PART-A

1. Write a program to implement RSA algorithm.

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
#include<string.h>
unsigned long modexp(unsigned long msg,unsigned long exp,unsigned long n)
{
    unsigned long i,k=1;
    for(i=0;i<exp;i++)
    k=(k*msg)%n;
    return k;
}
int main()
{
    unsigned long p,q,e,d,n,z,i,m,c;
    int len;
    char data[100];
    printf("enter the value of p & q such that p*q>255\n");
    scanf("%lu%lu",&p,&q);
    n=p*q;
    z=(p-1)*(q-1);
    for(i=1;i<z;i++)
    {
        if((z%i)==0)
        continue;
        else
         break;
    }
    e=i;
    printf("\nencryption key is=%lu",e);
    for(i=1;i<z;i++)
            if(((e*i-1)%z)==0)
                    break;
    d=i;
    printf("\ndecryption key is=%lu",d);
    printf("\nenter the msg:");
    scanf("%s",data);
    len=strlen(data);
    for(i=0;i<len;i++)
    {
```

```
            m=(unsigned long) data[i];
        c=modexp(m,e,n);
        printf("\nencrypted key and its representation is %lu\t%c\n",c,c);
        m=modexp(c,d,n);
        printf("\ndecrypted key and its representation is %lu\t%c\n",m,m);
    }
    printf("\n decrypted msg is %s\n%lu\n%lu",data,c,m);
}
```

OUTPUT

```
$ gcc 1.c
$ ./a.out
enter the value of p & q such that p*q>255
5
3
encryption key is=3
decryption key is=3
enter the msg:shashanka
encrypted key and its representation is 10
decrypted key and its representation is 10
encrypted key and its representation is 14
decrypted key and its representation is 14
encrypted key and its representation is 13
decrypted key and its representation is 7
encrypted key and its representation is 10
decrypted key and its representation is 10
encrypted key and its representation is 14
decrypted key and its representation is 14
encrypted key and its representation is 13
decrypted key and its representation is 7
encrypted key and its representation is 5
decrypted key and its representation is 5
encrypted key and its representation is 8
decrypted key and its representation is 2
encrypted key and its representation is 13
decrypted key and its representation is 7
 decrypted msg is shashanka
```

2. **Write a Program to find the shortest path in a network of 6 to 10 nodes.**
   (Dijkstra's algorithm is used here)

```
#include<iostream>
using namespace std;
class dj
{
        int n,cost[10][10],d[10],p[10],v[10];
        public: void read_matrix();
        void short_path(int);
        void display(int);
};
void dj::read_matrix()
{
        int i,j;
        cout<<"Enter the number of vertices\n";
        cin>>n;
        cout<<"Enter the cost adjacency matrix\n";
        for(i=0;i<n;i++)
           for(j=0;j<n;j++)
                cin>>cost[i][j];
}
void dj::short_path(int src)
{
        int i,j,min,u,s;
        for(i=0;i<n;i++)
          {
                d[i]=cost[src][i];
                v[i]=0;
                p[i]=src;
          }
        v[src]=1;
        for(i=0;i<n;i++)
          {
                min=99;
                u=0;
                for(j=0;j<n;j++)
                   {
                        if(!v[j])
                        if(d[j]<min)
                            {
                                min=d[j];
                                u=j;
```

```
                        }
                    }
                v[u]=1;
                for(s=0;s<n;s++)
                if(!v[s]&&(d[u]+cost[u][s]<d[s]))
                    {
                        d[s]=d[u]+cost[u][s];
                        p[s]=u;
                    }
            }
}
void dj::display(int src)
{
        int i,k,parent;
        for(i=0;i<n;i++)
            {
                if(i==src)
                continue;
                cout<<"The shortest path from "<<src<<" to "<<i<<" is "<<endl;
                k=i;
                cout<<k<<"<----";
                while(p[k]!=src)
                    {
                        cout<<p[k]<<"<---";
                        k=p[k];
                    }
                cout<<src<<endl;
                cout<<"and the distance is "<<d[i]<<endl;
            }
}
int main()
{
        int source;
        dj dij;
        dij.read_matrix();
        cout<<"enter the source"<<endl;
        cin>>source;
        dij.short_path(source);
        dij.display(source);
        return 0;
}
```

OUTPUT

```
$ g++ 2.cc -o 2
./2
Enter the number of vertices
6
Enter the cost adjacency matrix
0 2 1 99 99 1
2 0 99 2 1 99
1 99 0 99 2 2
99 2 99 0 1 5
99 1 2 1 0 99
1 99 2 5 99 0
enter the source
1
The shortest path from 1 to 0 is
0<----1
and the distance is 2
The shortest path from 1 to 2 is
2<----4<---1
and the distance is 3
The shortest path from 1 to 3 is
3<----1
and the distance is 2
The shortest path from 1 to 4 is
4<----1
and the distance is 1
The shortest path from 1 to 5 is
5<----0<---1
and the distance is 3
```

3. Write a program for error detecting code using CRC-CCITT (16-bits).

```
#include<stdio.h>
#include<string.h>
char
data[100],concatdata[117],src_crc[17],dest_crc[17],frame[120],divident[18],divisor[18]="10001
000000100001",res[17]="0000000000000000";

void crc_cal(int node)
{
        int i,j;
        for(j=17;j<=strlen(concatdata);j++)
        {
                if(divident[0]=='1')
                {
                        for(i=1;i<=16;i++)
                                if(divident[i]!=divisor[i])
                                        divident[i-1]='1';
                                else
                                divident[i-1]='0';
                }
                else
                {
                        for(i=1;i<=16;i++)
                                divident[i-1]=divident[i];
                }
                if(node==0)
                        divident[i-1]=concatdata[j];
                else
                        divident[i-1]=frame[j];
        }
        divident[i-1]='\0';
        printf("\ncrc is %s\n",divident);
        if(node==0)
        {
                strcpy(src_crc,divident);
        }
        else
                strcpy(dest_crc,divident);
}
```

```
 int main()
{
        int i,len,rest;
        printf("\n\t\t\tAT SOURCE NODE\n\nenter the data to be send :");
        gets(data);

        strcpy(concatdata,data);
        strcat(concatdata,"0000000000000000");
        for(i=0;i<=16;i++)
                        divident[i]=concatdata[i];

        divident[i+1]='\0';
        crc_cal(0);
        printf("\ndata is :\t");
        puts(data);
        printf("\nthe frame transmitted is :\t");
        printf("\n%s%s",data,src_crc);
        printf("\n\t\tSOURCE NODE TRANSMITTED THE FRAME ---->");
        printf("\n\n\n\n\t\t\tAT DESTINATION NODE\nenter the received frame:\t");
        gets(frame);
        for(i=0;i<=16;i++)
                divident[i]=frame[i];
        divident[i+1]='\0';
        crc_cal(1);
        if(strcmp(dest_crc,res)==0)
                printf("\nreceived frame is error free ");
        else
                printf("\nreceived frame has one or more error");
        return 1;
}
```

OUTPUT

```
$ gcc 3.c
$ ./a.out

                        AT SOURCE NODE

enter the data to be send :1101

crc is 1101000110101101

data is :       1101
```

the frame transmitted is :
11011101000110101101
SOURCE NODE TRANSMITTED THE FRAME ---->

AT DESTINATION NODE
enter the received frame: 11011101000110101101

crc is 0000000000000000

received frame is error free

~$ ./a.out

AT SOURCE NODE

enter the data to be send :110011

crc is 0000011000110000

data is : 110011

the frame transmitted is :
1100110000011000110000
SOURCE NODE TRANSMITTED THE FRAME ---->

AT DESTINATION NODE
enter the received frame: 1100110000011000110001

crc is 0000000000000001

received frame has one or more error

4. **Write a program for distance vector algorithm to find suitable path for transmission.**

```
#include<stdio.h>

struct rtable
{
        int dist[20],nextnode[20];
}table[20];

int cost[10][10],n;
void distvector()
{
        int i,j,k,count=0;
        for(i=0;i<n;i++)
        {
                for(j=0;j<n;j++)
                {
                        table[i].dist[j]=cost[i][j];
                        table[i].nextnode[j]=j;
                }
        }
 do
 {
          count=0;
                for(i=0;i<n;i++)
                {
                        for(j=0;j<n;j++)
                        {
                        for(k=0;k<n;k++)
                                {
                                        if(table[i].dist[j]>cost[i][k]+table[k].dist[j])
                                        {
                                                table[i].dist[j]=table[i].dist[k]+table[k].dist[j];
                                                table[i].nextnode[j]=k;
                                                count++;
                                        }
                                }
                        }
                }

