//Assignment udp in C

//A.Client

#include<sys/types.h>

#include<sys/socket.h>

#include<stdio.h>

#include<netinet/in.h>

#include <unistd.h>

#include<string.h>

#include<strings.h>

#include <arpa/inet.h>

#include <stdlib.h>

void main()

{

int b,sockfd,sin\_size,con,n,len;

char buff[256];

char buff1[256];

char ip[50];

char port[10];

struct sockaddr\_in servaddr;

// Socket call for socket creation: socket on sucess returns smallest integer value (3) otherwise -1

if((sockfd=socket(AF\_INET,SOCK\_STREAM,0))>=0)

printf("Socket Created Sucessfully \n");

//Initialize object parameters of server: family,ip,port

servaddr.sin\_family=AF\_INET;

servaddr.sin\_addr.s\_addr=inet\_addr("10.15.9.218");

servaddr.sin\_port=6000;

//initiate a connection on a socket from client to server using connect call

// connect call on sucess return 0 otnerwise -1

sin\_size = sizeof(servaddr);

if((con=connect(sockfd,(struct sockaddr \*) &servaddr, sin\_size))==0)

printf("connect sucessful \n");

bzero(buff,256); //null 256 bytes of buff

bzero(buff1,256); //null 256 bytes of buff1

//fgets call reads msg from user

printf("Enter Your Message:");

fgets(buff, 256,stdin);

len=strlen(buff);

//write call writes message on socket

write(sockfd,buff,len);

//printf("\n msg to server:%s",buff);

//read call reads message from socket

read(sockfd, buff1, 256);

printf("msg from server:%s \n",buff1);

//close call closes socket created and free assigned parameters

close(sockfd);

}

//A. Server

#include<sys/types.h>

#include<sys/socket.h>

#include<stdio.h>

#include<netinet/in.h>

#include <unistd.h>

#include<string.h>

#include <arpa/inet.h>

#include<netdb.h>

void main()

{

int b,sockfd,connfd,sin\_size,l,n,len;

char buff[256];

char buff1[256];

if((sockfd=socket(AF\_INET,SOCK\_STREAM,0))>=0) //socket creation

printf("socket created sucessfully \n"); //on success 0 otherwise -1

struct sockaddr\_in servaddr;

struct sockaddr\_in clientaddr;

//Assign server object parameters

servaddr.sin\_family=AF\_INET;

servaddr.sin\_addr.s\_addr=htonl(INADDR\_ANY);

servaddr.sin\_port=htons(6000);

//bind() assigns the

// address specified by addr to the socket referred to by the file

// descriptor sockfd. addrlen specifies the size, in bytes, of the

// address structure pointed to by addr. Traditionally, this operation is

// called â€œassigning a name to a socketâ€.

if((b=bind(sockfd, (struct sockaddr \*)&servaddr,sizeof(servaddr)))==0)

printf("bind Sucessful \n");

//listen for connections on a socket

l=listen(sockfd,5);

printf("listen Sucessful \n");

//accept call accepts connection from client

sin\_size = sizeof(clientaddr);

if((connfd=accept(sockfd,(struct sockaddr \*)&clientaddr,&sin\_size))>0);

printf("accept sucessful \n");

bzero(buff,256);

//bzero(buff1,256);

//len=strlen(buff);

//reads msg from socket

n=read(connfd, buff, 256);

printf("msg from client:%s\n",buff);

//writes msg on socket

printf("Enter Your Message:");

fgets(buff1, 256,stdin);

len=strlen(buff1);

write(connfd, buff1, len);

//printf("msg to client:%s \n",buff1);

//close call closes socket created and free assigned parameters

close(sockfd);

}

//B.Client

#include<sys/types.h>

#include<sys/socket.h>

#include<stdio.h>

#include<netinet/in.h>

#include <unistd.h>

#include<string.h>

#include<strings.h>

#include <arpa/inet.h>

#include <stdlib.h>

#define MAXBUFLEN 1000000

//#define buffsize 150

void main()

{

int b,sockfd,sin\_size,con,n,len;

//char buff[256];

if((sockfd=socket(AF\_INET,SOCK\_STREAM,0))>=0)

printf("Socket Created Sucessfully \n");

struct sockaddr\_in servaddr;

servaddr.sin\_family=AF\_INET;

servaddr.sin\_addr.s\_addr=inet\_addr("127.0.0.1");

servaddr.sin\_port=6007;

sin\_size = sizeof(struct sockaddr\_in);

if((connect(sockfd,(struct sockaddr \*) &servaddr, sin\_size))==0) //initiate a connection on a socket

printf("Connect Sucessful \n");

char buffer[10000];

char c[10000];

FILE \*fp;

//bzero(buffer,10000);

//bzero(c,10000);

read(sockfd, buffer, 10000);

fp= fopen("/home/ubuntu/Downloads/CNL/Assg2/Assg2\_b/receive.txt", "w+");

