion before it happens.

- **Retransmission Policy:** The sender retransmits a packet, if it feels that the packet it has sent is lost or corrupted.
- **Window Policy:** Selective reject window method is used. This method sends only the lost or damaged packets.
- **Acknowledgement Policy:** By sending fewer acknowledgements we can reduce load on the network. To implement it, several approaches can be used:
  - ♣ A receiver may send an acknowledgement only if it has a packet to be sent.
  - ♣ A receiver may send an acknowledgement when a timer expires.
  - ♣ A receiver may also decide to acknowledge only N packets at a time.
- **Discarding Policy:** A router may discard less sensitive packets when congestion is likely to happen.
- **Admission Policy:** first check the resource requirement of a flow before admitting it to the network.

**II) Closed Loop Congestion Control:** Try to remove the congestion after it happens.

- **Backpressure:** In which a congested node stops receiving data from the immediate upstream node or nodes.
- **Choke Point:** Packet sent by a node to the source to inform it of congestion.
- **Implicit Signaling:** The source guesses that there is a congestion somewhere in the network from other symptoms.
- **Explicit Signaling:** The node that experiences congestion can explicitly send a signal to the source or destination, the signal is included in the packets that carry data.

## 2.21 Protocol:

- Set of rules for communication over a network.

## 2G: 2<sup>nd</sup> Generation mobile network

- Introduced in 1993.
- Technology used: Digital cellular, **GSM (Global System for Mobiles)** and **GPRS (General Packet Radio Service)**
- Data rate capacity – 64 Kbps
- Services: Calling, SMS, Web browsing, E-mail

### 3G : 3<sup>rd</sup> Generation mobile network

- Introduced in the year 2001.
- It was a digital broadband and increased speed wireless network.
- Technology used: WCDMA (Wireless Code Division Multiple Access), UMTS (Universal Mobile Telecommunication System), EDGE (Enhanced Data rates for GSM Evolution).
- Data rate capacity : 144 Kbps to 2 Mbps
- Services: Fast communication, Video calls, Smartphones, Mobile TV, GPS (Global Positioning System)

### 4G: 4<sup>th</sup> Generation mobile network

- Introduced in the year 2009.
- It is a high speed and all IP wireless network.
- Technology used : LTE (Long Term Evolution), Wi-Fi (Wireless Fidelity)
- Data rate capacity : 100 Mbps to 1 Gbps
- Services: Mobile multimedia, Global mobile support, Good quality of services, High security, very fast communication, IP TV, Voice over Internet Protocol (VoIP), VoLTE, High speed real time streaming

**Wi-Fi:** Stands for **Wireless Fidelity**. It is a facility allowing computers, smartphones, or other devices to connect to the Internet or communicate with one another wirelessly within a particular area.

#### 2.22 Basic Network Tools:

**a. traceroute:** A *traceroute* (*tracert*) is a command which traces the path from one network to another.

*Syntax:*

```
tracert hostname
```

where **hostname** is the name of the server connection you are testing.

*Example:*

```
tracert kvongcbrd.com
```

```
C:\Windows\System32\cmd.exe
C:\Windows\System32>tracert kvongcbird.com
Tracing route to kvongcbird.com [182.18.149.234]
over a maximum of 30 hops:
  0  2 ms    1 ms    1 ms    192.168.43.1
  1  *        *        *        Request timed out.
  2  88 ms   36 ms   45 ms   10.72.95.56
  3  *        87 ms   36 ms   172.25.76.221
  4  93 ms   36 ms   46 ms   172.25.76.220
  5  59 ms   38 ms   45 ms   172.25.8.87
  6  *        *        *        Request timed out.
  7  *        *        *        Request timed out.
  8  *        *        *        Request timed out.
  9  103 ms  47 ms   54 ms   218.100.48.12
 10  *        *        *        Request timed out.
 11  *        *        *        Request timed out.
 12  92 ms   84 ms   78 ms   1.6.76.237
 13  110 ms  81 ms   75 ms   103.233.124.9
 14  258 ms  105 ms  82 ms   103.233.124.21
 15  150 ms  92 ms   90 ms   103.233.124.239
 16  *        134 ms  77 ms   static-182-18-149-234.ctrls.in [182.18.149.234]
 17  87 ms   116 ms  75 ms   static-182-18-149-234.ctrls.in [182.18.149.234]

Trace complete.
C:\Windows\System32>_
```

**b. ping** : ping command to test the availability of a networking device on a network.

If you receive a reply then the device is working OK , if you don't then the device is faulty, disconnected, switched off, incorrectly configured.

*Syntax:*

ping IP address

*Example:*

ping 192.168.0.1

**c. ipconfig** : Displays all current TCP/IP network configuration values and refresh Dynamic Host Configuration Protocol and Domain Name System settings.

*Syntax:*

ipconfig

*Example:*

ipconfig

```
C:\Windows\System32\cmd.exe
C:\Windows\System32>ipconfig
Windows IP Configuration

Wireless LAN adapter Local Area Connection* 5:
    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . :
Wireless LAN adapter Local Area Connection* 2:
    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . :
Ethernet adapter Ethernet:
    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . :
Wireless LAN adapter Wi-Fi:
    Connection-specific DNS Suffix  . :
    IPv6 Address . . . . . : 2409:4052:80e:1d51:3473:df94:d31a:448f
    Temporary IPv6 Address . . . . . : 2409:4052:80e:1d51:394b:c562:a600:1acf
    Link-local IPv6 Address . . . . . : fe80::3473:df94:d31a:448f%3
    IPv4 Address . . . . . : 192.168.43.160
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : fe80::20a6:cff:fe70:a919%3
    192.168.43.1
Tunnel adapter isatap.{EDF4BDE7-46A4-4B72-9F6A-365A7C12887B}:
    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . :
```

**d. nslookup :** The **nslookup** (which stands for *name server lookup*) command is a network utility program used to obtain information about internet servers. It finds name server information for domains by querying the Domain Name System.

*Syntax:*

nslookup domainname

*Example:*

nslookup www.kvsangathan.nic.in

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Windows\System32>nslookup www.kvsangathan.nic.in
Server:    Unknown
Address:   192.168.43.1

Non-authoritative answer:
Name:     kvsangathan.nic.in
Address:  164.100.229.94
Aliases:  www.kvsangathan.nic.in
```

**e. whois :** whois is a simple **command**-line utility that allows you to easily get information about a registered domain. It automatically connect to the right **WHOIS** server, according to the top-level domain name, and retrieve the WHOIS record of the domain. It supports both generic domains and country code domains.

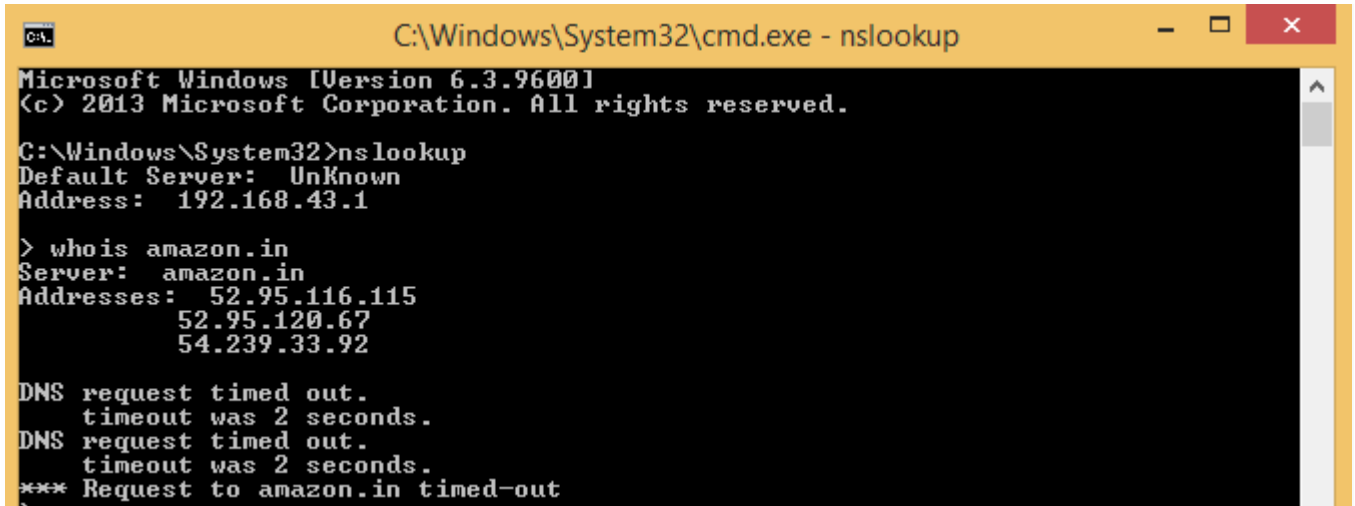
*Syntax:*

whois domainname

*Example:*

whois amazon.in

*Note:* Run this command after nslookup or along with nslookup command.



```
C:\Windows\System32\cmd.exe - nslookup
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Windows\System32>nslookup
Default Server: UnKnown
Address: 192.168.43.1

> whois amazon.in
Server: amazon.in
Addresses: 52.95.116.115
           52.95.120.67
           54.239.33.92

DNS request timed out.
           timeout was 2 seconds.
DNS request timed out.
           timeout was 2 seconds.
*** Request to amazon.in timed-out
```

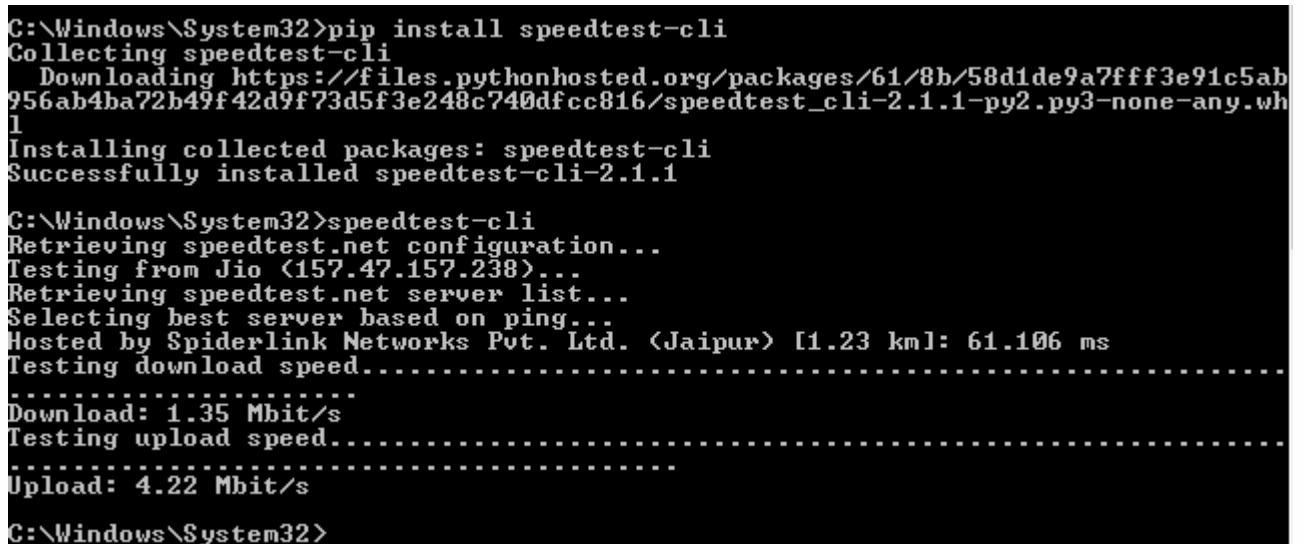
**f. speed-test :** To test the internet speed using command prompt, firstly you need to install the following using pip command.

pip install speedtest-cli

After successfully installation of above, run the following command.

speedtest-cli

*Example:*