        }while(count!=0);
}
```

```
int main()
{
	int i,j;
	printf("\nenter the no of vertices:\t");
	scanf("%d",&n);
	printf("\nenter the cost matrix\n");
	for(i=0;i<n;i++)
			for(j=0;j<n;j++)
				scanf("%d",&cost[i][j]);
	distvector();
	for(i=0;i<n;i++)
	{
		printf("\nstate value for router %c \n",i+65);
		printf("\ndestnode\tnextnode\tdistance\n");
		for(j=0;j<n;j++)
		{
			if(table[i].dist[j]==99)
				printf("%c\t\t-\t\tinfinite\n",j+65);
			else

	printf("%c\t\t%c\t\t%d\n",j+65,table[i].nextnode[j]+65,table[i].dist[j]);
	}	}
return 0;
 }
```

OUTPUT

```
$ gcc 4.c
$ ./a.out
enter the no of vertices:	6
enter the cost matrix
0 2 1 99 99 1
2 0 99 2 1 99
1 99 0 99 2 2
99 2 99 0 1 5
99 1 2 1 0 99
1 99 2 5 99 0
state value for router A
destnode	nextnode	distance
A		A		0
B		B		2
C		C		1
D		B		4
E		B		3
F		F		1
```

state value for router B

destnode	nextnode	distance
A	A	2
B	B	0
C	A	3
D	D	2
E	E	1
F	A	3

state value for router C

destnode	nextnode	distance
A	A	1
B	A	3
C	C	0
D	E	3
E	E	2
F	F	2

state value for router D

destnode	nextnode	distance
A	B	4
B	B	2
C	E	3
D	D	0
E	E	1
F	F	5

state value for router E

destnode	nextnode	distance
A	B	3
B	B	1
C	C	2
D	D	1
E	E	0
F	B	4

state value for router F

destnode	nextnode	distance
A	A	1
B	A	3
C	C	2
D	D	5
E	A	4
F	F	0

5. **Using TCP/IP sockets, write a client – server program to make the client send the file name and to make the server send back the contents of the requested file if present.**

SERVER

```
#include<stdio.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include <stdlib.h>
#include<string.h>
void error(char *msg)
{
        perror(msg);
        exit(1);
}
int main(int argc,char *argv[])
{
        int sockfd,newsockfd,portno,clilen,n,i=0;
        char buffer[256],c[2000],ch;
        struct sockaddr_in serv_addr,cli_addr;
        FILE *fd;
        if(argc < 2)
         {
                fprintf(stderr,"ERROR,no port provided\n");
                exit(1);
        }
         sockfd=socket(AF_INET,SOCK_STREAM,0);
        if(sockfd<0)
                error("ERROR opening socket");
        bzero((char*) &serv_addr,sizeof(serv_addr));
        portno=atoi(argv[1]);
        serv_addr.sin_family=AF_INET;
        serv_addr.sin_addr.s_addr=INADDR_ANY;
        serv_addr.sin_port=htons(portno);
        if(bind(sockfd,(struct sockaddr*)&serv_addr,sizeof(serv_addr))<0)
                error("ERROR on binding");
        listen(sockfd,5);
        clilen=sizeof(cli_addr);
        printf("SERVER:Waiting for client....\n");
         newsockfd=accept(sockfd,(struct sockaddr*) &cli_addr,&clilen);
```

```
	if(newsockfd<0)
		error("ERROR on accept");
	bzero(buffer,256);
	n=read(newsockfd,buffer,255);
	if(n<0)
		error("ERROR reading from socket");
	printf("SERVER:%s \n",buffer);
	if((fd=freopen(buffer,"r",stdin))!=NULL)
	{
		printf("SERVER:%s found! \n Transfering the contents ...\n",buffer);
	while((ch=getc(stdin))!=EOF)
		c[i++]=ch;
	c[i]='\0';
	printf("File content %s\n",c);
	n=write(newsockfd,c,1999);
	if(n<0)
		error("ERROR in writing to socket");
	}
	else
	{
		printf("SERVER:File not found!\n");
		n=write(newsockfd,"File not found!",15);
		if(n<0)
			error("ERROR writing to socket");
	}
	return 0;
}
```

CLIENT

```
#include<stdio.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<netdb.h>
#include<string.h>
#include <stdlib.h>
void error(char *msg)
{
	perror(msg);
	exit(0);
}
int main(int argc,char *argv[])
{
```

```
        int sockfd,portno,n;
        struct sockaddr_in serv_addr;
        struct hostent *server;
        char filepath[256],buf[3000];
        if(argc < 3)
        {
                fprintf(stderr,"usage %s hostname port\n",argv[0]);
                exit(0);
        }
        portno=atoi(argv[2]);
        sockfd=socket(AF_INET,SOCK_STREAM,0);
        if(sockfd<0)
                error("\nerror in opening socket");
        printf("\nclient online");
        server=gethostbyname(argv[1]);
        if(server==NULL)
        {
                fprintf(stderr,"error ,no such host");
                exit(0);
        }
        printf("\n server online");
        bzero((struct sockaddr_in *)&serv_addr,sizeof(serv_addr));
        serv_addr.sin_family=AF_INET;
        bcopy((char *)server->h_addr,(char *)&serv_addr.sin_addr.s_addr,server->h_length);
        serv_addr.sin_port=htons(portno);
        if(connect(sockfd,(struct sockaddr_in*)&serv_addr,sizeof(serv_addr))<0)
                error("error writing to socket");
        printf("\nclient:enter path with filename:\n");
        scanf("%s",filepath);
        n=write(sockfd,filepath,strlen(filepath));
        if(n<0)
                error("\nerror writing to socket");
        bzero(buf,3000);
        n=read(sockfd,buf,2999);
        if(n<0)
                error("\nerror reading to socket");
        printf("\nclient:displaying from socket");
        fputs(buf,stdout);
        return 0;
}
```

OUTPUT : **AT TERMINAL 1 : (SERVER)**

```
$ gcc 5server.c
./a.out 8000
SERVER:Waiting for client....
SERVER:hello.c
SERVER:hello.c found!
 Transfering the contents ...
File content hi
hello
uvce
```

AT TERMINAL 2 : (CLIENT)

```
$ gcc 5client.c
$ ./a.out 127.0.0.1 8000

client online
 server online
client:enter path with filename:
hello.c

client:displaying from sockethi
hello
uvce
```

6. Write a program for error detecting using Hamming Code.

```
#include<stdio.h>
#include<math.h>
void genhamcode();
void makeerror();
void correcterror();
int h[12];
int main()
{
        int i,ch;
        printf("\n enter the message in bits\n");
        for(i=1;i<12;i++)
                if(i==3||i==5||i==6||i==7||i==9||i==10||i==11)
                        scanf("%d",&h[i]);
        for(i=1;i<12;i++)
                printf("%d",h[i]);
        genhamcode();
        printf("\n do you want to make error\n(0 or 1)\n");
        scanf("%d",&ch);
        if(ch)
        {
                makeerror();
                correcterror();
        }
        else
                printf("\n no error");
        return(0);
}
void genhamcode()
{
        int temp,i;
        temp=h[3]+h[5]+h[7]+h[9]+h[11];
        (temp%2!=0)?(h[1]=1):(h[1]=0);
        temp=h[3]+h[6]+h[7]+h[10]+h[11];
        (temp%2!=0)?(h[2]=1):(h[2]=0);
        temp=h[5]+h[6]+h[7];
        (temp%2!=0)?(h[4]=1):(h[4]=0);
        temp=h[9]+h[10]+h[11];
        (temp%2!=0)?(h[8]=1):(h[8]=0);
        printf("\n transmitted codeword is:\n");
        for(i=1;i<12;i++)
        printf(" %d ",h[i]);
```

