LAB1

Q1

SERVER

#include<stdio.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<string.h>

#include<arpa/inet.h>

#include<unistd.h>

#include<stdlib.h>

#include<sys/socket.h>

int main()

{

int listenfd,clientfd,retval,rbyte,sbyte;

socklen\_t clilen;

int buff[3];

struct sockaddr\_in servaddr,cliaddr;

//socket creation

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

if(listenfd==-1){printf("\nSocket Error");exit(0);}

printf("\nSocket Success");

bzero(&servaddr,sizeof(servaddr));

servaddr.sin\_family=AF\_INET;

servaddr.sin\_port=htons(7891);

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

//Binding

retval=bind(listenfd,(struct sockaddr\*) &servaddr,sizeof(servaddr));

if(retval==-1){printf("\nBind Error");exit(0);}

printf("\nBind Success");

//listen

retval=listen(listenfd,2);

if( retval == -1 ){printf("\nListening Error");exit(0);}

printf("\nListen Success");

//Accepting connection

clilen=sizeof(cliaddr);

clientfd=accept(listenfd,(struct sockaddr\*) &cliaddr,&clilen);

if(clientfd==-1){ printf("\nAccept Error");close(listenfd);exit(0);}

printf("\nAccept Success\n");

while(1){

bzero(buff,sizeof(buff));

rbyte=recv(clientfd,buff,sizeof(buff),0);

if(rbyte==-1){printf("\nrecv error");close(clientfd);close(listenfd);exit(0);}

int n=buff[0],ele,choice=buff[1];

int arr[n+1];

if(choice==4){break;}

rbyte=recv(clientfd,arr,sizeof(arr),0);

if(rbyte==-1){printf("\nrecv error");close(clientfd);close(listenfd);exit(0);}

int ans[1024],pos=0,i,k,odd[100]={0},even[100]={0},o=0,e=0;

switch(choice)

{

case 1: pos=0,ele=buff[2];

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

{if(arr[i]==ele){pos=i+1;break;}}

if(i==n){pos=-1;}

ans[0]=pos;

break;

case 2:if(buff[2]==1)

{for(int i=0;i<n-1;i++)

{for(int j=0;j<n-1-i;j++)

{if(arr[j]>arr[j+1])

{int temp=arr[j];arr[j]=arr[j+1];arr[j+1]=temp;}}}}

else if(buff[2]==2)

{for(int i=0;i<n-1;i++)

{for(int j=0;j<n-1-i;j++)

{if(arr[j]<arr[j+1])

{int temp=arr[j];arr[j]=arr[j+1];arr[j+1]=temp;}}}}

for(int i=0;i<n;i++){ans[i]=arr[i];}

break;

case 3:

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

{if(arr[i]%2==0){even[e++]=arr[i];}

else {odd[o++]=arr[i];}}

ans[0]=o;ans[1]=e;

for(i=2,k=0;k<e;i++,k++){ans[i]=even[k];}k=0;

while(k<o){ans[i++]=odd[k++];}

break;

}

sbyte=send(clientfd,ans,sizeof(ans),0);

if(sbyte==-1){printf("\nSend error");close(clientfd);close(listenfd);exit(0);}

}

close(clientfd);

close(listenfd);

return 0;

}

CLIENT

#include<stdio.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<string.h>

#include<arpa/inet.h>

#include<unistd.h>

#include<sys/types.h>

#include<stdlib.h>

int main()

{

int clientfd,retval,rbyte,sbyte;

struct sockaddr\_in servaddr;

int buff[3]={0};

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

if(clientfd==-1){printf("\nSocket error");exit(0);}

printf("\nSocket Success");

bzero(&servaddr,sizeof(servaddr));

servaddr.sin\_family=AF\_INET;

servaddr.sin\_port=htons(7891);

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

retval=connect(clientfd,(struct sockaddr\*) &servaddr,sizeof(servaddr));

if(retval==-1){printf("\nConnect error");exit(0);}

printf("\nConnect Success");

bzero(buff,sizeof(buff));

int n;

printf("\nEnter number of integers=");

scanf("%d",&n);

int arr[n];

printf("Enter values=");

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

{

scanf("%d",&arr[i]);

}

while(1)

{

printf("\n1)search\n2)sort in ascending/descending)\n3)split into odd and even\n4)exit");

int choice,ele;

char ch;int c=1,chi;

buff[0]=n;

printf("\nEnter choice=");

scanf("%d",&choice);

switch(choice)

