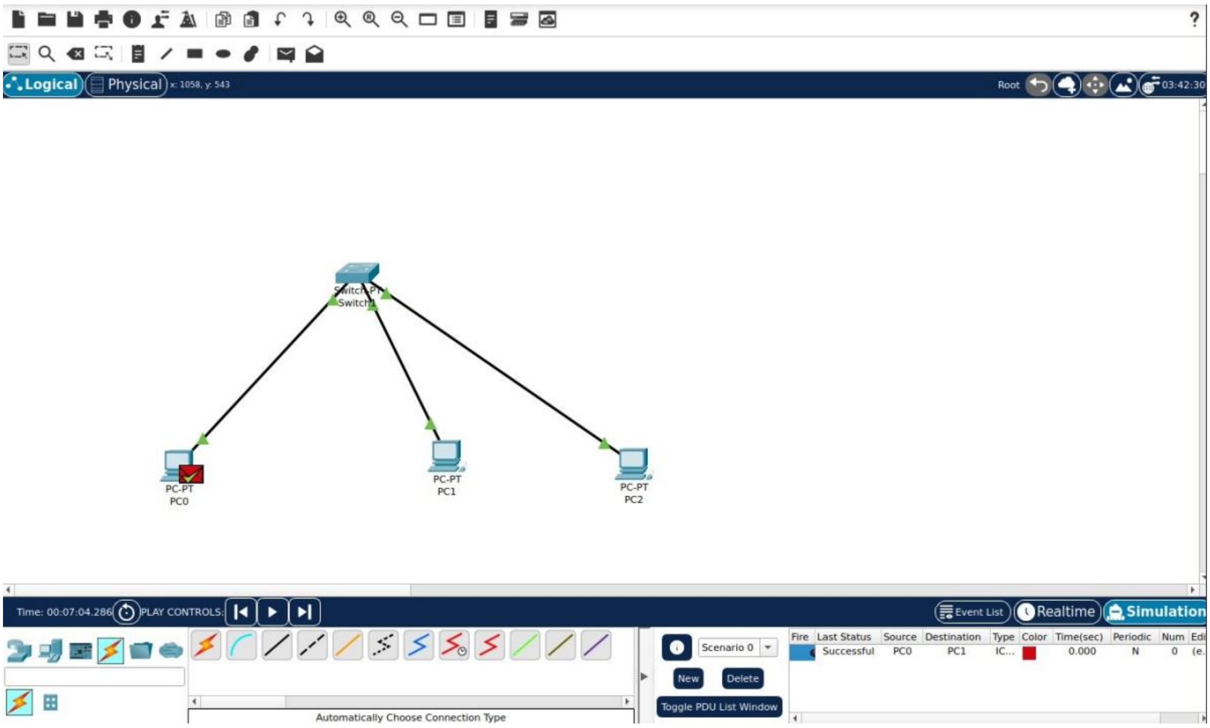
Assignment A.1

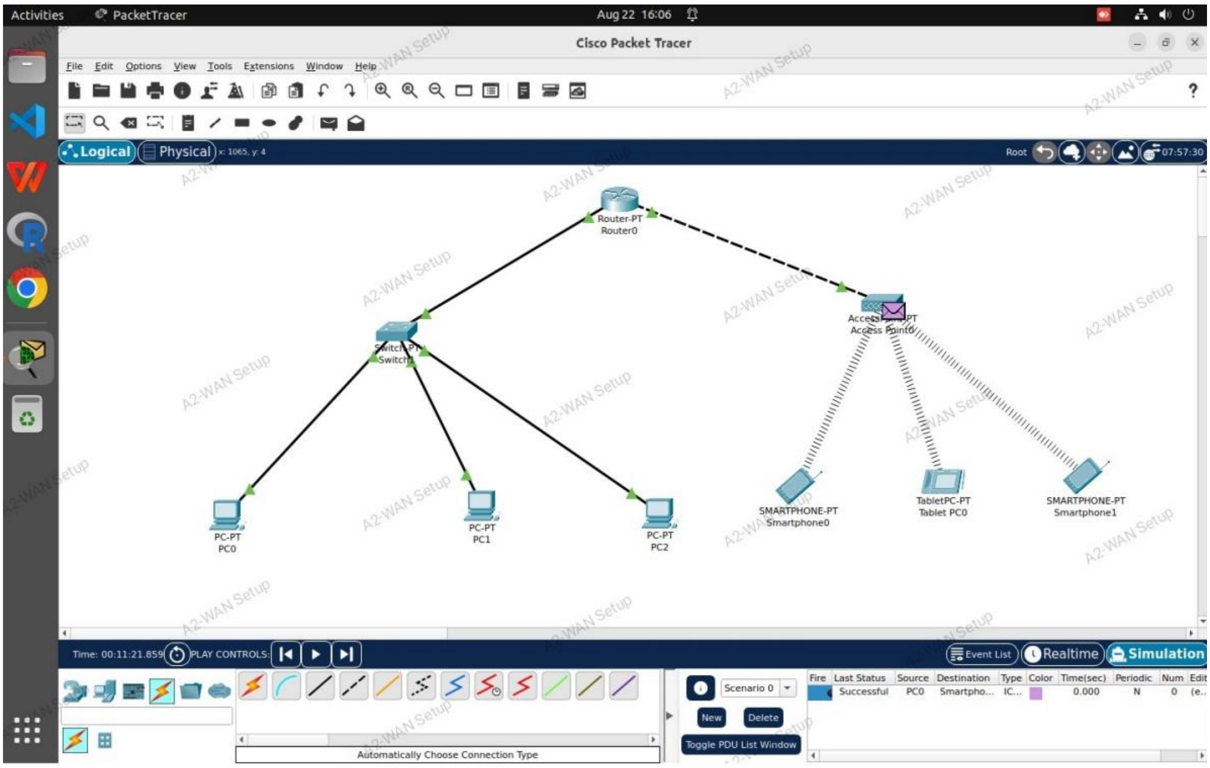
Output:-



Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edi
🟦	Successful	PC0	PC1	IC...	🟥	0.000	N	0	(e.

Assignment A3

Output:-



Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit
🟦	Successful	PC0	Smartpho...	IC...	🟪	0.000	N	0	(e...