```
C:\Windows\System32>pip install speedtest-cli
Collecting speedtest-cli
  Downloading https://files.pythonhosted.org/packages/61/8b/58d1de9a7fff3e91c5ab956ab4ba72b49f42d9f73d5f3e248c740dfcc816/speedtest_cli-2.1.1-py2.py3-none-any.whl
Installing collected packages: speedtest-cli
Successfully installed speedtest-cli-2.1.1

C:\Windows\System32>speedtest-cli
Retrieving speedtest.net configuration...
Testing from Jio (157.47.157.238)...
Retrieving speedtest.net server list...
Selecting best server based on ping...
Hosted by Spiderlink Networks Pvt. Ltd. (Jaipur) [1.23 km]: 61.106 ms
Testing download speed.....
.....
Download: 1.35 Mbit/s
Testing upload speed.....
.....
Upload: 4.22 Mbit/s

C:\Windows\System32>
```

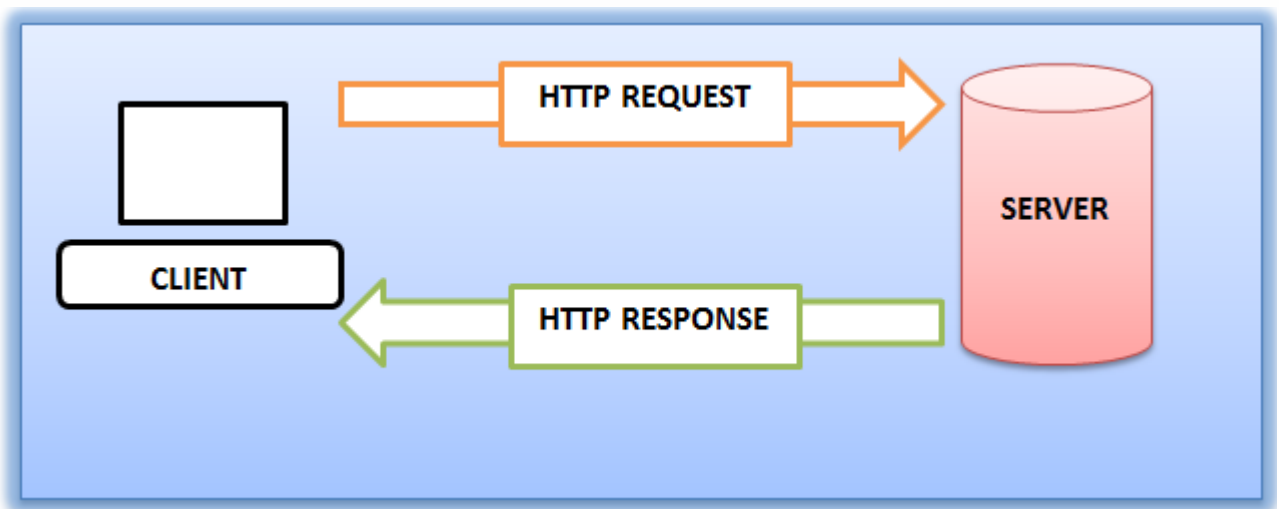


## 2.23 Application Layer:

The application layer contains a variety of protocols that are commonly needed by users. It provides a user interface that enables users to access the network and various services.

**2.23.1 HTTP (HyperText Transfer Protocol):** It is a protocol used to transfer the hypertext pages over internet.

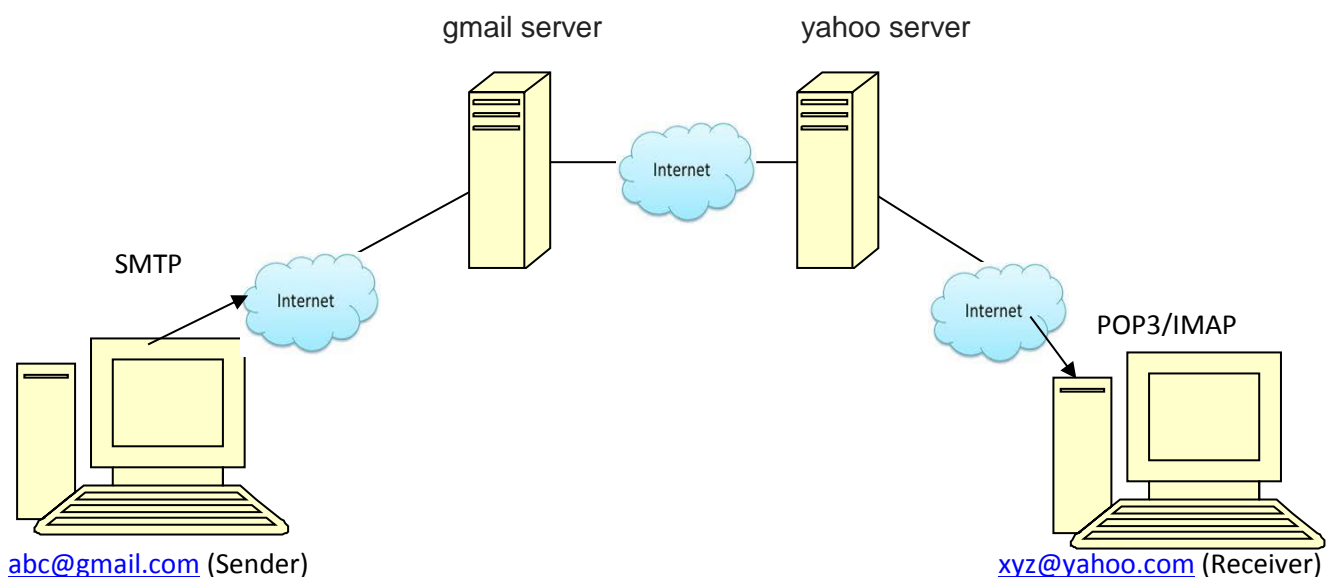
HTTP is implemented in two program: Client program and Server program. Both the programs executing on different end system, communicate to each other by exchanging HTTP messages.



**Fig : Client-Server HTTP communication**

## 2.23.2 E-Mail (Electronic Mail):

E-Mail is a method to send the messages in digital form. E-mail is a message that may contain text, files, images, or other attachments sent through a network to a specified individual or group of individuals.



**SMTP** (Simple Mail Transfer Protocol) is a protocol which is used to transfer the e-mail from sender side. This protocol is known as push protocol.

**POP3** (Post Office Protocol version 3): This protocol is used to access e-mail from the server to receiver. This protocol is known as pull protocol.

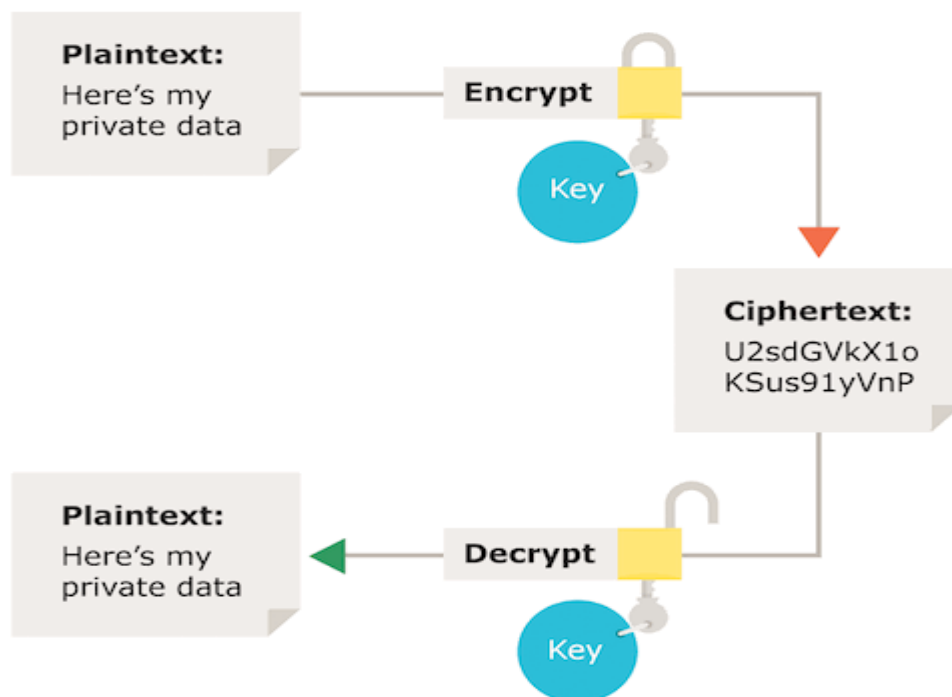
**IMAP** (Internet Message Access Protocol): It is a standard email protocol that stores email messages on a mail server, but allows the end user to view and manipulate the messages as though they were stored locally on the end user's computing device.

## 2.24 Secure Communication:

### 2.24.1 Encryption and Decryption:

**Encryption:** To convert a plain text (readable) into cipher text (Non-readable).

**Decryption:** To convert a cipher text (Non-readable) into plain text (Readable).



**Fig: Encryption and decryption**

### 2.24.2 HTTPS (HyperText Transfer Protocol Secure):

It is a variant of HTTP that adds a layer of security on the data in transit through a secure socket layer (SSL).

HTTPS enables encrypted communication and secure connection between a remote user and the primary web server.

HTTPS encrypts every data packet in transition using SSL encryption technique to avoid intermediary hackers and attackers to extract the content of the data.

Example:

<https://pythonschoolkvs.wordpress.com/>

**Digital certificate:** A digital certificate is issued by a certification authority (CA). Digital certificates are used with self-signatures and message encryption. Digital certificates are also known as public key certificates or identity certificates.

## **2.25 Network Applications:**

**2.25.1 Remote Desktop:** Remote desktop is a program that allows a user to connect to a computer in another location, see that computer's desktop and interact with it as if it were local. Example: Team Viewer, Any desk, Virtual Network Computing, GoToMyPC etc.

**2.25.2 Remote login:** A login that allows a user's computer to connect to a host computer via a network and to interact with that host.

**2.25.3 FTP (File Transfer Protocol):** It is used to transfer files from one computer to another computer.

**2.25.4 SCP (Session Control Protocol):** It is a session layer protocol. It creates multiple light-duty connections from a single TCP connection. It uses Secure Shell (SSH) for data transfer.

**2.25.5 SSH (Secure Shell):** Secure Shell (SSH) is a cryptographic network protocol for operating network services securely over an unsecured network. Secure Shell provides strong authentication and encrypted data communications between two computers connecting over an open network such as the internet.

**2.25.6 VoIP (Voice over Internet Protocol):** It is the phone service over the Internet.

**2.25.7 NFC (Near Field Communication):** NFC is a short-range high frequency wireless communication technology that enables the exchange of data between devices over about a 10 cm distance.

### 3.1 Django Introduction:

- Django is a free and open source web application framework written in Python.
- It is based on Model-View-Template (MVT) framework. Here Model means tables, View means logic and Template means HTML files to display the contents interactive.
- It is used for developing the web application.
- Django provides the concept of 'reusability'.

### 3.2 Installation of Django:

- Install Python 3.5 or upper version
- Connect your computer with internet and open Command Prompt & type the following command to install Django:

**pip install django**

- For verification of installation of Django. Open Command prompt and type the following command:

**django-admin --version**

It will display the installed version in computer.

### 3.3 Develop the Project and Web application:

#### 3.3.1 Create a Project:

- After Installation of Django, open command prompt window.
- Go to the directory where you want to create the project. Here, we shall create the project in D:\WebApplications directory.
- To create the project type the following command in command prompt.

**D:\WebApplications>django-admin startproject Vehicle**

