CRC – 1
CHECKSUM – 4
HAMMING CODE – 7
STOP AND WAIT – 10
GO AND BACK – 13
SELECTIVE REPEAT – 15
IP ADDRESSING – 19
TCP – 23
UDP – 28
NS-2 TCP – 32
NS-2 UDP – 40

Networking and Communication Lab Experiment

Name: Om Ashish Mishra
Registration Number: 16BCE0789
Slot: D2

CRC

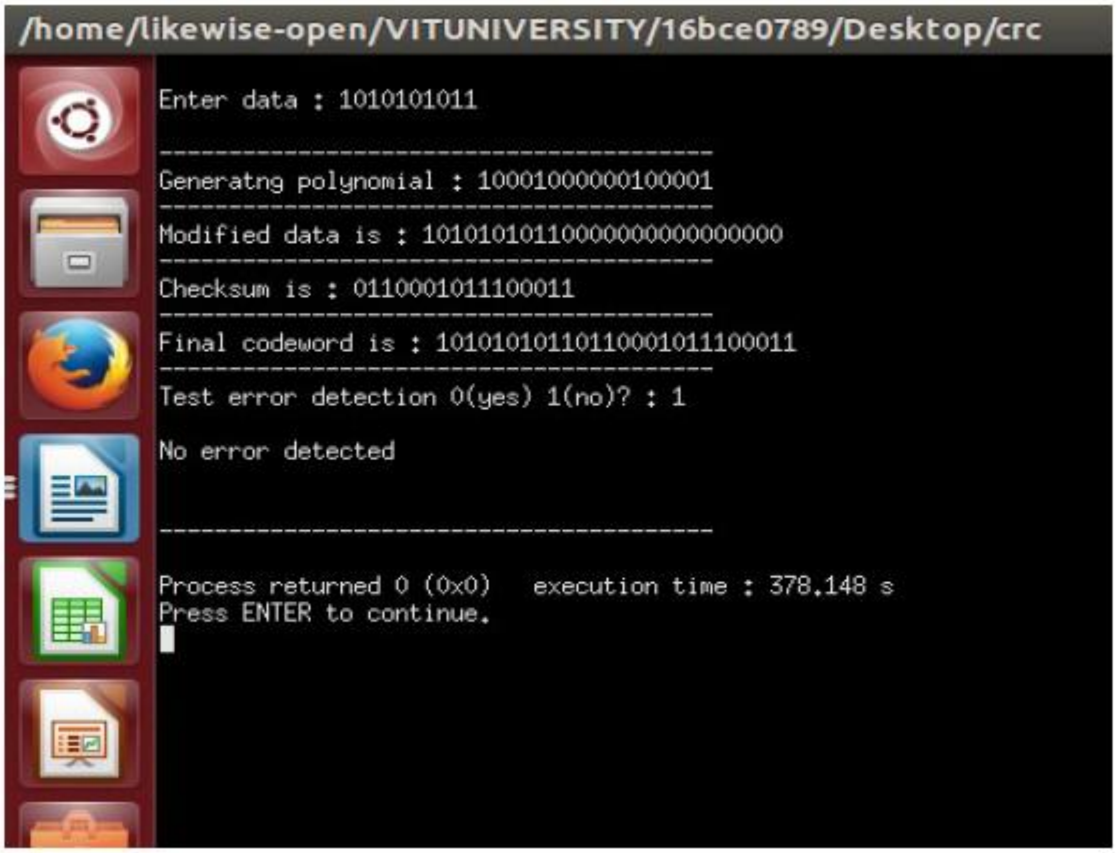
The Code

```
#include<stdio.h>
#include<string.h>
#define N strlen(g)
char t[28],cs[28],g[]="10001000000100001";
int a,e,c;
void exor(){
for(c = 1;c < N; c++)
cs[c] = (( cs[c] == g[c])?'0':'1');
}
void crc(){
for(e=0;e<N;e++)
cs[e]=t[e];
do{
if(cs[0]=='1')
exor();
for(c=0;c<N-1;c++)
cs[c]=cs[c+1];
cs[c]=t[e++];
}while(e<=a+N-1);
```

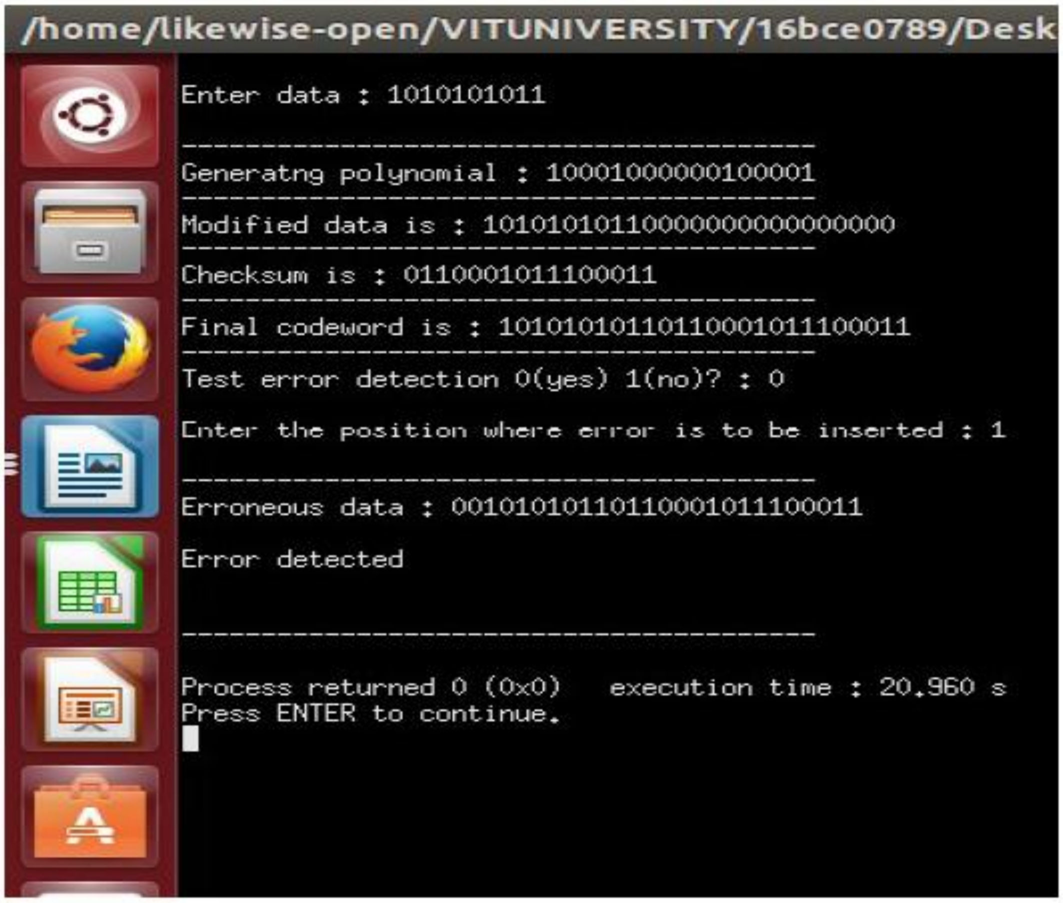
```
}
int main()
{
printf("\nEnter data : ");
scanf("%s",t);
printf("\n----------------------------------------");
printf("\nGeneratng polynomial : %s",g);
a=strlen(t);
for(e=a;e<a+N-1;e++)
t[e]='0';
printf("\n----------------------------------------");
printf("\nModified data is : %s",t);
printf("\n----------------------------------------");
crc();
printf("\nChecksum is : %s",cs);
for(e=a;e<a+N-1;e++)
t[e]=cs[e-a];
printf("\n----------------------------------------");
printf("\nFinal codeword is : %s",t);
printf("\n----------------------------------------");
printf("\nTest error detection 0(yes) 1(no)? : ");
scanf("%d",&e);
if(e==0)
{
do{
printf("\nEnter the position where error is to be inserted : ");
scanf("%d",&e);
}while(e==0 || e>a+N-1);
t[e-1]=(t[e-1]=='0')?'1':'0';
printf("\n----------------------------------------");
printf("\nErroneous data : %s\n",t);
}
crc();
for(e=0;(e<N-1) && (cs[e]!='1');e++);
if(e<N-1)
printf("\nError detected\n\n");
else
printf("\nNo error detected\n\n");
printf("\n----------------------------------------\n");
return 0;
```

```
}
```

The Output:



```
/home/likewise-open/VITUNIVERSITY/16bce0789/Desktop/crc

Enter data : 1010101011

----------------------------------------
Generatng polynomial : 10001000000100001
----------------------------------------
Modified data is : 10101010110000000000000000
----------------------------------------
Checksum is : 0110001011100011
----------------------------------------
Final codeword is : 10101010110110001011100011
----------------------------------------
Test error detection 0(yes) 1(no)? : 1

No error detected


----------------------------------------

Process returned 0 (0x0)   execution time : 378.148 s
Press ENTER to continue.
```



```
/home/likewise-open/VITUNIVERSITY/16bce0789/Desk
Enter data : 1010101011

----------------------------------------
Generatng polynomial : 10001000000100001
----------------------------------------
Modified data is : 10101010110000000000000000
----------------------------------------
Checksum is : 0110001011100011
----------------------------------------
Final codeword is : 10101010110110001011100011
----------------------------------------
Test error detection 0(yes) 1(no)? : 0

Enter the position where error is to be inserted : 1

----------------------------------------
Erroneous data : 00101010110110001011100011

Error detected


----------------------------------------

Process returned 0 (0x0)   execution time : 20.960 s
Press ENTER to continue.
```

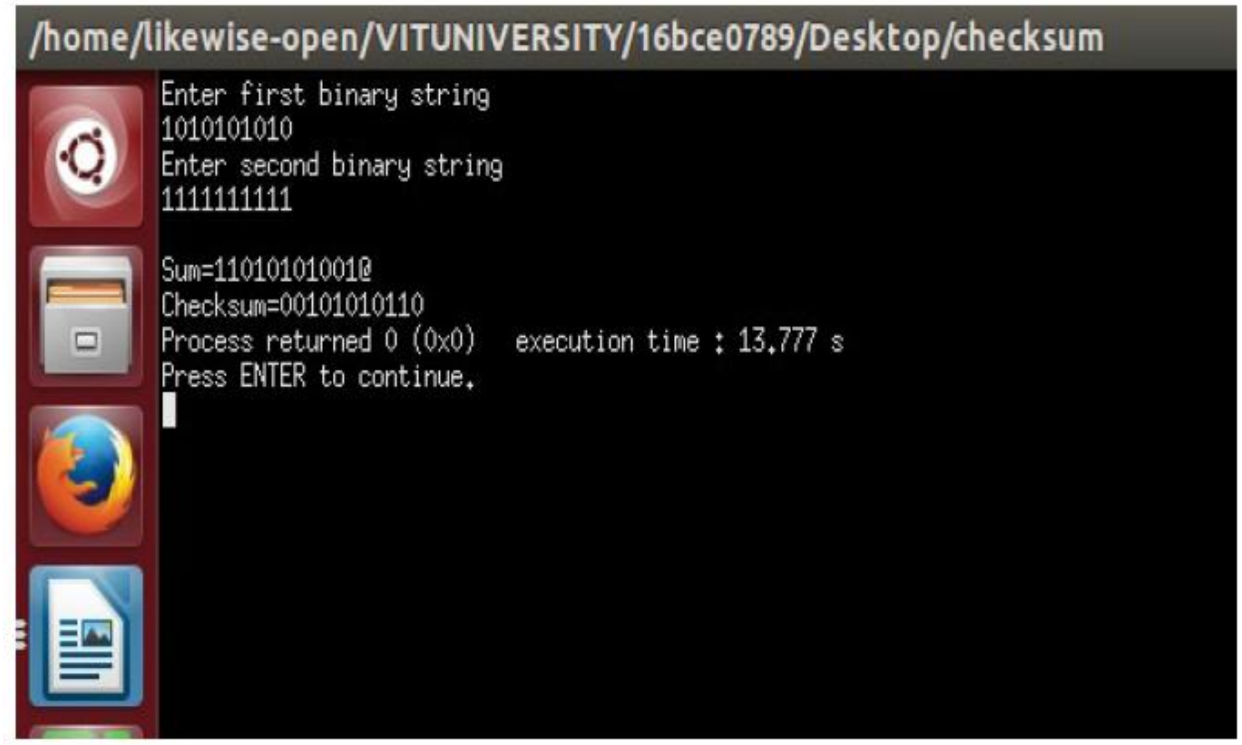
Check Sum

The Code:

```
#include<iostream>
#include<string.h>
using namespace std;
int main()
{
char a[20],b[20];
char sum[20],complement[20];
int i;
cout<<"Enter first binary string\n";
cin>>a;
cout<<"Enter second binary string\n";
cin>>b;
if(strlen(a)==strlen(b))
```

```
{
char carry='0';
int length=strlen(a);
for(i=length-1;i>=0;i--)
{
if(a[i]=='0' && b[i]=='0' && carry=='0')
{
sum[i]='0';
carry='0';
}
else if(a[i]=='0' && b[i]=='0' && carry=='1')
{
sum[i]='1';
carry='0';
}
else if(a[i]=='0' && b[i]=='1' && carry=='0')
{
sum[i]='1';
carry='0';
}
else if(a[i]=='0' && b[i]=='1' && carry=='1')
{
sum[i]='0';
carry='1';
}
else if(a[i]=='1' && b[i]=='0' && carry=='0')
{
sum[i]='1';
carry='0';
}
else if(a[i]=='1' && b[i]=='0' && carry=='1')
{
sum[i]='0';
carry='1';
}
else if(a[i]=='1' && b[i]=='1' && carry=='0')
{
sum[i]='0';
carry='1';
}
else if(a[i]=='1' && b[i]=='1' && carry=='1')
{
sum[i]='1';
carry='1';
}
else
break;
}
cout<<"\nSum="<<carry<<sum;
for(i=0;i<length;i++)
{
if(sum[i]=='0')
complement[i]='1';
else
complement[i]='0';
}
if(carry=='1')
carry='0';
else
carry='1';
cout<<"\nChecksum="<<carry<<complement;
}
else
cout<<"\nWrong input strings";
return 0;
}
```