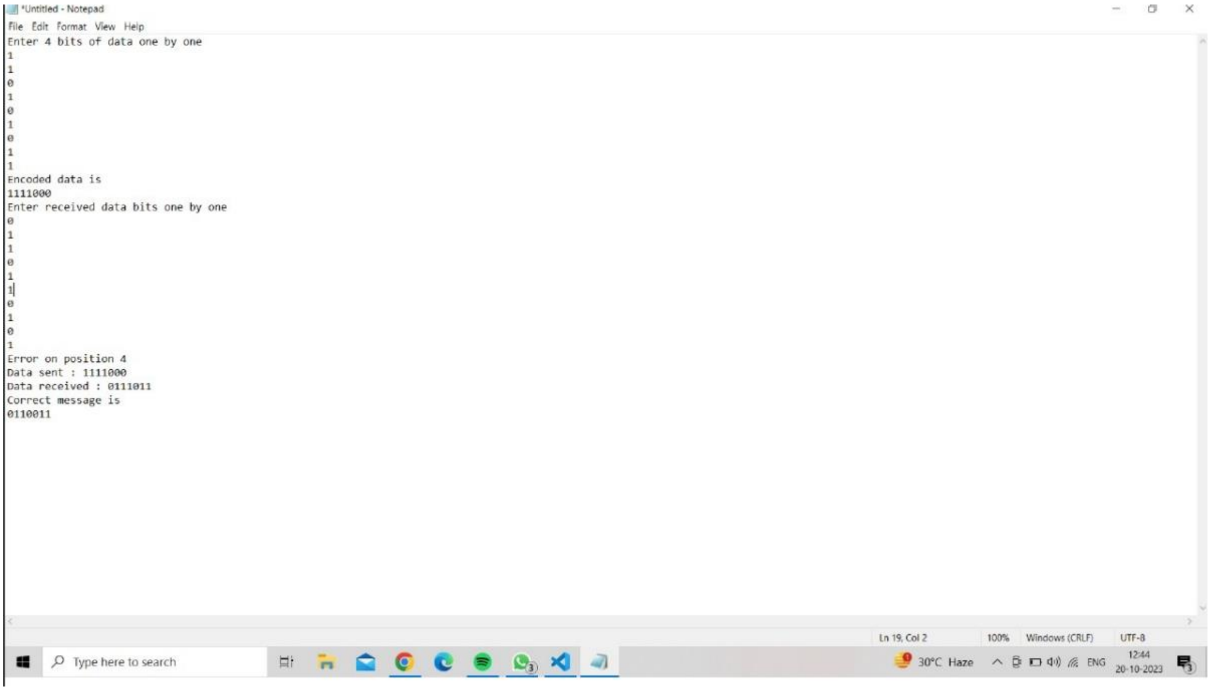
Assignment – A4

Input:-

Hamming code

```
#include<iostream>
 using namespace std;
 int main()
{
   int data[10];
   int dataatrec[10],c,c1,c2,c3,i;

   cout<<"Enter 4 bits of data one by one\n";
   cin>>data[0];
   cin>>data[1];
   cin>>data[2];
   cin>>data[4];

   //Calculation of even parity
   data[6]=data[0]^data[2]^data[4];
data[5]=data[0]^data[1]^data[4];
data[3]=data[0]^data[1]^data[2];

cout<<"\nEncoded data is\n";
for(i=0;i<7;i++)
       cout<<data[i];
cout<<"\n\nEnter received data bits one by one\n";
  for(i=0;i<7;i++)
    cin>>dataatrec[i];
 //max 1 bit error is detected
c1=dataatrec[6]^dataatrec[4]^dataatrec[2]^dataatrec[0];
c2=dataatrec[5]^dataatrec[4]^dataatrec[1]^dataatrec[0];
c3=dataatrec[3]^dataatrec[2]^dataatrec[1]^dataatrec[0];
c=c3*4+c2*2+c1 ;
 if(c==0) {
cout<<"\nNo error while transmission of data\n";
  }
else {
cout<<"\nError on position "<<c;
cout<<"\nData sent : ";
for(i=0;i<7;i++)
     cout<<data[i];
    cout<<"\nData received : ";
    for(i=0;i<7;i++)
     cout<<dataatrec[i];
    cout<<"\nCorrect message is\n";
    //if errorneous bit is 0 we complement it else vice versa
if(dataatrec[7-c]==0)
dataatrec[7-c]=1;
    else
dataatrec[7-c]=0;
for (i=0;i<7;i++) {
cout<<dataatrec[i];
}
}
return 0;
}
```

Output:-



```
Enter 4 bits of data one by one
1
1
0
1
0
1
0
1
1
Encoded data is
1111000
Enter received data bits one by one
0
1
1
0
1
1
0
1
0
1
Error on position 4
Data sent : 1111000
Data received : 0111011
Correct message is
0110011
```

CRC error detection

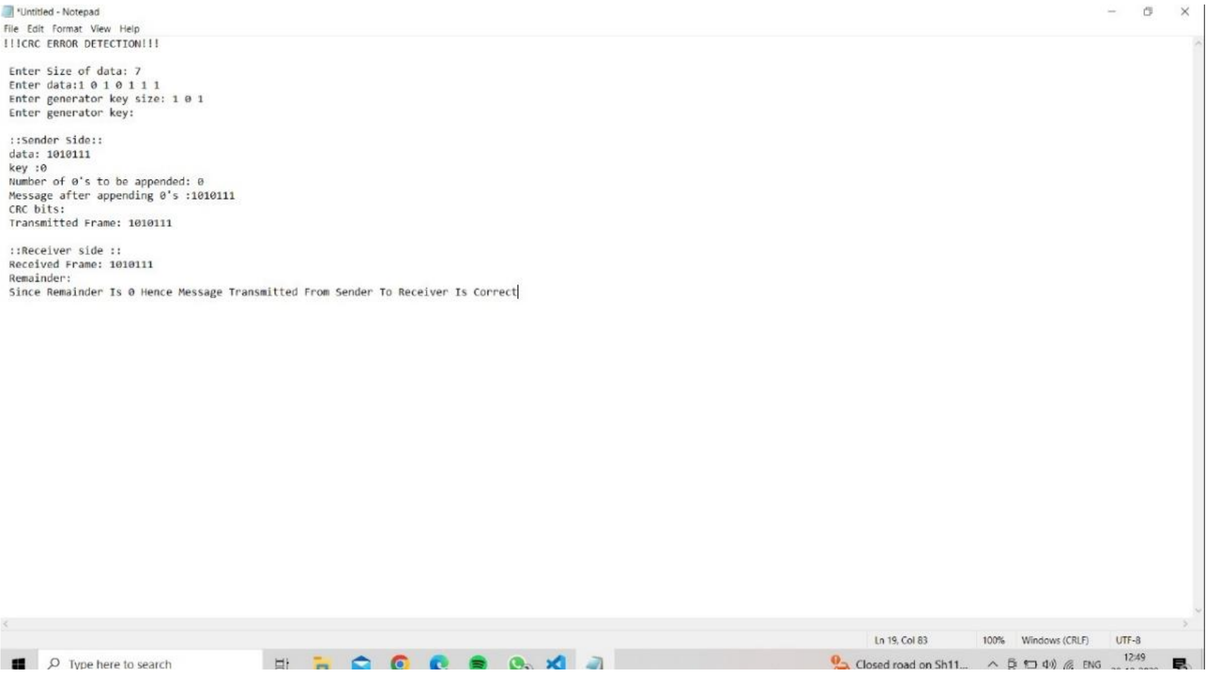
```
#include <iostream>
using namespace std;
int main() {
  int i,j,k;
    cout << "!!!CRC ERROR DETECTION!!!" << endl;
//Get Frame
        int fs; //frame size
        cout<<"\n Enter Size of data: ";
        cin>>fs;
  int f[20]; //frame data
 cout<<" Enter data:";
        for(i=0;i<fs;i++)//to read in frame data
        {
            cin>>f[i];
        }
//Get Generator
        int gs; //generator size
        cout<<"\n Enter generator key size: ";
        cin>>gs;
       int g[20]; //generator or divisor data
       cout<<"\n Enter generator key:";
        for(i=0;i<gs;i++)
        {
            cin>>g[i];
        }
       cout<<"\n\n ::Sender Side::";
        cout<<"\n data: ";
```

```
     for(i=0;i<fs;i++) //print frame
     {
        cout<<f[i];
     }
     cout<<"\n key :";
     for(i=0;i<gs;i++) //print generator/divisor
     {
        cout<<g[i];
     }
    //Append 0's before binary division
     int rs = gs-1; //remainder size
    cout<<"\n Number of 0's to be appended: "<<rs;
     for (i=fs;i<fs+rs;i++) //append zeros in frame
     {
        f[i]=0;
     }
    int temp[20];
     for(i=0;i<20;i++)//copy frame[] in temp[] for division purpose
     {
        temp[i]=f[i];
     }
    cout<<"\n Message after appending 0's :";
     for(i=0; i<fs+rs;i++)//print frame with appended zeros
     {
        cout<<temp[i];
     }
//binary Division
   for(i=0;i<fs;i++)
     {
        j=0; //beginning pos in generator
        k=i; //current bit pos in temp
```

```
      //check whether it is divisible or not
      if (temp[k]>=g[j])
      {
        for(j=0,k=i;j<gs;j++,k++)
        {
          if((temp[k]==1 && g[j]==1) || (temp[k]==0 && g[j]==0))
          {
            temp[k]=0;
          }
          else
          {
            temp[k]=1;
          }
        }
      }
    }
//CRC
    int crc[15];
    for(i=0,j=fs;i<rs;i++,j++)//copy last remainder bits as CRC
    {
      crc[i]=temp[j];
    }

    cout<<"\n CRC bits: ";
    for(i=0;i<rs;i++) //print CRC
    {
      cout<<crc[i];
    }

    cout<<"\n Transmitted Frame: ";
    int tf[15]; //f[]+crc[]
    for(i=0;i<fs;i++)
    {
      tf[i]=f[i];
    }
    for(i=fs,j=0;i<fs+rs;i++,j++)
    {
      tf[i]=crc[j];
    }
    for(i=0;i<fs+rs;i++)//print transmitted frame
    {
      cout<<tf[i];
    } cout<<"\n\n ::Receiver side :: ";
    cout<<"\n Received Frame: ";
    for(i=0;i<fs+rs;i++) //print received frame
    {
      cout<<tf[i];
    }
 for(i=0;i<fs+rs;i++) //copy tf[] in temp[] for division  purpose
    {
      temp[i]=tf[i];
    }
//Division
    for(i=0;i<fs;i++)
    {
      j=0;
      k=i;
      if (temp[k]>=g[j])
      {
        for(j=0,k=i;j<gs;j++,k++)
        {
          if((temp[k]==1 && g[j]==1) || (temp[k]==0 && g[j]==0))
```

```
                   {
                       temp[k]=0;
                 }
                   else
                   {
                       temp[k]=1;
                   }
                }
            }
        }
       cout<<"\n Remainder: ";
        int rrem[15]; //
        for (i=fs,j=0;i<fs+rs;i++,j++)
        {
            rrem[j]= temp[i];
        }
        for(i=0;i<rs;i++)
        {
            cout<<rrem[i];
        }
     int flag=0;
        for(i=0;i<rs;i++)
        {
            if(rrem[i]!=0)
            {
                flag=1;
                break;
            }
        }
if(flag==0)
        {
            cout<<"\n Since Remainder Is 0 Hence Message Transmitted From Sender To Receiver Is Correct";
        }
        else
        {
            cout<<"\n Since Remainder Is Not 0 Hence Message Transmitted From Sender To Receiver Contains Error";
        }
 return 0;
}
```

