**Q2**

**Server**

#include <stdio.h>

#include <netdb.h>

#include <netinet/in.h>

#include <stdlib.h>

#include <string.h>

#include <sys/socket.h>

#include <sys/types.h>

#define MAX 80

#define PORT 8080

#define SA struct sockaddr

// Function designed for chat between client and server.

void func(int sockfd)

{

char buff[MAX];

int n;

// infinite loop for chat

for (;;) {

bzero(buff, MAX);

// read the message from client and copy it in buffer

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

// print buffer which contains the client contents

printf("From client: %s\t To client : ", buff);

bzero(buff, MAX);

n = 0;

// copy server message in the buffer

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

;

// and send that buffer to client

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

// if msg contains "Exit" then server exit and chat ended.

if (strncmp("exit", buff, 4) == 0) {

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

break;

}

}

}

// Driver function

int main()

{

int sockfd, connfd, len;

struct sockaddr\_in servaddr, cli;

// socket create and verification

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));

// assign IP, PORT

servaddr.sin\_family = AF\_INET;

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

servaddr.sin\_port = htons(PORT);

// Binding newly created socket to given IP and verification

if ((bind(sockfd, (SA\*)&servaddr, sizeof(servaddr))) != 0) {

printf("socket bind failed...\n");

exit(0);

}

else

printf("Socket successfully binded..\n");

// Now server is ready to listen and verification

if ((listen(sockfd, 5)) != 0) {

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

exit(0);

}

else

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

len = sizeof(cli);

// Accept the data packet from client and verification

connfd = accept(sockfd, (SA\*)&cli, &len);

if (connfd < 0) {

printf("server accept failed...\n");

exit(0);

}

else

printf("server accept the client...\n");

// Function for chatting between client and server

func(connfd);

// After chatting close the socket

close(sockfd);

}

**Client**

#include <netdb.h>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <sys/socket.h>

#define MAX 80

#define PORT 8080

#define SA struct sockaddr

void func(int sockfd)

{

char buff[MAX];

int n;

for (;;) {

bzero(buff, sizeof(buff));

printf("Enter the string : ");

n = 0;

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

;

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

bzero(buff, sizeof(buff));

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

printf("From Server : %s", buff);

if ((strncmp(buff, "exit", 4)) == 0) {

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

break;

}

}

}

int main()

{

int sockfd, connfd;

struct sockaddr\_in servaddr, cli;

// socket create and varification

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));

// assign IP, PORT

servaddr.sin\_family = AF\_INET;

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

servaddr.sin\_port = htons(PORT);

// connect the client socket to server socket

if (connect(sockfd, (SA\*)&servaddr, sizeof(servaddr)) != 0) {

printf("connection with the server failed...\n");

exit(0);

}

else

printf("connected to the server..\n");

// function for chat

func(sockfd);

// close the socket

close(sockfd);

}

**Q3**

**Server**

#include<stdio.h>

#include<netinet/in.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<netdb.h>

#include<string.h>

#include<stdlib.h>

#define MAX 80

#define PORT 43454

#define SA struct sockaddr

void func(int sockfd)

{

char buff[MAX];

int n,clen;

struct sockaddr\_in cli;

clen=sizeof(cli);

for(;;)

{

bzero(buff,MAX);

recvfrom(sockfd,buff,sizeof(buff),0,(SA \*)&cli,&clen);

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

bzero(buff,MAX);

n=0;

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

sendto(sockfd,buff,sizeof(buff),0,(SA \*)&cli,clen);

if(strncmp("exit",buff,4)==0)

{

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

break;

}

}

}

int main()

{

int sockfd;

struct sockaddr\_in servaddr;

sockfd=socket(AF\_INET,SOCK\_DGRAM,0);

if(sockfd==-1)

{

printf("socket creation failed...\n");

exit(0);

}

else

printf("Socket successfully created..\n");

bzero(&servaddr,sizeof(servaddr));

servaddr.sin\_family=AF\_INET;

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

servaddr.sin\_port=htons(PORT);

if((bind(sockfd,(SA \*)&servaddr,sizeof(servaddr)))!=0)

{

printf("socket bind failed...\n");

exit(0);