```
D:\WebApplications>django-admin startproject Vehicle
```

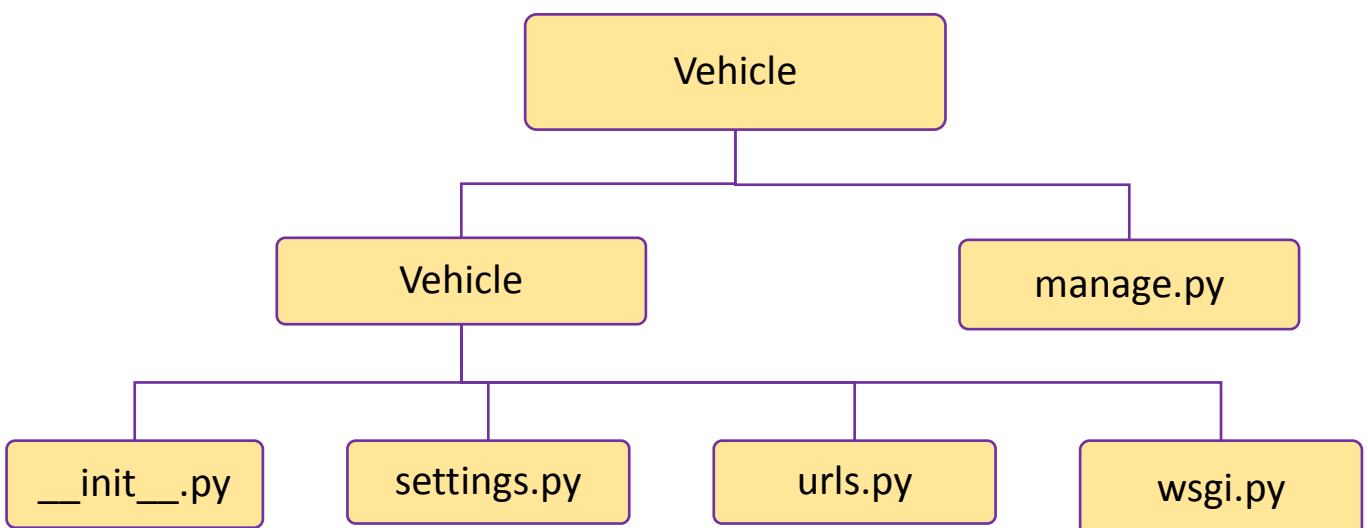
Here Vehicle is the name of Project. A folder with the name Vehicle will be created in D:\ WebApplications directory.

- **Vehicle** folder will have two things:
  - Vehicle folder (inner folder)
  - manage.py



- Vehicle subfolder has the following files:
  - `__init__.py`
  - `settings.py`
  - `urls.py`
  - `wsgi.py`

Now the directory structure is like this:



### 3.3.2 Run the Django Server:

- Now to start the Django server, Type following command in command prompt:  
**D:\WebApplications\Vehicle>python manage.py runserver**

```
D:\WebApplications\Vehicle>python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

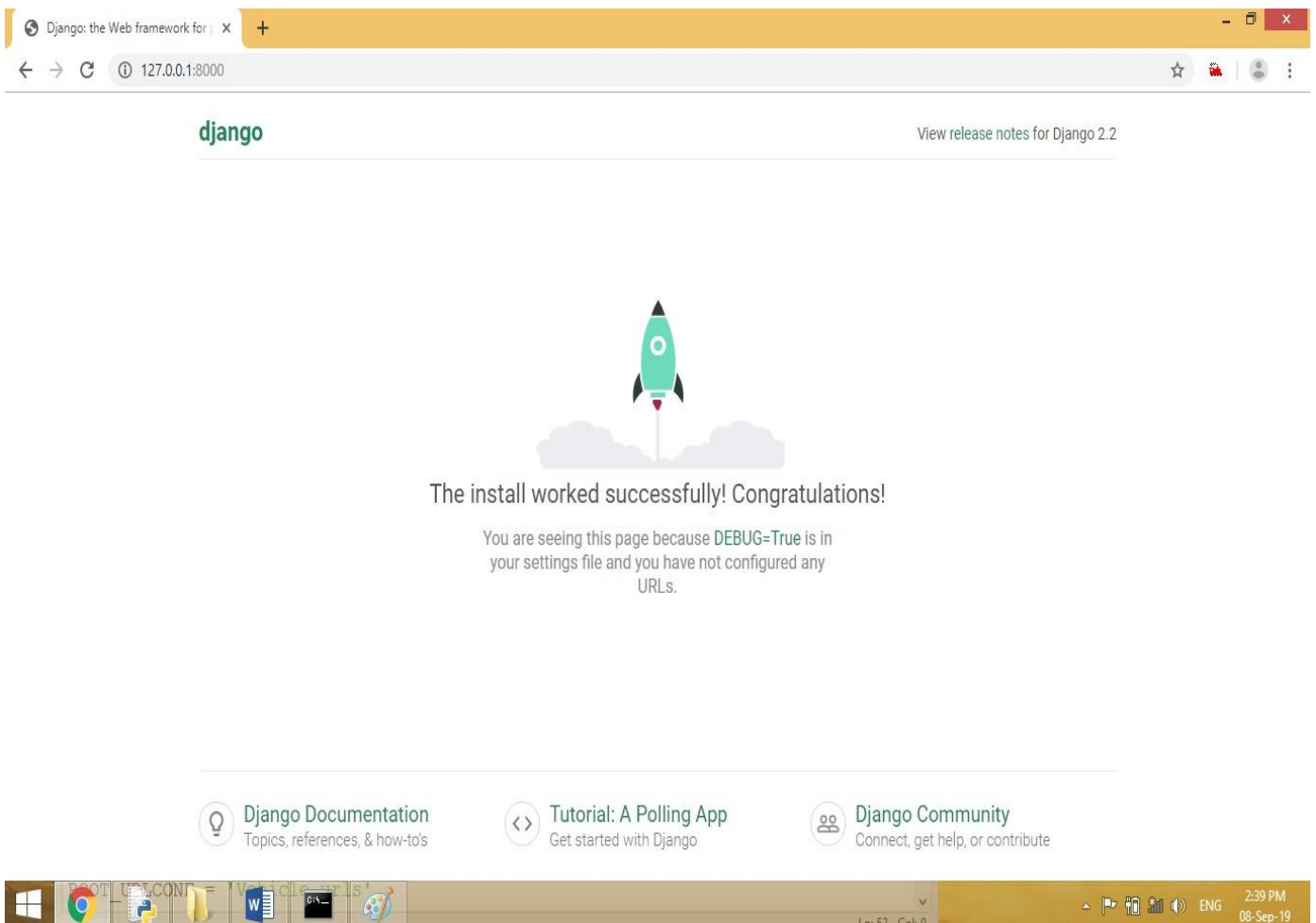
System check identified no issues (0 silenced).

You have 17 unapplied migration(s). Your project may not work properly until you
apply the migrations for app(s): admin, auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
September 08, 2019 - 14:35:13
Django version 2.2.1, using settings 'Vehicle.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

- After successfully execution of above command, it will provide an address, which is given below:

<http://127.0.0.1:8000/>

Copy the above link and paste it in web browser. The following screen will appear in your web browser. It means Django server is working properly.



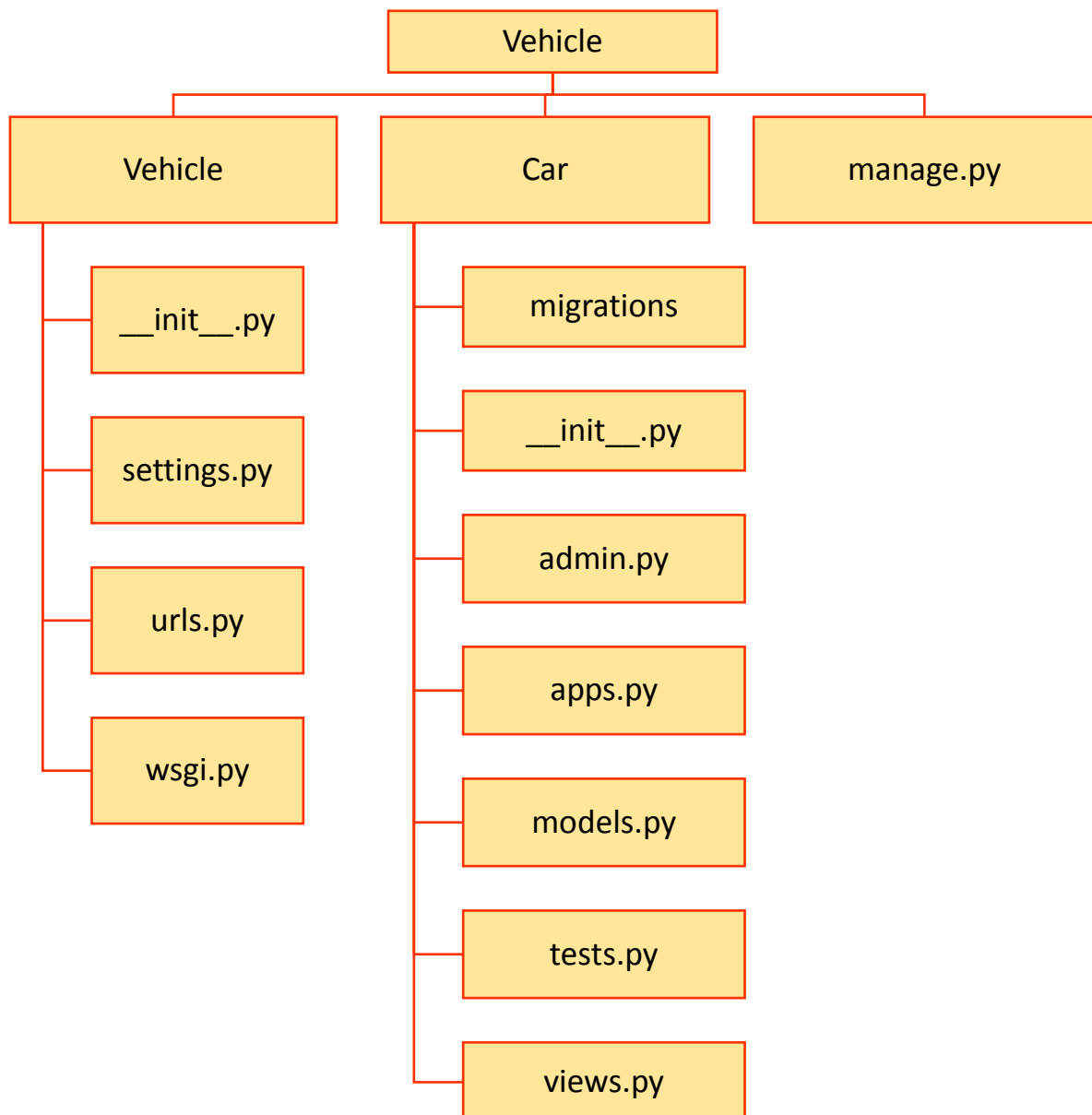
### 3.3.3 Create an Application:

- Now, open the command prompt. Go to the directory D:\WebApplications\Vehicle and create the application under Vehicle Project. Here the application name is Car. To create this application, type the following command :

**D:\WebApplications\Vehicle>python manage.py startapp Car**

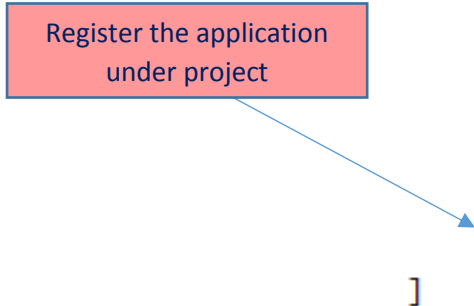
```
D:\WebApplications\Vehicle>python manage.py startapp Car
```

- After this, a folder will be created under Vehicle folder (Outer folder) and the structure of directory will look like this:



### 3.3.4 Register the Application in Project:

- Now register the created application **Car** under the project **Vehicle**. For this, we have to open **settings.py** file under **Vehicle** folder.



```
INSTALLED_APPS = [  
    'django.contrib.admin',  
    'django.contrib.auth',  
    'django.contrib.contenttypes',  
    'django.contrib.sessions',  
    'django.contrib.messages',  
    'django.contrib.staticfiles',  
    'Car',  
]
```

The application **Car** has registered under the project **Vehicle**.