Output:-



```
!!!CRC ERROR DETECTION!!!

 Enter Size of data: 7
 Enter data:1 0 1 0 1 1 1
 Enter generator key size: 1 0 1
 Enter generator key:

 ::Sender Side::
 data: 1010111
 key :0
 Number of 0's to be appended: 0
 Message after appending 0's :1010111
 CRC bits:
 Transmitted Frame: 1010111

 ::Receiver side ::
 Received Frame: 1010111
 Remainder:
 Since Remainder Is 0 Hence Message Transmitted From Sender To Receiver Is Correct
```

Assignment -A.5

Input:-

GO BACK N CLIENT PROGRAM

```
import java.io.*;

import java.net.*;

import java.math.*;

import java.util.*;


class testclient

{


public static void main(String args[])throws IOException

{

InetAddress addr=InetAddress.getByName("Localhost");

System.out.println(addr);


Socket connection=new Socket(addr,8111);


BufferedInputStream in=new BufferedInputStream(connection.getInputStream());

DataOutputStream out=new DataOutputStream(connection.getOutputStream());

Scanner scr=new Scanner(System.in);// this will be used to accept i/p from console


System.out.println(".......Client........");

System.out.println("Connect");

System.out.println("Enter the number of frames to be requested to the server");

int c=scr.nextInt();
```

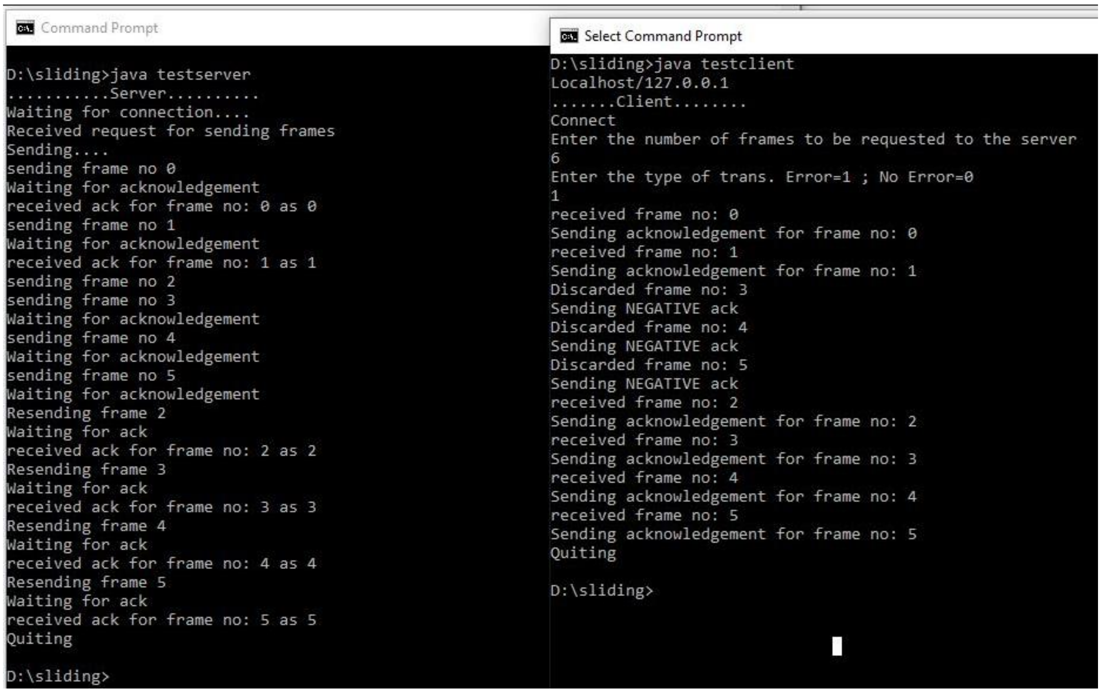
```
out.write(c);
out.flush();

System.out.println("Enter the type of trans. Error=1 ; No Error=0");
int choice=scr.nextInt();
out.write(choice);

int check=0;
int i=0;
int j=0;

if(choice==0)
{
for(j=0;j<c;++j)
{
i=in.read();
System.out.println("received frame no: "+i);
System.out.println("Sending acknowledgement for frame no: "+i);
out.write(i);
out.flush();
}
out.flush();
}
else
{
for(j=0;j<c;++j)
{
i=in.read();
if(i==check)
{
System.out.println("received frame no: "+i);
System.out.println("Sending acknowledgement for frame no: "+i);
out.write(i);
++check;
}
else
{
--j;
System.out.println("Discarded frame no: "+i);
System.out.println("Sending NEGATIVE ack");
out.write(-1);
}
out.flush();
}
}//end of else for error

in.close();
out.close();
System.out.println("Quiting");

}// end of main method
}// end of main class
```

GO BACK N SERVER PROGRAM

```
import java.io.*;
import java.net.*;
import java.util.*;
class testserver
{
public static void main(String args[])throws IOException
{
System.out.println("...........Server..........");
System.out.println("Waiting for connection....");
InetAddress addr=InetAddress.getByName("Localhost");
ServerSocket ss=new ServerSocket(8111);
Socket client=new Socket();
client=ss.accept();
BufferedInputStream in=new BufferedInputStream(client.getInputStream());
DataOutputStream out=new DataOutputStream(client.getOutputStream());
System.out.println("Received request for sending frames");
int p=in.read();
boolean f[]=new boolean[p];
int pc=in.read();
System.out.println("Sending....");
if(pc==0)
{
for(int i=0;i<p;++i)
{
System.out.println("sending frame number "+i);
out.write(i);
out.flush();
System.out.println("Waiting for acknowledgement");
try
{
Thread.sleep(7000);
}
catch(Exception e){}

int a=in.read();
System.out.println("received acknowledgement for frame "+i+" as "+a);
}
out.flush();
}
```

```
else
{
for(int i=0;i<p;++i)
{
if(i==2)
{
System.out.println("sending frame no "+i);
}
else
{
System.out.println("sending frame no "+i);
out.write(i);
out.flush();
System.out.println("Waiting for acknowledgement ");
try
{
Thread.sleep(7000);
}
catch(Exception e){}

int a=in.read();

if(a!=255)
{
System.out.println("received ack for frame no: "+i+" as "+a);
f[i]=true;
}
}// end of inner else
}// end of for

// check which frames have not been ack
for(int a=0;a<p;++a)
{
if(f[a]==false)
{
System.out.println("Resending frame "+a);
out.write(a);
out.flush();
System.out.println("Waiting for ack ");
try
{
Thread.sleep(5000);
}
catch(Exception e){}
int b=in.read();
System.out.println("received ack for frame no: "+a+" as "+b);
f[a]=true;
}
}
out.flush();
}// end of else which is for error
in.close();
out.close();
client.close();
ss.close();
System.out.println("Quiting");

}// end main method
}// end main class
```