}

else

printf("Socket successfully binded..\n");

func(sockfd);

close(sockfd);

}

**Client**

#include<sys/socket.h>

#include<netdb.h>

#include<string.h>

#include<stdlib.h>

#include<stdio.h>

#define MAX 80

#define PORT 43454

#define SA struct sockaddr

int main()

{

char buff[MAX];

int sockfd,len,n;

struct sockaddr\_in servaddr;

sockfd=socket(AF\_INET,SOCK\_DGRAM,0);

if(sockfd==-1)

{

printf("socket creation failed...\n");

exit(0);

}

else

printf("Socket successfully created..\n");

bzero(&servaddr,sizeof(len));

servaddr.sin\_family=AF\_INET;

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

servaddr.sin\_port=htons(PORT);

len=sizeof(servaddr);

for(;;)

{

printf("\nEnter string : ");

n=0;

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

sendto(sockfd,buff,sizeof(buff),0,(SA \*)&servaddr,len);

bzero(buff,sizeof(buff));

recvfrom(sockfd,buff,sizeof(buff),0,(SA \*)&servaddr,&len);

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

if(strncmp("exit",buff,4)==0)

{

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

break;

}

}

close(sockfd);

}

**Q4**

**Server**

#include <stdio.h>

#include <netdb.h>

#include <netinet/in.h>

#include <stdlib.h>

#include <string.h>

#include <sys/socket.h>

#include <sys/types.h>

#define MAX 80

#define PORT 8080

#define SA struct sockaddr

void func(int sockfd)

{

char buff[MAX];

int n;

for (;;) {

bzero(buff, MAX);

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

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

n=strlen(buff);

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

char rev[n];

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

rev[i]=buff[n-i-1];

}

write(sockfd, rev, n);

if (strncmp("exit", buff, 4) == 0) {

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

break;

}

}

}

int main()

{

int sockfd, connfd, len;

struct sockaddr\_in servaddr, cli;

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

if (sockfd == -1) {

printf("socket creation failed...\n");

exit(0);

}

else

printf("Socket successfully created..\n");

bzero(&servaddr, sizeof(servaddr));

servaddr.sin\_family = AF\_INET;

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

servaddr.sin\_port = htons(PORT);

if ((bind(sockfd, (SA\*)&servaddr, sizeof(servaddr))) != 0) {

printf("socket bind failed...\n");

exit(0);

}

else

printf("Socket successfully binded..\n");

if ((listen(sockfd, 5)) != 0) {

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

exit(0);

}

else

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

len = sizeof(cli);

// Accept the data packet from client and verification

connfd = accept(sockfd, (SA\*)&cli, &len);

if (connfd < 0) {

printf("server acccept failed...\n");

exit(0);

}

else

printf("server acccept the client...\n");

func(connfd);

close(sockfd);

}

**Client**

#include <netdb.h>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <sys/socket.h>

#define MAX 80

#define PORT 8080

#define SA struct sockaddr

void func(int sockfd)

{

char buff[MAX];

int n;

for (;;) {

bzero(buff, sizeof(buff));

printf("\nEnter the string : ");

n = 0;

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

;

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

bzero(buff, sizeof(buff));

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

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

if ((strncmp(buff, "exit", 4)) == 0) {

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

break;

}

}

}

int main()

{

int sockfd, connfd;

struct sockaddr\_in servaddr, cli;

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

if (sockfd == -1) {

printf("socket creation failed...\n");

exit(0);

}

else

printf("Socket successfully created..\n");

bzero(&servaddr, sizeof(servaddr));

servaddr.sin\_family = AF\_INET;

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

servaddr.sin\_port = htons(PORT);

if (connect(sockfd, (SA\*)&servaddr, sizeof(servaddr)) != 0) {

printf("connection with the server failed...\n");

exit(0);

}

else

printf("connected to the server..\n");

func(sockfd);

fclose(sockfd);

}

**Q5**

**Server**

#include<netinet/in.h>

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#include<sys/socket.h>

#include<unistd.h>

#define PORT 8090

int main()

