```
//SIMPLE CLIENT SERVER

//client:

#include<stdio.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<string.h>

int main(int argc,char **argv)

{

int len;

int sockfd,n;

struct sockaddr_in servaddr,cliaddr;

char str[1000];

char buff[1024];

sockfd=socket(AF_INET,SOCK_STREAM,0);

if(sockfd<0)

perror("cannot create socket");

bzero(&servaddr,sizeof(servaddr));

servaddr.sin_family=AF_INET;

servaddr.sin_addr.s_addr=inet_addr(argv[1]);

servaddr.sin_port=htons(7228);

connect(sockfd,(struct sockaddr*)&servaddr,sizeof(servaddr));

printf("Enter the message \t");

scanf("%s",buff);

n=write(sockfd,buff,sizeof(buff));
```

```
close(sockfd);
return 0;
}
//server
#include<stdio.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<string.h>
int main(int argc,char **argv)
{
int len;
int sockfd,newfd,n;
struct sockaddr_in servaddr,cliaddr;
char buff[1024];
char str[1000];
sockfd=socket(AF_INET,SOCK_STREAM,0);
if(sockfd<0)
perror("cannot create socket");
bzero(&servaddr,sizeof(servaddr));
servaddr.sin_family=AF_INET;
servaddr.sin_addr.s_addr=INADDR_ANY;
servaddr.sin_port=htons(7228);
if(bind(sockfd,(struct sockaddr*)&servaddr,sizeof(servaddr))<0)
perror("Bind error");
```

```
listen(sockfd,2);

len=sizeof(cliaddr);

newfd=accept(sockfd,(struct sockaddr*)&cliaddr,&len);

// printf("hi");

//Receiving the message

n=read(newfd,buff,sizeof(buff));

printf("\nReceived Message is \t%s",buff);

close(sockfd);

close(newfd);

return 0;

}




//EX3 Echo Server Using TCP

//echoclient

#include<stdio.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<netdb.h>

#define SERV_TCP_PORT 5035

int main(int argc,char*argv[])

{

    int sockfd;

    struct sockaddr_in serv_addr;
    struct hostent *server;

    char buffer[4096];

    sockfd=socket(AF_INET,SOCK_STREAM,0);

    serv_addr.sin_family=AF_INET;

    serv_addr.sin_addr.s_addr=inet_addr("127.0.0.1");

    serv_addr.sin_port=htons(SERV_TCP_PORT);

    printf("\nReady for sending...");

    connect(sockfd,(struct sockaddr*)&serv_addr,sizeof(serv_addr));

    printf("\nEnter the message to send\n");

    printf("\nClient: ");

    fgets(buffer,4096,stdin);

    write(sockfd,buffer,4096);

    printf("Serverecho:%s",buffer);

    printf("\n");

    close(sockfd);

    return 0;

}



//echoserver

#include<stdio.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<netdb.h>
```

```
#define SERV_TCP_PORT 5035

int main(int argc,char*argv[])

{

    int sockfd;

    struct sockaddr_in serv_addr;

    struct hostent *server;

    char buffer[4096];

    sockfd=socket(AF_INET,SOCK_STREAM,0);

    serv_addr.sin_family=AF_INET;

    serv_addr.sin_addr.s_addr=inet_addr("127.0.0.1");

    serv_addr.sin_port=htons(SERV_TCP_PORT);

    printf("\nReady for sending...");

    connect(sockfd,(struct sockaddr*)&serv_addr,sizeof(serv_addr));

    printf("\nEnter the message to send\n");

    printf("\nClient: ");

    fgets(buffer,4096,stdin);

    write(sockfd,buffer,4096);

    printf("Serverecho:%s",buffer);

    printf("\n");

    close(sockfd);

    return 0;

}
```

```
//EX5 Transfer Files

//tcpclient

#include<stdio.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<unistd.h>

#include<arpa/inet.h>

#include<string.h>

int main(int argc,char **argv)

{

int len;

int sockfd,newfd,n,a;

struct sockaddr_in servaddr,cliaddr;

char str[1000];

char buff[1024];

sockfd=socket(AF_INET,SOCK_STREAM,0);

if(sockfd<0)

		perror("cannot create socket");

bzero(&servaddr,sizeof(servaddr));

servaddr.sin_family=AF_INET;

servaddr.sin_addr.s_addr=inet_addr(argv[1]);

servaddr.sin_port=htons(atoi(argv[2]));