The Output:



```
/home/likewise-open/VITUNIVERSITY/16bce0789/Desktop/checksum
Enter first binary string
1010101010
Enter second binary string
1111111111

Sum=11010101001@
Checksum=00101010110
Process returned 0 (0x0)   execution time : 13.777 s
Press ENTER to continue.
```

Hamming Code:

The Code:

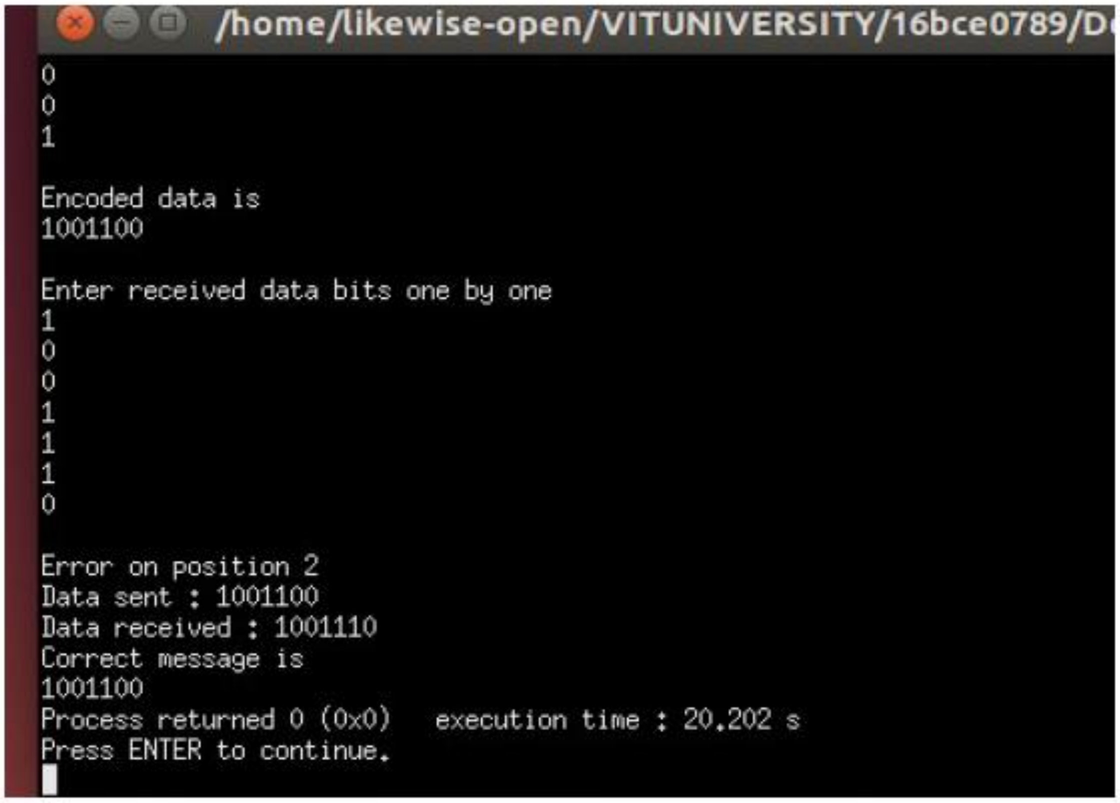
```
#include<iostream>
using namespace std;
int main() {
int data[10];
int dataatrec[10],c,c1,c2,c3,i;
cout<<"Enter 4 bits of data one by one\n";
cin>>data[0];
cin>>data[1];
cin>>data[2];
cin>>data[4];
//Calculation of even parity
data[6]=data[0]^data[2]^data[4];
data[5]=data[0]^data[1]^data[4];
data[3]=data[0]^data[1]^data[2];
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];
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(dataatrec[7-c]==0)
dataatrec[7-c]=1;
else
dataatrec[7-c]=0;
for (i=0;i<7;i++) {
cout<<dataatrec[i];
}
}
return 0;
}
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(dataatrec[7-c]==0)
dataatrec[7-c]=1;
else
dataatrec[7-c]=0;
for (i=0;i<7;i++) {
cout<<dataatrec[i];
```

```
}
}
return 0;
}
```

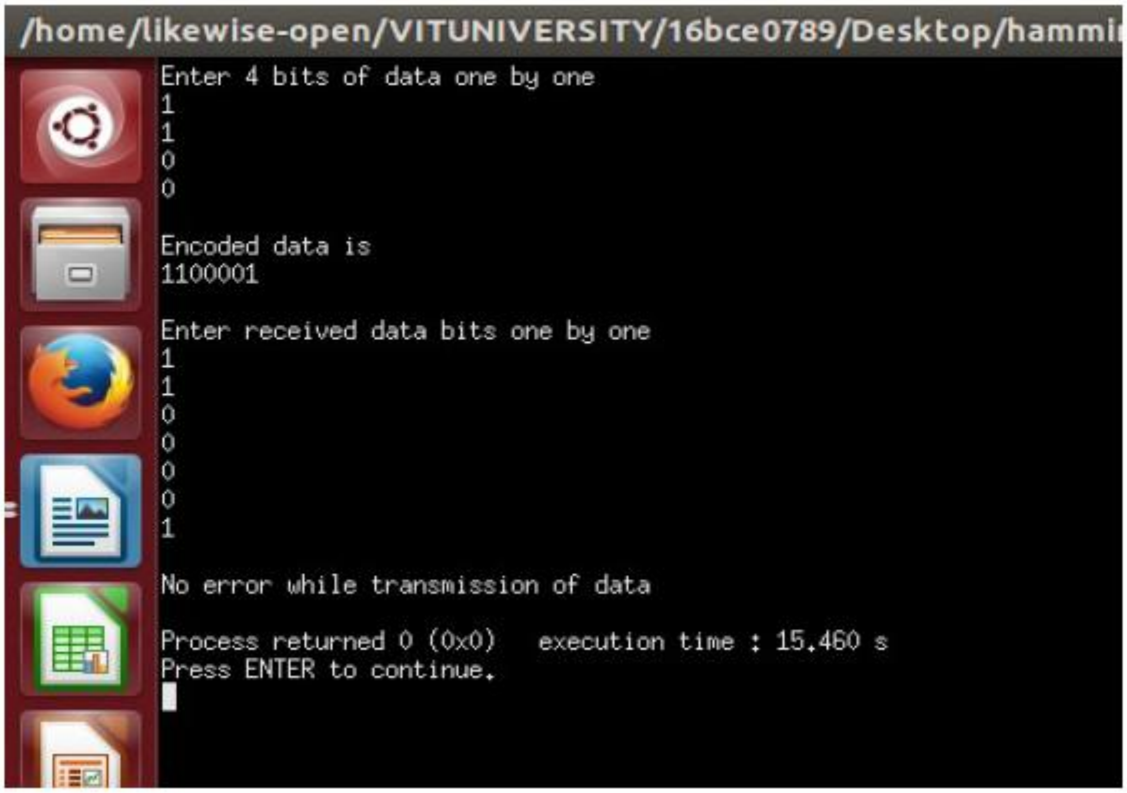
The Output:

With Error:



```
/home/likewise-open/VITUNIVERSITY/16bce0789/D
0
0
1

Encoded data is
1001100

Enter received data bits one by one
1
0
0
1
1
1
0

Error on position 2
Data sent : 1001100
Data received : 1001110
Correct message is
1001100
Process returned 0 (0x0)   execution time : 20.202 s
Press ENTER to continue.
```

Without Error:



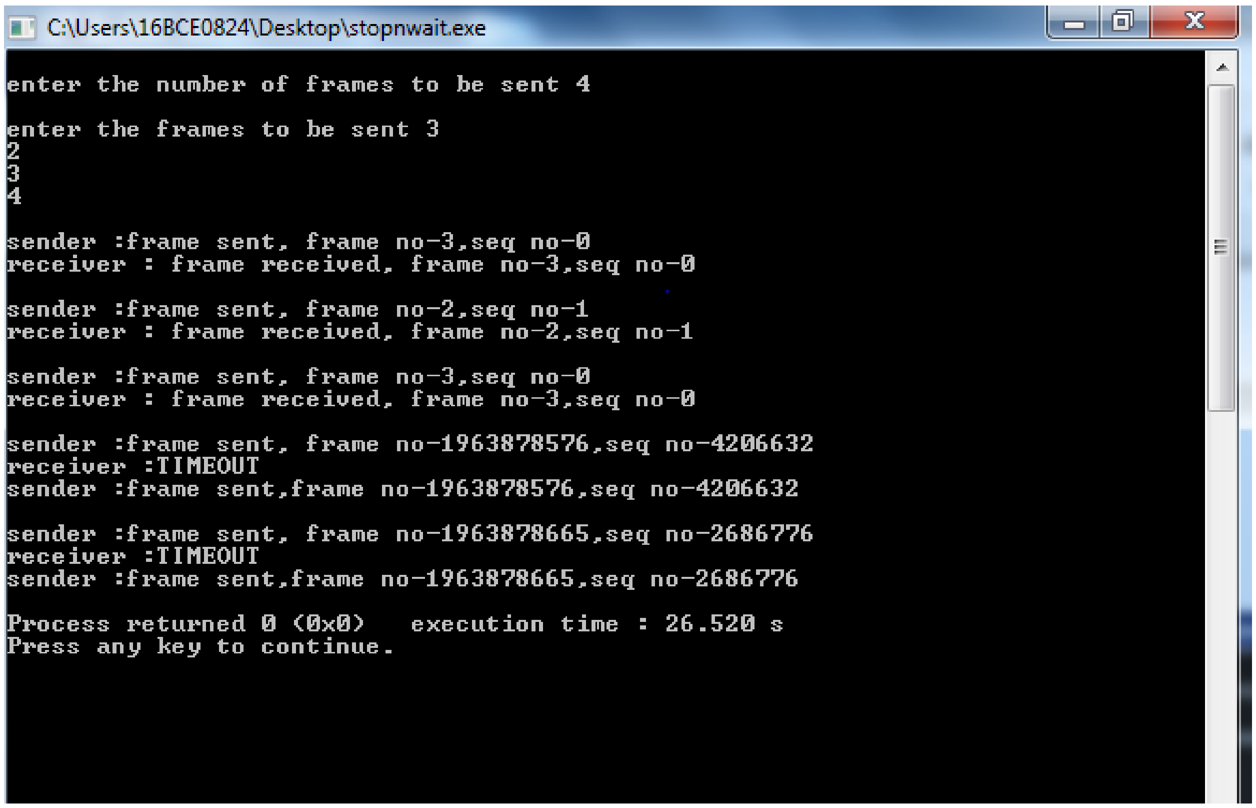
```
/home/likewise-open/VITUNIVERSITY/16bce0789/Desktop/hammi
Enter 4 bits of data one by one
1
1
0
0

Encoded data is
1100001

Enter received data bits one by one
1
1
0
0
0
0
1

No error while transmission of data

Process returned 0 (0x0)   execution time : 15.460 s
Press ENTER to continue.
```

Stop and wait :-

```
#include<stdlib.h>

#include<stdio.h>

int main()

{

int i,j,k,fnum,random,state=0;

printf("\nenter the number of frames to be sent ");
scanf("%d",&fnum);

int seq[fnum];

for(i=0;i<fnum;i++)

{
   if(state==0)

   {seq[i]=0;state=1;}

   else if(state==1)

   {seq[i]=1;state=0;}

}

int frames[fnum];

printf("\nenter the frames to be sent ");

for(i=0;i<fnum;i++)

scanf("%d",&frames[i]);

int rece[fnum];

srand(time(0));

random=rand()%100;

for(i=0;i<3;i++)

{printf("\nsender :frame sent, frame no-%d,seq no-%d",frames[i],seq[i]);
 printf("\nreceiver : frame received, frame no-%d,seq no-%d\n",frames[i],seq[i]);}

i++;

if(random%5==0 || random%3==0)


{

printf("\nsender :frame sent, frame no-%d,seq no-%d",frames[i],seq[i]);

printf("\nreceiver : garbled frame ");


printf("\nsender :frame sent,frame no-%d,seq no-%d\n",frames[i],seq[i]);}

else

{printf("\nsender :frame sent, frame no-%d,seq no-%d",frames[i],seq[i]);
```

```
printf("\nreceiver :TIMEOUT ");

printf("\nsender :frame sent,frame no-%d,seq no-%d\n",frames[i],seq[i]);}

for(i=4;i<fnum-2;i++)


{printf("\nsender :frame sent, frame no-%d,seq no-%d",frames[i],seq[i]);
 printf("\nreceiver : frame received, frame no-%d,seq no-%d\n",frames[i],seq[i]);

}

i++;

if(random%7==0 || random%13==0)


{

printf("\nsender :frame sent, frame no-%d,seq no-%d",frames[i],seq[i]);

printf("\nreceiver : garbled frame ");


printf("\nsender :frame sent,frame no-%d,seq no-%d\n",frames[i],seq[i]);

}

else

{

printf("\nsender :frame sent, frame no-%d,seq no-%d",frames[i],seq[i]);

printf("\nreceiver :TIMEOUT ");

printf("\nsender :frame sent,frame no-%d,seq no-%d\n",frames[i],seq[i]);}


return 0;

}
```

OUTPUT :