Output:-



```
D:\sliding>java testserver
...........Server..........
Waiting for connection....
Received request for sending frames
Sending....
sending frame no 0
Waiting for acknowledgement
received ack for frame no: 0 as 0
sending frame no 1
Waiting for acknowledgement
received ack for frame no: 1 as 1
sending frame no 2
sending frame no 3
Waiting for acknowledgement
sending frame no 4
Waiting for acknowledgement
sending frame no 5
Waiting for acknowledgement
Resending frame 2
Waiting for ack
received ack for frame no: 2 as 2
Resending frame 3
Waiting for ack
received ack for frame no: 3 as 3
Resending frame 4
Waiting for ack
received ack for frame no: 4 as 4
Resending frame 5
Waiting for ack
received ack for frame no: 5 as 5
Quiting

D:\sliding>
```

```
D:\sliding>java testclient
Localhost/127.0.0.1
.......Client........
Connect
Enter the number of frames to be requested to the server
6
Enter the type of trans. Error=1 ; No Error=0
1
received frame no: 0
Sending acknowledgement for frame no: 0
received frame no: 1
Sending acknowledgement for frame no: 1
Discarded frame no: 3
Sending NEGATIVE ack
Discarded frame no: 4
Sending NEGATIVE ack
Discarded frame no: 5
Sending NEGATIVE ack
received frame no: 2
Sending acknowledgement for frame no: 2
received frame no: 3
Sending acknowledgement for frame no: 3
received frame no: 4
Sending acknowledgement for frame no: 4
received frame no: 5
Sending acknowledgement for frame no: 5
Quiting

D:\sliding>
```

SELECTIVE REPEAT RECEIVER

```
import java.lang.System;
import java.net.*;
import java.io.*;
public class Client {
   static Socket connection;
 public static void main(String a[]) throws SocketException {
       try {
          int v[] = new int[8];
          //int g[] = new int[8];
          int n = 0;
          InetAddress addr = InetAddress.getByName("localhost");
//InetAddress addr = InetAddress.getByName("1.1.7.199");
          System.out.println(addr);
          connection = new Socket(addr, 8045);
          DataOutputStream out = new DataOutputStream(
                 connection.getOutputStream());
          DataInputStream in = new DataInputStream(
                 connection.getInputStream());
          int p = in.read();
          System.out.println("No of frame is:" + p);

          for (int i = 0; i < p; i++) {
             v[i] = in.read();
             System.out.println(v[i]);
             //g[i] = v[i];
          }
          v[5] = -1;
          for (int i = 0; i < p; i++)
           {
```

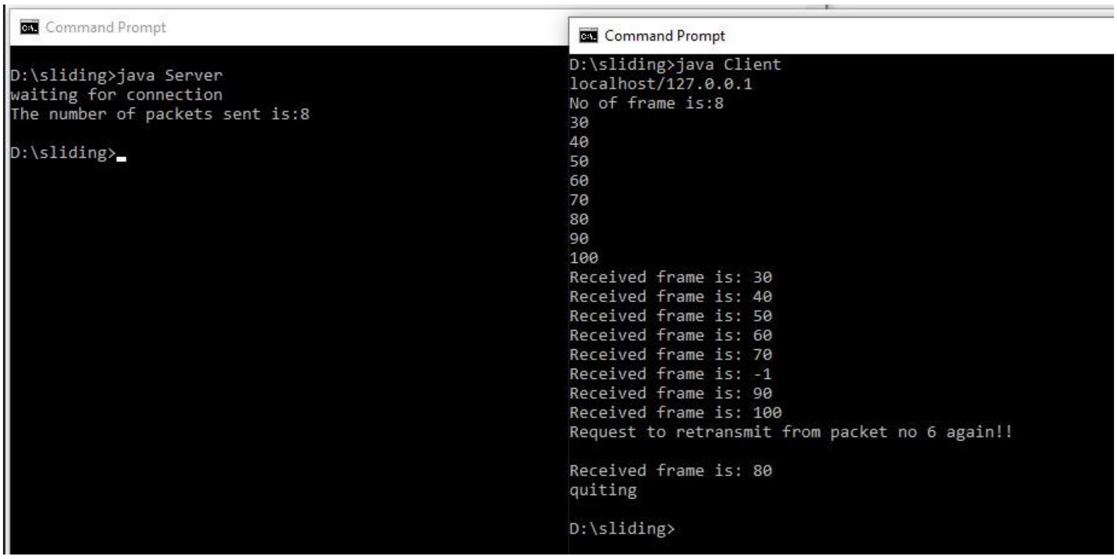
```
          System.out.println("Received frame is: " + v[i]);


        }
      for (int i = 0; i < p; i++)
        if (v[i] == -1) {
          System.out.println("Request to retransmit from packet no "
              + (i+1) + " again!!");
          n = i;
          out.write(n);
          out.flush();
        }

      System.out.println();


        v[n] = in.read();
        System.out.println("Received frame is: " + v[n]);
        System.out.println("quiting");
    } catch (Exception e) {
      System.out.println(e);
    }
}
```

SELECTIVE REPEAT SENDER

```
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.ServerSocket;
import java.net.Socket;
import java.net.SocketException;

public class Server {
```

```
static ServerSocket Serversocket;

static DataInputStream dis;

static DataOutputStream dos;


public static void main(String[] args) throws SocketException {


    try {

        int a[] = { 30, 40, 50, 60, 70, 80, 90, 100 };

        Serversocket = new ServerSocket(8045); //PORT NUMBER

        System.out.println("waiting for connection");


        Socket client = Serversocket.accept();

        dis = new DataInputStream(client.getInputStream());

        dos = new DataOutputStream(client.getOutputStream());

        System.out.println("The number of packets sent is:" + a.length);

        int y = a.length;

        dos.write(y);

        dos.flush();


        for (int i = 0; i < a.length; i++) {

            dos.write(a[i]);

            dos.flush();

        }


        int k = dis.read(); //


        dos.write(a[k]); //

        dos.flush();


    } catch (IOException e) {

        System.out.println(e);
        } finally {
            try {
                dis.close();
                dos.close();
            } catch (IOException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }

        }
    }
}
```

Output:-



```
D:\sliding>java Server
waiting for connection
The number of packets sent is:8

D:\sliding>
```

```
D:\sliding>java Client
localhost/127.0.0.1
No of frame is:8
30
40
50
60
70
80
90
100
Received frame is: 30
Received frame is: 40
Received frame is: 50
Received frame is: 60
Received frame is: 70
Received frame is: -1
Received frame is: 90
Received frame is: 100
Request to retransmit from packet no 6 again!!

Received frame is: 80
quiting

D:\sliding>
```