### 3.3.5 CRUD Operations in Web Application:

- In this web application we will perform the CRUD (Create, Read, Update, Delete ) operations.
- We shall store the data of different types of cars and perform the CRUD operations with the data stored in database.
- To Store the data, we will use **sqlite3** database system which is in-built in Django. So no need to install it separately.

### 3.3.6 Create the model in Django:

- Open the file **models.py** under **Car** web application to create a model and its fields. Models are considered as tables in MVT framework. An application may have multiple models. Write the following code in **models.py**:

```
from django.db import models  
  
# Create your models here.  
class CARTABLE(models.Model):  
    cbrand = models.CharField(max_length=50)  
    cdom = models.CharField(max_length=20)  
    cprice = models.CharField(max_length=20)  
    class Meta:  
        db_table='ctable'
```

The name of the database table to use for the model



- Here, **Meta class** is simply an inner class. In Django, the use of Meta class is simply to provide metadata to the ModelForm class.
- Django provides the facility of forms. It means we need not create any HTML form for table fields. Django denotes the table fields as a form. Now, create a file **forms.py** under Car folder.

This PC > Local Disk (D:) > WebApplications > Vehicle > Car >

Name	Date modified	Type
__pycache__	08-Sep-19 3:14 PM	File
migrations	08-Sep-19 2:42 PM	File
__init__	08-Sep-19 1:15 PM	Pyt
admin	08-Sep-19 1:15 PM	Pyt
apps	08-Sep-19 1:15 PM	Pyt
<b>forms</b>	08-Sep-19 3:18 PM	Pyt
models	08-Sep-19 3:14 PM	Pyt
tests	08-Sep-19 1:15 PM	Pyt
views	08-Sep-19 1:15 PM	Pyt

- Write the following code in forms.py file.

```
from django import forms
from Car.models import CARTABLE

class CARFORM (forms.ModelForm):
    class Meta:
        model = CARTABLE
        fields = "__all__"
```

- Now migrate the models in database. For this, we have to run the following command in command prompt:

**D:\WebApplications\Vehicle>python manage.py makemigrations**

```
D:\WebApplications\Vehicle>python manage.py makemigrations
Migrations for 'Car':
  Car\migrations\0001_initial.py
  - Create model CARTABLE
```

This command will create the model and make the changes if any.

- Now run another command to migrate the created model. In command prompt window, write the following command:
- **D:\WebApplications\Vehicle>python manage.py migrate**

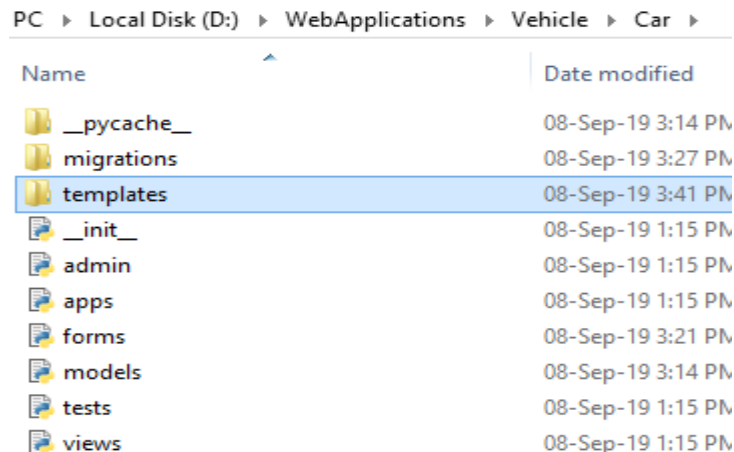
```
D:\WebApplications\Vehicle>python manage.py migrate
```

Now the created model is stored in the file **0001\_initial.py** under the folder migration in Car application.

- A table named as **ctable** has been created in database. It has following fields: (id, cbrand, cdom, cprice). id field is automatically created in Django, which has its own system generated values.

### 3.3.7 Creation of Templates:

- Create a folder named as **templates** under the application Car. templates folder will have the .html files.



Name	Date modified
__pycache__	08-Sep-19 3:14 PM
migrations	08-Sep-19 3:27 PM
templates	08-Sep-19 3:41 PM
__init__	08-Sep-19 1:15 PM
admin	08-Sep-19 1:15 PM
apps	08-Sep-19 1:15 PM
forms	08-Sep-19 3:21 PM
models	08-Sep-19 3:14 PM
tests	08-Sep-19 1:15 PM
views	08-Sep-19 1:15 PM

- Make the entry of created folder in **settings.py** file. The meaning of this step that we are informing the project that there is an application with the name **Car**, whose all .html pages are inside templates folder.

```
TEMPLATES = [  
    {  
        'BACKEND': 'django.template.backends.django.DjangoTemplates',  
        'DIRS': ['templates'],  
        'APP_DIRS': True,  
        'OPTIONS': {  
            'context_processors': [  
                'django.template.context_processors.debug',  
                'django.template.context_processors.request',  
                'django.contrib.auth.context_processors.auth',  
                'django.contrib.messages.context_processors.messages',  
            ],  
        },  
    ],  
]
```

- In our application, we will have index.html, edit.html and show.html files under templates folder.

### ➤ HTML coding for index.html:

In this file we'll create an HTML form to take the values from user.

(CREATE operation)

```
<html>
<head><title>Index Page</title></head>
<body>
<center><strong>
<h1>CAR DATA</h1>
</strong></center>
<center>

<form method = "POST" action="/home/">
{% csrf_token %}
<h2>Enter Details</h2> <br>

Car Brand: {{ form.cbrand }} <br>
Date of Manufacturing: {{ form.cdom }} <br>
Price: {{ form.cprice }} <br>

<button type="submit" >Submit</button>
</form>

</center> </body> </html>
```

**In above code**, csrf\_token is inbuilt feature of django application. csrf stands for Cross Site Request Forgery. It provides easy-to-use protection against Cross Site Request Forgeries.

### ➤ HTML coding for show.html:

In this file, we shall display the records on the webpage. To display records, we have to fetch the data from database and display them. ( READ Operation)

```
<html>
<head>
<title>Show Car Details</title>
</head>
<body>
<center><strong><u><h1>CAR DETAILS</h1></u></strong></center>
<center>
<table width=60% border=6">
<tr>
<th>Car Brand</th>
<th>Date of Manufacturing</th>
<th> Price</th>
```

```

<th colspan="2">Action</th>
</tr>

{% for ctable in ctables %}           # use of for loop to fetch multiple records
<tr>
<td>{{ ctable.cbrand }}</td>
<td>{{ ctable.cdom }}</td>
<td>{{ ctable.cprice }}</td>
<td> <a href = "/edit/{{ ctable.id }}">Edit </a></td>
<td> <a href="/delete/{{ ctable.id }}">Delete</a> </td>
</tr>
{% endfor %}
</table> </center>
<center><a href="/home">Add New Record</a></center>
</body> </html>

```

➤ **HTML coding for edit.html:**

To update the record. (UPDATE operation).

```

<html>
<head> <title> Edit Car Details </title></head>
<body>
<center>
<form method= "POST" action="/update/{{ ctable.id }}/" >
{% csrf_token %}
<h3>Update Details</h3>
Car Brand : <input type="text" name="cbrand" id="cbrand" value={{ ctable.cbrand }}>
<br>
Date of Manufacture :
<input type="text" name="cdom" id="cdom" value={{ ctable.cdom }}> <br>

Price:<input type="text" name="cprice" id="cprice" value={{ ctable.cprice }}> <br>
<button type="submit"> Update </button>
</form></center>
</body>
</html>

```

### 3.3.8 Write the logic in views.py:

- Now, open **views.py** under the application **Car**.
- **views.py** file will work as a controller. In this file we would write the logics in the form of functions and call the views according to their actions. So, it is an important file.
- Write the following code in **views.py** file:

```
from django.shortcuts import render, redirect
from Car.forms import CARFORM
from Car.models import CARTABLE

# Create your views here.
def create(request):          #CREATE operation
    if request.method == "POST":
        form = CARFORM(request.POST)
        if form.is_valid( ):
            try:
                form.save( )
                return redirect('/show')
            except:
                pass
    else:
        form = CARFORM( )
        return render(request,"index.html", {'form':form})

def show(request):          #READ operation
    ctables = CARTABLE.objects.all()
    return render(request,"show.html", {'ctables':ctables})

def edit(request, id):
    ctable = CARTABLE.objects.get(id=id)
    return render(request, "edit.html", {'ctable': ctable})

def update(request, id):    #UPDATE operation
    ctable = CARTABLE.objects.get(id=id)
    form = CARFORM(request.POST, instance=ctable)
    if form.is_valid( ):
        form.save( )
        return redirect('/show')
    return render(request, "edit.html", {'ctable': ctable})

def delete (request, id):   #DELETE operation
    ctable = CARTABLE.objects.get(id=id)
    ctable.delete( )
    return redirect('/show')
```

### 3.3.9 Write the logic in views.py

- Open **urls.py** under **Vehicle** subfolder and refer the views. Write the following code in **urls.py** file.

```
from django.contrib import admin
from django.urls import path
from Car import views

urlpatterns = [
    path('admin/', admin.site.urls),
    path('home/', views.create),
    path('show/', views.show),
    path('edit/<int:id>', views.edit),
    path('update/<int:id>', views.update),
    path('delete/<int:id>', views.delete),
]
```

- Now open command prompt and Write following command:  
**D:\WebApplications\Vehicle>python manage.py runserver**

```
D:\WebApplications\Vehicle>python manage.py runserver
```

After execution of above command, the following address will appear on command prompt:

<http://127.0.0.1:8000/>

Copy it and paste it into web browser. Press Enter. It will display the name of all urls. Now write the address of the show page. Means, write the address given below:

<http://127.0.0.1:8000/show>

## OUTPUT:

The screenshot shows a web browser window with the title 'Show Car Details' and the URL '127.0.0.1:8000/show/'. The main content is a table with the following structure:

Car Brand	Date of Manufacturing	Price	Action
-----------	-----------------------	-------	--------

Below the table, there is a link labeled 'Add New Record'.

The Windows taskbar at the bottom shows the time as 6:06 PM on 08-Sep-19.