connect(sockfd,(struct sockaddr*)&servaddr,sizeof(servaddr));
```

```
{



FILE *f,*f1;


if((f1=fopen("/student/csea061/Desktop/4/outt.txt","r"))==NULL)

        printf("Wrong File");

f=fopen("/student/csea061/Desktop/4/in.txt","r");

fscanf(f,"%s",buff);

write(sockfd,buff,sizeof(buff));

printf("the file was sent successfully");



close(sockfd);

return 0;

}

//tcpserver

#include<stdio.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<unistd.h>

#include<arpa/inet.h>

#include<string.h>

int main(int argc,char **argv)

{
```

```
int len;

int sockfd,newfd,n;

struct sockaddr_in servaddr,cliaddr;

char buff[1024];

char str[1000];

sockfd=socket(AF_INET,SOCK_STREAM,0);

if(sockfd<0)

        perror("cannot create socket");

bzero(&servaddr,sizeof(servaddr));

servaddr.sin_family=AF_INET;

servaddr.sin_addr.s_addr=INADDR_ANY;

servaddr.sin_port=htons(atoi(argv[1]));

if(bind(sockfd,(struct sockaddr*)&servaddr,sizeof(servaddr))<0)

        perror("Bind error");

listen(sockfd,2);

len=sizeof(cliaddr);

newfd=accept(sockfd,(struct sockaddr*)&cliaddr,&len);


FILE *fp;


read(newfd,buff,100);

printf("\nReceived Message is \t%s",buff);



fp=fopen("/student/csea061/Desktop/4/out.txt","w");
```

```
fprintf(fp,"%s",buff);

printf("\nthe file was received successfully");

printf("\nthe new file created");




close(sockfd);

close(newfd);

return 0;

}


//UDPclient

#include<stdio.h>

#include<sys/socket.h>

#include<sys/types.h>

#include<string.h>

#include<netinet/in.h>

#include<sys/types.h>

#include<sys/stat.h>

#include<fcntl.h>

#include<unistd.h>

#include<arpa/inet.h>


int main(int argc,char** argv)

{
```

```
int sockfd,filefd,len;

struct sockaddr_in servaddr,cliaddr;

char buff[5120],dest[1024],filename[64];

sockfd = socket(AF_INET,SOCK_DGRAM,IPPROTO_UDP);

if(sockfd < 0)

        {

         perror("Creation error");

        }

bzero(&servaddr,sizeof(servaddr));

servaddr.sin_family = AF_INET;

servaddr.sin_addr.s_addr = inet_addr(argv[1]);

servaddr.sin_port = htons(7228);

printf("Filename");

scanf("%s",filename);

strcpy(buff,filename);

len = sizeof(servaddr);

sendto(sockfd,buff,5120,0,(struct sockaddr*)&servaddr,len);

printf("Destination to save:");

scanf("%s",dest);

strcat(dest,filename);

if((filefd = creat(dest,S_IRWXU)) != -1)

        {

         sendto(sockfd,buff,5120,0,(struct sockaddr*)&servaddr,len);

         recvfrom(sockfd,buff,5120,0,(struct sockaddr*)&servaddr, &len);

         if(strcmp(buff,"ERROR")!=0)
```

```
                                {
                                 write(filefd,buff,strlen(buff));
                                }
                        else
                                {
                                printf("File not found...\n");
                                close(filefd);
                                }
                        }
                else
                        {
                         printf("%s\n",dest);
                         close(sockfd);
                         return 0;
                        }
}
//UDPserver
#include<stdio.h>
#include<sys/socket.h>
#include<sys/types.h>
#include<string.h>
#include<netinet/in.h>
#include<sys/types.h>
#include<sys/stat.h>
#include<fcntl.h>
```

```
#include<unistd.h>

#include<arpa/inet.h>


int main(int argc, char** argv)

{

        int sockfd,neffd,len,filefd,i;

        struct sockaddr_in servaddr,cliaddr;

        char buff[5120],filename[64],data;

        sockfd = socket(AF_INET,SOCK_DGRAM,IPPROTO_UDP);

        if(sockfd < 0)

                {