{

case 1:printf("\nElement to search=");

scanf("%d",&ele);

buff[1]=1;

buff[2]=ele;

break;

case 2:fflush(stdin);

printf("\nAscending(1)/Descending(2)=");

scanf("%d",&chi);

buff[1]=2;

//if(ch=='A'){buff[2]=1;}

//else if(ch=='D') {buff[2]=2;}

buff[2]=chi;

break;

case 3:buff[1]=3;buff[2]=0;break;

case 4:buff[1]=4;buff[2]=0;break;

default:printf("\nwrong choice");c=0;break;

}

if(c==1){

sbyte=send(clientfd,buff,sizeof(buff),0);

if(sbyte==-1){printf("\nsend Error");close(clientfd);exit(0);}

if(buff[1]==4){break;}

sbyte=send(clientfd,arr,sizeof(arr),0);

if(sbyte==-1){printf("\nsend Error");close(clientfd);exit(0);}

int ans[1024];

// bzero(buff,sizeof(buff));

rbyte=recv(clientfd,ans,sizeof(ans),0);

if(rbyte==-1){printf("\nRecv Error");close(clientfd);exit(0);}

int e,i,o;

// printf("\nData received=%s\n",buff);

switch(choice)

{

case 1:if(ans[0]==-1){printf("\nElement not found");}

else {printf("Element found at position=%d",ans[0]);}break;

case 2:printf("\nsorted elements=");

for(int i=0;i<n;i++){printf("%d ",ans[i]);}break;

case 3:e=ans[1];o=ans[0];

printf("even=");

for(i=2;i<2+e;i++){printf("%d ",ans[i]);}

printf("\nodd=");

while(i<2+e+o){printf("%d ",ans[i++]);}break;

}

}

}

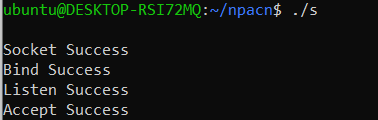
close(clientfd);

return 0;

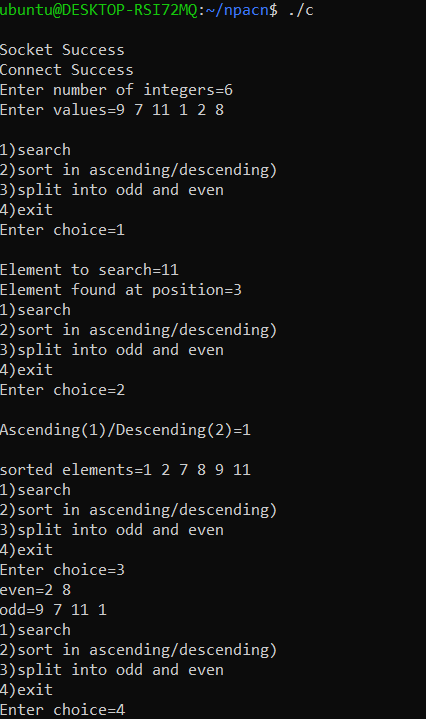
}

OUTPUT

SERVER



CLIENT



LAB-1

Q2(UDP)

SERVER

#include<stdio.h>

#include<stdlib.h>

#include<unistd.h>

#include<sys/socket.h>

#include<sys/types.h>

#include<string.h>

#include<arpa/inet.h>

#include<netinet/in.h>

int main(){

int sockfd,ret,clilen;

struct sockaddr\_in servaddr,cliaddr;

char buff[100],ans[3][100];

//socket creation

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

if(sockfd==-1){printf("Socket error");}

bzero(&servaddr,sizeof(servaddr));

servaddr.sin\_port=htons(9000);

servaddr.sin\_family=AF\_INET;

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

//binding

ret=bind(sockfd,(struct sockaddr\*)&servaddr,sizeof(servaddr));

if(ret==-1){printf("Bind error");}

clilen=sizeof(cliaddr);

while(1){

bzero(buff,sizeof(buff));

recvfrom(sockfd,buff,sizeof(buff),MSG\_WAITALL,(struct sockaddr\*)&cliaddr,&clilen);

printf("\nString received=%s\n",buff);

if(strncmp(buff,"Halt",4)==0){break;}

bzero(ans,sizeof(ans));

int i,len=0,vow=0;

char ch;

while(buff[len]!='\0'){len++;}

for(i=0;i<len/2;i++){

if(buff[i]!=buff[len-1-i]){strcpy(ans[0],"Not palin");break;}

}

if(i==len/2){strcpy(ans[0],"Is palin");}

ch=len+'0';

ans[1][0]=ch;

int a=0,e=0,o=0,u=0,t=0;

i=0;

for(t=0;t<len;t++)