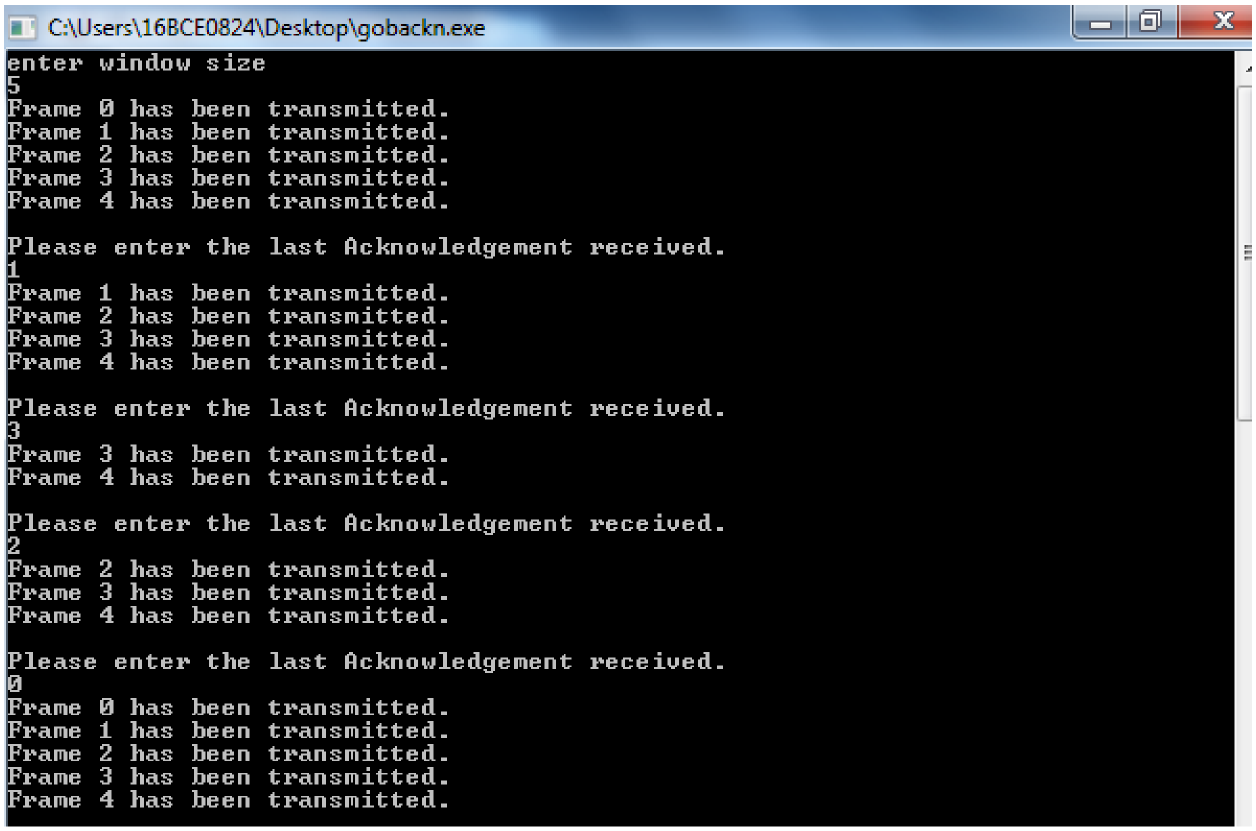
```
enter the number of frames to be sent 4

enter the frames to be sent 3
2
3
4

sender :frame sent, frame no-3,seq no-0
receiver : frame received, frame no-3,seq no-0

sender :frame sent, frame no-2,seq no-1
receiver : frame received, frame no-2,seq no-1

sender :frame sent, frame no-3,seq no-0
receiver : frame received, frame no-3,seq no-0

sender :frame sent, frame no-1963878576,seq no-4206632
receiver :TIMEOUT
sender :frame sent,frame no-1963878576,seq no-4206632

sender :frame sent, frame no-1963878665,seq no-2686776
receiver :TIMEOUT
sender :frame sent,frame no-1963878665,seq no-2686776

Process returned 0 (0x0)   execution time : 26.520 s
Press any key to continue.
```

Go and back

Code:

```
#include<stdio.h>
int main()
{
int windowsize,sent=0,ack,i;
printf("enter window size\n");
scanf("%d",&windowsize);
while(1)
{
for( i = 0; i < windowsize; i++)
{
printf("Frame %d has been transmitted.\n",sent);
sent++;
if(sent == windowsize)
break;
}
printf("\nPlease enter the last Acknowledgement received.\n");
scanf("%d",&ack);
if(ack == windowsize)
break;
else
sent = ack;
}
return 0;
}
```

OUTPUT


C:\Users\16BCE0824\Desktop\gobackn.exe

```
enter window size
5
Frame 0 has been transmitted.
Frame 1 has been transmitted.
Frame 2 has been transmitted.
Frame 3 has been transmitted.
Frame 4 has been transmitted.

Please enter the last Acknowledgement received.
1
Frame 1 has been transmitted.
Frame 2 has been transmitted.
Frame 3 has been transmitted.
Frame 4 has been transmitted.

Please enter the last Acknowledgement received.
3
Frame 3 has been transmitted.
Frame 4 has been transmitted.

Please enter the last Acknowledgement received.
2
Frame 2 has been transmitted.
Frame 3 has been transmitted.
Frame 4 has been transmitted.

Please enter the last Acknowledgement received.
0
Frame 0 has been transmitted.
Frame 1 has been transmitted.
Frame 2 has been transmitted.
Frame 3 has been transmitted.
Frame 4 has been transmitted.
```

Selective repeat:

Code:

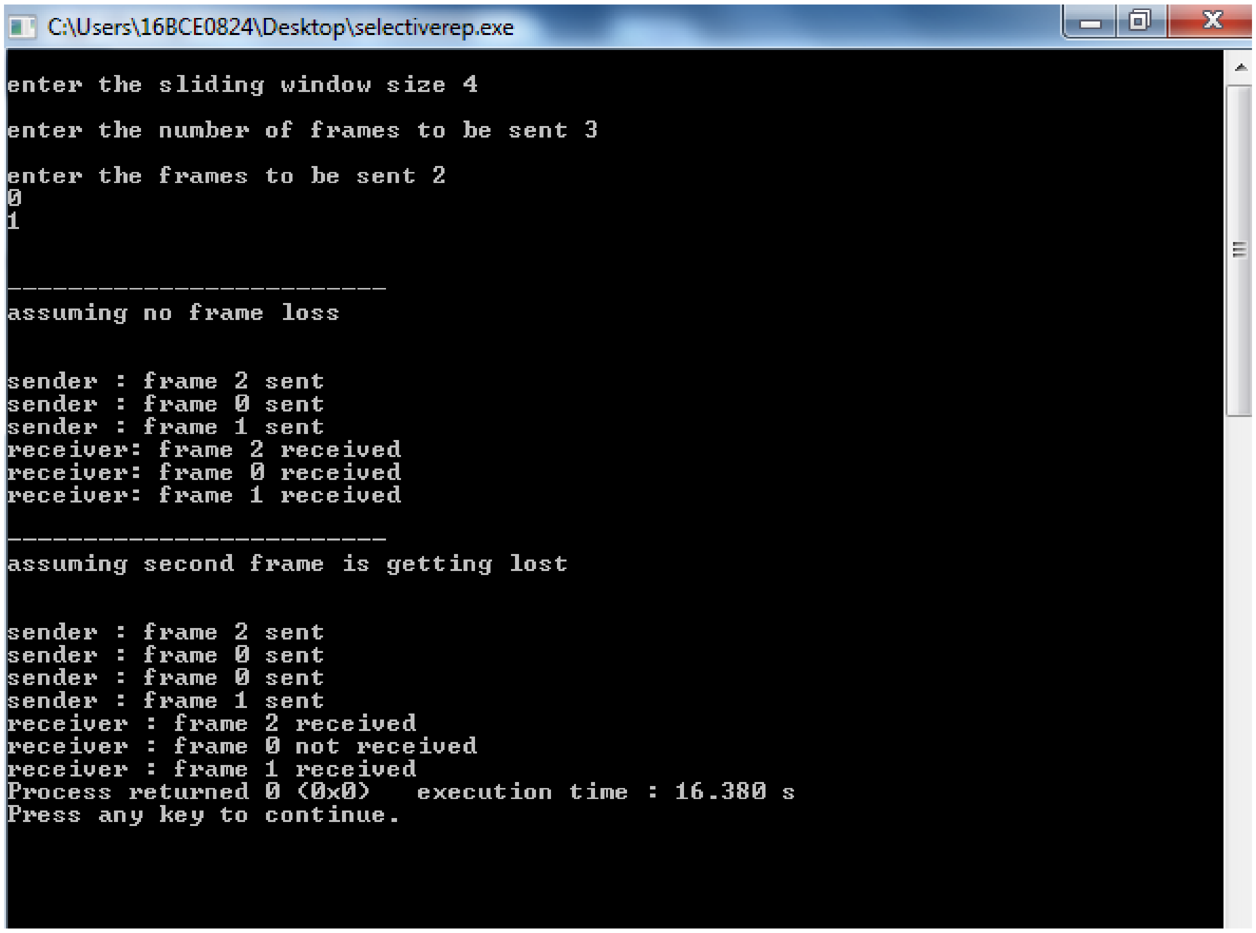
```
#include<stdio.h>
int main()
{int i,j=0,k,sw,fnum;
printf("\nenter the sliding window size ");
scanf("%d",&sw);
printf("\nenter the number of frames to be sent ");
scanf("%d",&fnum);
int frames[fnum];
printf("\nenter the frames to be sent ");
for(i=0;i<fnum;i++)
scanf("%d",&frames[i]);
printf("\n\n------------------------\nassuming no frame loss\n\n");
i=0;
while((j*sw)<fnum && i<fnum)
```

```
{
k=(i-(j*sw))/sw;
if(k==0)
{
printf("\nsender : frame %d sent ",frames[i]);
i++;
}
else
{
printf("\n\nwindow needs to be moved ");
j++;

}
}
i=0;j=0;
while((j*sw)<fnum && i<fnum)
{
k=(i-(j*sw))/sw;
if(k==0)
{
printf("\nreceiver: frame %d received ",frames[i]);
i++;
}
else
{
j++;
}
}
i=0;j=0;
printf("\n\n-------------------------\nassuming second frame is getting lost\n\n");
while((j*sw)<fnum && i<fnum)
{
k=(i-(j*sw))/sw;
```

```
if(k==0)
{
printf("\nsender : frame %d sent ",frames[i]);
if(i==1)
printf("\nsender : frame %d sent ",frames[i]);
i++;
}
else
{
printf("\n\nwindow needs to be moved ");
j++;
}
}
i=0,j=0;
while((j*sw)<fnum && i<fnum)
{
k=(i-(j*sw))/sw;
if(k==0)
{
if(i!=1)
printf("\nreceiver : frame %d received ",frames[i]);
if(i==1)
{
printf("\nreceiver : frame %d not received ",frames[i]);
}

i++;
}
else
{
//printf("\n\nwindow needs to be moved ");
if(i==3)
printf("\nreceiver : frame %d received ",frames[1]);
j++;
}
```

```
}
return 0;
}
```

OUTPUT



C:\Users\16BCE0824\Desktop\selectiverep.exe

```
enter the sliding window size 4

enter the number of frames to be sent 3

enter the frames to be sent 2
0
1


-------------------------
assuming no frame loss


sender : frame 2 sent
sender : frame 0 sent
sender : frame 1 sent
receiver: frame 2 received
receiver: frame 0 received
receiver: frame 1 received

-------------------------
assuming second frame is getting lost


sender : frame 2 sent
sender : frame 0 sent
sender : frame 0 sent
sender : frame 1 sent
receiver : frame 2 received
receiver : frame 0 not received
receiver : frame 1 received
Process returned 0 (0x0)   execution time : 16.380 s
Press any key to continue.
```

<u>The Question:</u>

An ISP is granted a block of addresses starting with 190.100.0.0/16 (65,536 addresses). The ISP needs to distribute these addresses to three groups of customers as follows:

a. The first group has 64 customers; each needs 256 addresses.

b. The second group has 128 customers; each needs 128 addresses.

c. The third group has 128 customers; each needs 64 addresses.

Design the sub blocks and find out how many addresses are still available after these allocations.

The Code:

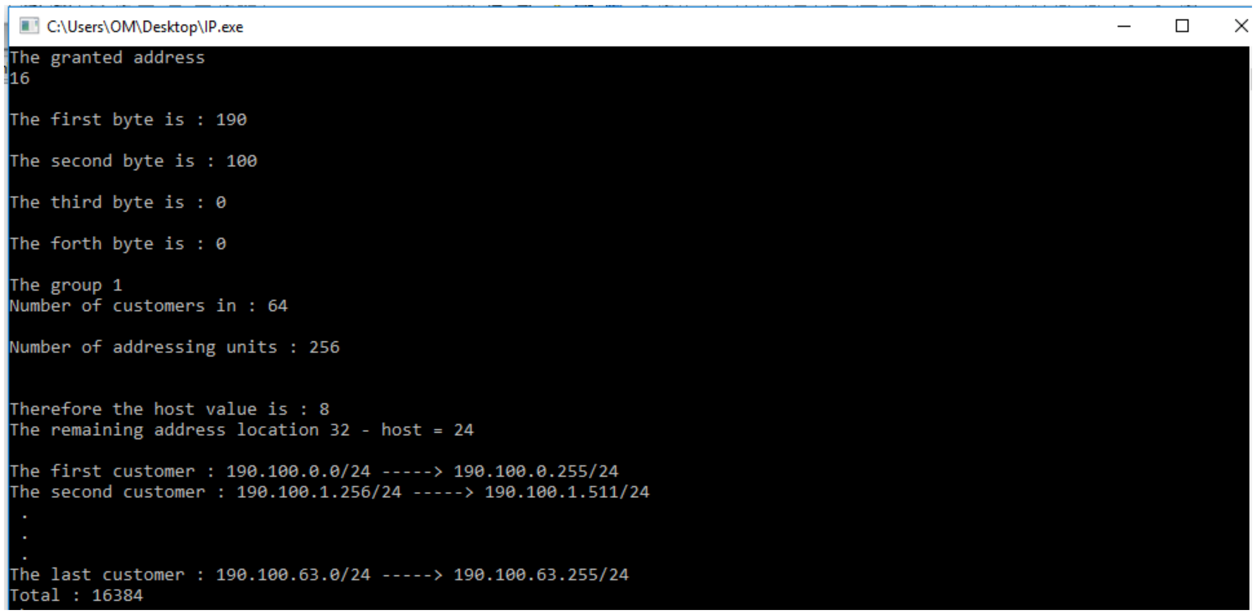
```
#include<stdio.h>

#include<math.h>

int main()

{

   int x,y,z,t,n;

   int a,b,b1;

   int c1,r1,c2,r2,c3,r3;

   int h1,h2,h3,k1,k2,k3;

   printf("The granted address\n");

   scanf("%d",&n);

   b= pow(2,n);

   b1=b;

   printf("\nThe first byte is : ");

   scanf("%d",&x);

   printf("\nThe second byte is : ");

   scanf("%d",&y);

   printf("\nThe third byte is : ");

   scanf("%d",&z);

   printf("\nThe forth byte is : ");

   scanf("%d",&t);
```

```
//--------------------------------------first group--------------------------------------------

   printf("\nThe group 1 ");

   printf("\nNumber of customers in : ");

   scanf("%d",&c1);

   printf("\nNumber of addressing units : ");

   scanf("%d",&r1);


   h1=log(r1)/log(2);

   printf("\n\nTherefore the host value is : %d",h1);

   k1=32-h1;

   printf("\nThe remaining address location 32 - host = %d",k1);

   printf("\n\nThe first customer : %d.%d.%d.%d/%d -----> %d.%d.%d.%d/%d",x,y,z,t,k1,x,y,z,t+(r1-1),k1);

   printf("\nThe second customer : %d.%d.%d.%d/%d -----> 
%d.%d.%d.%d/%d",x,y,z+1,t+r1,k1,x,y,z+1,t+r1+(r1-1),k1);

   printf("\n . \n . \n .");

   printf("\nThe last customer : %d.%d.%d.%d/%d -----> %d.%d.%d.%d/%d",x,y,z+(c1-1),t+(255-
r1),k1,x,y,z+(c1-1),255,k1);

   printf("\nTotal : %d",r1*c1);

//-------------------------------------second group-------------------------------------------

   printf("\nThe group 2 ");

   printf("\nNumber of customers in : ");

   scanf("%d",&c2);

   printf("\nNumber of addressing units : ");

   scanf("%d",&r2);


   h2=log(r2)/log(2);
```

```
    printf("\n\nTherefore the host value is : %d",h2);

    k2=32-h2;

    printf("\nThe remaining address location 32 - host = %d",k2);

    printf("\n\nThe first customer : %d.%d.%d.%d/%d ----->
%d.%d.%d.%d/%d",x,y,z+c1,t,k2,x,y,z+c1,t+(r2-1),k2);

    printf("\nThe second customer : %d.%d.%d.%d/%d ----->
%d.%d.%d.%d/%d",x,y,z+c1+1,t+r2,k2,x,y,z+1,t+r2+(r2-1),k2);

    printf("\n . \n . \n .");

    printf("\nThe last customer : %d.%d.%d.%d/%d -----> %d.%d.%d.%d/%d",x,y,z+c1+(c2-1),t+(255-
r2),k2,x,y,z+c1+(c2-1),255,k2);

    printf("\nTotal : %d",r2*c2);

//--------------------------------------third group-------------------------------------------

    printf("\nThe group 3 ");

    printf("\nNumber of customers in : ");

    scanf("%d",&c3);

    printf("\nNumber of addressing units : ");

    scanf("%d",&r3);


    h3=log(r3)/log(2);

    printf("\n\nTherefore the host value is : %d",h3);

    k3=32-h3;

    printf("\nThe remaining address location 32 - host = %d",k3);


    printf("\n\nThe first customer : %d.%d.%d.%d/%d ----->
%d.%d.%d.%d/%d",x,y,z+c1+c2,t,k3,x,y,z+c1+c2,t+(r3-1),k3);

    printf("\nThe second customer : %d.%d.%d.%d/%d ----->
%d.%d.%d.%d/%d",x,y,z+c1+c2+1,t+r3,k3,x,y,z+c1+c2+1,t+r3+(r3-1),k3);
```

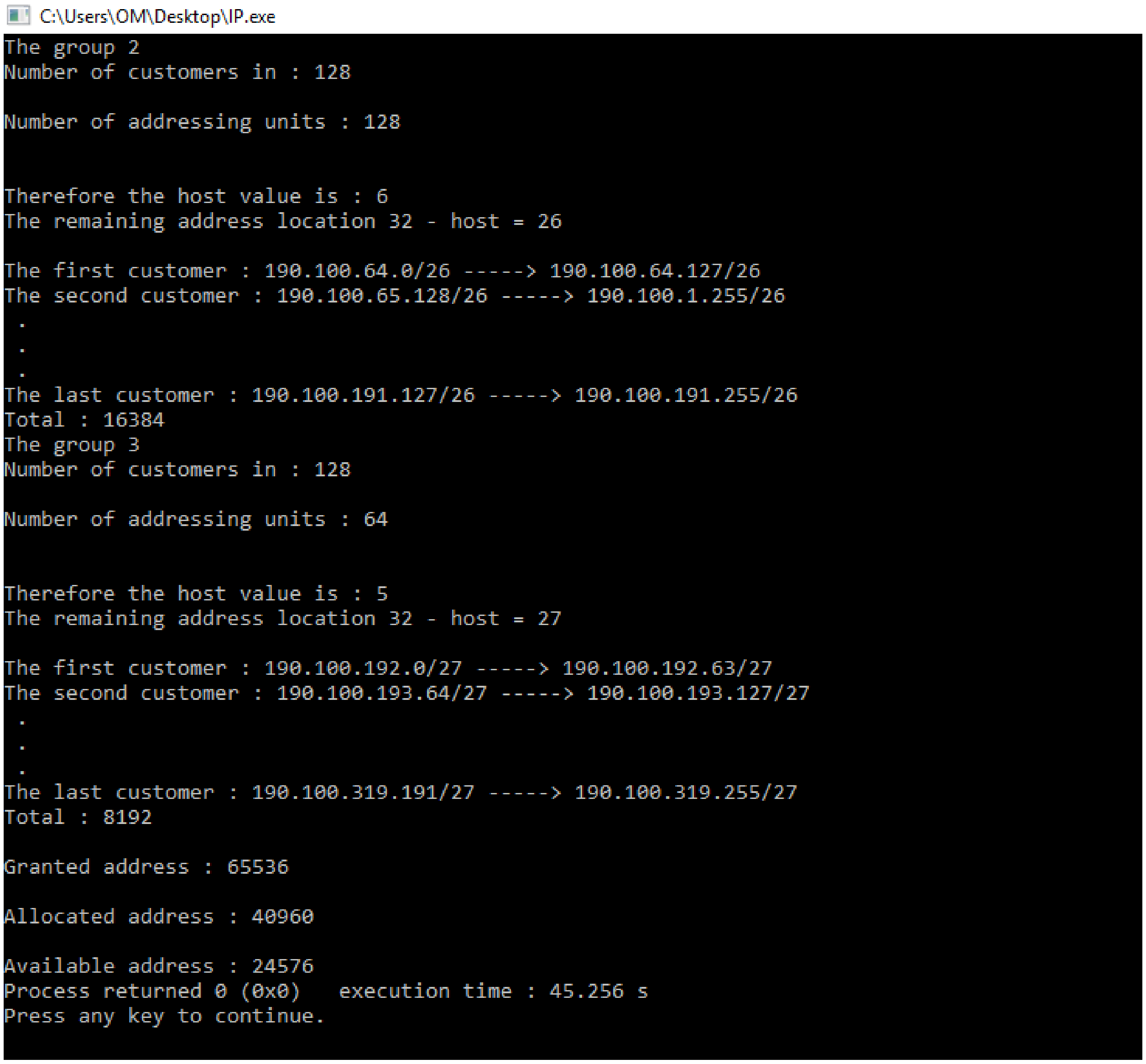
```
   printf("\n . \n . \n .");

   printf("\nThe last customer : %d.%d.%d.%d/%d -----> %d.%d.%d.%d/%d",x,y,z+c1+c2+(c3-1),t+(255-
r3),k3,x,y,z+c1+c2+(c3-1),255,k3);

   printf("\nTotal : %d",r3*c3);



   printf("\n\nGranted address : %d",b1);

   printf("\n\nAllocated address : ");

   a=r1*c1+r2*c2+r3*c3;

   printf("%d",a);



   printf("\n\nAvailable address : ");

   double c=b-a;

   printf("%d",c);

   return(0);

}
```

The Output:-



C:\Users\OM\Desktop\IP.exe

```
The granted address
16

The first byte is : 190

The second byte is : 100

The third byte is : 0

The forth byte is : 0

The group 1
Number of customers in : 64

Number of addressing units : 256


Therefore the host value is : 8
The remaining address location 32 - host = 24

The first customer : 190.100.0.0/24 -----> 190.100.0.255/24
The second customer : 190.100.1.256/24 -----> 190.100.1.511/24
 .
 .
 .
The last customer : 190.100.63.0/24 -----> 190.100.63.255/24
Total : 16384
```



```
C:\Users\OM\Desktop\IP.exe
The group 2
Number of customers in : 128

Number of addressing units : 128


Therefore the host value is : 6
The remaining address location 32 - host = 26

The first customer : 190.100.64.0/26 -----> 190.100.64.127/26
The second customer : 190.100.65.128/26 -----> 190.100.1.255/26
 .
 .
 .
The last customer : 190.100.191.127/26 -----> 190.100.191.255/26
Total : 16384
The group 3
Number of customers in : 128

Number of addressing units : 64


Therefore the host value is : 5
The remaining address location 32 - host = 27

The first customer : 190.100.192.0/27 -----> 190.100.192.63/27
The second customer : 190.100.193.64/27 -----> 190.100.193.127/27
 .
 .
 .
The last customer : 190.100.319.191/27 -----> 190.100.319.255/27
Total : 8192

Granted address : 65536

Allocated address : 40960

Available address : 24576
Process returned 0 (0x0)   execution time : 45.256 s
Press any key to continue.
```

TCP SOCKETS CHAT APPLICATION (SERVER & CLIENT) USING C

THE CODES:-

SERVER

```
#include<stdio.h>
#include<netinet/in.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<netdb.h>
#include<stdlib.h>
#include<string.h>
```

```
#define MAX 80
#define PORT 43454
#define SA struct sockaddr
void func(int sockfd)
{
char buff[MAX];
int n;
for(;;)
{
bzero(buff,MAX);
read(sockfd,buff,sizeof(buff));
printf("From client: %s\t To client : ",buff);
bzero(buff,MAX);
n=0;
while((buff[n++]=getchar())!='\n');
write(sockfd,buff,sizeof(buff));
if(strncmp("exit",buff,4)==0)
{
printf("Server Exit...\n");
break;
}
}
}
int main()
{
int sockfd,connfd,len;
struct sockaddr_in servaddr,cli;
sockfd=socket(AF_INET,SOCK_STREAM,0);
if(sockfd==-1)
{
printf("socket creation failed...\n");
exit(0);
}
else
printf("Socket successfully created..\n");
bzero(&servaddr,sizeof(servaddr));
servaddr.sin_family=AF_INET;
servaddr.sin_addr.s_addr=htonl(INADDR_ANY);
servaddr.sin_port=htons(PORT);
if((bind(sockfd,(SA*)&servaddr, sizeof(servaddr)))!=0)
{
```

```
printf("socket bind failed...\n");
exit(0);
}
else
printf("Socket successfully binded..\n");
if((listen(sockfd,5))!=0)
{
printf("Listen failed...\n");
exit(0);
}
else
printf("Server listening..\n");
len=sizeof(cli);
connfd=accept(sockfd,(SA *)&cli,&len);
if(connfd<0)
{
printf("server acccept failed...\n");
exit(0);
}
else
printf("server acccept the client...\n");
func(connfd);
close(sockfd);
}
```

CLIENT

```
#include<stdio.h>
#include<netinet/in.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<netdb.h>
#include<string.h>
#include<stdlib.h>
#define MAX 80
#define PORT 43454
#define SA struct sockaddr
void func(int sockfd)
{
char buff[MAX];
int n;
for(;;)
{
```

```
bzero(buff,sizeof(buff));
printf("Enter the string : ");
n=0;
while((buff[n++]=getchar())!='\n');
write(sockfd,buff,sizeof(buff));
bzero(buff,sizeof(buff));
read(sockfd,buff,sizeof(buff));
printf("From Server : %s",buff);
if((strncmp(buff,"exit",4))==0)
{
printf("Client Exit...\n");
break;
}
}
}
int main()
{
int sockfd,connfd;
struct sockaddr_in servaddr,cli;
sockfd=socket(AF_INET,SOCK_STREAM,0);
if(sockfd==-1)
{
printf("socket creation failed...\n");
exit(0);
}
else
printf("Socket successfully created..\n");
bzero(&servaddr,sizeof(servaddr));
servaddr.sin_family=AF_INET;
servaddr.sin_addr.s_addr=inet_addr("127.0.0.1");
servaddr.sin_port=htons(PORT);
if(connect(sockfd,(SA *)&servaddr,sizeof(servaddr))!=0)
{
printf("connection with the server failed...\n");
exit(0);
}
else
printf("connected to the server..\n");
func(sockfd);
close(sockfd);
}
```

OUTPUT
SERVER SIDE

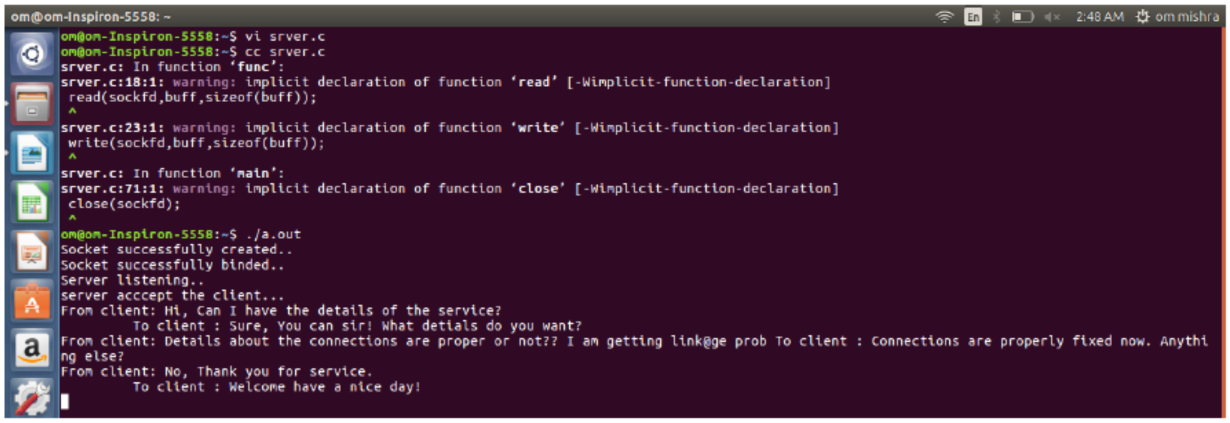
```
$ cc tcpchatserver.c
$ ./a.out
Socket successfully created..
Socket successfully binded..
Server listening..
server acccept the client...
From client: hai
To client : hello
From client: exit
To client : exit
Server Exit...
$
```