                 perror("Creation error");

                }

        bzero(&servaddr,sizeof(servaddr));

        servaddr.sin_family = AF_INET;

        servaddr.sin_addr.s_addr=inet_addr(argv[1]);

        servaddr.sin_port=htons(7228);

        if(bind(sockfd,(struct sockaddr*)&servaddr,sizeof(servaddr))<0);

                {

                 perror("Bind error");

                }

        len = sizeof(cliaddr);

        recvfrom(sockfd,filename,64,0,(struct sockaddr *)&cliaddr,&len);

        printf("Filename:%s\n",filename);

        if((filefd = open(filename,O_RDONLY))!= -1)
```

```
                    {
                     i=0;
                     strcpy(buff,"\0");
                     while(read(filefd,&data,sizeof(data))!= 0)
                              {
                                buff[i++]=data;
                              }
                     buff[i]='\0';
                     close(filefd);
                     printf("File sent...\n");
                    }
          else
                    {
                     strcpy(buff,"Error!!!!");
                    }
          sendto(sockfd,buff,5120,0,(struct sockaddr*)&cliaddr,len);
          close(sockfd);
          return 0;
}


//stopnwaitpro
//swclient
#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


//EVEN PARITY


int main(int argc,char *argv[])

{

char buff[MAX];

int n=0;

char buff1[MAX];

int sockfd,connfd;

char buffer[MAX];

int l;

char k='0';

int j;

int i=0;

int l1;


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(argv[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");


bzero(buff,sizeof(buff));


printf("Enter 16-bit data:");

while((buff[n++]=getchar())!='\n');
```

```
buff[n+1]='\n';

printf("%s",buff);


//-------------------------------------------------------------------------------------------


for(i=0;i<16;i+=4)

{

k='0';

bzero(buff1,sizeof(buff1));

for(j=0;j<4;j++)

{

 buff1[j]=buff[i+j];

 if(buff1[j]=='1' && k=='0')

   k='1';

 else if(buff1[j]=='1' && k=='1')

   k='0';

}

buff1[j]=k;

buff1[j+1]='\n';

printf("Buffer-%s",buff1);

printf("Do you want to introduce an error 1-YES 0-NO:");

scanf("%d",&l);

if(l==1)

{

 printf("Introduce error in which position:");
 scanf("%d",&l);
 if(buff1[l-1]=='0')
  buff1[l-1]='1';
 else
  buff1[l-1]='0';
}
lab1:

if(i==0 || i==8)
 l1=0;
else
 l1=1;

printf("Transmitting Frame %d-%s",l1,buff1);
write(sockfd,buff1,sizeof(buff1));
bzero(buffer,sizeof(buffer));

read(sockfd,buffer,sizeof(buffer));
if(buffer[0]=='N'&&buffer[1]=='A'&&buffer[2]=='C'&&buffer[3]=='K')
{
 if(buff1[l-1]=='0')
  buff1[l-1]='1';
 else
  buff1[l-1]='0';
 goto lab1;
```

```
}

}


close(sockfd);

}


//swserver

#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,z;

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

{

bzero(buff,MAX);

read(sockfd,buff,sizeof(buff));
```

```
printf("From client:%s\n",buff);

n=0;

for(int j=0;j<5;j++)

 if(buff[j]=='1')

  n++;

bzero(buff,MAX);


if(n%2!=0)

{

 z=0;

 strcpy(buff,"NACK-");

 printf("\nError in data\n");


}

else

{

 z=1;

 strcpy(buff,"ACK-");

}

int ack=i%2;

if(z==1 && ack==0)

 ack=1;

else if(z==1 && ack==1)

 ack=0;
```

```
printf("Transmitting %s%d",buff,ack);


write(sockfd,buff,sizeof(buff));

if(buff[0]=='N'&&buff[1]=='A'&&buff[2]=='C'&&buff[3]=='K')

 i--;

}

}

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);

}
```

```
//gobacknarq

//gbnarqclient

#include<stdio.h>

#include<sys/socket.h>

#include<fcntl.h>

#include<netinet/in.h>

#include<strings.h>

#include<sys/types.h>

#include<stdlib.h>