### OUTPUT-1 Empty Table, No Records

The screenshot shows a web browser window with the title 'Index Page' and the URL '127.0.0.1:8000/home/'. The main content is a form titled 'CAR DATA' with the sub-heading 'Enter Details'.

The form contains the following fields:

- Car Brand:
- Date of Manufacturing:
- Price:

A 'Submit' button is located below the Price field.

The Windows taskbar at the bottom shows the time as 6:09 PM on 08-Sep-19.

### OUTPUT-2 Adding the New Record

### CAR DETAILS

Car Brand	Date of Manufacturing	Price	Action	
Maruti	10/10/2015	750000	<a href="#">Edit</a>	<a href="#">Delete</a>
Hyundai	12/06/2017	675000	<a href="#">Edit</a>	<a href="#">Delete</a>
Audi	03/04/2018	4500000	<a href="#">Edit</a>	<a href="#">Delete</a>
Tata	28/02/2019	825000	<a href="#">Edit</a>	<a href="#">Delete</a>

[Add New Record](#)

### OUTPUT-3 Records in the table after addition

#### Update Details

Car Brand :   
Date of Manufacture :   
Price:

### OUTPUT-4 Update the Record



### 3.4 CSV and Flat Files:

CSV (Comma Separated Values). A csv file is a type of plain text file that uses specific structuring to arrange tabular data. csv is a common format for data interchange as it is compact, simple and general. Each line of the file is one line of the table. csv files have .csv as file extension.

Let us take a **data.csv** file which has the following contents:

Roll No., Name of student, stream, Marks

1, Anil, Arts, 426

2, Sujata, Science, 412

As you can see each row is a new line, and each column is separated with a comma. This is an example of how a CSV file looks like.

To work with csv files, we have to import the **csv module** in our program.

#### 3.4.1 Read a CSV file:

To read data from a CSV file, we have to use reader( ) function.

The reader function takes each row of the file and make a list of all columns.

#### CODE:

```
import csv
with open('C:\\data.csv','rt') as f:
    data = csv.reader(f)    #reader function to generate a reader object
    for row in data:
        print(row)
```

#### OUTPUT:

```
['Roll No.', 'Name of student', 'stream', 'Marks']
```

```
['1', 'Anil', 'Arts', '426']
```

```
['2', 'Sujata', 'Science', '412']
```

#### 3.4.2 Write data to a CSV file:

When we want to write data in a CSV file you have to use writer( ) function. To iterate the data over the rows (lines), you have to use the writerow( ) function.

#### CODE:

```
import csv
with open('C:\\data.csv', mode='a', newline='') as file:
    writer = csv.writer(file, delimiter=',', quotechar='"')
    #write new record in file
    writer.writerow(['3', 'Shivani', 'Commerce', '448'])
    writer.writerow(['4', 'Devansh', 'Arts', '404'])
```

## OUTPUT:

```
['Roll No.', 'Name of student', 'stream', 'Marks']  
['1', 'Anil', 'Arts', '426']  
['2', 'Sujata', 'Science', '412']  
['3', 'Shivani', 'Commerce', '448']  
['4', 'Devansh', 'Arts', '404']
```

When we shall open the file in notepad (Flat file) then the contents of the file will look like this:

```
Roll No.,Name of student,stream,Marks  
1,Anil,Arts,426  
2,Sujata,Science,412  
3,Shivani,Commerce,448  
4,Devansh,Arts,404
```

### 3.5 Creation of Web Application with csv files:

- Create a project. In this example, project name is Stock.

```
D:\WebApplications>django-admin startproject Stock
```

- Create an application under this project. Here name of applications is ComputerParts.

```
D:\WebApplications\Stock>python manage.py startapp ComputerParts
```

- Now, register the application in **settings.py** under Stock folder.
- Open **view.py**, and write following code:

```
from django.shortcuts import render  
from django.http import HttpResponse  
import csv  
  
# Create your views here.  
  
def WriteData(request):  
    response=HttpResponse(content_type='text/csv')  
    response['Content-Disposition']='attachment;filename="Test.csv"'  
    writer=csv.writer(response)  
    writer.writerow(['Part_ID','Name','Quantity','Price'])  
    writer.writerow(['101','Mouse','150','200'])  
    writer.writerow(['',' ',' ',' '])  
    writer.writerow(['102','Keyboard','80','450.50'])  
    writer.writerow(['103','Monitor','20'])  
    writer.writerow(['104','HDD',' ','4500'])  
    return response
```

- Open **urls.py**, and write following code:

```
from django.contrib import admin
from django.urls import path
from ComputerParts import views

urlpatterns = [
    path('admin/', admin.site.urls),
    path('csv', views.WriteData),
]
```

- Now open command prompt and run the server.

```
D:\WebApplications\Stock>python manage.py runserver
```

- After successful execution of above, write the given address in web browser.

<http://127.0.0.1:8000/csv>

- A csv file with the name Test.csv will be created automatically.

### 3.6 MySQL database connectivity with Python:

- Install python
- Install MySQL
- Install MySQL Driver using following command: (In Command Prompt):  
***pip install mysql-connector***  
*Note: Make sure your computer is connected with internet.*
- To verify, whether the connector is properly installed or not, open python shell and type the following command:  
>>>import mysql.connector  
>>>

If the command successfully runs (without any error), then the MySQL connector is successfully installed.

- Now, open MySQL and check the current user, by typing the following command in MySQL:

```
SELECT current_user();
```

```
Enter password: *****
Welcome to the MySQL monitor.
Your MySQL connection id is 4
Server version: 5.0.41-communit
Type 'help;' or '\h' for help.

mysql> select current_user();
+-----+
| current_user() |
+-----+
| root@localhost |
+-----+
1 row in set (0.00 sec)

mysql>
```

- Connect MySQL database with python. For this, open Python IDLE and write the following code in python file.

**CODE:**

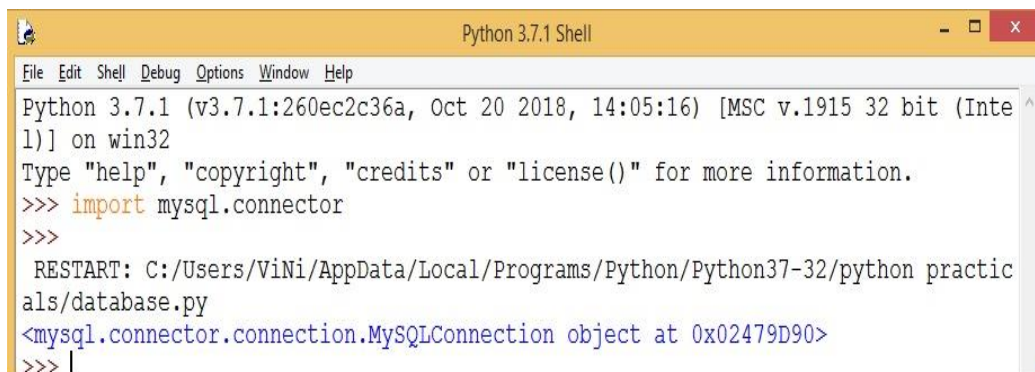
```
import mysql.connector

demodb=mysql.connector.connect(host="localhost",user="root", passwd="computer")

print(demodb)
```

If you get the following output, then the connection made successfully.

**OUTPUT:**



```
Python 3.7.1 Shell
File Edit Shell Debug Options Window Help
Python 3.7.1 (v3.7.1:260ec2c36a, Oct 20 2018, 14:05:16) [MSC v.1915 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> import mysql.connector
>>>
RESTART: C:/Users/ViNi/AppData/Local/Programs/Python/Python37-32/python practicals/database.py
<mysql.connector.connection.MySQLConnection object at 0x02479D90>
>>> |
```

- After making successful connection between python and MySQL, now create a database in MySQL through python. For that, write the following code in python:

```
import mysql.connector

demodb = mysql.connector.connect(host="localhost", user="root", passwd="computer")

democursor=demodb.cursor( )

democursor.execute("CREATE DATABASE EDUCATION")
```

- After successful execution of the following code, check in MySQL, whether EDUCATION database has been created or not. for that, write the following command in MySQL:

```
mysql> show databases ;
+-----+
| Database |
+-----+
| information_schema |
| education |
| mysql |
| test |
+-----+
4 rows in set (0.01 sec)

mysql>
```

- If you want to check the created database through python, write the following python code to show the present databases in MySQL.

```
import mysql.connector
demodb = mysql.connector.connect(host="localhost", user="root", passwd="computer")
democursor=demodb.cursor()
democursor.execute("SHOW DATABASES")
for i in democursor:
    print(i)
```

## OUTPUT:

```
RESTART: C:\Users\Vinil\
als\database.py
('information_schema',)
('education',)
('mysql',)
('test',)
>>> |
```

Here, we can see that EDUCATION database has been created.

## 3.7 Create a table in database:

### CODE:

```
import mysql.connector
demodb = mysql.connector.connect(host="localhost", user="root",
passwd="computer", database="EDUCATION")
democursor=demodb.cursor( )
democursor.execute("CREATE TABLE STUDENT (admn_no int primary key, sname
varchar(30), gender char(1), DOB date, stream varchar(15), marks float(4,2))")
```

To verify the table created or not, write the following code in python:

```
import mysql.connector
```

```
demodb = mysql.connector.connect(host="localhost", user="root",
passwd="computer", database="EDUCATION")
democursor = demodb.cursor( )
democursor.execute ("show tables")
for i in democursor:
    print(i)
```

#### OUTPUT:

```
>>>
  RESTART: C:\Users\ViNi\i
als\database.py
('student',)
>>>
```

### 3.8 Insert the data in the table:

```
import mysql.connector
demodb = mysql.connector.connect(host="localhost", user="root",
passwd="computer", database="EDUCATION")
democursor=demodb.cursor( )
democursor.execute("insert into student values (%s, %s, %s, %s, %s, %s)", (1245,
'Arush', 'M', '2003-10-04', 'science', 67.34))
demodb.commit( )
```

### 3.9 Fetch the data from table:

```
import mysql.connector
demodb = mysql.connector.connect(host="localhost", user="root",
passwd="computer", database="EDUCATION")
democursor=demodb.cursor( )
democursor.execute("select * from student")
for i in democursor:
    print(i)
```

#### OUTPUT:

```
RESTART: C:\Users\ViNi\AppData\Local\Programs\Python\Python37-32\python practic
als\database.py
(1245, 'Arush', 'M', datetime.date(2003, 10, 4), 'science', 67.34)
(1356, 'swati', 'F', datetime.date(2005, 11, 23), 'arts', 72.6)
>>> |
```

### 3.10 Update the record:

```
import mysql.connector
demodb = mysql.connector.connect(host="localhost", user="root",
passwd="computer", database="EDUCATION")
democursor=demodb.cursor( )
democursor.execute("update student set marks=55.68 where admn_no=1356")
demodb.commit( )
```

### 3.11 Delete a record:

```
import mysql.connector
demodb = mysql.connector.connect(host="localhost", user="root",
passwd="computer", database="EDUCATION")
democursor=demodb.cursor( )
democursor.execute("delete from student where admn_no=1356")
demodb.commit( )
```

### 3.12 SQL COMMANDS:

SQL commands are categorized into four sub languages:

- (i) Data Definition Language (DDL)
- (ii) Data Manipulation Language (DML)
- (iii) Transaction Control Language (TCL)
- (iv) Data Control Language (DCL)

(i) **Data Definition Language (DDL):** It consist the commands to create objects such as tables, views, indexes etc. in the database.