{if(buff[t]=='a'){a++;}

else if(buff[t]=='e'){e++;}

else if(buff[t]=='i'){i++;}

else if(buff[t]=='o'){o++;}

else if(buff[t]=='u'){u++;}}

ch=a+'0';ans[2][0]=ch;

ch=e+'0';ans[2][1]=ch;

ch=i+'0';ans[2][2]=ch;

ch=o+'0';ans[2][3]=ch;

ch=u+'0';ans[2][4]=ch;

//ans[2][0]=ch;

ret=sendto(sockfd,ans,sizeof(ans),0,(struct sockaddr\*)&cliaddr,sizeof(cliaddr));

if(ret==-1){printf("send error");close(sockfd);return 0;}

}

close(sockfd);

return 0;

}

CLIENT

#include<stdio.h>

#include<stdlib.h>

#include<sys/types.h>

#include<unistd.h>

#include<arpa/inet.h>

#include<netinet/in.h>

#include<string.h>

#include<sys/socket.h>

int main(){

int clientfd,ret,i=0;

struct sockaddr\_in servaddr;

char ans[3][100];

//socket creation

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

if(clientfd==-1){printf("Socket Error");}

char buff[100];

bzero(&servaddr,sizeof(servaddr));

servaddr.sin\_port=htons(9000);

servaddr.sin\_family=AF\_INET;

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

while(1)

{

bzero(buff,sizeof(buff));

printf("\nEnter string to check=");

scanf("%[^\n]%\*c",buff);

ret=sendto(clientfd,buff,strlen(buff),0,(struct sockaddr\*)&servaddr,sizeof(servaddr));

if(ret==-1){printf("Send error");close(clientfd);return 0;}

if(strncmp(buff,"Halt",4)==0){break;}

bzero(ans,sizeof(ans));

ret=recvfrom(clientfd,ans,sizeof(ans),0,NULL,NULL);

if(ret==-1){printf("Recevive error");close(clientfd);return 0;}

printf("Answer=%s",ans[0]);

printf("\nNumber of characters=%c",ans[1][0]);

for(i=0;i<5;i++)

{if(ans[2][i]!='0')

{

if(i==0){printf("\na=%c",ans[2][0]);}

if(i==1){printf("\ne=%c",ans[2][1]);}

if(i==2){printf("\ni=%c",ans[2][2]);}

if(i==3){printf("\no=%c",ans[2][3]);}

if(i==4){printf("\nu=%c",ans[2][4]);}

}

}

}

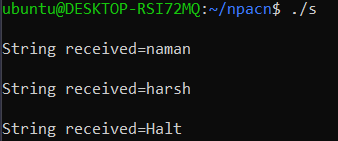
close(clientfd);

return 0;

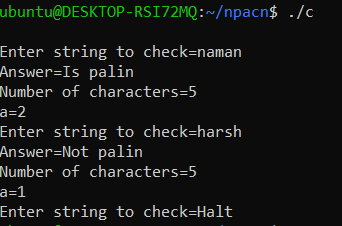
}

OUTPUT

SERVER



CLIENT



Q1(TCP)

SERVER

#include<stdio.h>

#include<stdlib.h>

#include<unistd.h>

#include<sys/socket.h>

#include<sys/types.h>

#include<string.h>

#include<arpa/inet.h>

#include<netinet/in.h>

int main(){

int sockfd,ret,clilen,clientfd,rbyte, sbyte;

struct sockaddr\_in servaddr,cliaddr;

char buff[100],ans[3][100];

//socket creation

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

if(sockfd==-1){printf("Socket error");}

bzero(&servaddr,sizeof(servaddr));

servaddr.sin\_port=htons(9000);

servaddr.sin\_family=AF\_INET;

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

//binding

ret=bind(sockfd,(struct sockaddr\*)&servaddr,sizeof(servaddr));

if(ret==-1){printf("Bind error");}

printf("\nBind done");

//listen

ret=listen(sockfd,2);

if(ret==-1){printf("Listen error");}

printf("\nListen done");