#include<string.h>


int main(int argcc,char * argv[])

{

        struct sockaddr_in addr;

        char a[20]="1",b[10],c[10],ch[2],ackk[3],f[3];

        int x=0,sockfd,s=0,p=0,pos,f_no=0,i,j;

        int LAR=-1,LFS=-1;

        sockfd=socket(AF_INET,SOCK_STREAM,0);

        if(sockfd<0)

        perror("Cannot create socket");

        bzero(&addr,sizeof(addr));

        addr.sin_family=AF_INET;

        addr.sin_addr.s_addr=inet_addr(argv[1]);

        addr.sin_port=htons(8080);

        printf("Enter 16-bit data\n");
```

```
while(strlen(a)!=16)

scanf("%s",a);

if(connect(sockfd,(struct sockaddr*)&addr,sizeof(addr))<0)

perror("Connection failed\n");

while(LAR !=3)

{

        if(LFS+1 <=3)

        printf("\nFrame %d- \n",LFS+1);

        x=(LFS+1)*4;

        s=0;

        strcpy(c,"\0");

        strncpy(b,a+x,4);

        p=0;

        while(s<4)

        {

                if(b[s++]=='1')

                p++;

        }

        if(p%2==0)

                b[s]='0';

        else

                b[s]='1';

                b[s+1]='\0';

printf("%s",b);

if(LFS+1 <=3)
```

```
{
        printf("\nDo you want to introduce error(y/n)\t:");
        scanf("%s",ch);
}
if(ch[0]=='y' && LFS+1 <=3)
{
        printf("Enter position\t:");
        scanf("%d",&pos);
        if(b[pos]=='0')
        b[pos]='1';
        else
        b[pos]='o';
}
for(i=0;i<5;i++)
{
        printf("%c",b[i]);
        if(i==3)
        printf("");
}
f[0]=(char)((LFS)+49);
f[1]='0';
strcat(b,f);
write(sockfd,b,strlen(b));
LFS++;
read(sockfd,c,6);
```

```
        if(!strncmp(c,"ack",3))
        {
        strcpy(ackk,c+3);
        LAR=atoi(ackk);
        printf("\nAck received for frame %d\n",LAR);
        }
        else
        {
        if(LFS==LAR+2)
        {
        LFS=LAR;
        }
        }
        }
        close(sockfd);
}


//gbnarqserver
#include<stdio.h>
#include<sys/socket.h>
#include<fcntl.h>
#include<netinet/in.h>
#include<stdlib.h>
#include<string.h>
```

```
#include<sys/types.h>

#include<strings.h>


int main()

{

char b[0],mess[20],temp[5];

struct sockaddr_in addr;

char ackmsg[7];

int RW=0,fl;

int x=0,cf=0,g,fd,sockfd,s=0,p=0,i,j,ackchar,q,m,ackno=-1,ll,count;


sockfd=socket(AF_INET,SOCK_STREAM,0);

strcpy(ackmsg,"ackxx\0");


if(sockfd<0)

perror("cannot creat socket");


bzero(&addr,sizeof(addr));

addr.sin_family=AF_INET;

addr.sin_addr.s_addr=INADDR_ANY;

addr.sin_port=htons(8080);

s=sizeof(addr);


        if(bind(sockfd,(struct sockaddr*)&addr,sizeof(addr))<0)

        perror("Bind error");
```

```
    }



      listen(sockfd,2);

                    if((fd=accept(sockfd,(struct sockaddr*)&addr,&s))<0)

                    {

                    printf("No connection\n");

                    return;

                    }


do

{

  ll=x=p=0;

  printf("\nReceiving Frame%d\n",RW);

  read(fd,b,7);

  for(i=0;i<6;++i)

   {

    printf("%c",b[i]);

    if(i==3||i==4)

    printf("");

     }


  while(x<5)

  {

  if(b[x++]=='1')

   p++;

    }
```

```
if(p%2!=0)

{

printf("\nError\n");

//write(fd."Nak",4);

count--;

}


else

{

g=RW;

printf("\nNo Error");

if(b[5]==(char)(RW+48))

{

RW++;

strncpy(temp,b,4);

strcat(mess,temp);

printf("Do you want to send ack?(0/1):\t");

scanf("%d",&ackchar);

if(ackchar==1)

{

cf=g;

ackmsg[3]=(char)((g)+48);

write(fd,ackmsg,5);
```

```
   }
   else fl++;
   }
   else fl++;
   }
   if(fl!=0)
   {
   write(fd,"Soc",5);
   fl=0;
   if(RW-cf==2)
   {
   RW=cf+1;
   mess[RW*4]='\0';
   }

}
}
while(RW<4);
printf("\n Message Received\t %s \n",mess);