**COMMANDS:** CREATE, ALTER, DROP, RENAME, TRUNCATE

(ii) **Data Manipulation Language (DML):** It is used for queries. It allows you to perform data manipulation e.g. retrieval, insertion, deletion, modification of data stored in database.

**COMMANDS:** SELECT, INSERT, UPDATE, DELETE

(iii) **Transaction Control Language (TCL):** This language allows you to manage and control the transaction.

**COMMANDS:** COMMIT, ROLLBACK

- NOTE:- Savepoint is also used in TCL.

(iv) **Data Control Language (DCL):** This language is used to control data and access to the databases. It is used for protecting the data from unauthorized access.

**COMMANDS:** GRANT, REVOKE

### 3.13 SQL QUERIES:

In MySQL, to create new tables or query, the user must specify the database. This database is called current database.

**To check the available database in MySQL:-**

SHOW databases;

**To access the database:-**

Syntax:- USE database-name;

Example: USE Education;

A database has tables. Tables have rows and columns.

**To show the available tables in the database:-**

SHOW tables;

**CREATING DATABASE:** In MySQL we can create the databases using CREATE DATABASE statement.

**Syntax:** CREATE DATABASE <database-name>;

**Example:** CREATE DATABASE Bank;

**OPENING DATABASE:**

**Syntax:** USE <database-name>;



**Example:** USE Bank;

**DROPPING DATABASES:** To remove the entire database we use the DROP DATABASE statement.

**Syntax:** DROP DATABASE <database-name>;

**Example:** DROP DATABASE Bank;

### Creating Tables in MySQL:

Tables are created using CTREATE TABLE command.

A table has rows and columns. The column name must be specified along the data type. Each table must have at least one column.

**Syntax:**

CREATE TABLE <table-name>(<column-name> <data-type> (size), <column-name> <data-type> (size), <column-name> <data-type> (size) );

**Example:**

CREATE TABLE EMPLOYEE(Ecode int(6), Ename varchar(30), Dept varchar(30), city varchar(25), sex char(1), DOB Date, salary float(12,2) );

Ecode	Ename	Dept	City	Sex	Dob	Salary

*Fig. : EMPLOYEE*

### Viewing a table structure:

DESC <table-name>;

**Example:** DESC EMPLOYEE;

**Or we can write**

DESCRIBE EMPLOYEE;

### Inserting data into table:

The rows(tuples) are added to relations(tables) using INSERT command.

**Syntax:**

INSERT INTO <table-name>[<column-name>] VALUES (<value1>, <value2>, .....);

**Example:** INSERT INTO EMPLOYEE VALUES(1001, 'Amit', 'production', 'Ahemedabad', 'M', '1988-08-22', 20000.00);

The **into** clause specifies the target table and the value clause specifies the data to be added to the new row of the table.

While inserting data into tables, following points should be taken care of:

- ✓ Character data should be enclosed within single quotes.
- ✓ NULL values are given as NULL, without any quotes.
- ✓ If no data is available for all the columns then the column list must be included, following the table name. Example: INSERT INTO EMPLOYEE(Ecode, Ename, salary) VALUES(1001, 'Amit', 20000.00);

After inserting the data , we created the following table:

<b>Ecode</b>	<b>Ename</b>	<b>Dept</b>	<b>City</b>	<b>Sex</b>	<b>DOB</b>	<b>Salary</b>
1001	Amit	Production	Ahemedabad	M	1988-08-22	38000.00
1002	Sanjeev	Marketing	New Delhi	M	1990-09-05	32000.00
1003	Imran	RND	Surat	M	1989-01-01	40000.00
1004	Harish	RND	Jaipur	M	1988-01-20	40050.00
1005	Neha	Marketing	Guwahati	F	1985-04-15	35000.00
1006	Dheeraj	Production	Mumbai	M	1984-03-02	39000.00
1007	Vikram	Marketing	Shimla	M	1990-10-10	31000.00
1008	Ashok	Marketing	Patna	M	1980-09-11	40000.00
1009	Priyanka	RND	Gurgaon	F	1990-07-23	40000.00
1010	Seema	Production	New Delhi	F	1989-05-16	37000.00
1011	Manish	Marketing	Guwahati	M	1980-02-07	39050.00

**Table: EMPLOYEE**

## **DROPPING A TABLE:**

To remove the entire structure of the table completely, we use the DROP TABLE command.

### **Syntax:**

DROP TABLE <table-name>;

**Example:**

DROP TABLE EMPLOYEE;

**MANIPULATING DATA OF A TABLE:-**

- (i) Retrieving data : SELECT command
- (ii) Inserting data : INSERT command
- (iii) Deleting data : DELETE command
- (iv) Modification : UPDATE command

(i) **SELECT Command:-** A SELECT command retrieves information from the database.

(ii) **INSERT Command:-** This command is used to insert the data in table.

NOTE:- We have already discussed about this command.

(iii) **DELETE Command:-** It means delete the information from the table. This command is used to remove the rows from the table.

- Specific rows are deleted when you specify the WHERE clause.
- All rows in the table are deleted if you omit the WHERE clause.

The syntax is:

```
DELETE FROM <table-name>
WHERE <condition> ;
```

Example:- Delete those rows whose department is production.

```
Solution:  DELETE FROM EMPLOYEE
           WHERE dept='production';
```

Example:- Delete all the records of EMPLOYEE table having salary less than 35000.

```
Solution:  DELETE FROM EMPLOYEE
           WHERE Salary<35000;
```

(iv) **UPDATE command:-** Values of a single column or group of columns can be updated.

The syntax is:-

```
UPDATE table-name
SET column_name=value
WHERE condition;
```

Example:- Change the salary of Vikram to 36000.

```
Solution:      UPDATE EMPLOYEE
                SET salary=36000
                WHERE Ename='Vikram';
```

Example:- Increase every employee salary by 10%.

```
Solution: UPDATE EMPLOYEE
                SET salary=salary+(salary*0.1);
```

## MAKING SIMPLE QUERIES:-

In SQL queries, we use three clauses mostly:-

- (i) SELECT:- What to select
- (ii) FROM:- Which Table
- (iii) WHERE:- Condition to satisfy

### (i) SELECT:-

A SELECT command is used to retrieve information from a table.

- ❖ If you want to select the all columns from a table, then we use the asterisk(\*) in SELECT clause.

Example: - SELECT \* FROM EMPLOYEE;

- ❖ To display specific columns of the table by specifying the column names, separated by commas.

Example: - SELECT Ecode, Ename, salary  
FROM EMPLOYEE;

### (ii) FROM:-

A FROM clause, specifies the table name that contains the columns.

### (iii) WHERE:-

A WHERE clause, specifies the condition.

```
Syntax:- SELECT column_name
           FROM table_name
           WHERE condition;
```

## SOME IMPORTANT POINTS:-

- ✓ SQL statements are not case sensitive.
  - ✓ To end the SQL command, we write the semicolon(;) at the end of a line followed by <ENTER>.
- **Selecting All Columns:-** To select all the columns, we use asterisk (\*) in SELECT statement.  
Example:-  

```
SELECT *
FROM EMPLOYEE;
```
  - **Selecting Specific Columns:-** To display the specific columns of the table, write columns name, separated by commas.  
Example:-  

```
SELECT Ecode, Ename, salary
FROM EMPLOYEE;
```
  - **Eliminating redundant data:-** The DISTINCT keyword eliminates duplicate rows from the results. DISTINCT is used in SELECT statement.  
Example:-  

```
SELECT DISTINCT(Dept)
FROM EMPLOYEE;
```

## ALL keyword:-

SQL allows us to use the keyword ALL to specify explicitly that duplicates are not removed.

Example:  

```
SELECT ALL Dept
FROM EMPLOYEE;
```

## Arithmetic Operations:-

The SELECT clause may also contain arithmetic expressions involving the operators +, -, \*, and / operating on constants or attributes.

Example:- Find the new salary of every employee increased by 25%.

```
SELECT Ename,salary,salary*0.25
FROM EMPLOYEE;
```

**COLUMN ALIAS:-** You can change a column heading by using a column alias.

Example:-  

```
SELECT Ename as Name
```

FROM EMPLOYEE;

### Examples of Queries:-

1. List the name and department of those employees where department is production.

Solution:-  
SELECT Ename, Dept  
FROM EMPLOYEE  
WHERE Dept='production';

2. Find the name and salary of those employees whose salary is more than 20000.

Solution:-  
SELECT Ename, salary  
FROM EMPLOYEE  
WHERE salary > 20000;

3. Display the name of those employees who live in New Delhi.

Solution:-  
SELECT Ename, city  
FROM EMPLOYEE  
WHERE city='New Delhi';

4. List the name of female employees in EMPLOYEE table.

Solution:-  
SELECT Ename  
FROM EMPLOYEE  
WHERE sex='F';

5. Display the name and department of those employees who work in surat and salary is greater than 25000.

Solution:-  
SELECT Ename, Dept  
FROM EMPLOYEE  
WHERE city='surat' and salary > 25000;

Or we can write this query in another way:

Solution:-  
SELECT Ename, Dept  
FROM EMPLOYEE  
WHERE city='surat' && salary > 25000;

6. Display the name of those female employees who work in Mumbai.

Solution:-  
SELECT Ename  
FROM EMPLOYEE  
WHERE sex='F' and city='Mumbai';

7. Display the name of those employees whose department is marketing or RND.

Solution:-  
SELECT Ename  
FROM EMPLOYEE  
WHERE Dept='marketing' OR Dept='RND';

8. List the name of employees who are not males.

```
Solution:-  SELECT  Ename, Sex
            FROM    EMPLOYEE
            WHERE   sex!= 'M';
```

### 3.14 Queries for special operators:-

- (i) BETWEEN :- Between two values
- (ii) IN :- Match a value in the list
- (iii) LIKE :- Match a character pattern
- (iv) IS NULL :- Value is null.

#### (i) BETWEEN :-

Example:- Find the name and salary of those employees whose salary is between 35000 and 40000.

```
Solution:-  SELECT  Ename, salary
            FROM    EMPLOYEE
            WHERE   salary BETWEEN 35000 and 40000;
```

Or we can write this query in another way:

```
SELECT  Ename, salary
FROM    EMPLOYEE
WHERE   salary > 35000 and salary < 40000;
```

#### (ii) IN :-

Example:- Find the name of those employees who live in guwahati, surat or jaipur city.

```
Solution:-  SELECT  Ename, city
            FROM    EMPLOYEE
            WHERE   city IN('Guwahati', 'Surat', 'Jaipur');
```

#### (iii) LIKE :-

% :- It represents any sequence of zero or more characters.

\_ :- Represents any single character.

Example:- Display the name of those employees whose name starts with 'M'.

```
Solution:-  SELECT  Ename
            FROM    EMPLOYEE
            WHERE   Ename LIKE 'M%';
```

Example:- Display the name of those employees whose department name ends with 'a'.

```
Solution:-    SELECT  Ename
              FROM    EMPLOYEE
              WHERE   Dept LIKE '%a';
```

Example:- List the name of employees whose name having 'e' as the second character.

```
Solution:-    SELECT  Ename
              FROM    EMPLOYEE
              WHERE   Ename LIKE '_e%';
```

#### (iv) IS NULL :-

Example:- List the name of employees not assigned to any department.

```
Solution:-    SELECT  Ename
              FROM    EMPLOYEE
              WHERE   Dept IS NULL;
```

### 3.15 ORDER BY clause:-

You can sort the result in a specific order using ORDER BY clause. The sorting can be done either in ascending or descending order. The default order is ascending.