CLIENT SIDE

```
$ cc tcpchatclient.c
$ ./a.out
Socket successfully created..
connected to the server..
Enter the string : hai
From Server : hello
Enter the string : exit
From Server : exit
Client Exit...
$
```

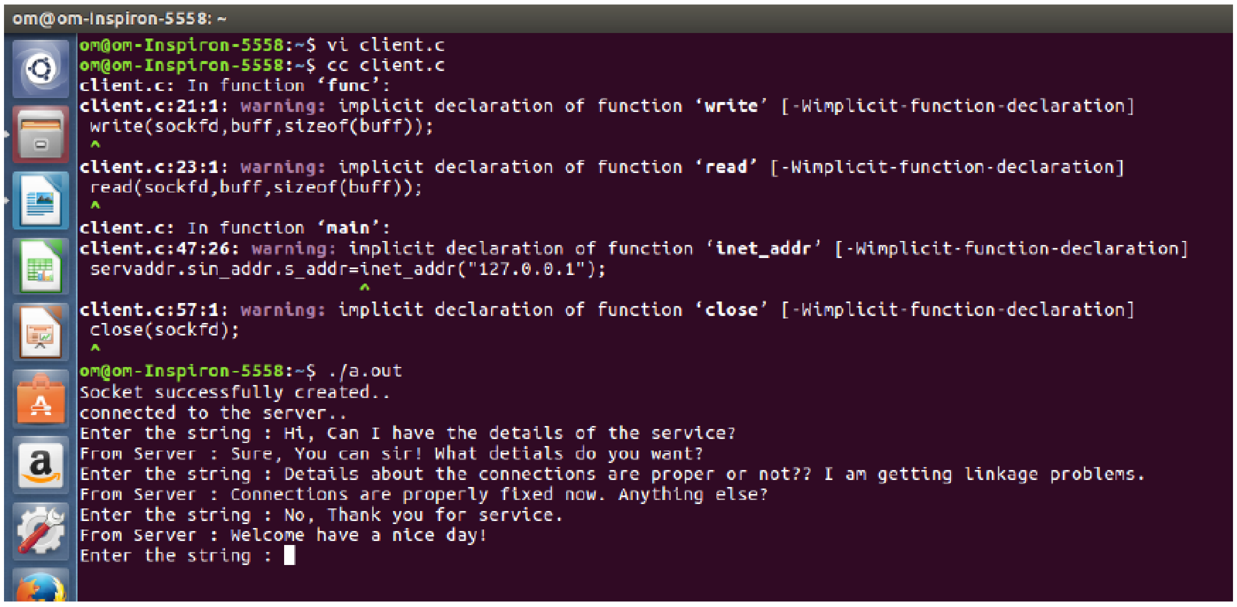
OUTPUT:

Server:



```
om@om-Inspiron-5558:~$ vi srver.c
om@om-Inspiron-5558:~$ cc srver.c
srver.c: In function ‘func’:
srver.c:18:1: warning: implicit declaration of function ‘read’ [-Wimplicit-function-declaration]
 read(sockfd,buff,sizeof(buff));
 ^
srver.c:23:1: warning: implicit declaration of function ‘write’ [-Wimplicit-function-declaration]
 write(sockfd,buff,sizeof(buff));
 ^
srver.c: In function ‘main’:
srver.c:71:1: warning: implicit declaration of function ‘close’ [-Wimplicit-function-declaration]
 close(sockfd);
 ^
om@om-Inspiron-5558:~$ ./a.out
Socket successfully created..
Socket successfully binded..
Server listening..
server acccept the client...
From client: Hi, Can I have the details of the service?
         To client : Sure, You can sir! What detials do you want?
From client: Details about the connections are proper or not?? I am getting link@ge prob To client : Connections are properly fixed now. Anythi
ng else?
From client: No, Thank you for service.
         To client : Welcome have a nice day!
```

Client:



```
om@om-Inspiron-5558: ~
om@om-Inspiron-5558:~$ vi client.c
om@om-Inspiron-5558:~$ cc client.c
client.c: In function ‘func’:
client.c:21:1: warning: implicit declaration of function ‘write’ [-Wimplicit-function-declaration]
 write(sockfd,buff,sizeof(buff));
 ^
client.c:23:1: warning: implicit declaration of function ‘read’ [-Wimplicit-function-declaration]
 read(sockfd,buff,sizeof(buff));
 ^
client.c: In function ‘main’:
client.c:47:26: warning: implicit declaration of function ‘inet_addr’ [-Wimplicit-function-declaration]
 servaddr.sin_addr.s_addr=inet_addr("127.0.0.1");
                          ^
client.c:57:1: warning: implicit declaration of function ‘close’ [-Wimplicit-function-declaration]
 close(sockfd);
 ^
om@om-Inspiron-5558:~$ ./a.out
Socket successfully created..
connected to the server..
Enter the string : Hi, Can I have the details of the service?
From Server : Sure, You can sir! What detials do you want?
Enter the string : Details about the connections are proper or not?? I am getting linkage problems.
From Server : Connections are properly fixed now. Anything else?
Enter the string : No, Thank you for service.
From Server : Welcome have a nice day!
Enter the string :
```

UDP SOCKETS CHAT APPLICATION (SERVER & CLIENT) USING C

THE CODES:-

SERVER

```
#include<stdio.h>
#include<netinet/in.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<netdb.h>
#include<string.h>
#include<stdlib.h>
#define MAX 80
#define PORT 43454
#define SA struct sockaddr
void func(int sockfd)
{
char buff[MAX];
int n,clen;
struct sockaddr_in cli;
clen=sizeof(cli);
for(;;)
{
bzero(buff,MAX);
recvfrom(sockfd,buff,sizeof(buff),0,(SA *)&cli,&clen);
printf("From client %s To client",buff);
bzero(buff,MAX);
n=0;
while((buff[n++]=getchar())!='\n');
sendto(sockfd,buff,sizeof(buff),0,(SA *)&cli,clen);
if(strncmp("exit",buff,4)==0)
{
printf("Server Exit...\n");
break;
}
}
}
int main()
{
int sockfd;
struct sockaddr_in servaddr;
sockfd=socket(AF_INET,SOCK_DGRAM,0);
if(sockfd==-1)
{
printf("socket creation failed...\n");
exit(0);
}
else
printf("Socket successfully created..\n");
bzero(&servaddr,sizeof(servaddr));
servaddr.sin_family=AF_INET;
servaddr.sin_addr.s_addr=htonl(INADDR_ANY);
servaddr.sin_port=htons(PORT);
if((bind(sockfd,(SA *)&servaddr,sizeof(servaddr)))!=0)
{
printf("socket bind failed...\n");
exit(0);
}
else
printf("Socket successfully binded..\n");
func(sockfd);
close(sockfd);
}
```

CLIENT

```
#include<sys/socket.h>
#include<netdb.h>
#include<string.h>
#include<stdlib.h>
#include<stdio.h>
```

```
#define MAX 80
#define PORT 43454
#define SA struct sockaddr
int main()
{
char buff[MAX];
int sockfd,len,n;
struct sockaddr_in servaddr;
sockfd=socket(AF_INET,SOCK_DGRAM,0);
if(sockfd==-1)
{
printf("socket creation failed...\n");
exit(0);
}
else
printf("Socket successfully created..\n");
bzero(&servaddr,sizeof(len));
servaddr.sin_family=AF_INET;
servaddr.sin_addr.s_addr=inet_addr("127.0.0.1");
servaddr.sin_port=htons(PORT);
len=sizeof(servaddr);
for(;;)
{
printf("\nEnter string : ");
n=0;
while((buff[n++]=getchar())!='\n');
sendto(sockfd,buff,sizeof(buff),0,(SA *)&servaddr,len);
bzero(buff,sizeof(buff));
recvfrom(sockfd,buff,sizeof(buff),0,(SA *)&servaddr,&len);
printf("From Server : %s\n",buff);
if(strncmp("exit",buff,4)==0)
{
printf("Client Exit...\n");
break;
}
}
close(sockfd);
}
```

OUTPUT

SERVER SIDE

```
$ cc udpchatserver.c
$ ./a.out
```

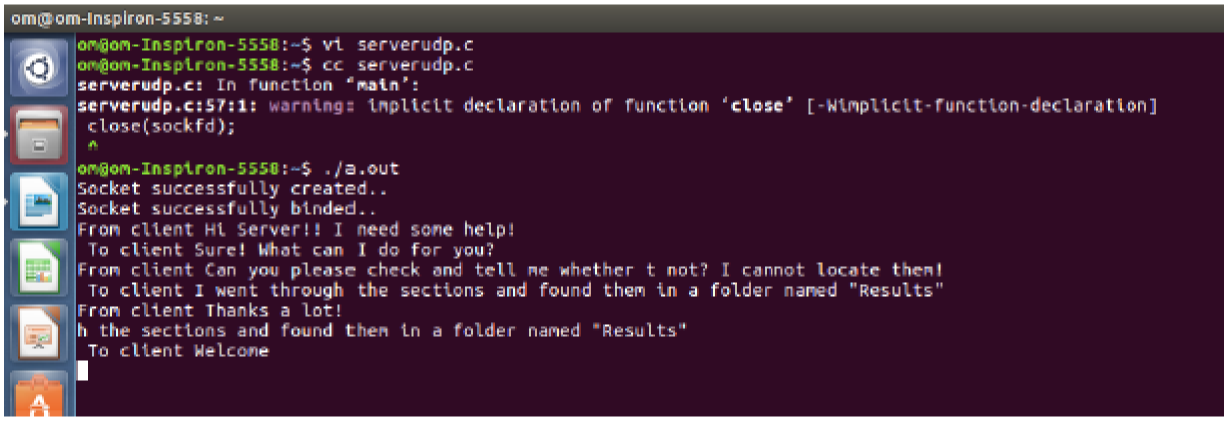
```
Socket successfully created..
Socket successfully binded..
From client hai
To client hello
From client exit
To client exit
Server Exit...
$
```

CLIENT SIDE

```
$ cc udpchatclient.c
$ ./a.out
Socket successfully created..
Enter string : hai
From Server : hello
Enter string : exit
From Server : exit
Client Exit...
$
```

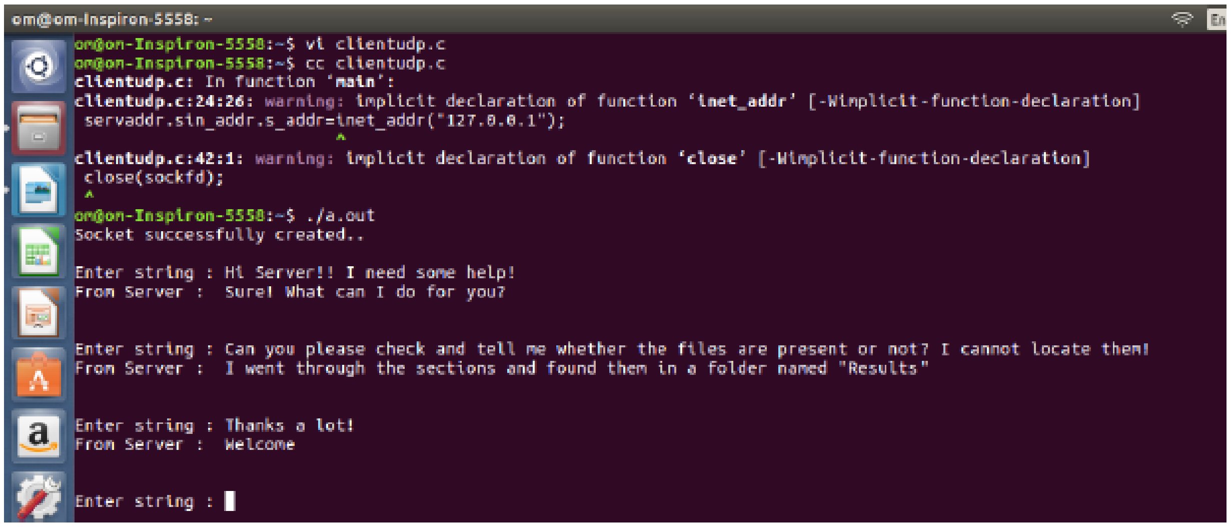
OUTPUT:

Server:



```
om@om-Inspiron-5558: ~
om@om-Inspiron-5558:~$ vi serverudp.c
om@om-Inspiron-5558:~$ cc serverudp.c
serverudp.c: In function ‘main’:
serverudp.c:57:1: warning: implicit declaration of function ‘close’ [-Wimplicit-function-declaration]
 close(sockfd);
 ^
om@om-Inspiron-5558:~$ ./a.out
Socket successfully created..
Socket successfully binded..
From client Hi Server!! I need some help!
 To client Sure! What can I do for you?
From client Can you please check and tell me whether t not? I cannot locate them!
 To client I went through the sections and found them in a folder named "Results"
From client Thanks a lot!
h the sections and found them in a folder named "Results"
 To client Welcome
```

Client:



```
om@om-Inspiron-5558: ~
om@om-Inspiron-5558:~$ vi clientudp.c
om@om-Inspiron-5558:~$ cc clientudp.c
clientudp.c: In function ‘main’:
clientudp.c:24:26: warning: implicit declaration of function ‘inet_addr’ [-Wimplicit-function-declaration]
 servaddr.sin_addr.s_addr=inet_addr("127.0.0.1");
                          ^
clientudp.c:42:1: warning: implicit declaration of function ‘close’ [-Wimplicit-function-declaration]
 close(sockfd);
 ^
om@om-Inspiron-5558:~$ ./a.out
Socket successfully created..

Enter string : Hi Server!! I need some help!
From Server :  Sure! What can I do for you?


Enter string : Can you please check and tell me whether the files are present or not? I cannot locate them!
From Server :  I went through the sections and found them in a folder named "Results"


Enter string : Thanks a lot!
From Server :  Welcome


Enter string : 
```

