Tcps.c

```
#include"sys/socket.h"
#include"netinet/in.h"
#include"stdio.h"
#include"string.h"
#include"stdlib.h"
int main()
{
	char buf[100];
	int k;
	socklen_t len;
	int sock_desc,temp_sock_desc;
	struct sockaddr_in server,client;
	sock_desc=socket(AF_INET,SOCK_STREAM,0);
	if(sock_desc==-1)
	printf("error in socket creation");
	server.sin_family=AF_INET;
	server.sin_addr.s_addr=INADDR_ANY;
	server.sin_port=3003;
	client.sin_family=AF_INET;
	client.sin_addr.s_addr=INADDR_ANY;
	client.sin_port=3003;
	k=bind(sock_desc,(struct sockaddr*)&server,sizeof(server));
	if(k==-1)
	printf("error in binding");
	k=listen(sock_desc,5);
	if(k==-1)
	printf("error in listening");
	len=sizeof(client);
	temp_sock_desc=accept(sock_desc,(struct sockaddr*)&client,&len);
	if(temp_sock_desc==-1)
	printf("error in temporary socket creation");
	k=recv(temp_sock_desc,buf,100,0);
	if(k==-1)
	printf("error in receiving");
	printf("message got from client is: %s",buf);
	close(temp_sock_desc);
	return 0;
}
```

tcpc.c

```
#include"sys/socket.h"
#include"netinet/in.h"
#include"stdio.h"
#include"string.h"
#include<stdlib.h>
int main()
{
        char buf[100];
        int k;
        int sock_desc;
        struct sockaddr_in client;
        sock_desc=socket(AF_INET,SOCK_STREAM,0);
        if(sock_desc==-1)
        printf("error in socket creation");
        client.sin_family=AF_INET;
        client.sin_addr.s_addr=INADDR_ANY;
        client.sin_port=3003;
        k=connect(sock_desc,(struct sockaddr*)&client,sizeof(client));
        if(k==-1)
        printf("error in connecting to server");
        printf("\nEnter data to be send:");
        fgets(buf,100,stdin);
        k=send(sock_desc,buf,100,0);
        printf("error in sending");
        close(sock_desc);
        return 0;
}
```

udps.c

```
#include <stdio.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <arpa/inet.h>
#include <sys/socket.h>
#include <netinet/in.h>
#define PORT 5000
#define MAXLINE 1000
int main()
{
	char buffer[MAXLINE];
	char message[MAXLINE];
	int sockfd, len;
	struct sockaddr_in servaddr, cliaddr;
	bzero(&servaddr, sizeof(servaddr));
	sockfd = socket(AF_INET, SOCK_DGRAM, 0);
	servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
	servaddr.sin_port = htons(PORT);
	servaddr.sin_family = AF_INET;
	bind(sockfd, (struct sockaddr*)&servaddr, sizeof(servaddr));
	len = sizeof(cliaddr);
	int n = recvfrom(sockfd, buffer, sizeof(buffer), 0, (struct sockaddr*)&cliaddr, &len);
	buffer[n] = '\0';
	puts(buffer);
	printf("Enter response message: ");
	fgets(message, sizeof(message), stdin);
	sendto(sockfd, message, strlen(message), 0, (struct sockaddr*)&cliaddr, sizeof(cliaddr));
	return 0;
}
```

udpc.c

```
#include <stdio.h>
#include <string.h>
#include <sys/types.h>
#include <arpa/inet.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <unistd.h>
#include <stdlib.h>
#define PORT 5000
#define MAXLINE 1000
int main()
{
	char buffer[MAXLINE];
	char message[MAXLINE];
	int sockfd, n;
	struct sockaddr_in servaddr;
	bzero(&servaddr, sizeof(servaddr));
	servaddr.sin_addr.s_addr = inet_addr("127.0.0.1");
	servaddr.sin_port = htons(PORT);
	servaddr.sin_family = AF_INET;
	sockfd = socket(AF_INET, SOCK_DGRAM, 0);
	if (connect(sockfd, (struct sockaddr *)&servaddr, sizeof(servaddr)) < 0)
		{
			printf("\nError: Connect Failed\n");
			exit(0);
		}
	printf("Enter message to send: ");
	fgets(message, sizeof(message), stdin);
	sendto(sockfd, message, strlen(message), 0, (struct sockaddr *)NULL, sizeof(servaddr));
	recvfrom(sockfd, buffer, sizeof(buffer), 0, (struct sockaddr *)NULL, NULL);
	printf("Response from server: %s\n", buffer);
	close(sockfd);
	return 0;
}
```