{

int m\_sockfd,c\_sockfd,val,val1;

struct sockaddr\_in address;

char str[100];

char str1[100];

char buff2[100];

int addrlen=sizeof(address);

char buffer[1024]={0};

char\* hello="welcome from server!!";

if((m\_sockfd=socket(AF\_INET,SOCK\_STREAM,0)) == 0)

{

perror("socket failed");

exit(EXIT\_FAILURE);

}

address.sin\_family=AF\_INET;

address.sin\_addr.s\_addr=INADDR\_ANY;

address.sin\_port=htons(PORT);

if(bind(m\_sockfd,(struct sockaddr\*)&address,sizeof(address)) < 0)

{

perror("bind failed");

exit(EXIT\_FAILURE);

}

if(listen(m\_sockfd,3)< 0)

{

perror("listen failed");

exit(EXIT\_FAILURE);

}

if((c\_sockfd=accept(m\_sockfd,(struct sockaddr\*)&address,(socklen\_t\*)&addrlen)) < 0)

{

perror("accept failed");

exit(EXIT\_FAILURE);

}

val=read(c\_sockfd,str,sizeof(str));

val1=read(c\_sockfd,str1,sizeof(str1));

int i,j,temp;

int l= strlen(str);

int l1=strlen(str1);

printf("\nFirst string sent by client:%s\n",str);

printf("\nSecond string sent by client:%s\n",str1);

//string comparison

char \*res;

if(strcmp(str,str1)==0)

res="yes";

else

res="no";

send(c\_sockfd,res,sizeof(res),0);

printf("string comparison result sent to client\n");

//string sorting

int len1=sizeof(buff2);

int val2=read(c\_sockfd,buff2,sizeof(buff2));

for (i=0;i<len1-1;i++)

{

for(j=i+1;j<len1;j++)

{

if(buff2[i]>buff2[j])

{

char temp=buff2[i];

buff2[i]=buff2[j];

buff2[j]=temp;

}

}

}

send(c\_sockfd,buff2,sizeof(buff2),0);

printf("string sorting result sent to client\n");

//string copy

char copied[100];

char buff3[100];

int val4=read(c\_sockfd,buff3,sizeof(buff3));

strcpy(buff3,copied);

send(c\_sockfd,copied,sizeof(copied),0);

printf("\n copied string sent to client\n");

return 0;

}

**Client**

#include<netinet/in.h>

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#include<sys/socket.h>

#include<unistd.h>

#define PORT 8090

int main()

{

int m\_sockfd=0,val;

struct sockaddr\_in address;

char str[100];

char str1[100];

char buff2[100];

char buff1[100];

char res[10];

char copied[100];

printf("enter string 1: ");

scanf("%s",str);

printf("enter string 2: ");

scanf("%s",str1);

char buffer[1024]={0};

if((m\_sockfd=socket(AF\_INET,SOCK\_STREAM,0)) < 0)

{

perror("socket failed");

exit(EXIT\_FAILURE);

}

memset(&address,'0',sizeof(address));

address.sin\_family=AF\_INET;

address.sin\_port=htons(PORT);

if(inet\_pton(AF\_INET,"127.0.0.1",&address.sin\_addr) < 0)

{

printf("\naddress not supported\n");

return -1;

}

if(connect(m\_sockfd,(struct sockaddr\*)&address,sizeof(address)) < 0)

{

printf("\n connection failed\n");

return -1;

}

int l= strlen(str);

int l1=strlen(str1);

send(m\_sockfd,str,sizeof(str),0);

send(m\_sockfd,str1,sizeof(str1),0);

val=read(m\_sockfd,res,10);

printf("string comparison result is:%s\n",res);

//string sorting

printf("enter string for sorting: ");

scanf("%s",buff2);

send(m\_sockfd,buff2,sizeof(buff2),0);

int val1=read(m\_sockfd,buff1,sizeof(buff2));

printf("first string after sorting is : %s\n",buff1);

//string copy

char buff3[100];

printf("enter string for copying: ");

scanf("%s",buff3);

send(m\_sockfd,buff3,sizeof(buff3),0);

int val3=read(m\_sockfd,copied,10);

printf(" string after copying is : %s\n",copied);

return 0;

}