return;
}

//Address resolution protocol
//arpclient
#include<stdio.h>
```

```
#include<sys/types.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<unistd.h>

#include<string.h>

#include <netdb.h>

int main(int argc,char **argv)

{

   char buff[1024],ip1[1024],ip[1024],mac[1024];

   int len,k=0,i,j=0;

   int sockfd,newfd,n,a,n1,n2;

   struct sockaddr_in servaddr,cliaddr;

   char str[1000];

   char buffer[1024],buf[1024],buff1[50];

   sockfd=socket(AF_INET,SOCK_STREAM,0);

if(sockfd<0)

   perror("cannot create socket");

  bzero(&servaddr,sizeof(servaddr));

   servaddr.sin_family=AF_INET;

   servaddr.sin_addr.s_addr=inet_addr(argv[1]);

   servaddr.sin_port=htons(atoi(argv[2]));

   connect(sockfd,(struct sockaddr*)&servaddr,sizeof(servaddr));

   printf("Enter the Ip Address");

   scanf("%s",ip);

   printf("Enter the MAC Address");
   scanf("%s",mac);

   read(sockfd,buff,sizeof(buff));

   puts(buff);

n=strlen(buff);

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

{

 if(buff[i]=='|')

   j++;

     if(j==2){

       ip1[k]=buff[i+1];

         k++;

}

}

if(strcmp(ip,ip1)==0)

 {

  printf("\nThis is ur client\n");

  strcat(buff,"|");

  strcat(buff,mac);

  printf("\nclient to server%s\n",buff);

  write(sockfd,buff,sizeof(buff));

  read(sockfd,buffer,sizeof(buffer));

  printf("\nReceived packets:%s\n",buffer);


}

}
```

```
//arpc1

#include<stdio.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<unistd.h>

#include<string.h>

#include <netdb.h>

int main(int argc,char **argv)

{

   char buff[1024],ip1[1024],ip[1024],mac[1024];

   int len,k=0,i,j=0;

   int sockfd,newfd,n,a,n1,n2;

   struct sockaddr_in servaddr,cliaddr;

   char str[1000];

   char buffer[1024],buf[1024],buff1[50];

   sockfd=socket(AF_INET,SOCK_STREAM,0);

if(sockfd<0)

   perror("cannot create socket");

  bzero(&servaddr,sizeof(servaddr));

   servaddr.sin_family=AF_INET;

   servaddr.sin_addr.s_addr=inet_addr(argv[1]);

   servaddr.sin_port=htons(atoi(argv[2]));
   connect(sockfd,(struct sockaddr*)&servaddr,sizeof(servaddr));
   printf("Enter the Ip Address");
   scanf("%s",ip);
   printf("Enter the MAC Address");
   scanf("%s",mac);
   read(sockfd,buff,sizeof(buff));
   puts(buff);
n=strlen(buff);
for(i=0;i<n;i++)
{
 if(buff[i]=='|')
   j++;
     if(j==2){
       ip1[k]=buff[i+1];
         k++;
}
}
if(strcmp(ip,ip1)==0)
 {
  printf("\nThis is ur client\n");
  strcat(buff,"|");
  strcat(buff,mac);
  printf("\nclient to server%s\n",buff);
  write(sockfd,buff,sizeof(buff));
  read(sockfd,buffer,sizeof(buffer));
  printf("\nReceived packets:%s\n",buffer);


}

}



//arpc2

#include<stdio.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<unistd.h>

#include<string.h>

#include <netdb.h>

int main(int argc,char **argv)

{

   char buff[1024],ip1[1024],ip[1024],mac[1024];

   int len,k=0,i,j=0;

   int sockfd,newfd,n,a,n1,n2;

   struct sockaddr_in servaddr,cliaddr;

   char str[1000];

   char buffer[1024],buf[1024],buff1[50];

   sockfd=socket(AF_INET,SOCK_STREAM,0);

if(sockfd<0)

   perror("cannot create socket");
```

```
 bzero(&servaddr,sizeof(servaddr));
  servaddr.sin_family=AF_INET;
  servaddr.sin_addr.s_addr=inet_addr(argv[1]);
  servaddr.sin_port=htons(atoi(argv[2]));
  connect(sockfd,(struct sockaddr*)&servaddr,sizeof(servaddr));
  printf("Enter the Ip Address");
  scanf("%s",ip);
  printf("Enter the MAC Address");
  scanf("%s",mac);
  read(sockfd,buff,sizeof(buff));
  puts(buff);
n=strlen(buff);
for(i=0;i<n;i++)
{
 if(buff[i]=='|')
  j++;
   if(j==2){
    ip1[k]=buff[i+1];
     k++;
}
}