multis.c

```
#include<stdio.h>
#include<stdlib.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<string.h>
#include<netinet/in.h>
#define PORT 4444
#define BUF_SIZE 2000
#define CLADDR_LEN 100
void main()
{
	struct sockaddr_in addr,cl_addr;
	int sockfd,len,ret,newsockfd;
	char buffer[BUF_SIZE];
	pid_t childpid;
	char clientAddr[CLADDR_LEN];
	sockfd=socket(AF_INET,SOCK_STREAM,0);
	if(sockfd<0)
		{
			printf("ERROR CREATING SOCKET\n");
			exit(1);
		}
	printf("SOCKET CREATED\n");
	memset(&addr,0,sizeof(addr));
	addr.sin_family=AF_INET;
	addr.sin_addr.s_addr=INADDR_ANY;
	addr.sin_port=PORT;
	ret=bind(sockfd,(struct sockaddr *)&addr,sizeof(addr));
	if(ret<0)
		{
			printf("error binding!\n");
			exit(1);
		}
	printf("binding done...\n");
	printf("waiting for a connection...\n");
	listen(sockfd,5);
	for(;;)
		{
			len=sizeof(cl_addr);
			newsockfd=accept(sockfd,(struct sockaddr *)&cl_addr,&len);
			if(newsockfd<0)
				{
					printf("error accepting connection!\n");
					exit(1);
				}
			printf("connection accepted...\n");
			inet_ntop(AF_INET,&(cl_addr.sin_addr),clientAddr,CLADDR_LEN);
			if((childpid=fork())==0)
				{
					close(sockfd);
```

```
                                        for(;;)
                                                {
                                                        memset(buffer,0,BUF_SIZE);
                                                        ret=recv(newsockfd,buffer,BUF_SIZE,0);
                                                        if(ret<0)
                                                                {
                                                                        printf("error receiving data!\n");
                                                                        exit(1);
                                                                }
                                                        printf("received data from %s:%s\
n",clientAddr,buffer);
                                                        ret=send(newsockfd,buffer,BUF_SIZE,0);
                                                        if(ret<0)
                                                                {
                                                                        printf("error sending data!\n");
                                                                        exit(1);
                                                                }
                                                        printf("sent data to %s:%s\n",clientAddr,buffer);
                                                }
                                        }
                                close(newsockfd);
                        }
}
```

multic.c

```
#include"stdio.h"
#include"stdlib.h"
#include"sys/types.h"
#include"sys/socket.h"
#include"string.h"
#include"netinet/in.h"
#include"netdb.h"
#define PORT 4444
#define BUF_SIZE 2000
int main(int argc,char**argv)
{
	struct sockaddr_in addr,cl_addr;
	int sockfd,ret;
	char buffer[BUF_SIZE];
	struct hostent * server;
	char * serverAddr;
	if(argc<2)
		{
			printf("usage: client<ip address>\n");
			exit(1);
		}
	serverAddr=argv[1];
	sockfd=socket(AF_INET,SOCK_STREAM,0);
	if(sockfd<0)
		{
			printf("error creating socket!\n");
			exit(1);
		}
	printf("Socket created...\n");
	memset(&addr,0,sizeof(addr));
	addr.sin_family=AF_INET;
	addr.sin_addr.s_addr=inet_addr(serverAddr);
	addr.sin_port=PORT;
	ret=connect(sockfd,(struct sockaddr *)&addr,sizeof(addr));
	if(ret<0)
		{
			printf("error connecting to the server!\n");
			exit(1);
		}
	printf("Connected to the server...\n");
	memset(buffer,0,BUF_SIZE);
	printf("enter your message(s):");
	while(fgets(buffer,BUF_SIZE,stdin)!=NULL)
		{
			ret=send(sockfd,buffer,BUF_SIZE,0);
			if(ret<0)
				{
					printf("Error sending data!\n\t-%s",buffer);
				}
			ret=recv(sockfd,buffer,BUF_SIZE,0);
```

```
                        if(ret<0)
                                {
                                        printf("error receiving data!\n");
                                }
                        else
                                {
                                        printf("received:");
                                        fputs(buffer,stdout);
                                        printf("\n");
                                }
                }
        return 0;
}
```