Example:- Display the list of employees in descending order of employee code.

```
Solution:-    SELECT  *
              FROM    EMPLOYEE
              ORDER BY  ecode DESC;
```

Example:- Display the employee code, name in ascending order of salary.

```
Solution:-    SELECT  Ecode, Ename, salary
              FROM    EMPLOYEE
              ORDER BY  salary asc;
```

Suppose that we wish to list the entire EMPLOYEE relation in descending order of salary. If several employees have the same salary, we order them in ascending order by employee code. We express this query in SQL as follows:-

```
SELECT  *
FROM    EMPLOYEE
ORDER BY  salary desc, Ecode asc;
```

### 3.16 Aggregate Functions:

Aggregate functions are functions that take a collection of values as input and return a single value. SQL offers five types of aggregate functions:-



- (i) Avg() :- To findout the average
- (ii) Min() :- Minimum value
- (iii) Max() :-Maximum value
- (iv) Sum() :-To calculate the total
- (v) Count() :- For counting

**NOTE:** - The input to sum ( ) and avg( ) must be a collection of numbers, but the other functions can operate on non numeric data types e.g.string.

**Q.1 Find the average salary of the employees in employee table.**

```
SELECT avg(salary)
FROM EMPLOYEE;
```

In some circumstance, we would like to apply the aggregate function not only to a single set of tuples, but also to a group of sets of tuples. We specify this wish in SQL using the group by clause.

The attributes given in the group by clause are used to form groups.

**Example: - Find the average salary at each department.**

```
Solution: - SELECT Dept, avg(salary)
FROM EMPLOYEE
group by Dept;
```

Output for this query

Dept	Avg(salary)
Production	38000.00
Marketing	35410.00
RND	40016.66

**Q.2 Find the minimum salary in EMPLOYEE table.**

```
Solution:- SELECT min(salary)
FROM EMPLOYEE;
```

**Q.3 Find the minimum salary of a female employee in EMPLOYEE table.**

```
Solution:- SELECT Ename, min(salary)
FROM EMPLOYEE
WHERE sex='F';
```

**Q.4 Find the maximum salary of a male employee in EMPLOYEE table.**

```
Solution:- SELECT Ename, max(salary)
FROM EMPLOYEE
WHERE sex='M';
```

**Q.5 Find the total salary of those employees who work in Guwahati city.**

```
Solution:- SELECT sum(salary)
```

```
FROM EMPLOYEE
WHERE city='Guwahati';
```

**Q.6 Find the total salary of all employees in EMPLOYEE relation.**

```
Solution:- SELECT sum(salary)
FROM EMPLOYEE;
```

**Q.7 Find the number of tuples in the EMPLOYEE relation.**

```
Solution:- SELECT count(*)
FROM EMPLOYEE;
```

**Q.8 Count the number of employees who work in RND department.**

```
Solution:- SELECT count(*)
FROM EMPLOYEE
WHERE Dept='RND';
```

**Q.9 Show the name of each department and minimum salary where minimum salary in the department is less than 8000.**

```
Solution: SELECT Dept, min(salary)
FROM EMPLOYEE
group by Dept
HAVING min(salary) < 8000;
```

**Note:** HAVING clause is often used with GROUP BY clause to apply filter condition for a group of rows.

**Q.10 Find maximum salary of each department and display the name of that department which has maximum salary more than 39000.**

```
Solution: SELECT Dept, max(salary)
FROM EMPLOYEE
group by Dept
HAVING max(salary)>39000;
```

**4.1 Intellectual Property Rights:** Intellectual property rights (IPR) is the term applied to the legal protection afforded to innovative and creative materials. It allows owner of IPR to gain from the use of the material and thereby to encourage innovation and creativity.

**IPR (Intellectual Property Rights) Issues:**

- Copyright law
- The law of confidence
- Patent law
- Design law
- Trademarks
- Copyright and computer programs
- Database copyright and the database right
- Criminal offences

**4.2 Plagiarism:** "The unauthorized use of someone's ideas, thoughts, expressions and the representation of them as one's own original work."

**4.3 Digital Rights Management:** Digital rights management (DRM) is a systematic approach to copyright protection for digital or electronic media. The purpose of DRM is to prevent unauthorized redistribution of digital media and restrict the ways consumers can copy content they've purchased.

**4.4 Digital License:** A license is an agreement that allows someone to copy, use, or resell the digital content. In digital rights management (DRM), acquiring a license to use protected copyrighted electronic material is essential.

- **Creative Commons license:** A Creative Commons (CC) license is one of several public copyright licenses that enable the free distribution of an otherwise copyrighted "work". A CC license is used when an author wants to give other people the right to share, use, and build upon a work that they (the author) have created.
- **GPL :** The GNU General Public License (GNU GPL or GPL) is a widely-used free software license, which guarantees end users the freedom to run, study, share and modify the software.

- **Apache License:** The Apache License is an open source software license released by the Apache Software Foundation (ASF). ... The Apache License allows you to freely use, modify, and distribute any Apache licensed product. However, while doing so, you're required to follow the terms of the Apache License.

#### 4.5 Open Source Software:

- Freely used and freely accessible
- Source code is open and freely available
- Less permissive than free software
- No payment required
- Example:
  - **Linux:** Free software and Open source Development
  - **Apache Server:** Open source web server
  - **MySql:** Open source database system
  - **Pango:** Open source framework
  - **Tomcat:** Open source servlet container
  - **PHP:** Open Source Programming Language
  - **OpenOffice:** Open office application suite
  - **Python:** interpreted, interactive programming language

**4.6 Open Data:** Open data is data that anyone can access, use and share without any copyright restrictions. Governments, businesses and individuals can use open data to bring about social, economic and environmental benefits.

Open data must also be licensed to allow free usage and be available for commercial use. It has the power to help improve services, grow economies and protect our planet.

#### *Example:*

- Government policies and implementation
- Money spent on government projects
- Surveys

#### 4.7 Privacy Laws:

**Privacy law** refers to the laws that deal with the regulating, storing, and using of personal information of individuals, which can be collected by governments, public or private organizations, or other individuals.

#### **Types of privacy laws:**

- (i) **General Privacy law:** related to personal information of an individual.
- (ii) **Specific Privacy law:** designed to regulate specific types of information. Example: Financial, Health, communication, information privacy law.

In India, there is a fundamental right to privacy under the Indian constitution, establishing that “The right to privacy is protected as an intrinsic part of the right to life and personal liberty”.

“Every individual in society irrespective of social class or economic status is entitled to the intimacy and autonomy which privacy protects. The pursuit of happiness is founded upon autonomy and dignity. Both are essential attributes of privacy which makes no distinction between the birth marks of individuals.”

The Information Technology (Amendment) Act, 2008 made changes to the Information Technology Act, 2000 and added the following two sections relating to Privacy:

- Section 43A, which deals with implementation of reasonable security practices for sensitive personal data or information and provides for the compensation of the person affected by wrongful loss or wrongful gain.
- Section 72A, which provides for imprisonment for a period up to three years and/or a fine up to Rs. 500,000 for a person who causes wrongful loss or wrongful gain by disclosing personal information of another person while providing services under the terms of lawful contract. A constitutional bench of the Supreme Court declared 'Privacy' as a fundamental right on 24 August 2017

#### **4.8 Fraud:**

- Fraud is intentional deception over internet to secure unfair or unlawful gain, or to deprive a victim of a legal right.
- Fraud can violate civil law, a criminal law.
- It refers to dishonestly cheating someone.

**4.9 Cyber Crime:** Criminal activities carried out by means of computers or the Internet. There are some examples of cybercrime:

(i) **Phishing:** Phishing is the fraudulent act of acquiring private and sensitive information (username and password) of a person or company through e-mail, malicious links etc.

(ii) **Illegal downloads:** Illegal downloading is obtaining files that you do not have the right to use from the Internet. Downloading of copyrighted files for which you do not have permission or licensed, is called illegal downloads.

(iii) **Child Pornography:** Sexual exploitation of children (under the age of 18). Child pornography is publishing and transmitting obscene material of children in electronic form. Child pornography is most often made by taking pictures, audio and video recording.

#### **Safeguards for children:**

- Never give address to people who you do not know.
- Never publish your personal information publicly.

- Do not open suspicious emails.
- Never visit porn, harmful websites.

Child pornography is a crime in India. Information Technology Act, 2000 & Indian Penal Code, 1860 provides protection from child pornography. Child is the person who is below the age of 18 years.

The newly passed Information Technology Bill is set to make it illegal to not only create and transmit child pornography in any electronic form, but even to browse it.

The punishment for a first offence of publishing, creating, exchanging, downloading or browsing any electronic depiction of children in “obscene or indecent or sexually explicit manner” can attract five years in jail and a fine of Rs. 10 lakh.

(iv) **Scams:** A scam is a term used to describe any fraudulent business or scheme that takes money or other goods from an unsuspecting person.

#### **Types of scams:**

- **Phishing:** fraudulent act of acquiring private and sensitive information
- **Auction Fraud:** someone may claim to be selling tickets for an upcoming concert that really are not official tickets.
- **Donation Scam:** A person claiming they have or have a child or someone they know with an illness and need financial assistance.
- **Catfish:** A person who creates a fake online profile with the intention of deceiving someone.
- **Cold call scam:** Someone claiming to be from technical support from a computer company, saying they have received information that your computer is infected with a virus, or hacked. They offer to remotely connect to your computer and fix the problem.
- **Chain main:** Usually harmless, this scam is usually spread through e-mail and tells people to forward the e-mail to all their friends to get money back from someone.
- **Online survey scams:** Online survey scams are survey sites that say they offer money or gift vouchers to participants.

**4.10 Cyber Forensics:** Cyber forensics is a technique of digital forensic science for investigation and analysis to collect evidence from a computer, mobile, internet or any electronic device to present evidence in court of law, against any criminal activity.

#### **4.11 Information Technology Act, 2000**

The Information Technology Act, 2000 (also known as IT Act-2000) is an Act of the Indian Parliament notified on 17 October 2000. It is the primary law in India dealing with cybercrime and electronic commerce.

The original Act contained 94 sections, divided in 19 chapters and 4 schedules. The laws apply to the whole of India. Persons of other nationalities can also be indicted under the law, if the crime involves a computer or network located in India.

The Act provides legal framework for electronic governance by giving recognition to electronic records and digital signatures.

Commission of cybercrime may be divided into three basic groups:

- Individual
- Organisation
- Society at Large
  
- **Against Individual**
  - Harassment via Emails
  - Cyber Stalking
  - Dissemination of obscene material
  - Defamation
  - Hacking/Cracking Indecent Exposure
  - Computer Vandalism
  - Transmitting a Virus
  - Network Trespassing
  - Unauthorized Control over Computer System
  - Hacking/Cracking
  
- **Against Organisation**
  - Hacking & Cracking
  - Possession of unauthorized Information
  - Cyber- Terrorism against Government Organization
  - Distribution of Pirated Software Etc
  
- **Against Society at Large**
  - Pornography
  - Polluting the youth through indecent exposure
  - Trafficking

## **Offenses UNDER THE IT ACT, 2000**

### **1. Tampering with computer source documents (Intellectual Property):**

**Section 65** of this Act provides that Whoever knowingly or intentionally conceals, destroys or alters or intentionally or knowingly causes another to conceal, destroy or alter any computer source code used for a computer, computer program, computer system or computer network, when the computer source code is required to be kept or maintained by law for the being time in force, shall be punishable.

**Punishment** : This is cognizable and non- bailable offense. Imprisonment up to 3 years and or fine up to two lakh rupees.