NS-2

```
#Create a simulator object

set ns [new Simulator]


#Define different colors for data flows (for NAM)

$ns color 1 Blue

$ns color 2 Red


#Open the NAM trace file

set nf [open out.nam w]

$ns namtrace-all $nf


#Define a 'finish' procedure

proc finish {} {

        global ns nf
        $ns flush-trace

        #Close the NAM trace file

        close $nf

        #Execute NAM on the trace file

        exec nam out.nam &

        exit 0

}


#Create four nodes

set n0 [$ns node]

set n1 [$ns node]

set n2 [$ns node]

set n3 [$ns node]


#Create links between the nodes

$ns duplex-link $n0 $n2 2Mb 10ms DropTail

$ns duplex-link $n1 $n2 2Mb 10ms DropTail

$ns duplex-link $n2 $n3 1.7Mb 20ms DropTail


#Set Queue Size of link (n2-n3) to 10

$ns queue-limit $n2 $n3 10


#Give node position (for NAM)

$ns duplex-link-op $n0 $n2 orient right-down

$ns duplex-link-op $n1 $n2 orient right-up
```

```
$ns duplex-link-op $n2 $n3 orient right


#Monitor the queue for link (n2-n3). (for NAM)

$ns duplex-link-op $n2 $n3 queuePos 0.5



#Setup a TCP connection

set tcp [new Agent/TCP]

$tcp set class_ 2

$ns attach-agent $n0 $tcp

set sink [new Agent/TCPSink]

$ns attach-agent $n3 $sink

$ns connect $tcp $sink

$tcp set fid_ 1


#Setup a FTP over TCP connection

set ftp [new Application/FTP]

$ftp attach-agent $tcp

$ftp set type_ FTP



#Setup a UDP connection

set udp [new Agent/UDP]

$ns attach-agent $n1 $udp

set null [new Agent/Null]
```

```
$ns attach-agent $n3 $null

$ns connect $udp $null

$udp set fid_ 2


#Setup a CBR over UDP connection

set cbr [new Application/Traffic/CBR]

$cbr attach-agent $udp

$cbr set type_ CBR

$cbr set packet_size_ 1000

$cbr set rate_ 1mb

$cbr set random_ false




#Schedule events for the CBR and FTP agents

$ns at 0.1 "$cbr start"

$ns at 1.0 "$ftp start"

$ns at 4.0 "$ftp stop"

$ns at 4.5 "$cbr stop"


#Detach tcp and sink agents (not really necessary)

$ns at 4.5 "$ns detach-agent $n0 $tcp ; $ns detach-agent $n3 $sink"


#Call the finish procedure after 5 seconds of simulation time

$ns at 5.0 "finish"
```

#Print CBR packet size and interval

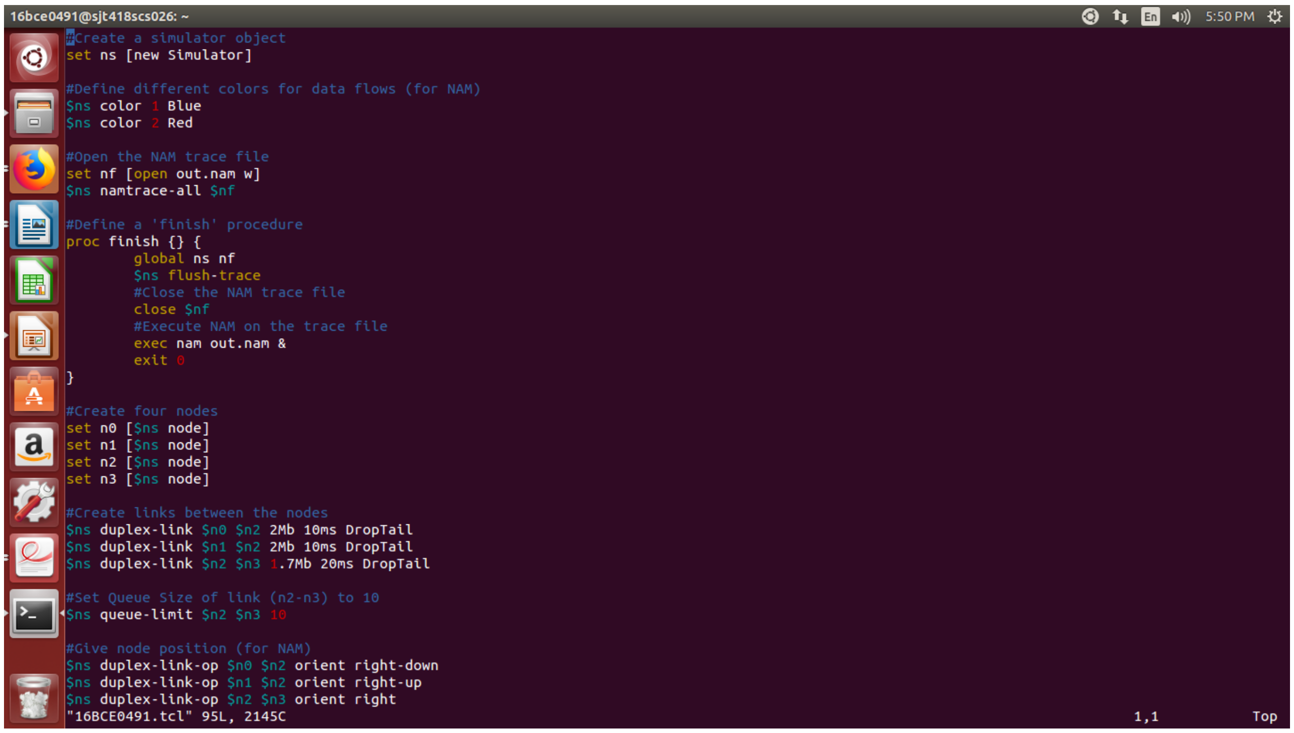
puts "CBR packet size = [$cbr set packet_size_]"

puts "CBR interval = [$cbr set interval_]"

#Run the simulation

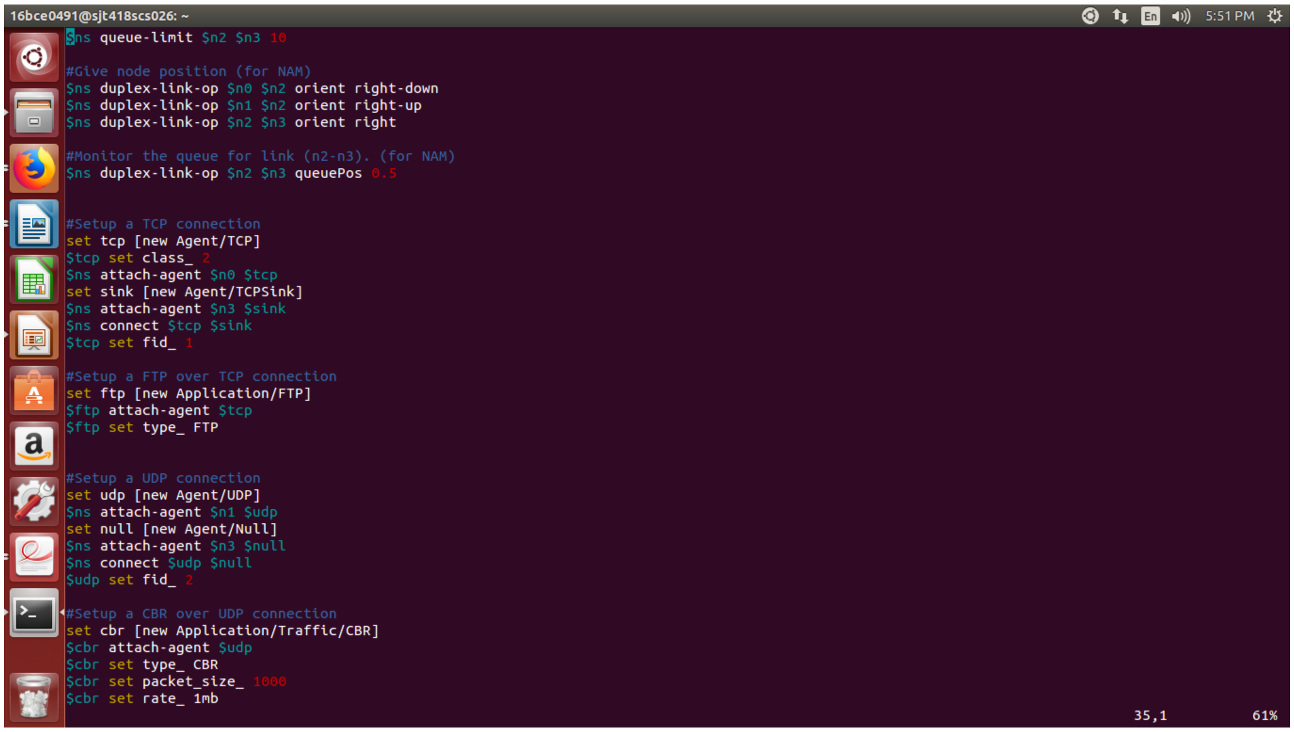
$ns run

screenshots:

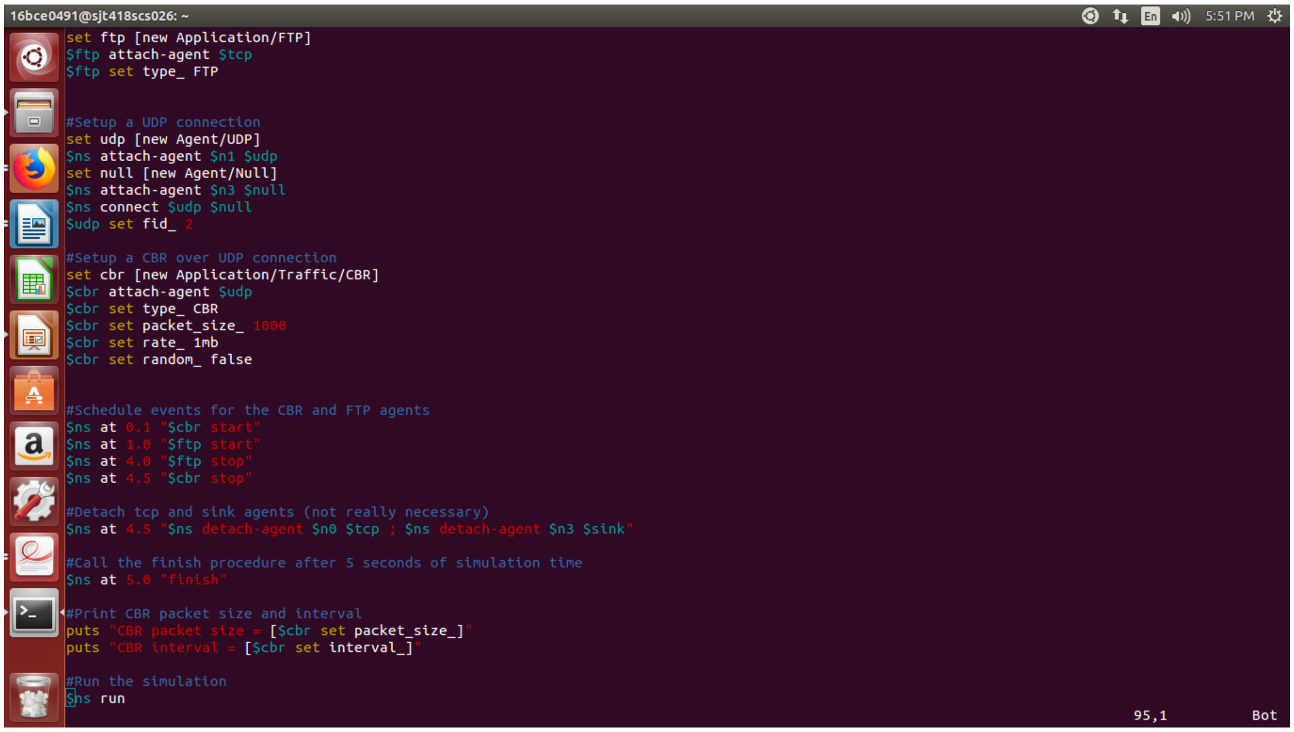


16bce0491@sjt418scs026: ~   5:50 PM

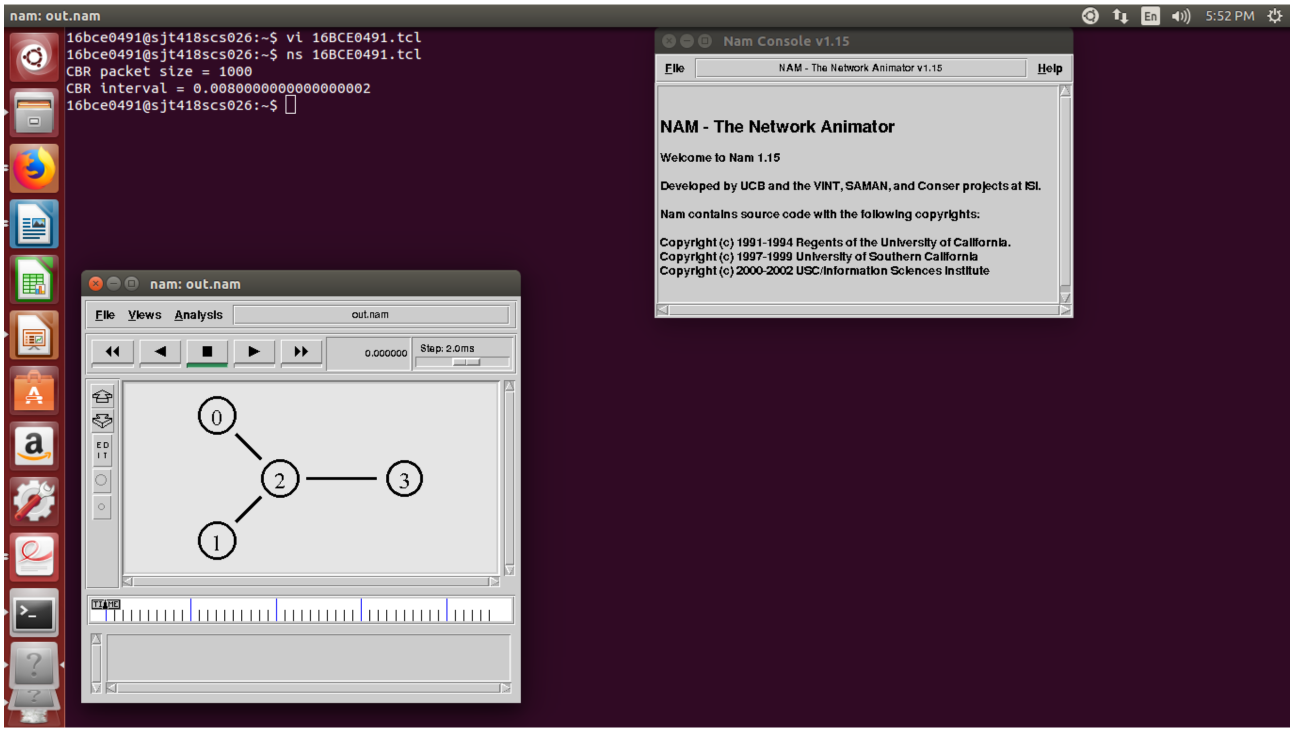
```
#Create a simulator object
set ns [new Simulator]

#Define different colors for data flows (for NAM)
$ns color 1 Blue
$ns color 2 Red

#Open the NAM trace file
set nf [open out.nam w]
$ns namtrace-all $nf

#Define a 'finish' procedure
proc finish {} {
        global ns nf
        $ns flush-trace
        #Close the NAM trace file
        close $nf
        #Execute NAM on the trace file
        exec nam out.nam &
        exit 0
}

#Create four nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]

#Create links between the nodes
$ns duplex-link $n0 $n2 2Mb 10ms DropTail
$ns duplex-link $n1 $n2 2Mb 10ms DropTail
$ns duplex-link $n2 $n3 1.7Mb 20ms DropTail

#Set Queue Size of link (n2-n3) to 10
$ns queue-limit $n2 $n3 10

#Give node position (for NAM)
$ns duplex-link-op $n0 $n2 orient right-down
$ns duplex-link-op $n1 $n2 orient right-up
$ns duplex-link-op $n2 $n3 orient right
"16BCE0491.tcl" 95L, 2145C                                            1,1           Top
```



```
16bce0491@sjt418scs026: ~                                                      5:51 PM
$ns queue-limit $n2 $n3 10

#Give node position (for NAM)
$ns duplex-link-op $n0 $n2 orient right-down
$ns duplex-link-op $n1 $n2 orient right-up
$ns duplex-link-op $n2 $n3 orient right

#Monitor the queue for link (n2-n3). (for NAM)
$ns duplex-link-op $n2 $n3 queuePos 0.5


#Setup a TCP connection
set tcp [new Agent/TCP]
$tcp set class_ 2
$ns attach-agent $n0 $tcp
set sink [new Agent/TCPSink]
$ns attach-agent $n3 $sink
$ns connect $tcp $sink
$tcp set fid_ 1

#Setup a FTP over TCP connection
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ftp set type_ FTP


#Setup a UDP connection
set udp [new Agent/UDP]
$ns attach-agent $n1 $udp
set null [new Agent/Null]
$ns attach-agent $n3 $null
$ns connect $udp $null
$udp set fid_ 2

#Setup a CBR over UDP connection
set cbr [new Application/Traffic/CBR]
$cbr attach-agent $udp
$cbr set type_ CBR
$cbr set packet_size_ 1000
$cbr set rate_ 1mb
                                                                       35,1          61%
```



```
16bce0491@sjt418scs026: ~                                                      5:51 PM
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ftp set type_ FTP


#Setup a UDP connection
set udp [new Agent/UDP]
$ns attach-agent $n1 $udp
set null [new Agent/Null]
$ns attach-agent $n3 $null
$ns connect $udp $null
$udp set fid_ 2

#Setup a CBR over UDP connection
set cbr [new Application/Traffic/CBR]
$cbr attach-agent $udp
$cbr set type_ CBR
$cbr set packet_size_ 1000
$cbr set rate_ 1mb
$cbr set random_ false


#Schedule events for the CBR and FTP agents
$ns at 0.1 "$cbr start"
$ns at 1.0 "$ftp start"
$ns at 4.0 "$ftp stop"
$ns at 4.5 "$cbr stop"

#Detach tcp and sink agents (not really necessary)
$ns at 4.5 "$ns detach-agent $n0 $tcp ; $ns detach-agent $n3 $sink"

#Call the finish procedure after 5 seconds of simulation time
$ns at 5.0 "finish"

#Print CBR packet size and interval
puts "CBR packet size = [$cbr set packet_size_]"
puts "CBR interval = [$cbr set interval_]"

#Run the simulation
$ns run
                                                                       95,1          Bot
```



```
16bce0491@sjt418scs026:~$ vi 16BCE0491.tcl
16bce0491@sjt418scs026:~$ ns 16BCE0491.tcl
CBR packet size = 1000
CBR interval = 0.0080000000000000002
16bce0491@sjt418scs026:~$ 
```

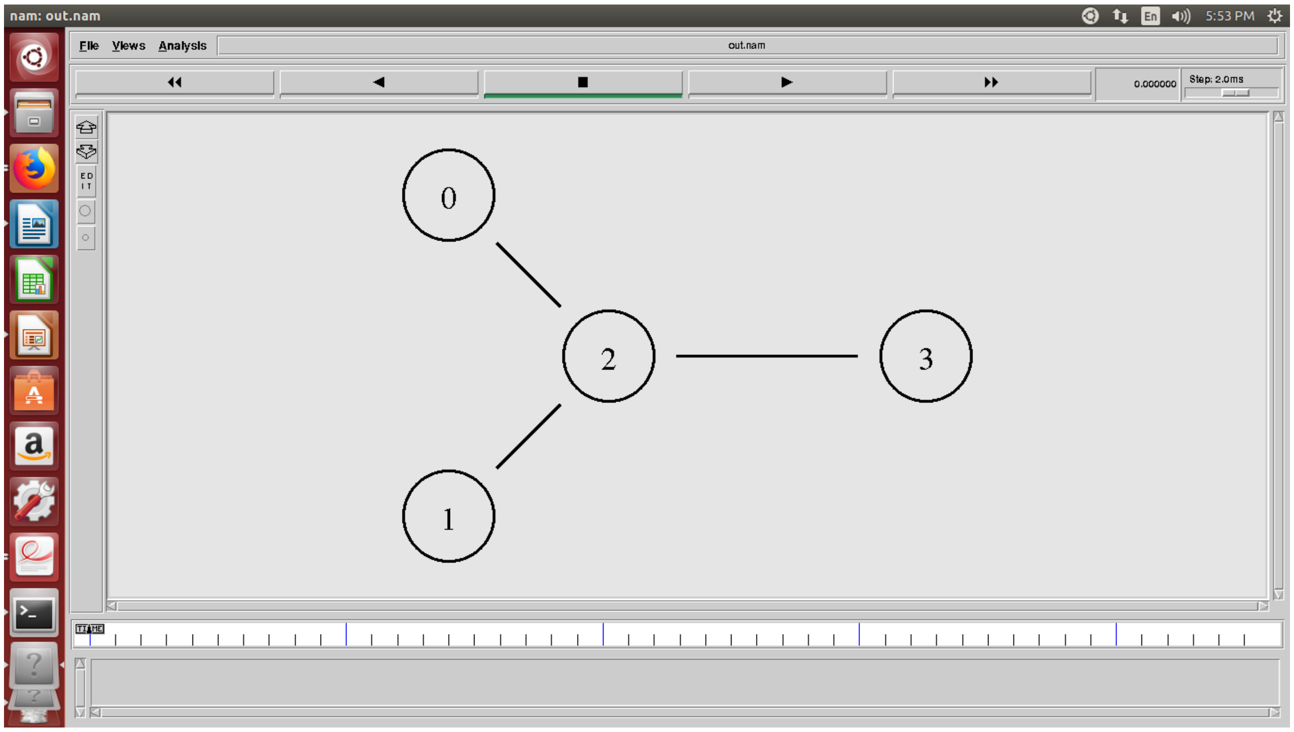
NAM - The Network Animator

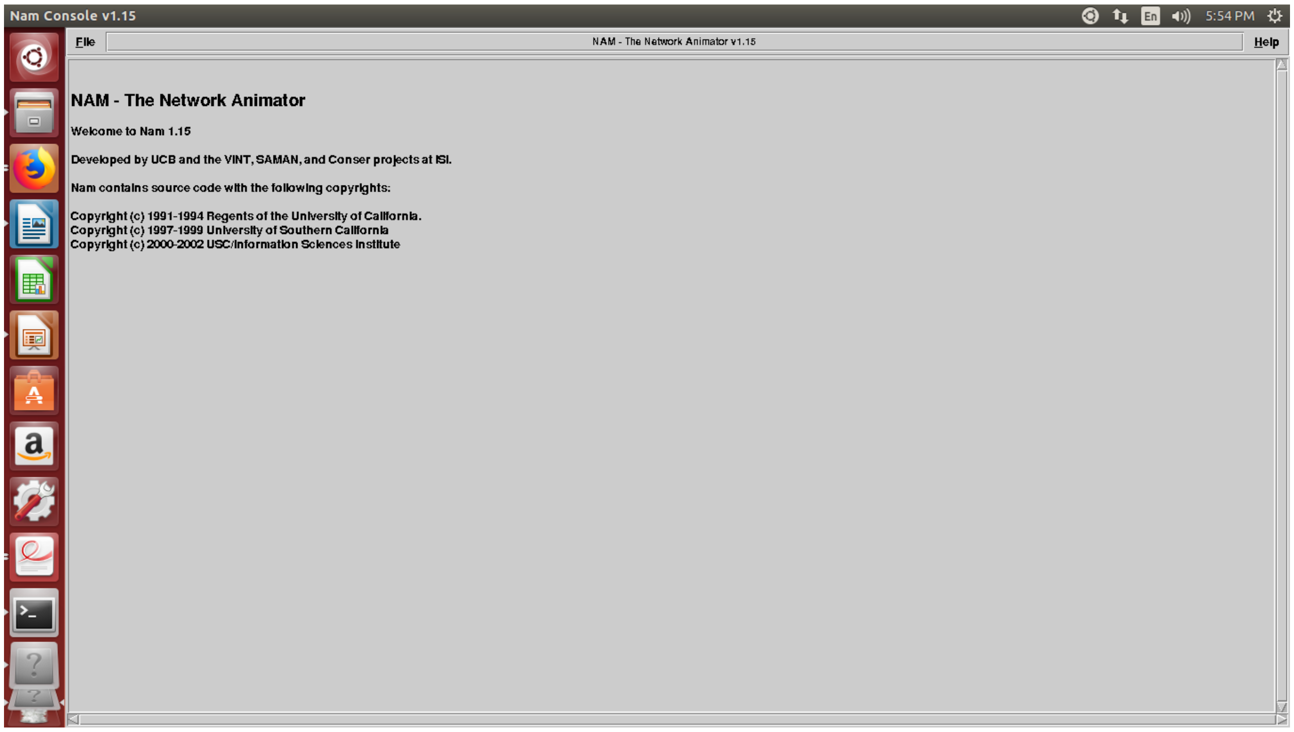
Welcome to Nam 1.15

Developed by UCB and the VINT, SAMAN, and Conser projects at ISI.

Nam contains source code with the following copyrights:

Copyright (c) 1991-1994 Regents of the University of California.
Copyright (c) 1997-1999 University of Southern California
Copyright (c) 2000-2002 USC/Information Sciences Institute





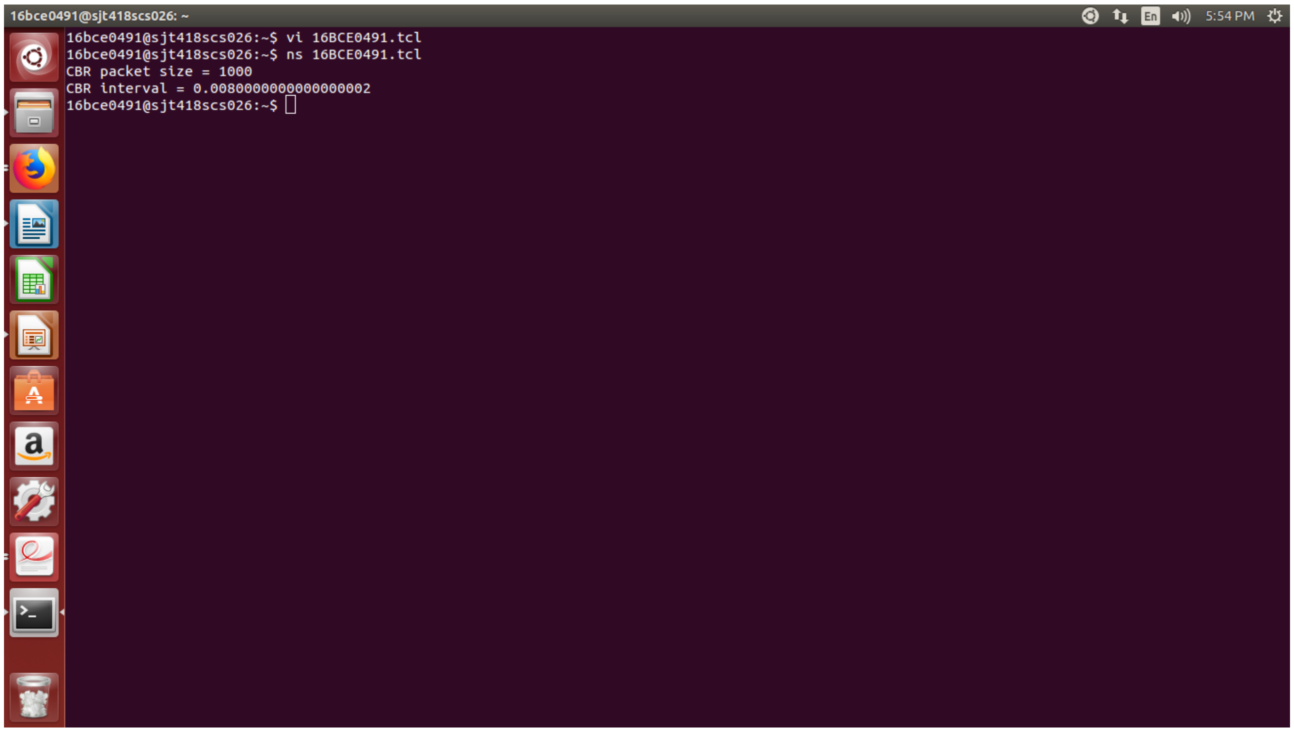
NAM - The Network Animator

Welcome to Nam 1.15

Developed by UCB and the VINT, SAMAN, and Conser projects at ISI.