gobackn_s.c

```
#include<stdio.h>
#include<string.h>
#include<sys/types.h>
#include<netinet/in.h>
#include<netdb.h>
#include<unistd.h>
struct frame
{
        int packet[40];
};
struct ack
{
        int acknowledge[40];
};

int main()
{
        int serversocket;
        struct sockaddr_in serveraddr,clientaddr;
        socklen_t len;
        struct frame f1;
        int windowsize, totalpackets, totalframes, i = 0, j = 0, framesend=0,framesreceived = 0, k, l,
buffer;
        struct ack acknowledgement;
   char req[50];
   serversocket = socket(AF_INET, SOCK_DGRAM, 0);
   bzero((char*)&serveraddr,sizeof(serveraddr));
   serveraddr.sin_family = AF_INET;
   serveraddr.sin_port = htons(5018);
   serveraddr.sin_addr.s_addr=INADDR_ANY;
   bind(serversocket,(struct sockaddr*)&serveraddr,sizeof(serveraddr));
   bzero((char*)&clientaddr,sizeof(clientaddr));
   len=sizeof(clientaddr);
   printf("\nWaiting for client connection");
   recvfrom(serversocket, req, sizeof(req), 0, (struct sockaddr*)&clientaddr, &len);
   printf("\nSending request for window size\n");
   sendto(serversocket, "REQUEST FOR WINDOW SIZE", sizeof("REQUEST FOR WINDOW
SIZE"), 0, (struct sockaddr*)&clientaddr,sizeof(clientaddr));
   printf("waiting for th window size\n");
   recvfrom(serversocket, (char*)&windowsize, sizeof(windowsize), 0, (struct
sockaddr*)&clientaddr, &len);
   printf("\n The window size obtained as :\t %d \n",windowsize);
   printf("\nObtaining packets from network layer\n");
   printf("\nTotal packets obtained: %d\n",(totalpackets=windowsize*5));
   printf("\nTotal frames or windows to be transmitted: %d\n",(totalframes=5));
   printf("\n sending total number of packets\n");
   sendto(serversocket, (char*)&totalpackets, sizeof(totalpackets), 0, (struct sockaddr*)&clientaddr,
sizeof(clientaddr));
   recvfrom(serversocket, req, sizeof(req), 0, (struct sockaddr*)&clientaddr, &len);
```

```
   printf("\nSending total number of frames\n");
   sendto(serversocket, (char*)&totalframes, sizeof(totalframes), 0, (struct sockaddr*)&clientaddr,
sizeof(clientaddr));
   recvfrom(serversocket, req, sizeof(req), 0, (struct sockaddr*)&clientaddr, &len);
   printf("\nPress Enter to start the process\n");
   fgets(req,2,stdin);
   while(i<totalpackets)
        {
                bzero((char*)&f1,sizeof(f1));
                printf("\nInitializing the transmit buffer\n");
                printf("\nThe frame to be send is %d with packets:",framesend);
                buffer=1;
                j=0;
                while(j<windowsize && i<totalpackets)
                        {
                                printf("%d",i);
                                f1.packet[j]=i;
                                j++;
                                i++;
                        }
                printf("sending frame %d\n",framesend);
                sendto(serversocket, (char*)&f1, sizeof(f1), 0, (struct sockaddr*)&clientaddr,
sizeof(clientaddr));
                printf("Waiting for the acknowledgement\n");
                recvfrom(serversocket, (char*)&acknowledgement, sizeof(acknowledgement), 0,
(struct sockaddr*)&clientaddr, &len);
                j=0;
                k=0;
                l=buffer;
                while(j<windowsize && l<totalpackets)
                        {
                                if(acknowledgement.acknowledge[j]==-1)
                                        {
                                                printf("\nNegative acknowledgement received for
packet: %d\n",f1.packet[j]);
                                                printf("\nRetransitting from packet: %d\
n",f1.packet[j]);
                                                i=f1.packet[j];
                                                k=1;
                                                break;
                                        }
                                if(k==0)
                                        {
                                                printf("\nPositive acknowledgement received for all
packets, within the frame :%d\n",framesend);
                                        }
                                framesend++;
                                printf("\nPress enter to proceed\n");
                                fgets(req,2,stdin);
                        }
                printf("\nAll frames sends successfully\n closing connection with the client\n");
                close(serversocket);
```

```
        }
    }
```