## **2. Hacking with the computer system:**

**Section 66** provides that-

(1) Whoever with the intent to cause or knowing that he is likely to cause wrongful loss or damage to the public or any person destroys or deletes or alters any information residing in a computer resource or diminishes its value or utility or affects it injuriously by any means, commits hacking.

(2) Whoever commits hacking shall be punished with imprisonment up to three years, or with fine which may extend up to two lakh rupees, or with both.

**Punishment:** Imprisoned up to three years and fine which may extend up to two lakh rupees or with both.

## **3. Publishing of obscene information in electronic form:**

**Section 67** of this Act provides that whoever publishes or transmits or causes to be published in the electronic form, any material which is about posting obscene, defamatory and annoying message about a person.

**Punishment:** On first conviction with imprisonment of either description for a term which may extend to five years and with fine which may extend to one lakh rupees and in the event of a second or subsequent conviction with imprisonment of either description for a term which may extend to ten years and also with fine which may extend to two lakh rupees.

## **4. Power of Controller to give directions:**

**Section 68** of this Act provides that :

(1) The Controller may, by order, direct a Certifying Authority or any employee of such Authority to take such measures or cease carrying on such activities as specified in the order if those are necessary to ensure compliance with the provisions of this Act, rules or any regulations made thereunder.

(2) Any person who fails to comply with any order under sub-section (1) shall be guilty of an offense and shall be liable on conviction to imprisonment for a term not exceeding three years or to a fine not exceeding two lakh rupees or to both.

**Punishment:** The offense under this section is non-bailable & cognizable. Imprisonment up to a term not exceeding three years or fine not exceeding two lakh rupees.

## **5. Directions of Controller to a subscriber to extend facilities to decrypt information:**

**Section 69** provides that-

(1) If the Controller is satisfied that it is necessary or expedient so to do in the interest of the sovereignty or integrity of India, the security of the State, friendly relations with foreign States or public order or for preventing incitement to the commission of any cognizable offense; for reasons to be recorded in writing, by order, direct any agency of the Government to intercept any information transmitted through any computer resource.



(2) The subscriber or any person in charge of the computer resource shall, when called upon by any agency which has been directed under sub-section (1), extend all facilities and technical assistance to decrypt the information.

(3) The subscriber or any person who fails to assist the agency referred to in subsection shall be punished with imprisonment for a term which may extend to seven years. Punishment: Imprisonment for a term which may extend to seven years. The offense is cognizable and non-bailable.

## **6. Protected System:**

**Section 70** of this Act provides that –

(1) The appropriate Government may, by notification in the Official Gazette, declare that any computer, computer system or computer network to be a protected system.

(2) The appropriate Government may, by order in writing, authorize the persons who are authorized to access protected systems notified under sub-section (1).

(3) Any person who secures access or attempts to secure access to a protected system in contravention of the provision of this section shall be punished with imprisonment of either description for a term which may extend to ten years and shall also be liable to fine.

**Punishment:** The imprisonment which may extend to ten years and fine.

## **7. Penalty for misrepresentation:**

**Section 71** provides that-

(1) Whoever makes any misrepresentation to, or suppresses any material fact from, the Controller or the Certifying Authority for obtaining any license or Digital Signature Certificate, as the case may be, shall be punished with imprisonment for a term which may extend to two years, or with fine which may extend to one lakh rupees, or with both.

**Punishment:** Imprisonment which may extend to two years or fine may extend to one lakh rupees or with both.

## **8. Penalty for breach of confidentiality and privacy:**

Section 72 provides that- Save as otherwise provide in this Act or any other law for the time being in force, any person who, in pursuance of any of the powers conferred under this Act, rules or regulation made thereunder, has secured access to any electronic record, book, register, correspondence, information, document or other material without the consent of the person concerned discloses such material to any other person shall be punished with imprisonment for a term which may extend to two years, or with fine which may extend to one lakh rupees, or with both.

**Punishment:** Term which may extend to two years or fine up to one lakh rupees or with both.

## **9. Penalty for publishing Digital Signature Certificate false in certain particulars:**

**Section 73** provides that –

(1) No person shall publish a Digital Signature Certificate or otherwise make it available to any other person with the knowledge that-

- (a) The Certifying Authority listed in the certificate has not issued it; or
  - (b) The subscriber listed in the certificate has not accepted it; or
  - (c) The certificate has been revoked or suspended unless such publication is for the purpose of verifying a digital signature created prior to such suspension or revocation.
- (2) Any person who contravenes the provisions of sub-section (1) shall be punished with imprisonment for a term which may extend to two years, or with fine which may extend to one lakh rupees, or with both.

**Punishment:** Imprisonment of a term of which may extend to two Years or fine may extend to 1 lakh rupees or with both.

### **10. Publication for fraudulent purpose:**

**Section 74** provides that- Whoever knowingly creates, publishes or otherwise makes available a Digital Signature Certificate for any fraudulent or unlawful purpose.

**Punishment:** Imprisonment for a term up to two years or fine up to one lakh or both.

### **11. Act to apply for offense or contravention committed outside India:**

**Section 75** provides that-

(1) Subject to the provisions of sub-section (2), the provisions of this Act shall apply also to any offense or contravention committed outside India by any person irrespective of his nationality.

For the purposes of sub-section (1), this Act shall apply to an offense or Contravention committed outside India by any person if the act or conduct constituting the offense or contravention involves a computer, computer system or computer network located in India.

### **12. Confiscation:**

**Section 76** provides that- Any computer, computer system, floppies, compact disks, tape drives or any other accessories related thereto, in respect of which any provisions of this Act, rules, orders or regulations made thereunder has been or is being contravened, shall be liable to confiscation. :

Provided that where it is established to the satisfaction of the court adjudicating the confiscation that the person in whose possession, power or control of any such computer, computer system, floppies, compact disks, tape drives or any other accessories relating thereto is found is not responsible for the contravention of the provisions of this Act, rules orders or regulations made thereunder, the court may, instead of making an order for confiscation of such computer, computer system, floppies, compact disks, tape drives or any other accessories related thereto, make such other order authorized by this Act against the person contravening of the provisions of this Act, rules, orders or regulations made thereunder as it may think fit.

### 13. Penalties or confiscation not to interfere with other punishments:

**Section 77** provides that – No penalty imposed or confiscation made under this Act shall prevent the imposition of any other punishment to which the person affected thereby is liable under any other law for the time being in force.

#### Power to investigate offenses:

**Section 78** provides that – Notwithstanding anything contained in the Code of Criminal Procedure, 1973, a police officer not below the rank of Deputy Superintendent of Police shall investigate any offense under this Act.

### 4.12 Technology and Society:

Technology impacts the environment, people and the society as a whole. The way we use technology determines if its impacts are positive to the society or negative.

Positive Impacts	Negative Impacts
Can send and access information anytime, anywhere.	Threat of online predators like hackers, viruses, fraud, scams etc.
Long distance communication	Loss of personal contact
Automation	Overdependence
Improved transportation	Natural resource Depletion
Improved Education and learning process	Increased pollution
Improvement in agriculture	Wastage of time with online games and social media
Improvement in health facilities	Less physical growth
Business and banking	Impact of ethics and human values
Online shopping	Global warming
Satellite communication	Nuclear bomb and weapons

#### Cultural Impact of technology:

**1. Tradition:** The influence of IT on religious practices has mainly been to the effect of making information about them more accessible.

**2. Government:** It helps government to improve its services to citizens.

Collection of data of citizens may help government to provide better services according to their need. But it may lead fraud and scam, if data is not preserved securely.

**3. Defence:** Defence capabilities are improved.

**4. Commercial business:** Able to extract information what the customer wants. Easy to transport the products.

5. Encourages innovation and creativity

6. Improves on human resource management.

7. Ethics, human values, customs and culture degraded.

### 4.13 E-waste Management:

E-waste stands for Electronic waste. E-waste is actually the old electronic items that people simply give away to garbage trucks that are then dumped into landfill. Electronics have a number of harmful elements that react with air and water to create problems of e-waste such as water, air and soil pollution as well as problems that affect human beings in the form of diseases. E-waste management is required to overcome this problems.

**Definition:** Proper disposal of electronic equipment and its impact on environment, is known as e-waste management. It includes reuse, resale, recycling of electronic devices.

#### E-waste management rules, 2011:

Extended Producer Responsibility (EPR) principle will apply Collection of E-waste :

- Generated during manufacturing
- Generated from the end of life products
- Such E Wastes are channelized to a registered dismantler or recycler
- Individual identification code for product tracking
- Provide contact details of dealers and authorized collection centers to consumers
- Finance and organise the system
- Ensure safe transportation, storage
- Submit annual return

#### How to manage e-waste:

- Buy less
- Follow the EPR strategy : The EPR is an environment protection strategy that makes the producer responsible for the entire life cycle of the product, specially for take back, recycle and final disposal of the product
- Organize what you have
- Donate your e-waste
- Use buy-back method
- Learn about your local recycling options
- Live in the cloud
- Follow RRR (Reduce, Reuse, Recycle)
  - **Reduce:** Reduce the use of electronic items.
  - **Reuse:**
    - Plastic
    - Metal
    - Glass
    - Printed Circuit board
    - Hard Drives

- Cartridges
- Batteries
- **Recycle:**
  - Picking Shed : Sort the items manually. Remove batteries
  - Disassemble
  - First size reduction process
  - Second size reduction process
  - Over band magnet
  - Separation of metallic and non-metallic components
  - Water separation

**4.14 Identity Theft:** when someone uses your personal information to pretend to be you without your knowledge and commit a crime or fraud.

**How to prevent Identity Theft:**

- Never share your password or account number over email or message.
- Do not follow links from e-mails when performing financial transaction.
- Be aware of callers, pop-ups, websites or emails asking for personal information.
- Use firewall and reliable anti-virus software.
- Never share your personal information in public domain.
- Regularly change the passwords of your account.

**4.15 Unique Ids:** Unique ids include fingerprints, hand geometry, earlobe geometry, retina and iris patterns, voice waves, DNA, and signatures.

**Iris Pattern:** Iris recognition is an automated method of biometric identification that uses mathematical pattern-recognition techniques on video images of one or both of the irises of an individual's eyes, whose complex patterns are unique, stable, and can be seen from some distance.

**4.16 Biometrics:** Electronic device which is used for authentication and identification of a person using his/her biological and behavioural characteristics.

**4.17 Gender and disability issues while teaching and using computers:**

The low participation by women in computer science courses in secondary and sr. secondary education is an important equity issue in science education. In addition to the increasingly intense need for more highly skilled people in the IT sector, women are missing out on many of today's most attractive career opportunities. Equally importantly, the IT field is missing out on the broader range of perspectives and talents that would result from significantly increased participation by women.

To overcome this stereo type situation, the following points to be noted:

Transform pink software by creating gender neutral software that challenges and appeals to a variety of students;

- Look to girls and women to fill the IT job shortage: encourage girls into computing by using technology in a broad range of subjects to attract a more diverse group of students.
- Prepare tech-savvy teachers: empowered teachers will empower students.
- Educate girls to be designers, not just users, ensuring they have opportunities to fully explore the potential of technology.
- Change the public face of computing so girls have a realistic image of computer professionals, and understand the importance of communication and team work in this field.
- Create a family computer, placed where the entire family has access, and computer activities are associated within a social context, and not equated to isolation.
- Set a new standard for gender equity that seeks equal contributions to innovations in technology and equal mastery of the analytical and computing skills required to make these contributions.