//accepting connection

clilen=sizeof(cliaddr);

clientfd=accept(sockfd,(struct sockaddr\*)&cliaddr,&clilen);

if(clientfd==-1){printf("Accept error");}

printf("\naccept done");

while(1){

bzero(buff,sizeof(buff));

rbyte=recv(clientfd,buff,sizeof(buff),0);

if(rbyte==-1){printf("receive error");close(sockfd);close(clientfd);return 0;}

printf("\nString received=%s\n",buff);

if(strncmp(buff,"Halt",4)==0){break;}

bzero(ans,sizeof(ans));

int i,len,vow;

char ch;

len=0; vow=0;

while(buff[len]!='\0'){len++;}

for(i=0;i<len/2;i++){

if(buff[i]!=buff[len-1-i]){strcpy(ans[0],"Not palin");break;}

}

if(i==len/2){strcpy(ans[0],"Is palin");}

ch=len+'0';

ans[1][0]=ch;

int a=0,e=0,o=0,u=0,t=0;

i=0;

for(t=0;t<len;t++)

{if(buff[t]=='a'){a++;}

else if(buff[t]=='e'){e++;}

else if(buff[t]=='i'){i++;}

else if(buff[t]=='o'){o++;}

else if(buff[t]=='u'){u++;}}

ch=a+'0';ans[2][0]=ch;

ch=e+'0';ans[2][1]=ch;

ch=i+'0';ans[2][2]=ch;

ch=o+'0';ans[2][3]=ch;

ch=u+'0';ans[2][4]=ch;

sbyte=send(clientfd,ans,sizeof(ans),0);

if(sbyte==-1){printf("send error");close(sockfd);close(clientfd);return 0;}

}

close(sockfd);

return 0;

}

CLIENT

#include<stdio.h>

#include<stdlib.h>

#include<sys/types.h>

#include<unistd.h>

#include<arpa/inet.h>

#include<netinet/in.h>

#include<string.h>

#include<sys/socket.h>

int main(){

int clientfd,ret,i=0;

struct sockaddr\_in servaddr;

char ans[3][100];

//socket creation

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

if(clientfd==-1){printf("Socket Error");}

char buff[100];

bzero(&servaddr,sizeof(servaddr));

servaddr.sin\_port=htons(9000);

servaddr.sin\_family=AF\_INET;

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

ret=connect(clientfd,(struct sockaddr\*)&servaddr,sizeof(servaddr));

if(ret==-1){printf("\nconnect error");}

while(1)

{

bzero(buff,sizeof(buff));

printf("\nEnter string to check=");

scanf("%s",buff);

ret=send(clientfd,buff,strlen(buff),0);

if(ret==-1){printf("Send error");close(clientfd);return 0;}

if(strncmp(buff,"Halt",4)==0){break;}

bzero(ans,sizeof(ans));

ret=recv(clientfd,ans,sizeof(ans),0);

if(ret==-1){printf("Recevive error");close(clientfd);return 0;}

printf("Answer=%s",ans[0]);

printf("\nNumber of characters=%c",ans[1][0]);

for(i=0;i<5;i++)

{if(ans[2][i]!='0')

{

if(i==0){printf("\na=%c",ans[2][0]);}

if(i==1){printf("\ne=%c",ans[2][1]);}

if(i==2){printf("\ni=%c",ans[2][2]);}

if(i==3){printf("\no=%c",ans[2][3]);}

if(i==4){printf("\nu=%c",ans[2][4]);}

}

}

}

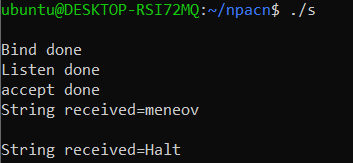
close(clientfd);

return 0;

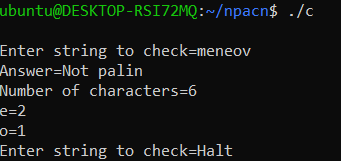
}

OUPUT

SERVER



CLIENT



LAB-2

Q1

SERVER

#include <stdio.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<string.h>

#include<arpa/inet.h>

#include<unistd.h>

#include<stdlib.h>

#include<sys/socket.h>

int main()

{

int listenfd,clientfd,retval,rbyte,sbyte;

socklen\_t clilen;

char buff[2][102];

int choice;

struct sockaddr\_in servaddr,cliaddr;

//socket creation

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

if(listenfd==-1){printf("\nSocket Error");exit(0);}

printf("\nSocket Success");

bzero(&servaddr,sizeof(servaddr));

servaddr.sin\_family=AF\_INET;

servaddr.sin\_port=htons(7891);