gobackn_c.c

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

struct frame {
   int packet[40];
};

struct ack {
   int acknowledge[40];
};

int main() {
   int clientsocket;
   struct sockaddr_in serveraddr;
   socklen_t len;
   struct hostent *server;
   struct frame f1;
   int windowsize, totalpackets, totalframes, i = 0, j = 0, framesreceived = 0, k, l, buffer;
   struct ack acknowledgement;
   char req[50];

   clientsocket = socket(AF_INET, SOCK_DGRAM, 0);
   if (clientsocket < 0) {
      perror("ERROR opening socket");
      exit(1);
   }

   bzero((char*)&serveraddr, sizeof(serveraddr));
   serveraddr.sin_family = AF_INET;
   serveraddr.sin_port = htons(5018);

   server = gethostbyname("127.0.0.1");
   if (server == NULL) {
      fprintf(stderr, "ERROR, no such host\n");
      exit(0);
   }

   bcopy((char*)server->h_addr, (char*)&serveraddr.sin_addr.s_addr, sizeof(server->h_addr));

   printf("Sending request to the server\n");
   sendto(clientsocket, "HI IAM CLIENT", strlen("HI IAM CLIENT"), 0, (struct
sockaddr*)&serveraddr, sizeof(serveraddr));

   printf("\nWaiting for reply\n");
   recvfrom(clientsocket, req, sizeof(req), 0, (struct sockaddr*)&serveraddr, &len);
```

```
    printf("\nThe server has to send:\t%s\n", req);

    printf("\nEnter the window size\n");
    scanf("%d", &windowsize);

    printf("\nSending window size\n");
    sendto(clientsocket, (char*)&windowsize, sizeof(windowsize), 0, (struct sockaddr*)&serveraddr,
sizeof(serveraddr));

    printf("\nWaiting for the server response\n");
    recvfrom(clientsocket, (char*)&totalpackets, sizeof(totalpackets), 0, (struct
sockaddr*)&serveraddr, &len);
    printf("\nTotal packets are :\t%d\n", totalpackets);
    sendto(clientsocket, "RECEIVED", strlen("RECEIVED"), 0, (struct sockaddr*)&serveraddr,
sizeof(serveraddr));

    recvfrom(clientsocket, (char*)&totalframes, sizeof(totalframes), 0, (struct
sockaddr*)&serveraddr, &len);
    printf("\nTotal number of frames or windows are:\t%d\n", totalframes);
    sendto(clientsocket, "RECEIVED", strlen("RECEIVED"), 0, (struct sockaddr*)&serveraddr,
sizeof(serveraddr));

    printf("\nStarting the process of receiving\n");
    while (i < totalpackets) {
        printf("\nInitializing the received buffer\n");
        printf("\nThe expected frame is %d with packets:", framesreceived);

        j = 0;
        buffer = i;
        while (j < windowsize && i < totalpackets) {
            printf("%d", i);
            i++;
            j++;
        }

        printf("\nWaiting for the frame\n");
        recvfrom(clientsocket, (char*)&f1, sizeof(f1), 0, (struct sockaddr*)&serveraddr, &len);
        printf("\nReceived frame %d\n\nEnter -1 to send negative acknowledgement for the following
packets\n", framesreceived);

        j = 0;
        i = buffer;
        k = 0;
        l = buffer;

        while (j < windowsize && i < totalpackets) {
            printf("\nPacket:%d\n", f1.packet[j]);
            scanf("%d", &acknowledgement.acknowledge[j]);

            if (acknowledgement.acknowledge[j] == -1) {
                if (k == 0) {
                    i = f1.packet[j];
```

```
                k = 1;
            }
        }
        j++;
        l++;
    }

    framesreceived++;
    sendto(clientsocket, (char*)&acknowledgement, sizeof(acknowledgement), 0, (struct
sockaddr*)&serveraddr, sizeof(serveraddr));
  }

  printf("\nAll frames received successfully\nClosing connection with the server\n");
  close(clientsocket);
  return 0;
}
```

selectiver_s.cpp

```
#include<iostream>
#include<stdio.h>
#include <strings.h>
#include<sys/types.h>
#include<netinet/in.h>
#include<netdb.h>
#define cls() printf("33[H33[J")

struct frame

{

 int packet[40];

};

struct ack

{

 int acknowledge[40];