/\* Read and display data \*/

fwrite(buffer, 1,strlen(buffer) + 1, fp);

//fseek(fp, 0, SEEK\_SET);

//fread(c, strlen(buffer)+1, 1, fp);

printf("Received File Contents :%s \n", buffer);

fclose(fp);

close(sockfd);

}

//B.Server

#include<sys/types.h>

#include<sys/socket.h>

#include<stdio.h>

#include<netinet/in.h>

#include <unistd.h>

#include<string.h>

#include <arpa/inet.h>

#include <stdlib.h>

#define MAXBUFLEN 1000000

void main()

{

int b,sockfd,connfd,sin\_size,l,n,len;

if((sockfd=socket(AF\_INET,SOCK\_STREAM,0))>=0) //socket creation

printf("Socket Created Sucessfully \n"); //on success 0 otherwise -1

struct sockaddr\_in servaddr;

struct sockaddr\_in clientaddr;

servaddr.sin\_family=AF\_INET;

servaddr.sin\_addr.s\_addr=inet\_addr("127.0.0.1");

servaddr.sin\_port=6007;

if((bind(sockfd, (struct sockaddr \*)&servaddr,sizeof(servaddr)))==0) //bind() assigns the

// address specified by addr to the socket referred to by the file

// descriptor sockfd. addrlen specifies the size, in bytes, of the

// address structure pointed to by addr. Traditionally, this operation is

// called â€œassigning a name to a socketâ€.

printf("Bind Sucessful \n");

if((listen(sockfd,5))==0) //listen for connections on a socket

printf("Listen Sucessful \n");

sin\_size = sizeof(struct sockaddr\_in);

if((connfd=accept(sockfd,(struct sockaddr \*)&clientaddr,&sin\_size))>0)

printf("Accept Sucessful \n");

char buffer[100];

char c[10000] = "this is file transfer program";

//char source[MAXBUFLEN + 1];

//bzero(buffer,10000);

FILE \*fp;

fp= fopen("/home/ubuntu/Downloads/CNL/Assg2/Assg2\_b/send.txt", "w+");

//fp = fopen("file.txt", "w+");

/\* Write data to the file \*/

fwrite(c, 1, strlen(c) + 1, fp);

/\* Seek to the beginning of the file \*/

fseek(fp, 0, SEEK\_SET);

/\* Read and display data \*/

fread(buffer, 1,strlen(c)+1, fp);

// fclose(fp);

write(connfd, buffer, strlen(buffer));

printf("Sent File Contents: %s\n", buffer);

fclose(fp);

close(sockfd);

}

//C.Client

#include<sys/types.h>

#include<sys/socket.h>

#include<stdio.h>

#include<netinet/in.h>

#include <unistd.h>

#include<string.h>

#include<strings.h>

#include <arpa/inet.h>

//#define buffsize 150

void main()

{

int b,sockfd,sin\_size,con,n,len;

//char buff[256];

char operator;

int op1,op2,result;

if((sockfd=socket(AF\_INET,SOCK\_STREAM,0))>0)

printf("socket created sucessfully\n");

//printf("%d\n", sockfd);

struct sockaddr\_in servaddr;

servaddr.sin\_family=AF\_INET;

servaddr.sin\_addr.s\_addr=inet\_addr("127.0.0.1");

servaddr.sin\_port=6006;

sin\_size = sizeof(struct sockaddr\_in);

if((con=connect(sockfd,(struct sockaddr \*) &servaddr, sin\_size))==0); //initiate a connection on a socket

printf("connect sucessful\n");

printf("Enter operation:\n +:Addition \n -: Subtraction \n /: Division \n\*:Multiplication \n");

scanf("%c",&operator);

printf("Enter operands:\n");

scanf("%d %d", &op1, &op2);

write(sockfd,&operator,10);

write(sockfd,&op1,sizeof(op1));

write(sockfd,&op2,sizeof(op2));

read(sockfd,&result,sizeof(result));

printf("Operation result from server=%d\n",result);

close(sockfd);

}

//C.Server  
#include<sys/types.h>

#include<sys/socket.h>

#include<stdio.h>

#include<netinet/in.h>

#include <unistd.h>

#include<string.h>

#include <arpa/inet.h>

void main()

{

int b,sockfd,connfd,sin\_size,l,n,len;

char operator;

int op1,op2,result;

if((sockfd=socket(AF\_INET,SOCK\_STREAM,0))>0)

printf("socket created sucessfully\n"); //socket creation

//printf("%d\n", sockfd); //on success 0 otherwise -1

struct sockaddr\_in servaddr;

struct sockaddr\_in clientaddr;

servaddr.sin\_family=AF\_INET;

servaddr.sin\_addr.s\_addr=inet\_addr("127.0.0.1");

servaddr.sin\_port=6006;

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

printf("bind sucessful\n"); //bind() assigns the

// address specified by addr to the socket referred to by the file

// descriptor sockfd. addrlen specifies the size, in bytes, of the

// address structure pointed to by addr. Traditionally, this operation is

// called â€œassigning a name to a socketâ€.