```
}

void makeerror()
{
        int pos,i;
        printf("\n enter the position you want to make error\n");
        scanf("%d",&pos);
        if(h[pos]==1)
                h[pos]=0;
        else
                h[pos]=1;
        printf("\n Error occured and the error codeword is\n");
        for(i=1;i<12;i++)
                printf(" %d ",h[i]);
}
void correcterror()
{
        int r1,r2,r4,r8,i,errpos;
        r1=(h[1]+h[3]+h[5]+h[7]+h[9]+h[11])%2;
        r2=(h[2]+h[3]+h[6]+h[7]+h[10]+h[11])%2;
        r4=(h[4]+h[5]+h[6]+h[7])%2;
        r8=(h[8]+h[9]+h[10]+h[11])%2;
        errpos=r8*8+r4*4+r2*2+r1*1;
        printf("\n Error occured in pos %d\n",errpos);
        printf("\n\n..... correction starts now.....\n");
        if(h[errpos]==1)
                h[errpos]=0;
        else
                h[errpos]=1;
        printf("\n Original codeword is :");
        for(i=1;i<12;i++)
                printf(" %d ",h[i]);
}
```

OUTPUT

```
$ gcc 6.c
$ ./a.out

 enter the message in bits
1 1 0 1
^Z
[4]+  Stopped                 ./a.out
shashanka@shashanka-945GCM-S2L:~$ ./a.out

 enter the message in bits
```

```
1 1 0 1 0 1 0
00101010010
 transmitted codeword is:
 1  1  1  0  1  0  1  1  0  1  0
 do you want to make error
(0 or 1)
1

 enter the position you want to make error
10

 Error occured and the error codeword is
 1  1  1  0  1  0  1  1  0  0  0
 Error occured in pos 10


..... correction starts now.....

 Original codeword is : 1  1  1  0  1  0  1  1  0  1  0
```

7. Write a Program to implement sliding window protocol.

SENDER

```
#include<sys/socket.h>
#include<sys/types.h>
#include<netinet/in.h>
#include<netdb.h>
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
#include<unistd.h>
#include<errno.h>
int main()
{
        int sock,bytes_received,connected,true=1,i=1,s,f=0,sin_size;
        char send_data[1024],data[1024],c,fr[30]=" ";
        struct sockaddr_in server_addr,client_addr;
        if((sock=socket(AF_INET,SOCK_STREAM,0))==-1)
        {
                perror("Socket not created");
                exit(1);
        }
        if(setsockopt(sock,SOL_SOCKET,SO_REUSEADDR,&true,sizeof(int))==-1)
        {
                perror("Setsockopt");
                exit(1);
        }
        server_addr.sin_family=AF_INET;
        server_addr.sin_port=htons(17000);
        server_addr.sin_addr.s_addr=INADDR_ANY;
        if(bind(sock,(struct sockaddr *)&server_addr,sizeof(struct sockaddr))==-1)
        {
                perror("Unable to bind");
                exit(1);
        }
        if(listen(sock,5)==-1)
        {
                perror("Listen");
                 exit(1);
        }
        fflush(stdout);
```

```
	sin_size=sizeof(struct sockaddr_in);
	connected=accept(sock,(struct sockaddr *)&client_addr,&sin_size);
	while(strcmp(fr,"exit")!=0)
	{
		printf("Enter Data Frame %d:(Enter exit for End):",i);
		scanf("%s",fr);
		send(connected,fr,strlen(fr),0);
		recv(sock,data,1024,0);
		if(strlen(data)!=0)
		printf("I got an acknowledgment : %s\n",data);
		fflush(stdout);
		i++;
	}
	close(sock);
	return(0);
}
```

RECEIVER

```
#include<sys/socket.h>
#include<sys/types.h>
#include<netinet/in.h>
#include<netdb.h>
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
#include<unistd.h>
#include<errno.h>
int main()
{
	int sock,bytes_received,i=1;
	char receive[30];
	struct hostent *host;
	struct sockaddr_in server_addr;
	 host=gethostbyname("127.0.0.1");
	if((sock=socket(AF_INET,SOCK_STREAM,0))==-1)
	{
		perror("Socket not created");
		exit(1);
	}
	printf("Socket created");
	server_addr.sin_family=AF_INET;
	server_addr.sin_port=htons(17000);
	server_addr.sin_addr=*((struct in_addr *)host->h_addr);
	bzero(&(server_addr.sin_zero),8);
	if(connect(sock,(struct sockaddr *)&server_addr,sizeof(struct sockaddr))==-1)
```

```
        {
                perror("Connect");
                exit(1);
        }
        while(1)
        {
                bytes_received=recv(sock,receive,20,0);
                receive[bytes_received]='\0';
                if(strcmp(receive,"exit")==0)
                 {
                        close(sock);
                        break;
                }
                 else
                {
                        if(strlen(receive)<10)
                        {
                                printf("\nFrame %d data %s received\n",i,receive);
                                send(0,receive,strlen(receive),0);
                        }
                        else
                        {
                                send(0,"negative",10,0);
                        }
                        i++;
                }
         }
        close(sock);
        return(0);
}
```

OUTPUT **At terminal 1**

SENDER

```
$ gcc 7sender.c
./a.out
Enter Data Frame 1:(Enter exit for End):computer
I got an acknowledgment : ġ
Enter Data Frame 2:(Enter exit for End):networks
I got an acknowledgment : ġ
Enter Data Frame 3:(Enter exit for End):lab
I got an acknowledgment : ġ
Enter Data Frame 4:(Enter exit for End):exit
I got an acknowledgment : ġ
```

At terminal 2

$ gcc 7receiver.c
$ ./a.out
Socket created
Frame 1 data computer received

Frame 2 data networks received

Frame 3 data lab received

8. **Write a program to implement FIFO-Client and FIFO-Server to transfer files.**

SERVER

```
#include<stdio.h>
#include<stdlib.h>
#include<errno.h>
#include<string.h>
#include<fcntl.h>
#include<sys/types.h>
#include<sys/stat.h>
#include<unistd.h>
#define FIFO1_NAME "fifo1"
#define FIFO2_NAME "fifo2"
int main()
{
        char p[100],f[100],c[300],ch;
        int num,num2,f1,fd,fd2,i=0;

        mknod(FIFO1_NAME,S_IFIFO |0666,0);
        mknod(FIFO2_NAME,S_IFIFO |0666,0);

        printf("\nSERVER ONLINE");
        fd=open(FIFO1_NAME,O_RDONLY);
        printf("client online\nwaiting for request\n\n");
        while(1)
        {
                if((num=read(fd,p,100))==-1)
                        perror("\nread error");
                else
                {
                        p[num]='\0';
                        if((f1=open(p,O_RDONLY))<0)
                        {
                                printf("\nserver: %s not found",p);
                                exit(1);
                        }
                        else
                        {
                          printf("\nserver:%s found \ntranfering the contents",p);
                                stdin=fdopen(f1,"r");
                                while((ch=getc(stdin))!=EOF)
```

```
                                        c[i++]=ch;
                                c[i]='\0';
                                printf("\nfile contents %s\n ",c);
                                fd2=open(FIFO2_NAME,O_WRONLY);
                                if(num2=write(fd2,c,strlen(c))==-1)
                                        perror("\ntranfer error");
                                else
                                        printf("\nserver :tranfer completed");
                        }
                        exit(1);
                }
        }
}
```

CLIENT

```
#include<stdio.h>
#include<stdlib.h>
#include<errno.h>
#include<string.h>
#include<fcntl.h>
#include<sys/types.h>
#include<sys/stat.h>
#include<unistd.h>
#define FIFO1_NAME "fifo1"
#define FIFO2_NAME "fifo2"
int main()
{
        char p[100],f[100],c[3000];
        int num,num2,f1,fd,fd2;
        mknod(FIFO1_NAME,S_IFIFO|0666,0);
        mknod(FIFO2_NAME,S_IFIFO|0666,0);
        printf("\n waiting for server...\n");
        fd=open(FIFO1_NAME,O_WRONLY);
        printf("\n SERVER ONLINE !\n CLIENT:Enter the path\n");
        while(gets(p),!feof(stdin))
        {
                if((num=write(fd,p,strlen(p)))==-1)
                perror("write error\n");
                else
                {
                        printf("Waiting for reply....\n");
                        fd2=open(FIFO2_NAME,O_RDONLY);
                        if((num2=read(fd2,c,3000))==-1)
                                perror("Transfer error!\n");
                        else
```

```
                        {
                                printf("File recieved!   displaying the contents:\n");
                                if(fputs(c,stdout)==EOF)
                                        perror("print error\n");
                                 exit(1);
                        }
                }
        }
}
```

OUTPUT : **AT TERMINAL 1 : (SERVER)**

```
$ gcc 8server.c
$ ./a.out
SERVER ONLINEclient online
waiting for request

server:hello.c found
tranfering the contents
file contents hi
hello
uvce

server :tranfer completed
```

AT TERMINAL 2 : (CLIENT)

```
$ gcc 8client.c
$ ./a.out

 waiting for server...