};

int main()

{

 int serversocket;

 sockaddr_in serveraddr,clientaddr;

 socklen_t len;

 int windowsize,totalpackets,totalframes,framessend=0,i=0,j=0,k,l,m,n,repacket[40];

 ack acknowledgement;

 frame f1;

 char req[50];


 serversocket=socket(AF_INET,SOCK_DGRAM,0);


 bzero((char*)&serveraddr,sizeof(serveraddr));
```

```
serveraddr.sin_family=AF_INET;
serveraddr.sin_port=htons(5018);
serveraddr.sin_addr.s_addr=INADDR_ANY;

bind(serversocket,(sockaddr*)&serveraddr,sizeof(serveraddr));

bzero((char*)&clientaddr,sizeof(clientaddr));
len=sizeof(clientaddr);
printf("\nWaiting for client connection.\n");
recvfrom(serversocket,req,sizeof(req),0,(sockaddr*)&clientaddr,&len);
printf("\nThe client connection obtained.\t%s\n",req);
printf("\nSending request for window size.\n");
sendto(serversocket,"REQUEST FOR WINDOWSIZE.",sizeof("REQUEST FOR
WINDOWSIZE."),0,(sockaddr*)&clientaddr,sizeof(clientaddr));
printf("\nWaiting for the windowsize.\n");
recvfrom(serversocket,(char*)&windowsize,sizeof(windowsize),0,(sockaddr*)&clientaddr,&len);
cls();
printf("\nThe windowsize obtained as:\t%d\n",windowsize);
printf("\nObtaining packets from network layer.\n");
printf("\nTotal packets obtained:\t%d\n",(totalpackets=windowsize*2));
printf("\nTotal frames or windows to be transmitted:\t%d\n",(totalframes=2));
printf("\nSending total number of packets.\n");
sendto(serversocket,(char*)&totalpackets,sizeof(totalpackets),0,
(sockaddr*)&clientaddr,sizeof(clientaddr));
recvfrom(serversocket,req,sizeof(req),0,(sockaddr*)&clientaddr,&len);
printf("\nSending total number of frames.\n");
```

```
 sendto(serversocket,(char*)&totalframes,sizeof(totalframes),0,
(sockaddr*)&clientaddr,sizeof(clientaddr));

 recvfrom(serversocket,req,sizeof(req),0,(sockaddr*)&clientaddr,&len);

 printf("\nPRESS ENTER TO START THE PROCESS.\n");

 fgets(req,2,stdin);

 cls();


j=0;

l=0;

 while( l<totalpackets)

 {

  bzero((char*)&f1,sizeof(f1));

  printf("\nInitialising the transmit buffer.\n");

  printf("\nThe frame to be send is %d with packets:\t",framessend);


  for(m=0;m<j;m++)

  {

   printf("%d  ",repacket[m]);

   f1.packet[m]=repacket[m];

  }


  while(j<windowsize && i<totalpackets)

  {

   printf("%d  ",i);

   f1.packet[j]=i;

   i++;

   j++;
```

```
  }
  printf("\nSending frame %d\n",framessend);
  sendto(serversocket,(char*)&f1,sizeof(f1),0,(sockaddr*)&clientaddr,sizeof(clientaddr));
  printf("\nWaiting for the acknowledgement.\n");
  recvfrom(serversocket,(char*)&acknowledgement,sizeof(acknowledgement),0,
(sockaddr*)&clientaddr,&len);
  cls();
  j=0;
  k=0;
  m=0;
  n=l;
  while(m<windowsize && n<totalpackets)
  {
   if(acknowledgement.acknowledge[m]==-1)
   {
    printf("\nNegative acknowledgement received for packet: %d\n",f1.packet[m]);
    k=1;
    repacket[j]=f1.packet[m];
    j++;
   }
   else
   {
    l++;
   }
   m++;
   n++;
  }
```

```
 if(k==0)
 {
printf("\nPositive acknowledgement received for all packets within the frame: %d\n",framessend);
 }

 framessend++;
 printf("\nPRESS ENTER TO PROCEED……\n");
 fgets(req,2,stdin);
 cls();
 }
printf("\nAll frames send successfully.\n\nClosing connection with the client.\n");
//close(serversocket);
}
```

selectiver_c.cpp

```
#include<iostream>
#include<stdio.h>
#include <strings.h>
#include<sys/types.h>
#include<netinet/in.h>
#include<netdb.h>
#define cls() printf("33[H33[J")