Nam contains source code with the following copyrights:

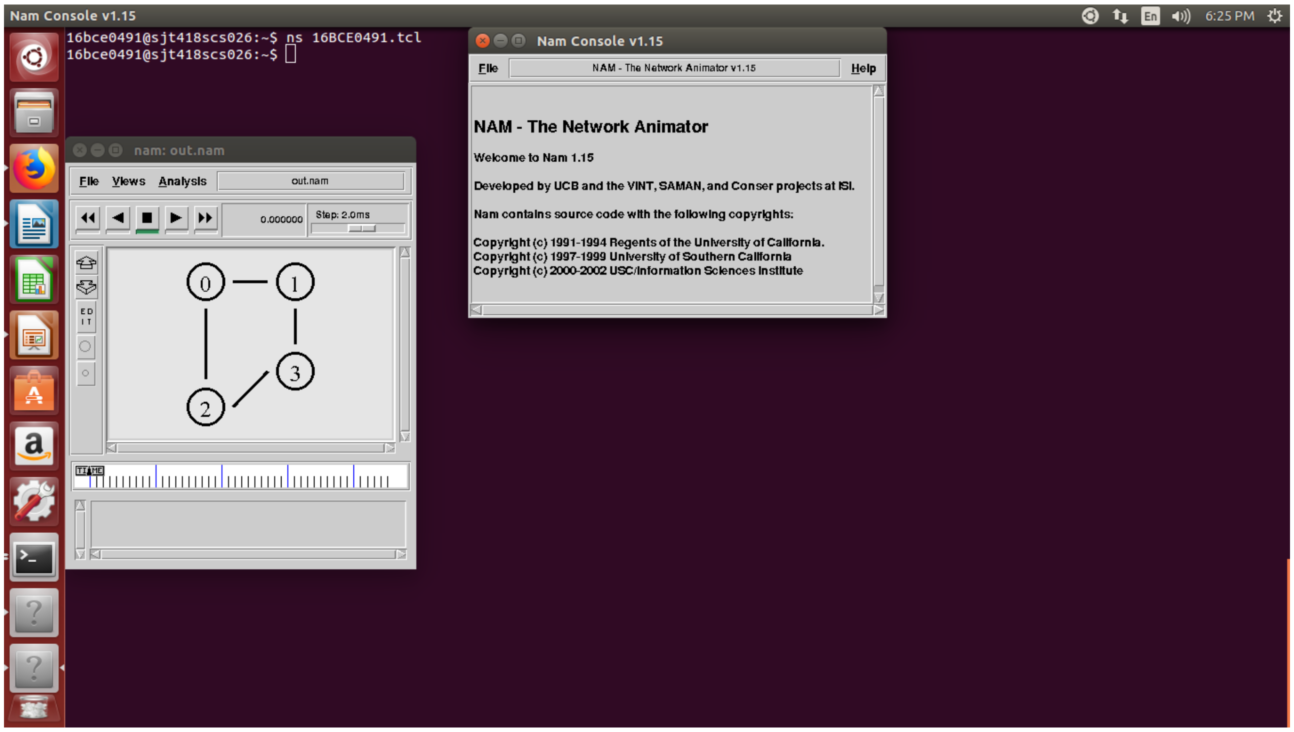
Copyright (c) 1991-1994 Regents of the University of California.
Copyright (c) 1997-1999 University of Southern California
Copyright (c) 2000-2002 USC/Information Sciences Institute



```
16bce0491@sjt418scs026:~$ vi 16BCE0491.tcl
16bce0491@sjt418scs026:~$ ns 16BCE0491.tcl
CBR packet size = 1000
CBR interval = 0.0080000000000000002
16bce0491@sjt418scs026:~$ 
```

```
#Create a simulator object
set ns [new Simulator]

#Define different colors for data flows (for NAM)
$ns color 1 Blue
$ns color 2 Red

#Open the NAM trace file
set nf [open out.nam w]
$ns namtrace-all $nf

#Define a 'finish' procedure
proc finish {} {
        global ns nf
        $ns flush-trace
        #Close the NAM trace file
        close $nf
        #Execute NAM on the trace file
        exec nam out.nam &
        exit 0
}

#Create four nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]

#Create links between the nodes
$ns duplex-link $n0 $n1 2Mb 10ms DropTail
$ns duplex-link $n1 $n3 2Mb 10ms DropTail
$ns duplex-link $n3 $n2 1.7Mb 20ms DropTail
$ns duplex-link $n2 $n0 1.7Mb 20ms DropTail

#Set Queue Size of link (n2-n3) to 10
#$ns queue-limit $n0 $n3 10
#$ns queue-limit $n1 $n2 10

#Give node position (for NAM)
$ns duplex-link-op $n0 $n2 orient down
$ns duplex-link-op $n0 $n1 orient right
$ns duplex-link-op $n2 $n3 orient right
$ns duplex-link-op $n1 $n3 orient down

#Monitor the queue for link (n2-n3). (for NAM)
#$ns duplex-link-op $n0 $n3 queuePos 0.5
#$ns duplex-link-op $n1 $n2 queuePos 0.5


#Setup a udp connection
set udp0 [new Agent/UDP]
$udp0 set class_ 1
$ns attach-agent $n0 $udp0
```

```
#Setup a CBR over UDP connection
set cbr0 [new Application/Traffic/CBR]
$cbr0 attach-agent $udp0
$cbr0 set type_ CBR
$cbr0 set packet_size_ 1000
$cbr0 set rate_ 1mb
$cbr0 set random_ false


#Setup a udp connection
set udp1 [new Agent/UDP]
$udp0 set class_ 2
$ns attach-agent $n1 $udp1

set cbr1 [new Application/Traffic/CBR]
$cbr1 attach-agent $udp1
$cbr1 set type_ CBR
$cbr1 set packet_size_ 1000
$cbr1 set rate_ 1mb
$cbr1 set random_ false


set null0 [new Agent/Null]
$ns attach-agent $n2 $null0


set null1 [new Agent/Null]
$ns attach-agent $n3 $null1



$ns connect $udp0 $null1
$ns connect $udp1 $null0



#Schedule events for the CBR and FTP agents
$ns at 0.5 "$cbr0 start"
$ns at 1.0 "$cbr1 start"
$ns at 4.0 "$cbr1 stop"
$ns at 4.5 "$cbr0 stop"

#Call the finish procedure after 5 seconds of simulation time
$ns at 5.0 "finish"

#Print CBR packet size and interval
#puts "CBR packet size = [$cbr set packet_size_]"
#puts "CBR interval = [$cbr set interval_]"

#Run the simulation
$ns run
```



```
16bce0491@sjt418scs026:~$ ns 16BCE0491.tcl
16bce0491@sjt418scs026:~$ 
```

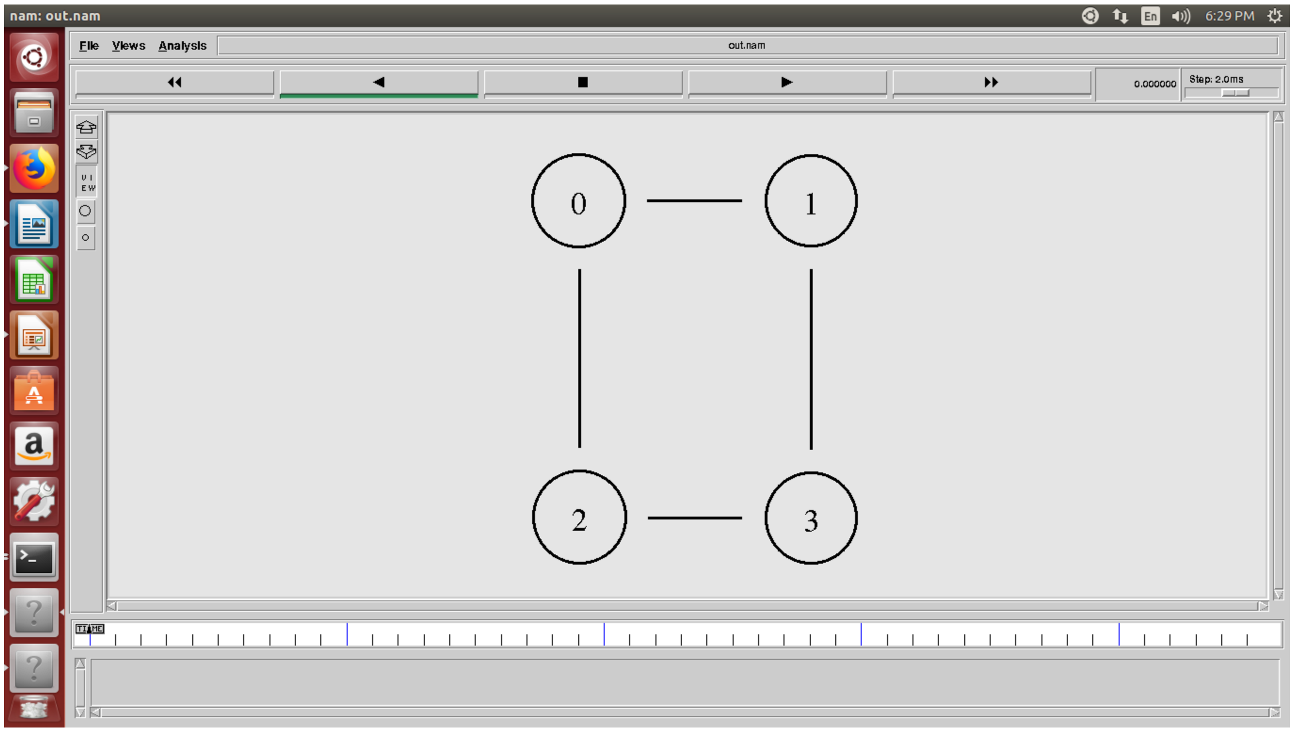
NAM - The Network Animator

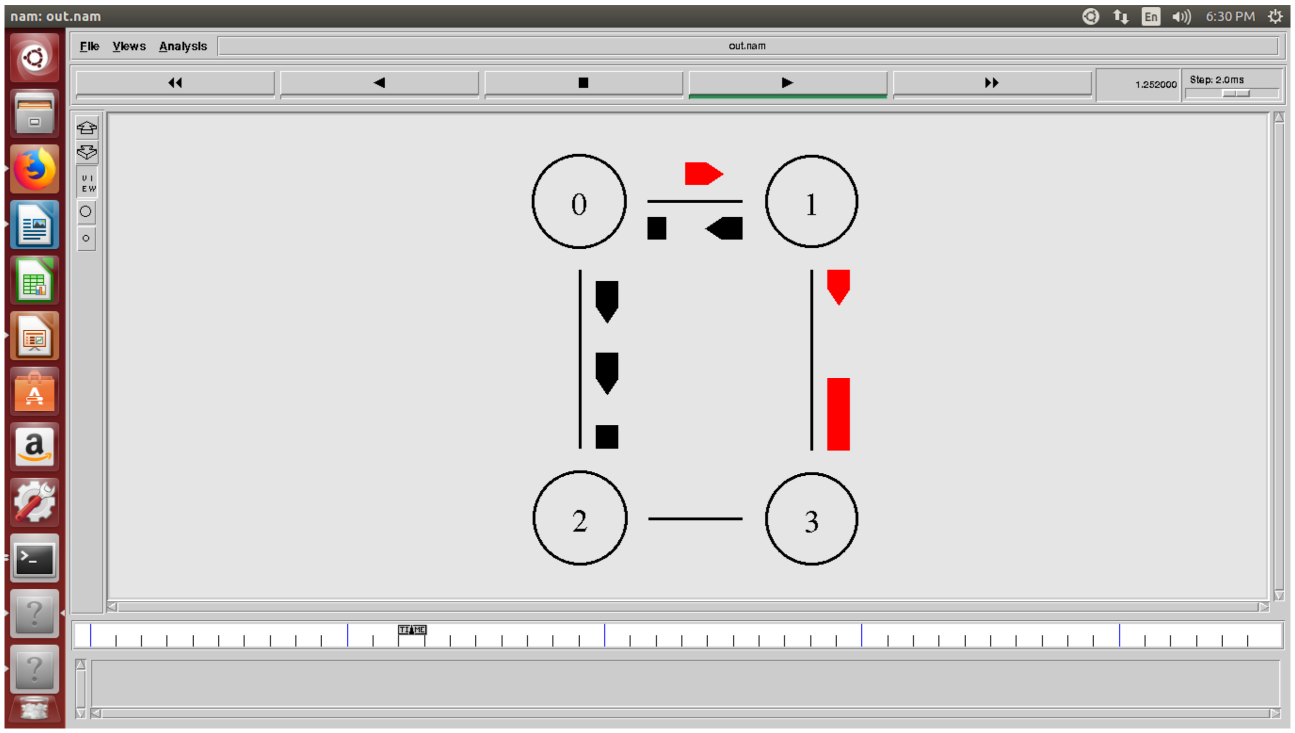
Welcome to Nam 1.15

Developed by UCB and the VINT, SAMAN, and Conser projects at ISI.

Nam contains source code with the following copyrights:

Copyright (c) 1991-1994 Regents of the University of California.
Copyright (c) 1997-1999 University of Southern California
Copyright (c) 2000-2002 USC/Information Sciences Institute




nam: out.nam
6:30 PM
File
Views
Analysis
out.nam
1.252000
Step: 2.0ms
0
1
2
3
TIME