 SERVER ONLINE !
 CLIENT:Enter the path
hello.c
Waiting for reply....
File recieved!   displaying the contents:
hi
hello
uvce
```

9. Using UDP Sockets write client server program to transfer files.

SERVER

```
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<netdb.h>
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
void error(char *msg)
{
        perror(msg);
        exit(0);
}
int main(int argc, char *argv[])
{
        int sock, length, fromlen, n;
        struct sockaddr_in server;
        struct sockaddr_in from;
        char buf[1024];
        if (argc < 2)
        {
                fprintf(stderr, "ERROR, no port provided\n");
                exit(0);
        }
        Sock=socket(AF_INET, SOCK_DGRAM, 0);
        if (sock < 0)
        {
                error("Opening socket");
        }
        length = sizeof(server);
        bzero(&server,length);
        server.sin_family=AF_INET;
        server.sin_addr.s_addr=INADDR_ANY;
        server.sin_port=htons(atoi(argv[1]));
        if (bind(sock,(struct sockaddr *)&server,length)<0)
        {
                error("binding");
        }
        fromlen = sizeof(struct sockaddr_in);
        while (1)
        {
```

```
                n = recvfrom(sock,buf,1024,0,(struct sockaddr *)&from,&fromlen);
                if (n < 0)
                {
                        error("recvfrom");
                }
                write(1,"Received a datagram: ",21);
                write(1,buf,n);
                n = sendto(sock,"Got your message\n",17,
                0,(struct sockaddr *)&from,fromlen);
                if (n < 0)
                {
                        error("sendto");
                }
        }
}
```

CLIENT

```
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<arpa/inet.h>
#include<netdb.h>
#include<stdio.h>
#include<string.h>
#include <stdlib.h>
void error(char *);
int main(int argc, char *argv[])
{
        int sock, length, n;
        struct sockaddr_in server, from;
        struct hostent *hp;
        char buffer[256];
        if (argc != 3)
        {
                printf("Usage: server port\n");
                exit(1);
        }
        sock= socket(AF_INET, SOCK_DGRAM, 0);
        if(sock<0)
        {
                error("socket");
        }
        server.sin_family=AF_INET;
        hp=gethostbyname(argv[1]);
```

```
        if(hp==0)
        {
                error("Unknown host");
         }
        bcopy((char *)hp->h_addr,(char *)&server.sin_addr,hp->h_length);
        server.sin_port = htons(atoi(argv[2]));
        length=sizeof(struct sockaddr_in);
        printf("Please enter the message: ");
        bzero(buffer,256);
        fgets(buffer,255,stdin);
        n=sendto(sock,buffer,strlen(buffer),0,&server,length);
        if (n < 0)
        {
                error("Sendto");
        }
         n = recvfrom(sock,buffer,256,0,&from, &length);
         if (n < 0)
        {
                error("recvfrom");
        }
        write(1,"Got an ack: ",12);
        write(1,buffer,n);
}
void error(char *msg)
{
        perror(msg);
        exit(0);
}
```

OUTPUT : **AT TERMINAL 1 : (SERVER)**

```
$ gcc 9server.c
~$ ./a.out 8080
Received a datagram: network
Received a datagram: lab
```

AT TERMINAL 2 : (CLIENT)

```
$ gcc 9client.c
$ ./a.out localhost 8080
Please enter the message: network
Got an ack: Got your message
$ ./a.out localhost 8080
Please enter the message: lab
Got an ack: Got your message
```

10.Write a program to implement Diffie-Hellman key Exchange.

```
#include <stdio.h>
#include <math.h>
void main()
{
	int q,alpha,xa,xb,ya,yb,ka,kb, x,y,z,count,ai[20][20];
	printf("Enter a Prime Number \"q\":");
	scanf("%d",&q);
	printf("Enter a No \"xa\" which is lessthan value of q:");
	scanf("%d",&xa);
	printf("Enter a No \"xb\" which is lessthan value of q:");
	scanf("%d",&xb);
	for(x=0;x<q-1;x++) //Primitive Root Calculation
		for(y=0;y<q-1;y++)
			ai[x][y] = ((int)pow(x+1,y+1))%q;
	for(x=0;x<q-1;x++)
	{
		count = 0;
		for(y=0;y<q-2;y++)
		{
			for(z=y+1;z<q-1;z++)
			if(ai[x][y] == ai[x][z])
			{
				count = 1;
				break;
			}
			if(count == 1)
			 break;
		}
		if (count == 0 )
		{
			alpha = x+1;
			break;
		}
	 }
	 printf("alpha = %d\n",alpha);
	ya = ((int)pow(alpha,xa))%q; yb = ((int)pow(alpha,xb))%q;
	ka = ((int)pow(yb,xa))%q; kb = ((int)pow(ya,xb))%q;
	 printf("ya = %d\nyb = %d\nka = %d\nkb = %d\n",ya,yb,ka,kb);
	if(ka == kb) printf("The keys exchanged are same");
	else printf("The keys exchanged are not same");
}
```

OUTPUT

```
$ gcc 10.c -lm
$ ./a.out
Enter a Prime Number "q":11
Enter a No "xa" which is lessthan value of q:7
Enter a No "xb" which is lessthan value of q:5
alpha = 2
ya = 7
yb = 10
ka = 10
kb = 10
The keys exchanged are same
```

11.Write a program to implement Congestion Control using leaky bucket.

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<sys/types.h>
#include<error.h>
#include<sys/stat.h>
#include<unistd.h>
#define min(x,y)((x)<(y)?(x):(y))
#define max(x,y)((x)>(y)?(x):(y))
#define MAX 25
int main()
{
        int cap,oprt,cont,i=0,inp[MAX],ch,nsec,drop;
        printf("LEAKY BUCKET ALGORITM\n");
        printf("\nEnter the bucket size:\n");
        scanf("%d",&cap);
        printf("\nEnter the output rate:");
        scanf("%d",&oprt);
        do
        {
                printf("\nEnter the number of packets entering at %d seconds\n",i+1);
                scanf("%d",&inp[i]);
                 i++;
                printf("\nEnter 1 to insert packet or 0 to quit\n");
                scanf("%d",&ch);
        }
        while(ch);
        nsec=i;
        printf("\n(SECOND):(PACK RECVD):(PACK SENT):(PACK LEFT IN BUCKET):(PACK
dROPPED)\n");
        cont=0;
        drop=0;
        for(i=0;i<nsec;i++)
        {
                cont+=inp[i];
                if(cont>cap)
                {
                        drop=cont-cap;
                        cont=cap;
                }
                printf("(%d): ",i+1);
```

```
                printf("\t\t(%d): ",inp[i]);
                printf("\t\t(%d): ",min(cont,oprt));
                cont=cont-min(cont,oprt);
                printf("\t\t(%d)",cont);
                printf("\t\t(%d)\n",drop);
        }
        for(;cont!=0;i++)
        {
                if(cont>cap)
                        cont=cap;
                drop=0;
                printf("(%d): ",i+1);
                printf("\t\t(0): ");
                printf("\t\t(%d): ",min(cont,oprt));
                cont=cont-min(cont,oprt);
                printf("\t\t(%d)",cont);
                printf("\t\t(%d)\n",drop);
        }
        return(0);
}
```

OUTPUT

```
$ gcc 11.c
$ ./a.out
LEAKY BUCKET ALGORITM

Enter the bucket size:
10

Enter the output rate:4

Enter the number of packets entering at 1 seconds
6

Enter 1 to insert packet or 0 to quit
1

Enter the number of packets entering at 2 seconds
8

Enter 1 to insert packet or 0 to quit
1

Enter the number of packets entering at 3 seconds
12
```

Enter 1 to insert packet or 0 to quit
1

Enter the number of packets entering at 4 seconds
20

Enter 1 to insert packet or 0 to quit
1

Enter the number of packets entering at 5 seconds
4

Enter 1 to insert packet or 0 to quit
0

(SECOND):(PACK RECVD):(PACK SENT):(PACK LEFT IN BUCKET):(PACK dROPPED)