Assignment -B.6

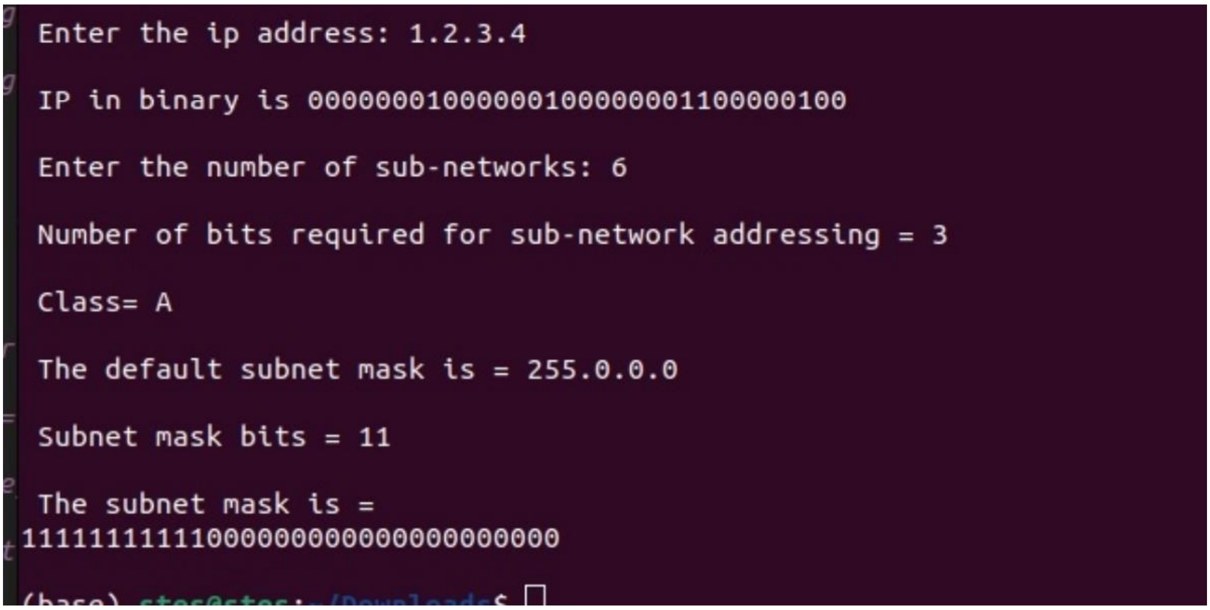
Input:-

```
import java.util.Scanner;

import java.net.InetAddress;

class Subnet{

public static void main(String args[])

 {

   Scanner sc = new Scanner(System.in);

   System.out.print("\n Enter the ip address: ");

   String ip = sc.nextLine();

   String split_ip[] = ip.split("\\.");//SPlit the string after every.

   String split_bip[] = new String[422]; //split binary ip

   String bip = "";

   for(int i=0;i<4;i++)

   {

// "18" => 18 => 10010 => 00010010

       split_bip[i] = appendZeros(Integer.toBinaryString(Integer.parseInt(split_ip[i])));

       bip += split_bip[i];

   }

   System.out.println("\n IP in binary is "+bip);

   System.out.print("\n Enter the number of sub-networks: ");

   int n = sc.nextInt();


//Calculation of mask

/*eg if address = 120, log 120/log 2 gives log to the base 2 => 6.9068, ceil gives us upper integer */

   int bits = (int)Math.ceil(Math.log(n)/Math.log(2));

System.out.println("\n Number of bits required for sub-network addressing = "+bits);
```

```
 int[] arr=new int[32]; //arr[] is mask after subnetting
 for(int i=0;i<32;i++)    {    arr[i]=1;    }
int cc = Integer.parseInt(split_ip[0]); //class check
 String mask = null;
 int x =0;
 if(cc>0 && cc<224)
    {
       if(cc<128)
       {
       mask = "255.0.0.0";
           x = 8+bits;
       System.out.println("\n Class= A");
       }
       if(cc>127 && cc<192)
       {
           mask = "255.255.0.0";
            x = 16+bits;
       System.out.println("\n Class= B");
       }
       if(cc>191)
       {
           mask = "255.255.255.0";
            x = 24+bits;
       System.out.println("\n Class= C");
       }
    for(int i=x;i<32;i++)
           {
             arr[i]=0;
           }
    }
```

```
   System.out.println("\n The default subnet mask is = "+mask);
   System.out.println("\n Subnet mask bits = "+x);
   System.out.println("\n The subnet mask is = ");
      for(int i=0;i<32;i++)
      {
         System.out.print(arr[i]);
      }
   System.out.println("\n");
 }//main() ends

static String appendZeros(String s)
 {
   String temp = new String("00000000");
   return temp.substring(s.length())+ s;
 }
}
```

Output:-



```
 Enter the ip address: 1.2.3.4

 IP in binary is 00000001000000100000001100000100

 Enter the number of sub-networks: 6

 Number of bits required for sub-network addressing = 3

 Class= A

 The default subnet mask is = 255.0.0.0

 Subnet mask bits = 11

 The subnet mask is =
11111111111000000000000000000000

(base) stos@stos:~/Downloads$
```

Assignment -B.7

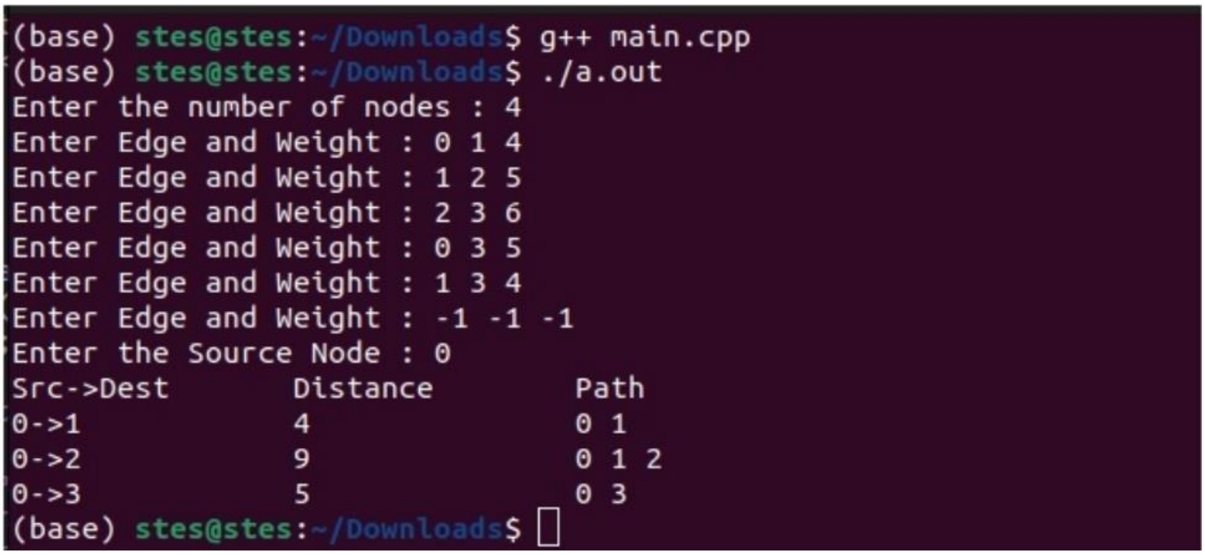
Input:-

```
#include <iostream>
using namespace std;
int N;
int graph[10][10];
int dist[10];
bool visited[10];
int parent[10];
void createGraph()
{
   int i,j,max,u,v,w;
   cout<<"Enter the number of nodes : ";
   cin>>N;
   for(i=0;i<=N;i++)
    for(j=0;j<=N;j++)
     graph[i][j]=0;
   max=N*(N+1);
   for(i=0;i<max;i++)
   {
    cout<<"Enter Edge and Weight : ";
    cin>>u>>v>>w;
    if(u==-1)    break;
    else
    {
      graph[u][v]=w;
      graph[v][u]=w;
    }
   }
```

```
}

int minDistance()
{
   int min = 10000, minDist;
  for (int v = 0; v < N; v++)
      if (visited[v] == false && dist[v] <= min)
      {
         min = dist[v];
         minDist = v;
      }
   return minDist;
}

void printPath(int j)
{
   if (parent[j]==-1)
      return;
   printPath(parent[j]);
   cout<<j<<" ";
}

void dijkstra()
{
   int src;
   cout<<"Enter the Source Node : ";
   cin>>src;
   for (int i = 0; i < N; i++)
   {
      parent[0] = -1;
      dist[i] = 10000;
```

```
            visited[i] = false;
        }
        dist[src] = 0;
        for (int count = 0; count < N-1; count++)
        {
            int u = minDistance();
            visited[u] = true;
            for (int v = 0; v < N; v++)
                if (!visited[v] && graph[u][v] &&
                    dist[u] + graph[u][v] < dist[v])
                {
                    parent[v]  = u;
                    dist[v] = dist[u] + graph[u][v];
                }
        }
        cout<<"Src->Dest\tDistance\tPath"<<endl;
        for (int i = 1; i < N; i++)
        {
            cout<<src<<"->"<<i<<"\t\t"<<dist[i]<<"\t\t"<<src<<" ";
            printPath(i);
            cout<<endl;
        }
    }
    int main()
    {
        createGraph();
        dijkstra();
        return 0;
    }
```