if(strcmp(ip,ip1)==0)
 {
 printf("\nThis is ur client\n");
 strcat(buff,"|");
```

```
  strcat(buff,mac);
  printf("\nclient to server%s\n",buff);
  write(sockfd,buff,sizeof(buff));
  read(sockfd,buffer,sizeof(buffer));
  printf("\nReceived packets:%s\n",buffer);

}
}

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

int main(int argc,char *argv[])
{
   char ip[1024],ip1[20],mac[20],d[1024];
   int len;
   int sockfd,newfd,n,a,n1=0,i;
   struct sockaddr_in servaddr,cliaddr;
```

```
    char buff[1024];

   char str[1000];

   char buffer[1024],buf[1024],buff1[50];

   sockfd=socket(AF_INET,SOCK_STREAM,0);

if(sockfd<0)

   perror("cannot create socket");

bzero(&servaddr,sizeof(servaddr));

servaddr.sin_family=AF_INET;

servaddr.sin_addr.s_addr=INADDR_ANY;

servaddr.sin_port=htons(atoi(argv[1]));

if(bind(sockfd,(struct sockaddr*)&servaddr,sizeof(servaddr))<0)

   perror("Bind error");

   listen(sockfd,2);



printf("Enter the Data:");

scanf("%s",d);

printf("Enter the Source IP Address:");

scanf("%s",ip);

printf("%s",ip);

printf("Enter the Destination IP Address");

scanf("%s",ip1);

n=strlen(ip1);

printf("%d",n);

printf("Enter the MAC Address");
```

```
scanf("%s",mac);

strcat(ip,"|");

strcat(ip,mac);

strcat(ip,"|");

strcat(ip,ip1);

puts(ip);


while(1)

{

if(fork()==0)

{

len=sizeof(cliaddr);

newfd=accept(sockfd,(struct sockaddr*)&cliaddr,&len);

write(newfd,ip,sizeof(ip));

}

else

{

read(newfd,buf,sizeof(buf));

                        //printf("\nReceived message is %s\n",buf);

strcat(buf,"|");

strcat(buf,d);

write(newfd,buf,sizeof(buf));

close(newfd);

return;

            }
}


}


//subnetting

//subclient

#include<stdio.h>

#include<stdlib.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include <unistd.h>

#include<arpa/inet.h>

#include<string.h>

int main(int argc,char **argv)

{

char subnet[20];int i=0,j=0,k,b,a;

printf("Enter a subnet addr :");

scanf("%s",subnet);


printf("Trying to connect");


char packet[60],match[40],destaddr[40];

        int len;char flag[10];

        int sockfd,newfd,n;
```

```
	struct sockaddr_in servaddr,cliaddr;
	sockfd=socket(AF_INET,SOCK_STREAM,0);
	if(sockfd<0)
		perror("cannot create socket");
	bzero(&servaddr,sizeof(servaddr));
	servaddr.sin_family=AF_INET;
	servaddr.sin_addr.s_addr=inet_addr("127.0.0.1");
	servaddr.sin_port=htons(atoi(argv[1]));
	connect(sockfd,(struct sockaddr*)&servaddr,sizeof(servaddr));
//	n=write(sockfd,subnetaddr,sizeof(subnetaddr));
	//sleep(3);
	n=read(sockfd,match,sizeof(match));
	printf("\nConnection established: %s\n",match);
printf("\n %s", match);
printf("\n %s",subnet);
if(strcmp(match,subnet)==0)
{
	printf("\npacket can be send");
	strcpy(flag,"yes");
}
//printf("%d",flag);
a=write(sockfd,flag,sizeof(flag));

if(strcmp(flag,"yes")==0)
{
```

```
n=read(sockfd,packet,sizeof(packet));

printf("\n Recieved packet : %s ", packet);

}

else

        return 0;

close(sockfd);



}


//subserver

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#include <string.h>

#include <unistd.h>

#include<arpa/inet.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<netinet/in.h>

int main(int argc,char **argv)

{

char str[50],c[10],add[10],subm[50],def[50],class,sub1[4][3],sub[4][3];

char dest[20],dest1[40][40],data[20],pack[40],Add1[40],Add2[20],Add3[20],Add4[20];

int dec_sub[4],dec_dest[4];
```

```
char str1[20],subnetaddr[20];

int a,m,k,dec=0,rem,bin=0,i=0,j=0,base=1,AND1[10];

printf("Enter the network address:\n");

scanf("%s",str);


printf("Enter the number of subnets:");

scanf("%d",&a);




while(str[i]!='.')