(1):	(6):	(4):	(2)	(0)
(2):	(8):	(4):	(6)	(0)
(3):	(12):	(4):	(6)	(8)
(4):	(20):	(4):	(6)	(16)
(5):	(4):	(4):	(6)	(16)
(6):	(0):	(4):	(2)	(0)
(7):	(0):	(2):	(0)	(0)

PART-B

1. **Simulate a three nodes point – to – point network with duplex links between them. Set the queue size and vary the bandwidth and find the number of packets dropped.**

TCL file

```
set ns [ new Simulator ]
set tf [ open lab1.tr w ]
$ns trace-all $tf
set nf [ open lab1.nam w ]
$ns namtrace-all $nf

# The below code is used to create the nodes.

set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]

#This is used to give color to the packets.

$ns color 1 "red"
$ns color 2 "blue"
$n0 label "Source/udp0"
$n1 label "Source/udp1"
$n2 label "Router"
$n3 label "Destination/Null"

#Vary the below Bandwidth and see the number of packets dropped.

$ns duplex-link $n0 $n2 10Mb 300ms DropTail
$ns duplex-link $n1 $n2 10Mb 300ms DropTail
$ns duplex-link $n2 $n3 1Mb 300ms DropTail

#The below code is used to set the queue size b/w the nodes

$ns set queue-limit $n0 $n2 10
$ns set queue-limit $n1 $n2 10
$ns set queue-limit $n2 $n3 5
```

#The below code is used to attach an UDP agent to n0, UDP #agent to n1 and null agent to n3.

```
set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 attach-agent $udp0
set null3 [new Agent/Null]
$ns attach-agent $n3 $null3

set udp1 [new Agent/UDP]
$ns attach-agent $n1 $udp1

set cbr1 [new Application/Traffic/CBR]
$cbr1 attach-agent $udp1
```

#The below code sets the udp0 packets to red and udp1 #packets to blue color

```
$udp0 set class_ 1
$udp1 set class_ 2
```

#The below code is used to connect the agents.

```
$ns connect $udp0 $null3
$ns connect $udp1 $null3
```

#The below code is used to set the packet size to 500

```
$cbr1 set packetSize_ 500Mb
```

#The below code is used to set the interval of the packets, #i.e., Data rate of the packets. if the data rate is high #then packets drops are high.

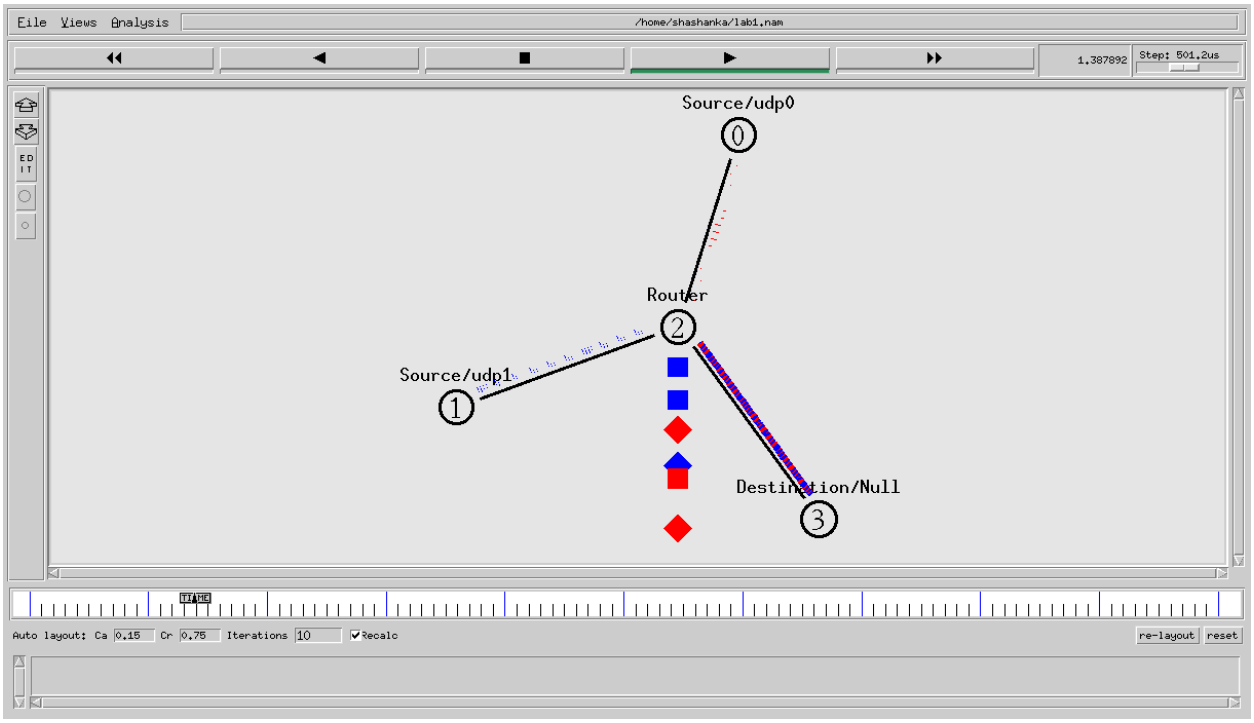
```
$cbr1 set interval_ 0.005
proc finish { } {
        global ns nf tf
        $ns flush-trace
        exec nam lab1.nam &
        close $tf
        close $nf
        exit 0
}
$ns at 0.1 "$cbr0 start"
$ns at 0.1 "$cbr1 start"
$ns at 10.0 "finish"
$ns run
```

awk file

```
BEGIN{
count=0
}
{
        if($1=="d") #d stands for the packets drops.
        count++
}
END{
printf("The Total no of Packets Dropped due to Congestion : %d\n\n", count)
}
```

OUTPUT

$ ns lab1.tcl



awk -f lab1.awk lab1.tr

The Total no of Packets Dropped due to Congestion : 750

2. **Simulate a FOUR node point-to-point network with the links connected as follows:**
   n0-n2, n1-n2 and n2-n3. Apply TCP agent between n0-n3and UDP between n1-n3. Apply relevant applications over TCP and UDP agents changing the parameter and determine the number of packets sent by TCP/UDP.

TCL file

```
set ns [new Simulator]
set tf [open lab2.tr w]
$ns trace-all $tf
set nf [open lab2.nam w]
$ns namtrace-all $nf
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]

# The below code is used to set the color and name's to the    #nodes.

$ns color 1 "red"
$ns color 2 "blue"
$n0 label "source/TCP"
$n1 label "source/UDP"
$n2 label "Router"
$n3 label "destination"
$ns duplex-link $n0 $n2 100Mb 1ms DropTail
$ns duplex-link $n1 $n2 100Mb 1ms DropTail
$ns duplex-link $n2 $n3 100Mb 1ms DropTail

# The below code is used to set the color and labels to the #links.

$ns duplex-link-op $n0 $n2 color "green"
$ns duplex-link-op $n0 $n2 label "from 0-2"
$ns duplex-link-op $n1 $n2 color "green"
$ns duplex-link-op $n1 $n2 label "from 1-2"
$ns duplex-link-op $n2 $n3 color "green"
$ns duplex-link-op $n2 $n3 label "from 2-3"

# The below code is used create TCP and UDP agents and the
# traffic ftp & cbr respectively.
```