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

//Binding

retval=bind(listenfd,(struct sockaddr\*) &servaddr,sizeof(servaddr));

if(retval==-1){printf("\nBind Error");exit(0);}

printf("\nBind Success");

//listen

retval=listen(listenfd,2);

if( retval == -1 ){printf("\nListening Error");exit(0);}

printf("\nListen Success");

//accepting connection

clilen=sizeof(cliaddr);

clientfd=accept(listenfd,(struct sockaddr\*) &cliaddr,&clilen);

if(clientfd==-1){ printf("\nAccept Error");close(listenfd);exit(0);}

printf("\nAccept Success");

//receiving file name

char fname[100];

recv(clientfd,fname,sizeof(fname),0);

FILE \*f1 =fopen(fname,"r");

if (f1==NULL){send(clientfd,"File not found",14,0);close(clientfd); close(listenfd); return 0;}

else{send(clientfd,"File found",11,0);}

while(1){

bzero(buff,sizeof(buff));

rbyte=recv(clientfd,&choice,sizeof(choice),0);

if(rbyte==-1){printf("\nrecv error");close(clientfd);close(listenfd);exit(0);}

if(choice==4){return 0;}

rbyte=recv(clientfd,buff,sizeof(buff),0);

if(rbyte==-1){printf("\nrecv error");close(clientfd);close(listenfd);exit(0);}

int i,found=0,j,st[256]={0};

char word[100];

char ans[100],ch;

//menu option wise code

switch(choice)

{

case 1: fseek(f1,0,SEEK\_SET);

while(fscanf(f1,"%s",word)!=EOF)

{

if(strcmp(word,buff[0])==0) {found++;}

}

if(found>0){ch=found+'0'; ans[0]=ch;}

else{bcopy(ans,"String not found",17);}

break;

case 2: fseek(f1,0,SEEK\_SET);

FILE \*f2=fopen("new.txt","w");

//printf("case 2=%s",buff[1]);

while(fscanf(f1,"%s",word)!=EOF)

{

if(strcmp(word,buff[0])==0){found=1;fputs(buff[1],f2);fputs(" ",f2);}

else{fputs(word,f2);fputs(" ",f2);}

}

fclose(f2);

if(found==1){

strcpy(ans,"String replaced");

char temp[]="cp new.txt ";

strcat(temp,fname);

int result=system(temp);

}

else{strcpy(ans,"String not found");}

break;

case 3: fseek(f1,0,SEEK\_SET);

FILE \*f3=fopen("new\_arrange.txt","w");

while((ch=fgetc(f1))!=EOF)

{

int x=ch;

st[x]++;

}

for(i=0;i<256;i++){

for(j=0;j<st[i];j++){

ch=i;fputc(ch,f3);}

}

fclose(f3);

strcpy(ans,"Reordering done");

break;

case 4:break;

}

if(choice==4){break;}

sbyte=send(clientfd,ans,sizeof(ans),0);

if(sbyte==-1){printf("\nSend error");close(clientfd);close(listenfd);exit(0);}

}

fclose(f1);

close(clientfd);

close(listenfd);

return 0;

}

CLIENT

#include<stdio.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<string.h>

#include<arpa/inet.h>

#include<unistd.h>

#include<sys/types.h>

#include<stdlib.h>

int main()

{

int clientfd,retval,rbyte,sbyte;

struct sockaddr\_in servaddr;

char buff[2][102];

//socket creation

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

if(clientfd==-1){printf("\nSocket error");exit(0);}

printf("\nSocket Success");

bzero(&servaddr,sizeof(servaddr));

servaddr.sin\_family=AF\_INET;

servaddr.sin\_port=htons(7891);

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

//connection establishment

retval=connect(clientfd,(struct sockaddr\*) &servaddr,sizeof(servaddr));

if(retval==-1){printf("\nConnect error");exit(0);}

printf("\nConnect Success");

char fname[100];

bzero(buff,sizeof(buff));

char msg[100];

//taking and sending file name

bzero(fname,sizeof(fname));

printf("\nEnter file name=");

scanf("%s",fname);

send(clientfd,fname,sizeof(fname),0);

recv(clientfd,msg,sizeof(msg),0);

printf("%s",msg);

if(strncmp(msg,"File not found",14)==0){close(clientfd); return 0;}

int c=1;

char ans[1024];

//menu

while(1)

{

c=1;

bzero(buff,sizeof(buff));

bzero(ans,sizeof(ans));

printf("\n1)search\n2)replace\n3)reorder\n4)exit");

int choice;

printf("\nEnter choice=");

scanf("%d",&choice);

switch(choice)