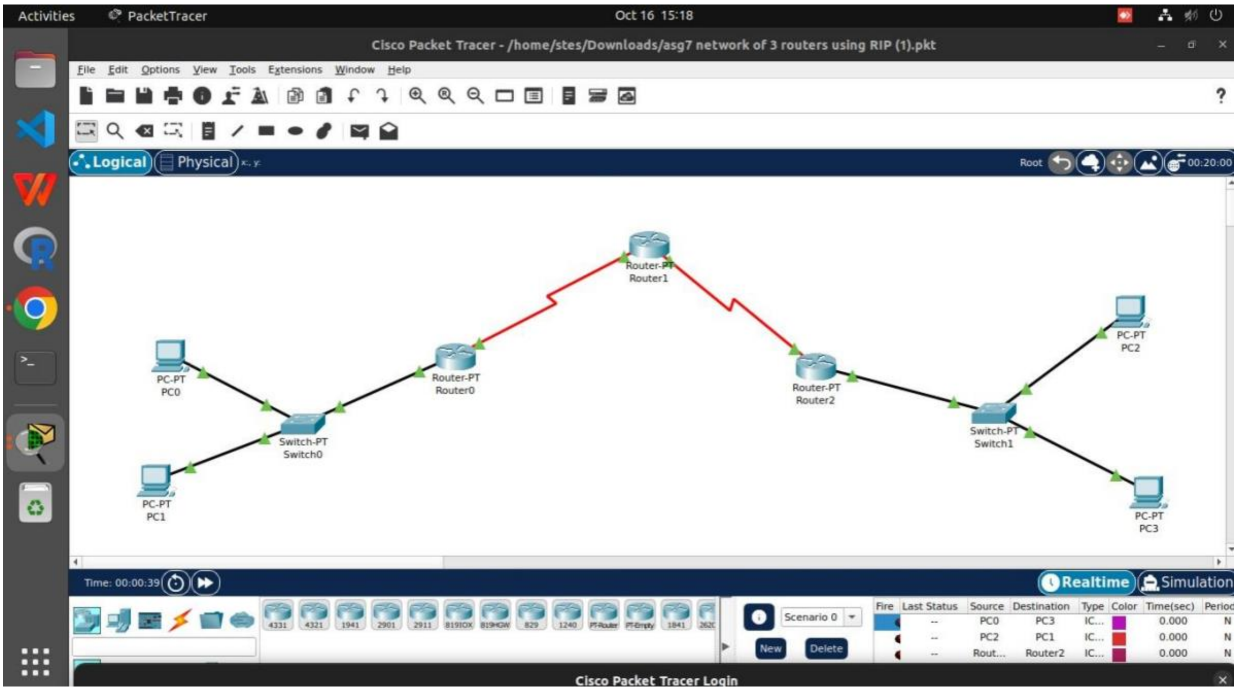
Output:-



```
(base) stes@stes:~/Downloads$ g++ main.cpp
(base) stes@stes:~/Downloads$ ./a.out
Enter the number of nodes : 4
Enter Edge and Weight : 0 1 4
Enter Edge and Weight : 1 2 5
Enter Edge and Weight : 2 3 6
Enter Edge and Weight : 0 3 5
Enter Edge and Weight : 1 3 4
Enter Edge and Weight : -1 -1 -1
Enter the Source Node : 0
Src->Dest       Distance        Path
0->1            4               0 1
0->2            9               0 1 2
0->3            5               0 3
(base) stes@stes:~/Downloads$ 
```

Assignment -B.8

Output:-



Cisco Packet Tracer - /home/stes/Downloads/asg7 network of 3 routers using RIP (1).pkt

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Period
●	--	PC0	PC3	IC...	🟪	0.000	N
●	--	PC2	PC1	IC...	🟥	0.000	N
●	--	Rout...	Router2	IC...	🟥	0.000	N

Assignment -B.9

Input:-

CLIENT PROGRAM

```
#include<sys/types.h>
#include<sys/socket.h>
#include<stdio.h>
#include<netinet/in.h>
#include <unistd.h>
#include<string.h>
#include<strings.h>
#include <arpa/inet.h>
//#define buffsize  150
void main()
{
int b,sockfd,sin_size,con,n,len;
//char buff[256];
char msg[25];
int connfd;
char operator,ch;
int op1,op2,result;
FILE *fp;
if((sockfd=socket(AF_INET,SOCK_STREAM,0))>0)
printf("socket created sucessfully\n");
//printf("%d\n", sockfd);
struct sockaddr_in servaddr;
servaddr.sin_family=AF_INET;
servaddr.sin_addr.s_addr=inet_addr("127.0.0.1"); //LOOPBACK ADDR
servaddr.sin_port=6006;
sin_size = sizeof(struct sockaddr_in);
if((con=connect(sockfd,(struct sockaddr *) &servaddr, sin_size))==0); //initiate a connection on a socket
printf("connect sucessful\n");
//1 send and receive hello
write(sockfd,"HELLO FROM CLIENT", sizeof("HELLO FROM CLIENT"));
read(sockfd, &msg,sizeof(msg));
printf("%s\n",msg);
//2 To receive file
fp = fopen("b.txt", "w");
read(sockfd,&ch,sizeof(ch));
while(ch!=EOF)
{
   fputc(ch,fp);
   read(sockfd,&ch,sizeof(ch));
}
printf("FILE RECEIVED\n");
fclose(fp);
//3 maths ops
printf("Enter operation:\n +:Addition \n -: Subtraction \n /: Division \n*:Multiplication \n");
scanf("%c",&operator);
printf("Enter operands:\n");
scanf("%d %d", &op1, &op2);
write(sockfd,&operator,10);
write(sockfd,&op1,sizeof(op1));
write(sockfd,&op2,sizeof(op2));
read(sockfd,&result,sizeof(result));
printf("Operation result from server=%d\n",result);
close(sockfd);
```