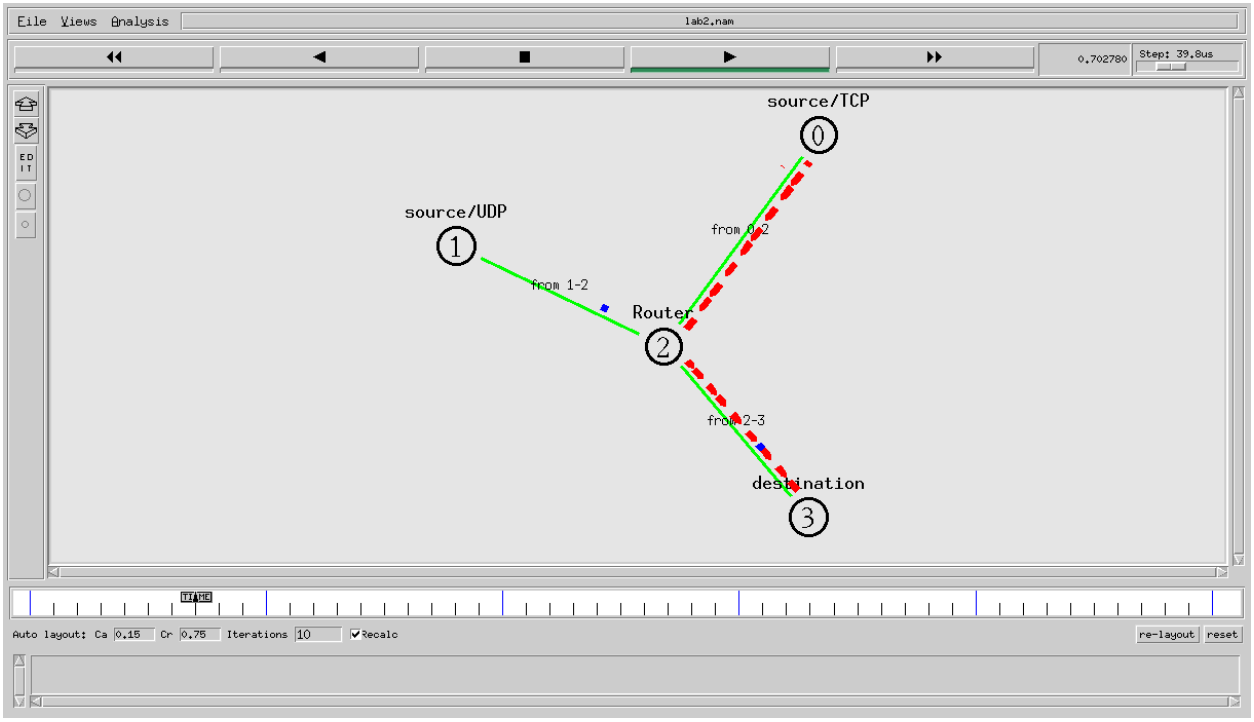
```
set tcp0 [new Agent/TCP]
$ns attach-agent $n0 $tcp0
set ftp0 [new Application/FTP]
$ftp0 attach-agent $tcp0
set sink3 [new Agent/TCPSink]
$ns attach-agent $n3 $sink3

set udp1 [new Agent/UDP]
$ns attach-agent $n1 $udp1
set cbr1 [new Application/Traffic/CBR]
$cbr1 attach-agent $udp1
set null3 [new Agent/Null]
$ns attach-agent $n3 $null3

#The below code is used to set the packet size of ftp and #udp.
$ftp0 set packetSize_ 500
$ftp0 set interval_ 0.001

#The below code is used to increase the data rate(if the #interval is more then the more
number of packets goes to #destination).

$cbr1 set packetSize_ 500
$cbr1 set interval_ 0.001
#This code is used give a color red->tcp and blue ->udp.
$tcp0 set class_ 1
$udp1 set class_ 2

# The below code is used connect the agents.

$ns connect $tcp0 $sink3
$ns connect $udp1 $null3
proc finish { } {
global ns nf tf
$ns flush-trace
exec nam lab2.nam &
close $nf
close $tf
exit 0
}
$ns at 0.1 "$cbr1 start"
$ns at 0.2 "$ftp0 start"
$ns at 5.0 "finish"
$ns run
```

awk file

```
BEGIN{
#include<stdio.h>
tcp=0;
udp=0;
}
{
 if($1=="r"&&$3=="2"&&$4=="3"&& $5=="tcp")
   tcp++;
 if($1=="r"&&$3=="2"&&$4=="3"&&$5=="cbr")
   udp++;
}
END{
printf("\n Total number of packets sent by TCP : %d\n",tcp);
printf("\n Total number of packets sent by UDP : %d\n",udp);
}
```

OUTPUT

$ ns lab2.tcl



$ awk -f lab2.awk lab2.tr

Total number of packets sent by TCP : 22928

Total number of packets sent by UDP : 4898

3. **Simulate the transmission of ping messages over a network topology consisting of 6 nodes and find the number of packets dropped due to congestion.**

TCL file

```
set ns [new Simulator]
#create trace file
set trace_file [open lab3.tr w]
$ns trace-all $trace_file
#create nam trace file
set nam_file [open lab3.nam w]
$ns namtrace-all $nam_file
# Creating nodes.
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]
$n0 label "Ping0"
$n4 label "Ping4"
$n1 label "Ping1"
$n5 label "Ping5"
$ns color 1 "blue"
$ns color 2 "orange"
#establish communication links
$ns duplex-link $n0 $n2 0.5mb 10ms DropTail
$ns duplex-link $n1 $n2 0.5mb 10ms DropTail
$ns duplex-link $n2 $n3 0.5mb 10ms DropTail
$ns duplex-link $n3 $n4 0.5mb 10ms DropTail
$ns duplex-link $n3 $n5 0.5mb 10ms DropTail
# connect the ping agents
set ping0 [new Agent/Ping]
$ns attach-agent $n0 $ping0
set ping4 [new Agent/Ping]
$ns attach-agent $n4 $ping4
set ping1 [new Agent/Ping]
$ns attach-agent $n1 $ping1
set ping5 [new Agent/Ping]
$ns attach-agent $n5 $ping5
$ping0 set packetSize_ 500
$ping0 set interval_ 0.001
```

```
$ping1 set packetSize_ 500
$ping1 set interval_ 0.001
$ping4 set packetSize_ 500
$ping4 set interval_ 0.001
$ping5 set packetSize_ 500
$ping5 set interval_ 0.001
set udp0 [new Agent/UDP]
set null [new Agent/Null]
$ns attach-agent $n0 $udp0
$ns attach-agent $n4 $null
set cbr [new Application/Traffic/CBR]
$cbr set packetSize_ 512
$cbr set interval_ 0.001
$cbr attach-agent $udp0
$ns connect $udp0 $null
$ping0 set class_ 1
$ping1 set class_ 2
#ping the receiver from other nodes
$ns connect $ping0 $ping4
$ns connect $ping1 $ping5
#The below function is executed when the ping agent receives a reply from the destination
Agent/Ping instproc recv {from rtt} {
$self instvar node_
puts " The node [$node_ id] received a reply from $from with round trip time of $rtt ms"
}
#define finish procedure
proc finish { } {
global ns nam_file trace_file

$ns flush-trace
exec nam lab3.nam &

close $trace_file
close $nam_file
exit 0
}
#schedule events to start sending the ping packets
$ns at 0.1 "$ping0 send"
$ns at 0.2 "$ping0 send"
$ns at 0.3 "$ping0 send"
$ns at 0.4 "$ping0 send"
$ns at 0.5 "$ping0 send"
$ns at 0.6 "$ping0 send"
$ns at 0.7 "$ping0 send"
```

```
$ns at 0.8 "$ping0 send"
$ns at 0.9 "$ping0 send"
$ns at 1.0 "$ping0 send"
$ns at 0.2 "$cbr start"
$ns at 4.0 "$cbr stop"
$ns at 0.1 "$ping1 send"
$ns at 0.2 "$ping1 send"
$ns at 0.3 "$ping1 send"
$ns at 0.4 "$ping1 send"
$ns at 0.5 "$ping1 send"
$ns at 0.6 "$ping1 send"
$ns at 0.7 "$ping1 send"
$ns at 0.8 "$ping1 send"
$ns at 0.9 "$ping1 send"
$ns at 1.0 "$ping1 send"
$ns at 5.5 "finish"
$ns run
```

Awk file

```
BEGIN{
#include<stdio.h>
count=0
}
{
 if($1=="d")
 count++
}
END{
 printf("The Total no of Packets Dropped due toCongestion:%d ", count)
}
```

OUTPUT

```
$ ns lab3.tcl

 The node 0 received a reply from 4 with round trip time of 108.0 ms

 The node 1 received a reply from 5 with round trip time of 116.0 ms

 The node 0 received a reply from 4 with round trip time of 108.0 ms

 The node 1 received a reply from 5 with round trip time of 116.0 ms

 The node 1 received a reply from 5 with round trip time of 122.3 ms
```