{


        c[i]=str[i];

        i++;

}

if(strcmp(c,"0")>0 && strcmp(c,"128")<0 )

        class = 'A';

else if(strcmp(c,"127")>0 && strcmp(c,"192")<0 )

        class = 'B';

else if(strcmp(c,"191")>0 && strcmp(c,"224")<0 )

        class = 'C';

else if(strcmp(c,"223")>0 && strcmp(c,"240")<0 )

        class = 'D';

else if(strcmp(c,"239")>0 && strcmp(c,"255")<0 )

        class = 'E';
```

```
printf("\nclass : %c\n",class);


while(1)

{

if(pow(2,j)>=a)

{

        k=j;

        break;

}

j++;

}

i=k;

for(m=0;m<8;m++)

{

  if(i>0)

  {

        add[m]='1';

        i--;

  }

  else

        add[m]='0';

}

for(i=0; i<8; i++)

{
        bin = bin * 10 + ( add[i] - '0' );

}

printf("\nones : %d",bin);

while(bin>0)

{

rem=bin%10;

dec=dec+rem*base;

bin=bin/10;

base=base*2;

}

def[0]='\0';

sprintf(def,"%d",dec);

subm[0]='\0';

if(class == 'A')

{



        strcpy(subm,"255.");

        strcat(subm,def);



        strcat(subm,".0.0");

}

else if(class == 'B')

{

        strcpy(subm,"255.255.");

        strcat(subm,def);
```

```
		subm[strlen(subm)]='\0';

		strcat(subm,".0");

	}

	else

	{

		strcpy(subm,"255.255.255.");

		strcat(subm,def);


	}

	printf("\nSubnet mask : %s ",subm);

	k=0;

	for(i=0;subm[k]!='\0';)

	{

		for(j=0;subm[k]!='.'&&subm[k]!='\0';j++)

			{

			sub1[i][j]=subm[k];

			k++;

			}

	sub[i][j]='\0';

	strcpy(sub[i],sub1[i]);

	dec_sub[i]=atoi(sub[i]);

	i++;

	k++;

	}

	sub[i][j]='\0';
```

```
printf("\nEnter the destination ip:\n");

scanf("%s",dest);


printf("\nIts Splitting:\n");


k=0;

for(i=0;dest[k]!='\0';i++)

{

        for(j=0;dest[k]!='.'&&dest[k]!='\0';j++)

                {

                dest1[i][j]=dest[k];

                        k++;


}

if(dest[k]!='\0')

        k++;

dest1[i][j]='\0';

}

printf("\nAgain:\n");

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

{

        for(j=0;dest1[i][j]!='\0';j++)

{

                printf("%c",dest1[i][j]);

}
printf("\n");


}

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

{

dec_dest[i]=atoi(dest1[i]);

AND1[i]=dec_sub[i]&dec_dest[i];

printf("AND: %d ",AND1[i]);

}

sprintf(Add1,"%d",AND1[0]);

sprintf(Add2,"%d",AND1[1]);

sprintf(Add3,"%d",AND1[2]);

sprintf(Add4,"%d",AND1[3]);




strcat(Add1,".");

strcat(Add1,Add2);

strcat(Add1,".");

strcat(Add1,Add3);

strcat(Add1,".");

strcat(Add1,Add4);

printf("\nAddr : %s" ,Add1);

printf("\nData :");

scanf("%s",data);
```

```
strcpy(pack,dest);

strcat(pack,"||");

strcat(pack,data);

printf("packet: %s",pack);




char resp[50],match[4];

        int len;char flag[10];int b;

        int sockfd,newfd,n;

        struct sockaddr_in servaddr,cliaddr;

        sockfd=socket(AF_INET,SOCK_STREAM,0);

        if(sockfd<0)

                        perror("cannot create socket");

        bzero(&servaddr,sizeof(servaddr));

        servaddr.sin_family=AF_INET;

        servaddr.sin_addr.s_addr=INADDR_ANY;

        servaddr.sin_port=htons(atoi(argv[1]));

        if(bind(sockfd,(struct sockaddr*)&servaddr,sizeof(servaddr))<0)

                perror("Bind error");

        listen(sockfd,2);

        len=sizeof(cliaddr);

while(1){

        newfd=accept(sockfd,(struct sockaddr*)&cliaddr,&len);

        printf("\nConnection done ");
```

```
               n=write(newfd,Add1,sizeof(Add1));
     b=read(newfd,flag,sizeof(flag));
if(strcmp(flag,"yes")==0)

{

a=write(newfd,pack,sizeof(pack));