                                        //structure definition for accepting the packets.

struct frame

{

 int packet[40];

};

                        //structure definition for constructing the acknowledgement frame

struct ack

{

 int acknowledge[40];

};


int main()

{

 int clientsocket;

 sockaddr_in serveraddr;

 socklen_t len;

 hostent * server;

 frame f1;

 int windowsize,totalpackets,totalframes,i=0,j=0,framesreceived=0,k,l,m,repacket[40];

 ack acknowledgement;
```

```
char req[50];

 clientsocket=socket(AF_INET,SOCK_DGRAM,0);

 bzero((char*)&serveraddr,sizeof(serveraddr));

serveraddr.sin_family=AF_INET;

serveraddr.sin_port=htons(5018);

server=gethostbyname("127.0.0.1");

bcopy((char*)server->h_addr,(char*)&serveraddr.sin_addr.s_addr,sizeof(server->h_addr));

                                        //establishing the connection.

printf("\nSending request to the client.\n");

sendto(clientsocket,"HI I AM CLIENT.",sizeof("HI I AM CLIENT."),0,
(sockaddr*)&serveraddr,sizeof(serveraddr));

 printf("\nWaiting for reply.\n");

recvfrom(clientsocket,req,sizeof(req),0,(sockaddr*)&serveraddr,&len);

printf("\nThe server has send:\t%s\n",req);

                                        //accepting window size from the user.

printf("\nEnter the window size:\t");

scanf("%d",&windowsize);

                                       //sending the window size.

printf("\n\nSending the window size.\n");

sendto(clientsocket,(char*)&windowsize,sizeof(windowsize),0,
(sockaddr*)&serveraddr,sizeof(serveraddr));

cls();

                                        //collecting details from server.

printf("\nWaiting for the server response.\n");

recvfrom(clientsocket,(char*)&totalpackets,sizeof(totalpackets),0,(sockaddr*)&serveraddr,&len);

printf("\nThe total packets are:\t%d\n",totalpackets);
```

```
 sendto(clientsocket,"RECEIVED.",sizeof("RECEIVED."),0,
(sockaddr*)&serveraddr,sizeof(serveraddr));

  recvfrom(clientsocket,(char*)&totalframes,sizeof(totalframes),0,(sockaddr*)&serveraddr,&len);

 printf("\nThe total frames/windows are:\t%d\n",totalframes);

 sendto(clientsocket,"RECEIVED.",sizeof("RECEIVED."),0,
(sockaddr*)&serveraddr,sizeof(serveraddr));

                                        //starting the process.

 printf("\nStarting the process of receiving.\n");

 j=0;

 l=0;

 while(l<totalpackets)

 {                                      //initialising the receive buffer.

   printf("\nInitialising the receive buffer.\n");

   printf("\nThe expected frame is %d with packets:  ",framesreceived);

     for(m=0;m<j;m++)

   {                                //readjusting for packets with negative acknowledgement.

    printf("%d  ",repacket[m]);

   }

   while(j<windowsize && i<totalpackets)

   {

    printf("%d  ",i);

    i++;

    j++;

   }


   printf("\n\nWaiting for the frame.\n");

                                        //accepting the frame.
```

```
recvfrom(clientsocket,(char*)&f1,sizeof(f1),0,(sockaddr*)&serveraddr,&len);

printf("\nReceived frame %d\n\nEnter -1 to send negative acknowledgement for the following packets.\n",framesreceived);

                                        //constructing the acknowledgement frame.

j=0;

m=0;

k=l;

while(m<windowsize && k<totalpackets)

{

 printf("\nPacket: %d\n",f1.packet[m]);

                                        //accepting acknowledgement from the user.

 scanf("%d",&acknowledgement.acknowledge[m]);

  if(acknowledgement.acknowledge[m]==-1)

 {

  repacket[j]=f1.packet[m];

 j++;

 }

 else

 {

  l++;

 }

 m++;

 k++;

}

framesreceived++;

                                        //sending acknowledgement to the server.
```

```
 sendto(clientsocket,(char*)&acknowledgement,sizeof(acknowledgement),0,
(sockaddr*)&serveraddr,sizeof(serveraddr));

  cls();

 }

  printf("\nAll frames received successfully.\n\nClosing connection with the server.\n");

 //close(clientsocket);

}
```