{

case 1:printf("\nString to search=");

scanf("%s",buff[0]);

break;

case 2:printf("\nEnter string to replace=");

scanf("%s",buff[0]);

printf("Enter string to replace with=");

scanf("%s",buff[1]);

break;

case 3:printf("\nrearraging..\n"); break;

case 4:break;

default:printf("\nwrong choice");c=0;break;

}

if(c==1){

sbyte=send(clientfd,&choice,sizeof(choice),0);

if(sbyte==-1){printf("\nsend Error");close(clientfd);exit(0);}

if(choice==4){break;}

sbyte=send(clientfd,buff,sizeof(buff),0);

if(sbyte==-1){printf("\nsend Error");close(clientfd);exit(0);}

rbyte=recv(clientfd,ans,sizeof(ans),0);

if(rbyte==-1){printf("\nRecv Error");close(clientfd);exit(0);}

printf("Answer=%s",ans);

}

}

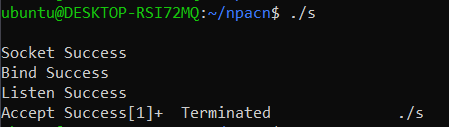
close(clientfd);

return 0;

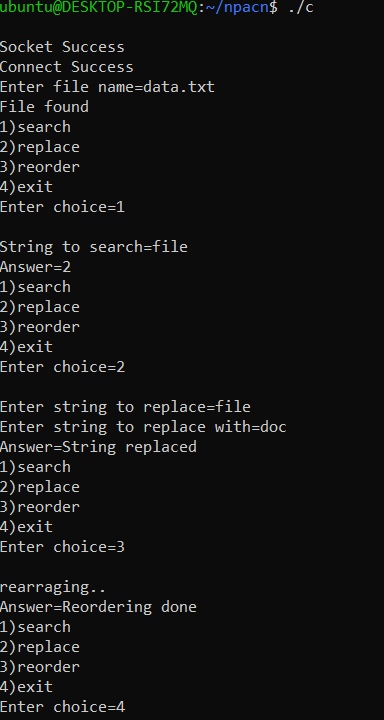
}

OUTPUT

SERVER



CLIENT



LAB-3

Q1

SERVER

#include <stdio.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<string.h>

#include<arpa/inet.h>

#include<unistd.h>

#include<stdlib.h>

#include<sys/socket.h>

#include<sys/wait.h>

int main()

{

int listenfd,clientfd,retval,rbyte,sbyte,pid;

socklen\_t clilen;

char buff[1024];

struct sockaddr\_in servaddr,cliaddr;

//socket creation

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

if(listenfd==-1){printf("\nSocket Error");exit(0);}

printf("\nSocket Success");

bzero(&servaddr,sizeof(servaddr));

servaddr.sin\_family=AF\_INET;

servaddr.sin\_port=htons(7891);

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

//binding

retval=bind(listenfd,(struct sockaddr\*) &servaddr,sizeof(servaddr));

if(retval==-1){printf("\nBind Error");exit(0);}

printf("\nBind Success");

//listen

retval=listen(listenfd,2);

if( retval == -1 ){printf("\nListening Error");exit(0);}

printf("\nLsiten Success");

//accepting connection

clilen=sizeof(cliaddr);

clientfd=accept(listenfd,(struct sockaddr\*) &cliaddr,&clilen);

if(clientfd==-1){ printf("\nAccept Error");close(listenfd);exit(0);}

printf("\nAccept Success");

//full duplex communication

while(1){

bzero(buff,sizeof(buff));

pid=fork();

//child

if(pid==0){

close(listenfd);

printf("\nI am the child process with pid=%d and ppid=%d",getpid(),getppid());

bzero(buff,sizeof(buff));

rbyte=recv(clientfd,buff,sizeof(buff),0);

if(rbyte==-1){printf("\nrecv error");close(clientfd);close(listenfd);exit(0);}

printf("\n Server side received=%s\n",buff);

if(strncmp(buff,"Exit",4)==0){break;}

exit(0);

}

if(pid!=0){

//parent process

printf("\nI am the parent process with pid=%d and ppid=%d",getpid(),getppid());

bzero(buff,sizeof(buff));

printf("\nEnter string to send=");

gets(buff);

sbyte=send(clientfd,buff,strlen(buff)+1,0);

if(sbyte==-1){printf("\nSend error");close(clientfd);close(listenfd);exit(0);}

if(strncmp(buff,"Exit",4)==0){break;}

}

}

close(clientfd);

close(listenfd);

return 0;