The node 1 received a reply from 5 with round trip time of 128.6 ms

The node 1 received a reply from 5 with round trip time of 134.9 ms

The node 1 received a reply from 5 with round trip time of 141.2 ms

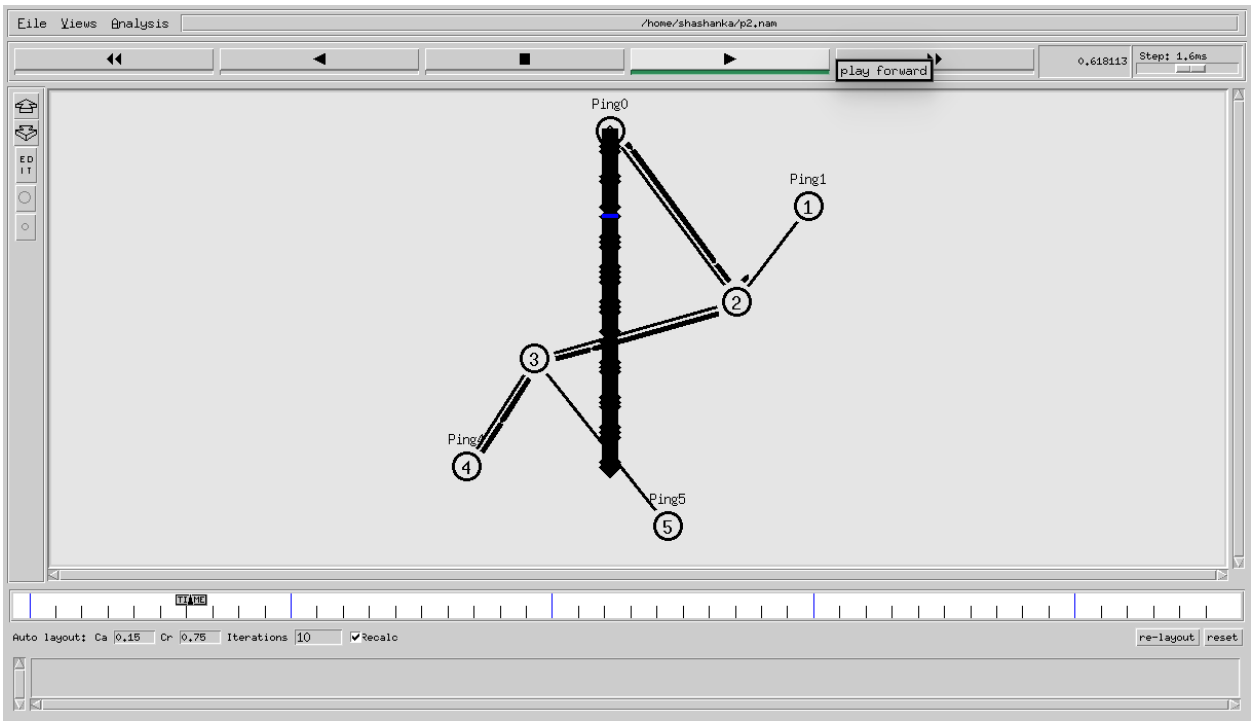
The node 1 received a reply from 5 with round trip time of 155.7 ms

The node 1 received a reply from 5 with round trip time of 162.0 ms

The node 1 received a reply from 5 with round trip time of 168.3 ms

The node 1 received a reply from 5 with round trip time of 174.6 ms

The node 0 received a reply from 4 with round trip time of 581.1 ms



$ awk -f lab3.awk lab3.tr

The Total no of Packets Dropped due toCongestion:3297

4. **Simulate an Ethernet LAN using n nodes and set multiple traffic nodes and plot congestion window for different source / destination**

TCL File

```
set ns [new Simulator]

set tf [open lab4.tr w]
$ns trace-all $tf

set nf [open lab4.nam w]
$ns namtrace-all $nf

set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]

$ns make-lan "$n0 $n1 $n2 $n3" 10mb 10ms LL Queue/DropTail  Mac/802_3

set tcp0 [new Agent/TCP]
$ns attach-agent $n0 $tcp0
set ftp0 [new Application/FTP]
$ftp0 attach-agent $tcp0
set sink3 [new Agent/TCPSink]
$ns attach-agent $n3 $sink3

$ns connect $tcp0 $sink3
set tcp2 [new Agent/TCP]
$ns attach-agent $n2 $tcp2
set ftp2 [new Application/FTP]
$ftp2 attach-agent $tcp2

set sink1 [new Agent/TCPSink]
$ns attach-agent $n1 $sink1

$ns connect $tcp2 $sink1

######To trace the congestion window##########
set file1 [open file1.tr w]
$tcp0 attach $file1
$tcp0 trace cwnd_
$tcp0 set maxcwnd_ 10
```

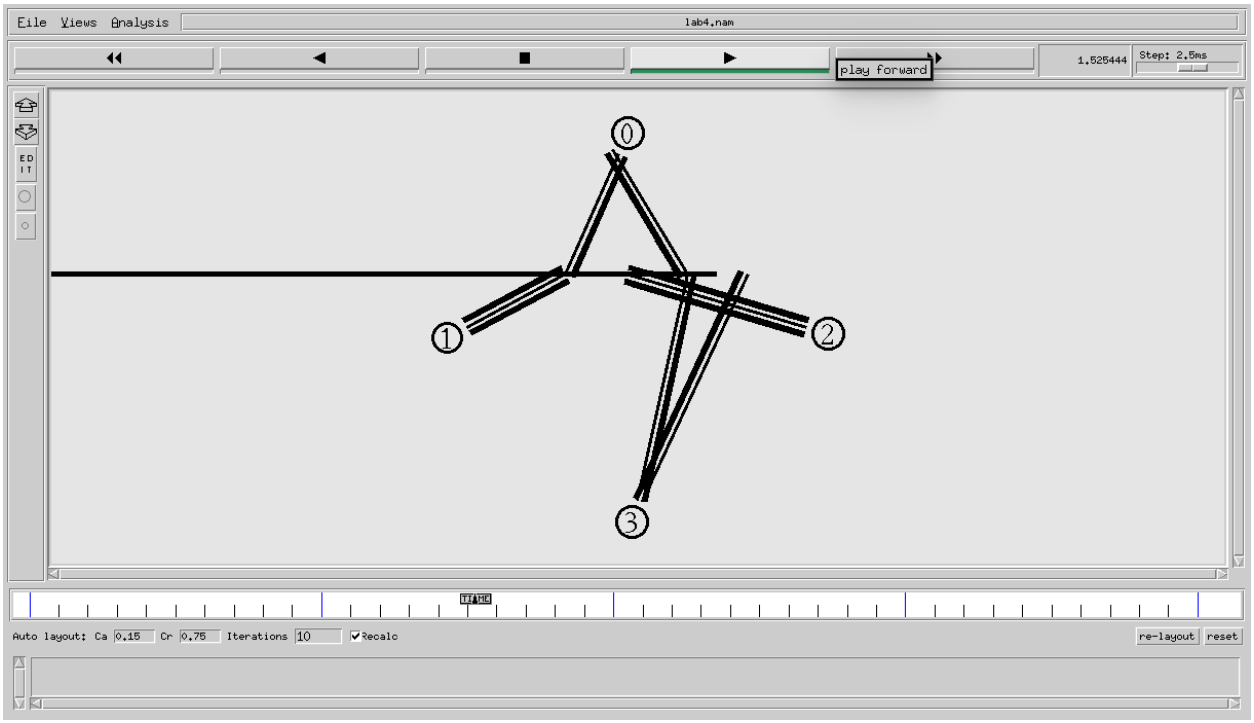
```
set file2 [open file2.tr w]
$tcp2 attach $file2
$tcp2 trace cwnd_

proc finish { } {
global nf tf ns
$ns flush-trace
exec nam lab4.nam &
close $nf
close $tf
exit 0
}

$ns at 0.1 "$ftp0 start"
$ns at 1.5 "$ftp0 stop"

$ns at 2 "$ftp0 start"
$ns at 3 "$ftp0 stop"
$ns at 0.2 "$ftp2 start"
$ns at 2 "$ftp2 stop"

$ns at 2.5 "$ftp2 start"
$ns at 4 "$ftp2 stop"
$ns at 5.0 "finish"
$ns run
```