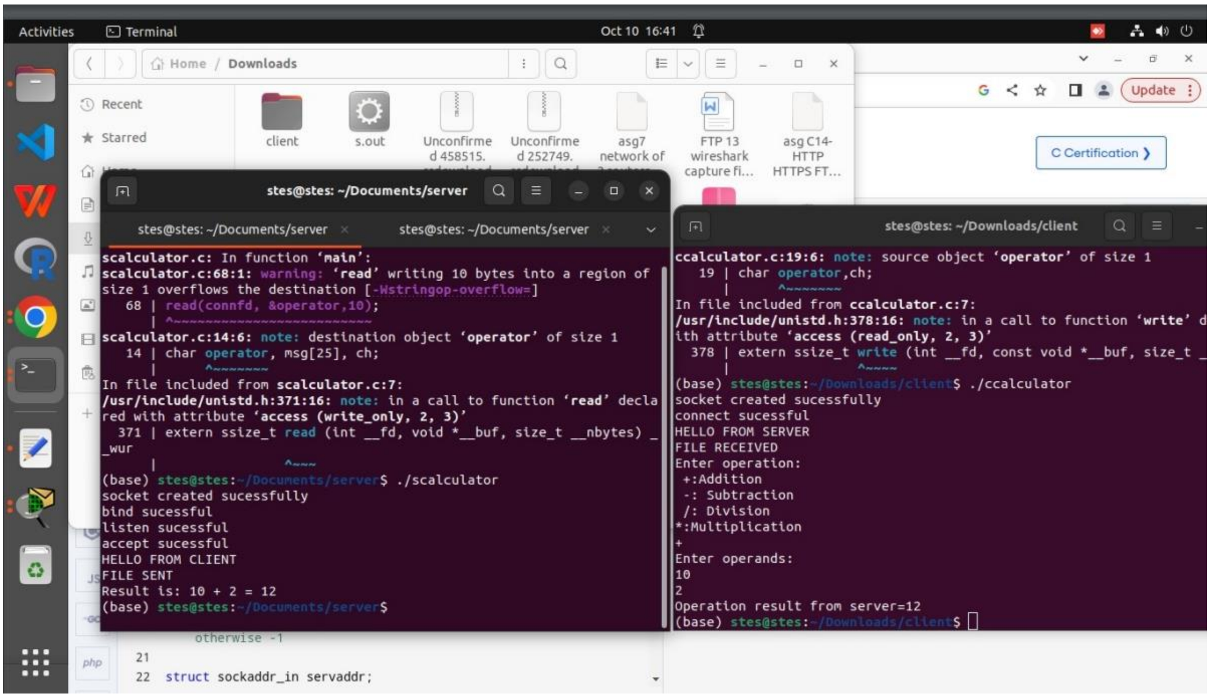
```
}
```

SERVER PROGRAM

```
#include<sys/types.h>
#include<sys/socket.h>
#include<stdio.h>
#include<netinet/in.h>
#include <unistd.h>
#include<string.h>
#include <arpa/inet.h>
void main()
{
int b,sockfd,connfd,sin_size,l,n,len;
char operator, msg[25], ch;
int op1,op2,result;
FILE * fp;

if((sockfd=socket(AF_INET,SOCK_STREAM,0))>0)
printf("socket created sucessfully\n");  //socket creation
//printf("%d\n", sockfd);                //on success 0 otherwise -1
struct sockaddr_in servaddr;
struct sockaddr_in clientaddr;
servaddr.sin_family=AF_INET;
servaddr.sin_addr.s_addr=inet_addr("127.0.0.1");
servaddr.sin_port=6006;
if((bind(sockfd, (struct sockaddr *)&servaddr,sizeof(servaddr)))==0)
printf("bind sucessful\n");   //bind() assigns the
   // address specified by addr to the socket referred to by the file
    // descriptor sockfd. addrlen specifies the size, in bytes, of the
   // address structure pointed to by addr. Traditionally, this operation is
    // called “assigning a name to a socket”.
```

```
//printf("%d\n",b);


if((listen(sockfd,5))==0) //listen for connections on a socket,5 requests server can handle
printf("listen sucessful\n");
//printf("%d\n",l);



sin_size = sizeof(struct sockaddr_in);
if((connfd=accept(sockfd,(struct sockaddr *)&clientaddr,&sin_size))>0);//connection failed descriptor
printf("accept sucessful\n");
//To receive hello msg from client
read(connfd, &msg,sizeof(msg));
printf("%s\n",msg);
write(connfd,"HELLO FROM SERVER", sizeof("HELLO FROM SERVER"));
//To send file
fp = fopen("a.txt", "r");
ch=fgetc(fp);
while(ch!=EOF)
{
    write(connfd,&ch,sizeof(ch));
    ch=fgetc(fp);
}
write(connfd,&ch,sizeof(ch));
printf("FILE SENT\n");
fclose(fp);
//Maths operations
read(connfd, &operator,10);
read(connfd,&op1,sizeof(op1));
read(connfd,&op2,sizeof(op2));
switch(operator) {
        case '+': result=op1 + op2;
         printf("Result is: %d + %d = %d\n",op1, op2, result);
         break;
        case '-':result=op1 - op2;
                printf("Result is: %d - %d = %d\n",op1, op2, result);
                break;
        case '*':result=op1 * op2;
                printf("Result is: %d * %d = %d\n",op1, op2, result);
                break;
        case '/':result=op1 / op2;
                printf("Result is: %d / %d = %d\n",op1, op2, result);
                break;
        default:
                printf("ERROR: Unsupported Operation");
    }
  write(connfd,&result,sizeof(result));
close(sockfd);
}
```

Output:-



```
scalculator.c: In function ‘main’:
scalculator.c:68:1: warning: ‘read’ writing 10 bytes into a region of
size 1 overflows the destination [-Wstringop-overflow=]
   68 | read(connfd, &operator,10);
      | ^~~~~~~~~~~~~~~~~~~~~~~~~~
scalculator.c:14:6: note: destination object ‘operator’ of size 1
   14 | char operator, msg[25], ch;
      |      ^~~~~~~~
In file included from scalculator.c:7:
/usr/include/unistd.h:371:16: note: in a call to function ‘read’ decla
red with attribute ‘access (write_only, 2, 3)’
  371 | extern ssize_t read (int __fd, void *__buf, size_t __nbytes) _
_wur
      |                ^~~~
(base) stes@stes:~/Documents/server$ ./scalculator
socket created sucessfully
bind sucessful
listen sucessful
accept sucessful
HELLO FROM CLIENT
FILE SENT
Result is: 10 + 2 = 12
(base) stes@stes:~/Documents/server$
```

```
ccalculator.c:19:6: note: source object ‘operator’ of size 1
   19 | char operator,ch;
      |      ^~~~~~~~
In file included from ccalculator.c:7:
/usr/include/unistd.h:378:16: note: in a call to function ‘write’ d
ith attribute ‘access (read_only, 2, 3)’
  378 | extern ssize_t write (int __fd, const void *__buf, size_t _
      |                ^~~~~
(base) stes@stes:~/Downloads/client$ ./ccalculator
socket created sucessfully
connect sucessful
HELLO FROM SERVER
FILE RECEIVED
Enter operation:
 +:Addition
 -: Subtraction
 /: Division
*:Multiplication
+
Enter operands:
10
2
Operation result from server=12
(base) stes@stes:~/Downloads/client$
```

```
     otherwise -1
21
22  struct sockaddr_in servaddr;
```

Assignment -B.10

Input:-

socser2.c

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
#include<arpa/inet.h>
#include<sys/socket.h>
#define BUFLEN 503
#define PORT 8885
void die(char *s)
{
   perror(s);
   exit(1);
}
int main(void)
{
   struct sockaddr_in si_me, si_other;
    int s, i,j, slen = sizeof(si_other) , recv_len;
   char buf[BUFLEN];
    if ((s=socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP)) == -1)
   {
      die("socket");
   }
   memset((char *) &si_me, 0, sizeof(si_me));
    si_me.sin_family = AF_INET;
   si_me.sin_port = htons(PORT);
   si_me.sin_addr.s_addr = htonl(INADDR_ANY);
```

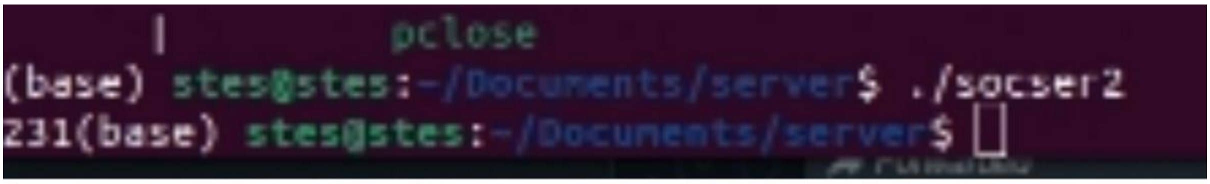
```
if( bind(s , (struct sockaddr*)&si_me, sizeof(si_me) ) == -1)
{
    die("bind");
}
  char fname[20];
FILE *fp;
recv_len = recvfrom(s, buf, 20, 0, (struct sockaddr *) &si_other, &slen);
char fna[100];
strcpy(fna,buf);
//printf("------%s",buf);
memset(buf,0,503);
recv_len = recvfrom(s, buf, 20, 0, (struct sockaddr *) &si_other, &slen);
//printf("%s",buf);
 int len= strlen(fna);
//printf("%d",len);
for(j=len-1;j>=0;j--)
{
    if(fna[j]=='.')
    {
        fna[j-1]='1';
    }
}
unsigned long mm = atoi(buf);
fp=fopen(fna,"wb");
int itr=1;
memset(buf,0,503);
while(itr*503<mm)
{
    if ((recv_len = recvfrom(s, buf, 503, 0, (struct sockaddr *) &si_other, &slen)) == -1)
    {
            die("recvfrom()");
        }
        fwrite(buf,503, 1, fp);
        memset(buf,0,503);
        itr++;
    }
    printf("%d",(mm%503));
    recv_len = recvfrom(s, buf, (mm%503), 0, (struct sockaddr *) &si_other, &slen);
    fwrite(buf,(mm%503), 1, fp);
    memset(buf,0,503);
    fclose(fp);
    close(s);
     return 0;
}
```

soccli2.c

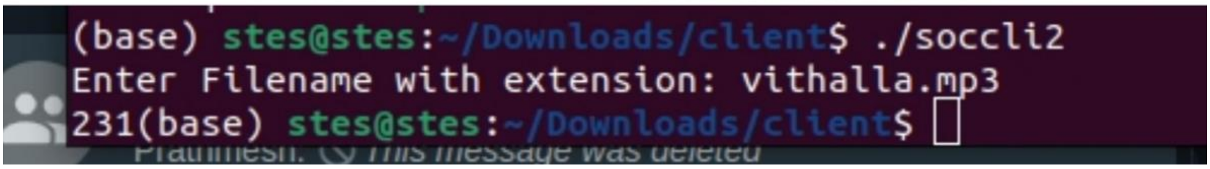
```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
#include<arpa/inet.h>
#include<sys/socket.h>
#define SERVER "127.0.0.1"
#define BUFLEN 503
#define PORT 8885
void die(char *s)
{
   perror(s);
   exit(1);
}
unsigned long fsize(char* file)
{
   FILE * f = fopen(file, "r");
   fseek(f, 0, SEEK_END);
   unsigned long len = (unsigned long)ftell(f);
   fclose(f);
   return len;
}
int main(void)
{
   struct sockaddr_in si_other;
   int s, i, slen=sizeof(si_other);
   char buf[BUFLEN];
   char message[BUFLEN];
```