}

CLIENT

#include<stdio.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<string.h>

#include<arpa/inet.h>

#include<unistd.h>

#include<sys/types.h>

#include<stdlib.h>

#include<sys/wait.h>

int main()

{

int clientfd,retval,rbyte,sbyte,pid;

struct sockaddr\_in servaddr;

char buff[1024];

//socket creation

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

if(clientfd==-1){printf("\nSocket error");exit(0);}

printf("\nSocket Success");

bzero(&servaddr,sizeof(servaddr));

servaddr.sin\_family=AF\_INET;

servaddr.sin\_port=htons(7891);

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

//connection

retval=connect(clientfd,(struct sockaddr\*) &servaddr,sizeof(servaddr));

if(retval==-1){printf("\nConnect error");exit(0);}

printf("\nConnect Success");

//full duplex communcation

while(1){

bzero(buff,sizeof(buff));

pid=fork();

//child process

if(pid==0){

printf("\nI am the child process with pid=%d and ppid=%d",getpid(),getppid());

printf("\nEnter string to send=");

gets(buff);

sbyte=send(clientfd,buff,strlen(buff)+1,0);

if(sbyte==-1){printf("\nsend Error");close(clientfd);exit(0);}

if(strncmp(buff,"Exit",4)==0){close(clientfd);break;}

exit(0);

}

//parent process

if(pid!=0){

printf("\nI am the parent process with pid=%d and ppid=%d",getpid(),getppid());

bzero(buff,sizeof(buff));

rbyte=recv(clientfd,buff,sizeof(buff),0);

if(rbyte==-1){printf("\nRecv Error");close(clientfd);exit(0);}

printf("\nData received=%s\n",buff);

if(strncmp(buff,"Exit",4)==0){close(clientfd);break;}

}

}

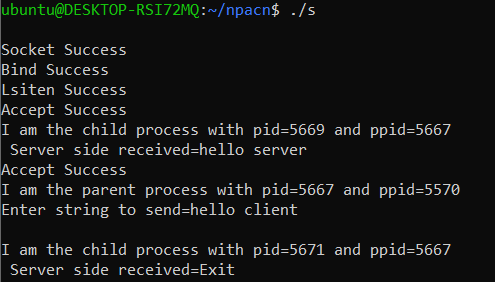
close(clientfd);

return 0;

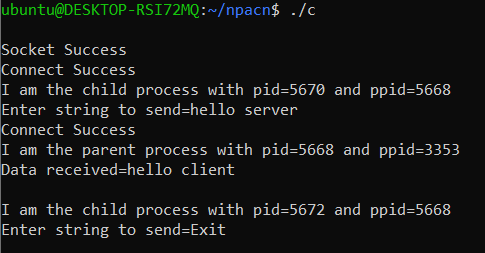
}

OUTPUT

SERVER



CLIENT



Q2

SERVER

#include<stdio.h>

#include<stdlib.h>

#include<unistd.h>

#include<sys/socket.h>

#include<sys/types.h>

#include<string.h>

#include<arpa/inet.h>

#include<netinet/in.h>

void swap(char \*x, char \*y)

{

char temp;

temp = \*x;

\*x = \*y;

\*y = temp;

}

void permutation(char \*a, int l, int r)

{

int i;

if (l == r)

printf("%s\n", a);

else

{

for (i = l; i <= r; i++)

{

swap((a+l), (a+i));

permutation(a, l+1, r);

swap((a+l), (a+i));

}

}

}

int main(){

int len=0,sockfd,ret,clilen;

struct sockaddr\_in servaddr,cliaddr;

char buff[100];

//socket creation

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

if(sockfd==-1){printf("Socket error");}

bzero(&servaddr,sizeof(servaddr));

servaddr.sin\_port=htons(9000);

servaddr.sin\_family=AF\_INET;

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

//binding

ret=bind(sockfd,(struct sockaddr\*)&servaddr,sizeof(servaddr));

if(ret==-1){printf("Bind error");}

clilen=sizeof(cliaddr);

while(1){

bzero(buff,sizeof(buff));

recvfrom(sockfd,buff,sizeof(buff),MSG\_WAITALL,(struct sockaddr\*)&cliaddr,&clilen);

printf("\nString received=%s\n",buff);

if(strncmp(buff,"Halt",4)==0){break;}

printf("\nPermutations are=");

len=0;

while(buff[len]!='\0'){len++;}

permutation(buff,0,len-1);