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

if((listen(sockfd,5))==0) //listen for connections on a socket

printf("listen sucessful\n");

//printf("%d\n",l);

sin\_size = sizeof(struct sockaddr\_in);

if((connfd=accept(sockfd,(struct sockaddr \*)&clientaddr,&sin\_size))>0);

printf("accept sucessful\n");

//printf("%d\n",connfd);

read(connfd, &operator,10);

read(connfd,&op1,sizeof(op1));

read(connfd,&op2,sizeof(op2));

switch(operator) {

case '+': result=op1 + op2;

printf("Result is: %d + %d = %d\n",op1, op2, result);

break;

case '-':result=op1 - op2;

printf("Result is: %d - %d = %d\n",op1, op2, result);

break;

case '\*':result=op1 \* op2;

printf("Result is: %d \* %d = %d\n",op1, op2, result);

break;

case '/':result=op1 / op2;

printf("Result is: %d / %d = %d\n",op1, op2, result);

break;

default:

printf("ERROR: Unsupported Operation");

}

write(connfd,&result,sizeof(result));

close(sockfd);

}

//D.Client

#include<sys/types.h>

#include<sys/socket.h>

#include<stdio.h>

#include<netinet/in.h>

#include <unistd.h>

#include<string.h>

#include<strings.h>

#include <arpa/inet.h>

#include<math.h>

//#define buffsize 150

void main()

{

int b,sockfd,sin\_size,con,n,len;

//char buff[256];

double angle,result;

char op;

if((sockfd=socket(AF\_INET,SOCK\_STREAM,0))>0)

printf("socket created sucessfully\n");

//printf("%d\n", sockfd);

struct sockaddr\_in servaddr;

servaddr.sin\_family=AF\_INET;

servaddr.sin\_addr.s\_addr=inet\_addr("127.0.0.1");

servaddr.sin\_port=6666;

sin\_size = sizeof(servaddr);

if((con=connect(sockfd,(struct sockaddr \*) &servaddr, sin\_size))==0); //initiate a connection on a socket

printf("connect sucessful\n");

printf("Enter operation:\n 1:sin \n 2:cos\n 3:tan \n ");

scanf("%c",&op);

printf("Enter angle in degree:");

scanf("%lf",&angle);

write(sockfd,&op,1);

write(sockfd,&angle,sizeof(angle));

read(sockfd,&result,sizeof(result));

printf("\n Operation result from server=%lf\n",result);

close(sockfd);

}

//D.Server

#include<sys/types.h>

#include<sys/socket.h>

#include<stdio.h>

#include<netinet/in.h>

#include <unistd.h>

#include<string.h>

#include <arpa/inet.h>

#include<math.h>

#define PI 3.14159265

void main()

{

int b,sockfd,connfd,sin\_size,l,n,len;

char op;

double angle1;

double result,val;

if((sockfd=socket(AF\_INET,SOCK\_STREAM,0))>0)

printf("socket created sucessfully\n"); //socket creation

//printf("%d\n", sockfd); //on success 0 otherwise -1

struct sockaddr\_in servaddr;

struct sockaddr\_in clientaddr;

servaddr.sin\_family=AF\_INET;

servaddr.sin\_addr.s\_addr=inet\_addr("127.0.0.1");

servaddr.sin\_port=6666;

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

printf("bind sucessful\n"); //bind() assigns the

// address specified by addr to the socket referred to by the file

// descriptor sockfd. addrlen specifies the size, in bytes, of the

// address structure pointed to by addr. Traditionally, this operation is

// called â€œassigning a name to a socketâ€.

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

if((listen(sockfd,5))==0) //listen for connections on a socket

printf("listen sucessful\n");

//printf("%d\n",l);

sin\_size = sizeof(clientaddr);

if((connfd=accept(sockfd,(struct sockaddr \*)&clientaddr,&sin\_size))>0);

printf("accept sucessful\n");

val = PI / 180;

read(connfd, &op,1);

//printf("\n op=%d",op);

read(connfd, &angle1, sizeof(angle1));

//printf("\n angle is=%lf \n",angle1);

switch(op) {

case '1':

result=sin(angle1\*val);

printf("sin(%lf)=%lf ",angle1,result);

break;

case '2':result=cos(angle1\*val);

printf("cos(%lf) =%lf ",angle1,result);

break;

case '3':result=tan(angle1\*val);

printf("tan(%lf) = %lf",angle1,result);

break;

default:

printf("ERROR: Unsupported Operation");

}

write(connfd,&result,sizeof(result));

close(connfd);

close(sockfd);

}