```
if ( (s=socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP)) == -1)
{
    die("socket");
}

memset((char *) &si_other, 0, sizeof(si_other));
si_other.sin_family = AF_INET;
si_other.sin_port = htons(PORT);

if (inet_aton(SERVER , &si_other.sin_addr) == 0)
{
    fprintf(stderr, "inet_aton() failed\n");
    exit(1);
}
char fname[20];
printf("Enter Filename with extension: ");
scanf("%s",&fname);
sendto(s, fname, 20 , 0 , (struct sockaddr *) &si_other, slen);
memset(message,0,503);
unsigned long siz = fsize(fname);
printf("%ld",(siz % 503));
char str[10];
sprintf(str, "%d", siz);
sendto(s, str, 20 , 0 , (struct sockaddr *) &si_other, slen);
FILE *f;
f=fopen(fname,"rb");
memset(message,0,503);
fread(message, 503,1,f);

int itr =1;
while(itr*503<siz){
     if (sendto(s, message, 503 , 0 , (struct sockaddr *) &si_other, slen)==-1)
    {
      die("sendto()");
    }
    memset(message,0,503);
    fread(message, 503,1,f);
    itr++;
  }
  fread(message, (siz % 503),1,f);
  sendto(s, message, (siz % 503) , 0 , (struct sockaddr *) &si_other, slen);
  memset(message,0,503);
  fclose(f);
  close(s);
  return 0;
}
```

Output:-



```
     |          pclose
(base) stes@stes:~/Documents/server$ ./socser2
231(base) stes@stes:~/Documents/server$ 
```



```
(base) stes@stes:~/Downloads/client$ ./soccli2
Enter Filename with extension: vithalla.mp3
231(base) stes@stes:~/Downloads/client$ 
```

Prathmesh: This message was deleted

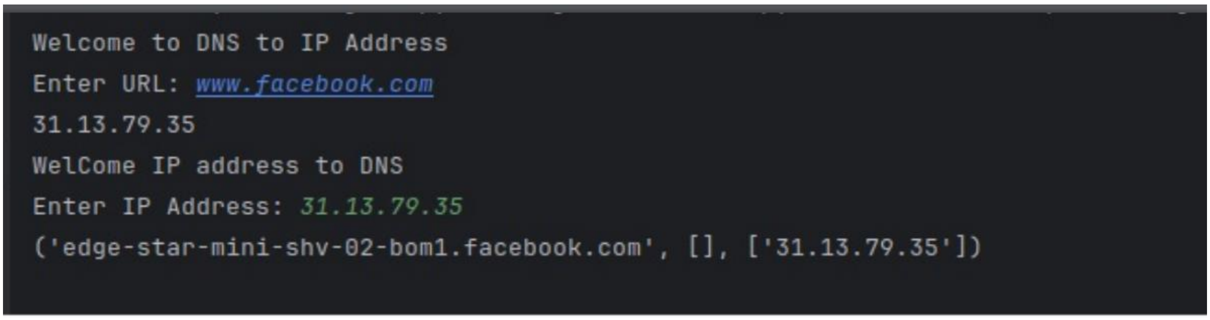
Assignment -C.11

Input:-

```
import socket
print 'Welcome to DNS to IP Address'
URL=raw_input('Enter URL: ')
addr1 = socket.gethostbyname(URL)
print(addr1)
print 'WelCome IP address to DNS'
IP=raw_input('Enter IP Address: ')
addr6=socket.gethostbyaddr(IP)
print addr6
```

```
***D:\>python DNS.py
Welcome to DNS to IP Address
Enter URL: www.google.com
142.250.67.132
WelCome IP address to DNS
Enter IP Address: 142.250.67.132
('bom12s06-in-f4.1e100.net', [], ['142.250.67.132'])
D:\>python DNS.py
Welcome to DNS to IP Address
Enter URL: sinhgad.edu
115.113.155.115
WelCome IP address to DNS
Enter IP Address: 115.113.155.115
('115.113.155.115.static-pune.vsnl.net.in', [], ['115.113.155.115'])

D:\>***
```

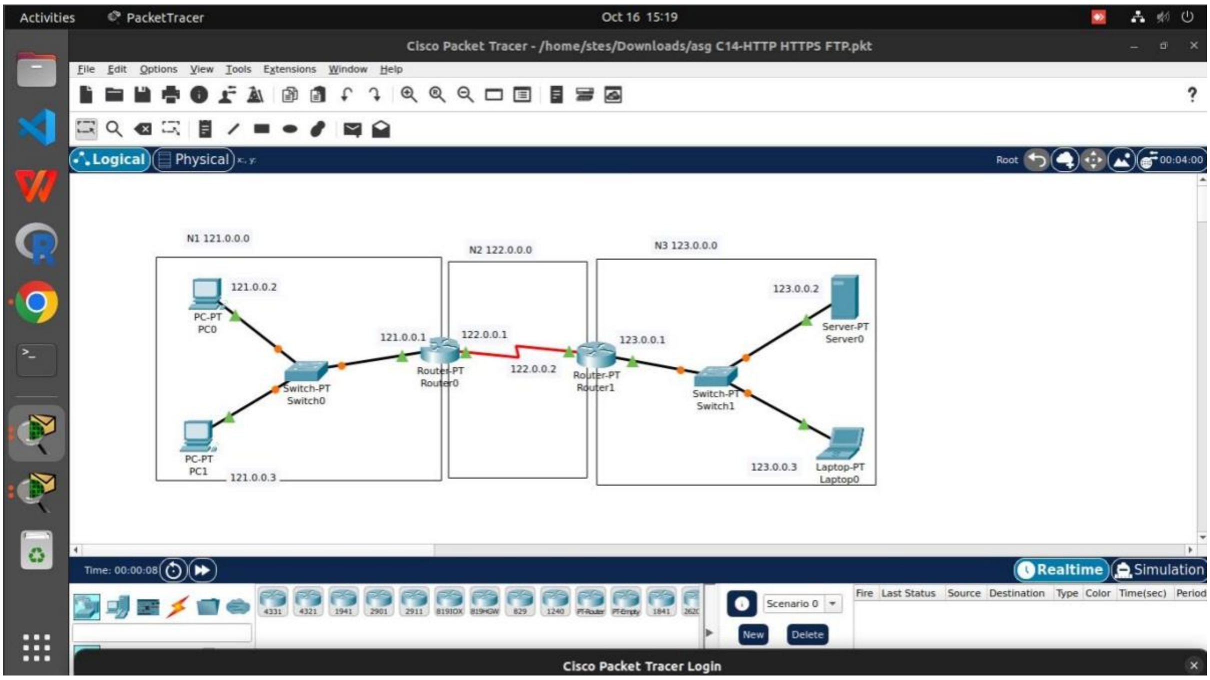
Output:-



```
Welcome to DNS to IP Address
Enter URL: www.facebook.com
31.13.79.35
WelCome IP address to DNS
Enter IP Address: 31.13.79.35
('edge-star-mini-shv-02-bom1.facebook.com', [], ['31.13.79.35'])
```

Assignment -C.13

Output:-


Activities
PacketTracer
Oct 16 15:19
Cisco Packet Tracer - /home/stes/Downloads/asg C14-HTTP HTTPS FTP.pkt
File Edit Options View Tools Extensions Window Help
Logical
Physical
Root
00:04:00
N1 121.0.0.0
N2 122.0.0.0
N3 123.0.0.0
121.0.0.2
PC-PT
PC0
PC-PT
PC1
121.0.0.3
Switch-PT
Switch0
121.0.0.1
Router-PT
Router0
122.0.0.1
122.0.0.2
Router-PT
Router1
123.0.0.1
Switch-PT
Switch1
123.0.0.2
Server-PT
Server0
123.0.0.3
Laptop-PT
Laptop0
Time: 00:00:08
Realtime
Simulation
Scenario 0
New
Delete
Fire
Last Status
Source
Destination
Type
Color
Time(sec)
Periodic
Cisco Packet Tracer Login