break;
}
}
close(sockfd);
close(newfd);


}




//Domain name server
//DNS client
#include<stdio.h>
#include<sys/stat.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<arpa/inet.h>
```

```
#include<netinet/in.h>

#include<string.h>

int main(int argc,char **argv)

{

int len;

int sockfd,n,i;

struct sockaddr_in servaddr,cliaddr;

char str[1000];

char buff[1024];

char rv[1024];

sockfd=socket(AF_INET,SOCK_DGRAM,0);

if(sockfd<0)

perror("cannot create socket");

servaddr.sin_family=AF_INET;

servaddr.sin_addr.s_addr=inet_addr(argv[1]);

servaddr.sin_port=htons(atoi(argv[2]));

printf("Enter the server name :");

scanf("%s",buff);

len=sizeof(servaddr);

sendto(sockfd,buff,sizeof(buff),0,( struct sockaddr*)&servaddr,sizeof(servaddr));

recvfrom(sockfd,str,sizeof(str),0,(struct sockaddr*)&servaddr,&len);

printf("The IP address is :%s",str);

}


//DNS server
```

```
#include<stdio.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<string.h>

#include <unistd.h>

#include<stdlib.h>

#include <arpa/inet.h>

#include <netdb.h>

int main(int argc,char **argv)

{

int len,flag=0;

int sockfd,newfd,n,y,y1,y2,i;

struct sockaddr_in servaddr,cliaddr;

char rv[100],rv1[100];

char buff[100][100],buff1[100];

char str[100][100],str1[100];

sockfd=socket(AF_INET,SOCK_DGRAM,0);

if(sockfd<0)

perror("cannot create socket");

bzero(&servaddr,sizeof(servaddr));

servaddr.sin_family=AF_INET;

servaddr.sin_addr.s_addr=INADDR_ANY;

servaddr.sin_port=htons(atoi(argv[1]));

if(bind(sockfd,(struct sockaddr*)&servaddr,sizeof(servaddr))<0)
```

```
perror("Bind error");

printf("Enter the Number\n");

scanf("%d",&n);

for(i=0;i<n;i++){

 printf("Enter the Server name:");

 scanf("%s",str[i]);

 printf("Enter the IP Address");

 scanf("%s",buff[i]);

}

printf("Lookup Table Details\n");

printf("S_name\t IP\n");

for(i=0;i<n;i++){

 printf("%s\t",str[i]);

 printf("%s\t",buff[i]);

 printf("\n");


}

label:

{

printf("1.Update\t 2.Modify\n");

printf("Enter ur Choice\n");

scanf("%d",&y);

switch(y){

  case 1:

     printf("Enter the server name ");
     scanf("%s",str[n]);
     printf("Enter the IP address");
     scanf("%s",buff[n]);
     n++;
     printf("Updated Lookup Table Details\n");
     printf("S_name\t IP\n");
     for(i=0;i<n;i++){
     printf("%s\t",str[i]);
     printf("%s\t",buff[i]);
      printf("\n");
}
      break;
  case 2:
    flag=0;
    printf("Enter the domain name");
    scanf("%s",rv);
      for(i=0;i<n;i++){
           if(strcmp(str[i],rv)==0)
              y1=i;
}
    printf("Enter the valid IP Adress");
    scanf("%s",rv1);
    for(i=0;i<n;i++){
       if(strcmp(rv1,buff[i])==0){
          printf("Given ip adress was already exit\n");
```

```
         flag=1;
        //break;
       }
}
      if(flag!=1){
         strcpy(buff[y1],rv1);

     printf("Modified Lookup Table Details\n");
     printf("S_name\t IP\n");
     for(i=0;i<n;i++){
     printf("%s\t",str[i]);
     printf("%s\t",buff[i]);
     printf("\n");
}
   break;
}
}
}printf("Do you want to continue this process 1/0");
scanf("%d",&y2);
if(y2==1){
goto label;
}
       len=sizeof(cliaddr);
      recvfrom(sockfd,buff1,sizeof(buff1),0,(struct sockaddr*)&cliaddr,&len);
       for(i=0;i<n;i++){
        if(strcmp(str[i],buff1)==0){

          strcpy(str1,buff[i]);

            sendto(sockfd,str1,sizeof(str1),0,(struct sockaddr*)&cliaddr,sizeof(cliaddr));


}

}

}
```