Awk file

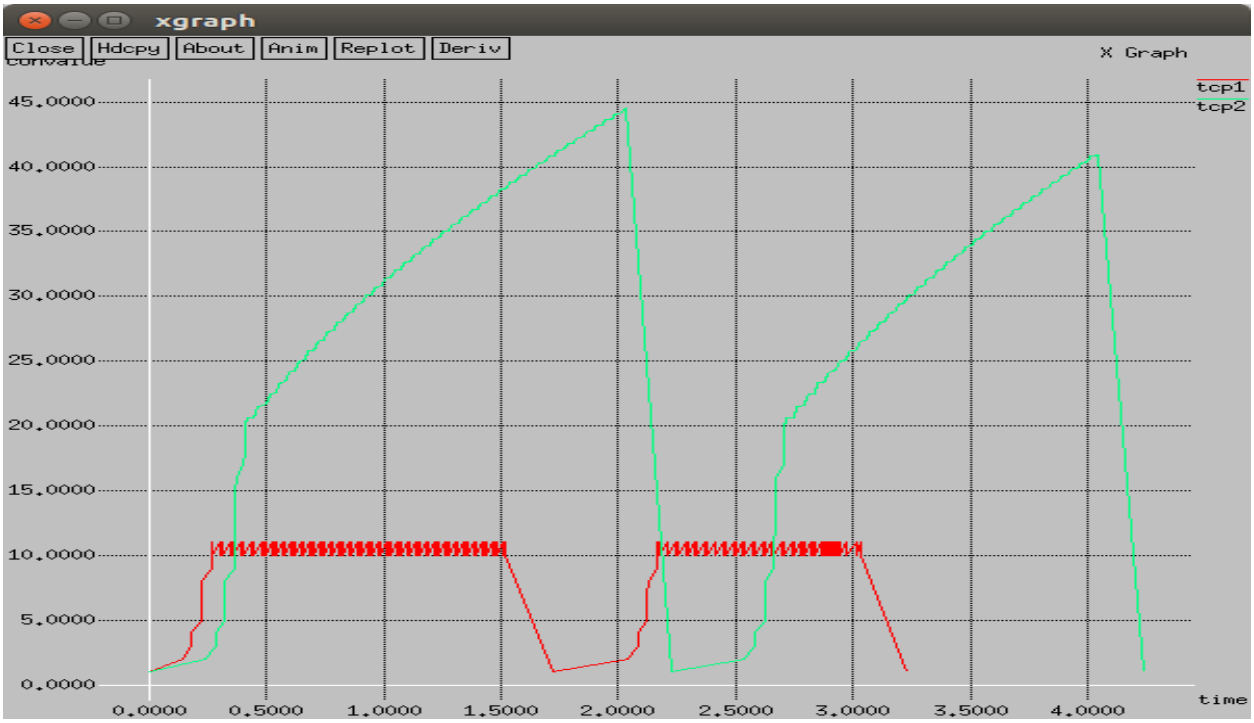
```
BEGIN{
#include<stdio.h>
}
{
 if($6=="cwnd_")
   printf("%f \t %f \n", $1,$7);
}
END{
puts "DONE"
}
```

OUTPUT

```
$ ns lab4.tcl
```



$ awk -f lab4.awk file1.tr>tcp1
$ awk -f lab4.awk file2.tr>tcp2
$ xgraph -x "time" -y "convalue" tcp1 tcp2



5. Simulate the Ethernet LAN using n nodes (6-10), change error rate and data rate and compare throughput

TCL File

```
set ns [new Simulator]
set tf [open lab5.tr w]
$ns trace-all $tf
set nf [open lab5.nam w]
$ns namtrace-all $nf

$ns color 1 "red"

set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]
set n6 [$ns node]

$n1 label "Source/UDP"
$n3 label "Error Node"
$n5 label "Destination"
#The below code is used to create a two Lans (Lan1 and     #Lan2).

$ns make-lan "$n0 $n1 $n2 $n3" 10Mb 10ms LL Queue/DropTail Mac/802_3
$ns make-lan "$n4 $n5 $n6" 10Mb 10ms LL Queue/DropTail Mac/802_3

#The below code is used to connect node n3 of lan1 and n6 of #lan2.
$ns duplex-link $n3 $n6 100Mb 10ms DropTail

set udp1 [new Agent/UDP]
$ns attach-agent $n1 $udp1
set cbr1 [ new Application/Traffic/CBR]
$cbr1 attach-agent $udp1
set null5 [new Agent/Null]
$ns attach-agent $n5 $null5

$ns connect $udp1 $null5
$cbr1 set packetSize_ 1000
$cbr1 set interval_ 0.0001 ;# This is the data rate. Change
                    ;# this to increase the rate.
```

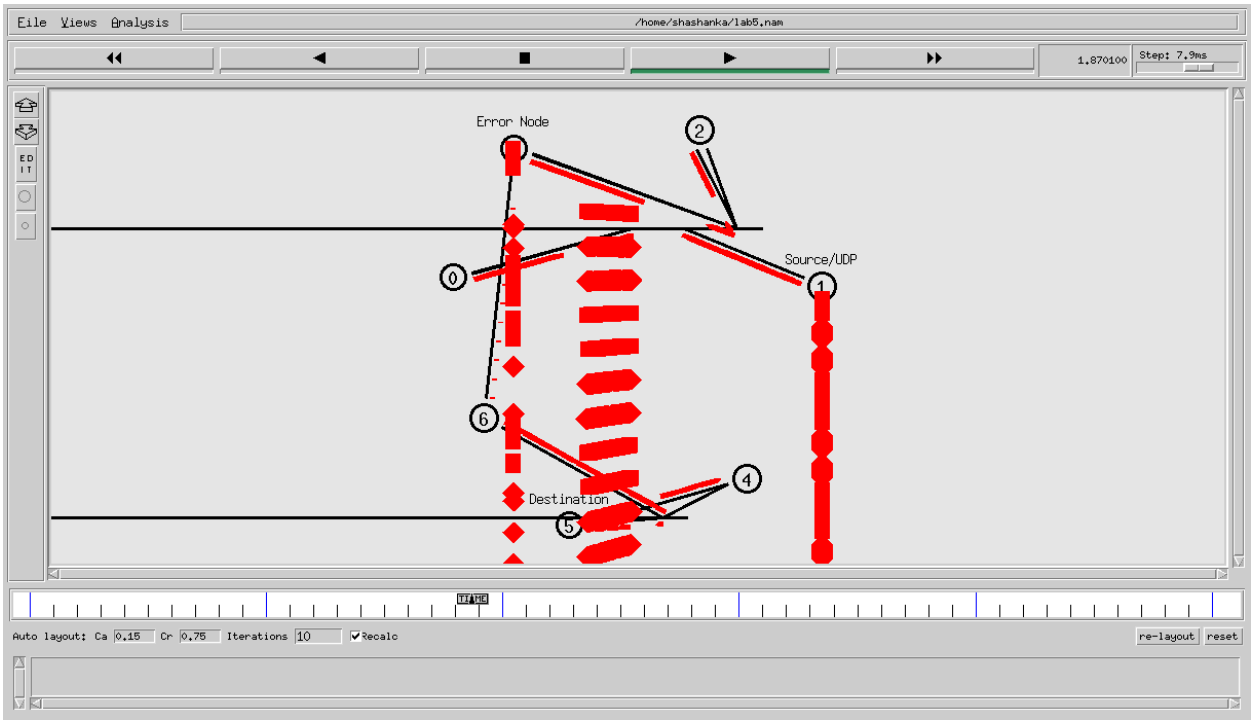
```
$udp1 set class_ 1

# The below code is used to add an error model between the #nodes n3 and n6.

set err [new ErrorModel]
$ns lossmodel $err $n3 $n6
$err set rate_ 0.2   ;# This is the error rate. Change this
                ;#rate to add errors between n3 and n6.
proc finish { } {
global nf ns tf
exec nam lab5.nam &
close $nf
close $tf
exit 0
}
$ns at 5.0 "finish"
$ns at 0.1 "$cbr1 start"
$ns run
```

Awk File

```
BEGIN{
#include <stdio.h>
pkt=0;
time=0
}
{
        if($1="r" && $3=="8" && $4=="5")
        {
                pkt=pkt+$6
                time=$2
        }
}
END{
  printf(" Throughput: %fMbps\n\n",(pkt/time)*(8/1000000));
}
```

OUTPUT

```
$ ns lab5.tcl
```



$ awk -f lab5.awk lab5.tr
Throughput: 7.623383Mbps