}

close(sockfd);

return 0;

}

CLIENT

#include<stdio.h>

#include<stdlib.h>

#include<sys/types.h>

#include<unistd.h>

#include<arpa/inet.h>

#include<netinet/in.h>

#include<string.h>

#include<sys/socket.h>

int main(){

int clientfd,ret,i=0;

struct sockaddr\_in servaddr;

//socket creation

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

if(clientfd==-1){printf("Socket Error");}

char buff[100];

bzero(&servaddr,sizeof(servaddr));

servaddr.sin\_port=htons(9000);

servaddr.sin\_family=AF\_INET;

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

while(1)

{

bzero(buff,sizeof(buff));

printf("\nEnter string=");

scanf("%s",buff);

ret=sendto(clientfd,buff,strlen(buff),0,(struct sockaddr\*)&servaddr,sizeof(servaddr));

if(ret==-1){printf("Send error");close(clientfd);return 0;}

if(strncmp(buff,"Halt",4)==0){break;}

}

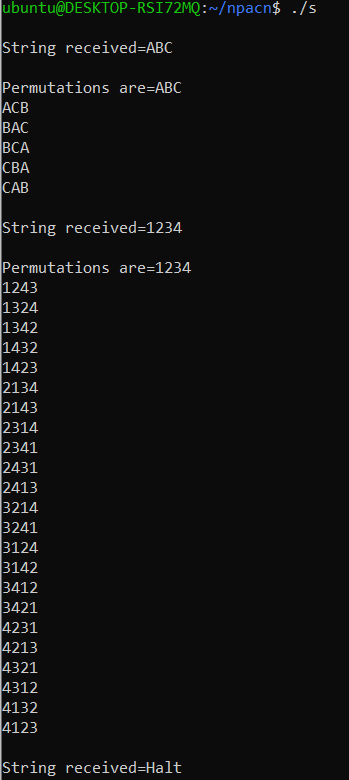
close(clientfd);

return 0;

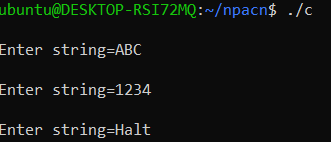
}

OUTPUT

SERVER



CLIENT



Q3

Server

#include <stdio.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<string.h>

#include<arpa/inet.h>

#include<unistd.h>

#include<stdlib.h>

#include<sys/socket.h>

#include<sys/wait.h>

int main()

{

int listenfd,clientfd,retval,rbyte,sbyte,pid;

socklen\_t clilen;

char buff[1024];

struct sockaddr\_in servaddr,cliaddr;

//socket creation

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

if(listenfd==-1){printf("\nSocket Error");exit(0);}

printf("\nSocket Success");

bzero(&servaddr,sizeof(servaddr));

servaddr.sin\_family=AF\_INET;

servaddr.sin\_port=htons(7891);

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

//binding

retval=bind(listenfd,(struct sockaddr\*) &servaddr,sizeof(servaddr));

if(retval==-1){printf("\nBind Error");exit(0);}

printf("\nBind Success");

//listen

retval=listen(listenfd,2);

if( retval == -1 ){printf("\nListening Error");exit(0);}

printf("\nListen Success");

//accepting connection

clilen=sizeof(cliaddr);

clientfd=accept(listenfd,(struct sockaddr\*) &cliaddr,&clilen);

if(clientfd==-1){ printf("\nAccept Error");close(listenfd);exit(0);}

printf("\nAccept Success");

char num[1024]="",alph[1024]="",ch,temp;

int n=0,a=0;

bzero(buff,sizeof(buff));

rbyte=recv(clientfd,buff,sizeof(buff),0);

if(rbyte==-1){printf("\nrecv error");close(clientfd);close(listenfd);exit(0);}

for(int i=0;buff[i]!='\0';i++){

ch=buff[i];

if(ch>='0' && ch<='9'){num[n++]=ch;}

else if(ch>='a' && ch<='z'){alph[a++]=ch;}

}

pid=fork();

//child process sorting in ascending order

if(pid==0){

close(listenfd);

for(int i=0;i<n-1;i++){

for(int j=0;j<n-1-i;j++){

if(num[j]>num[j+1]){temp=num[j];num[j]=num[j+1];num[j+1]=temp;}

}

}

sbyte=send(clientfd,num,sizeof(num),0);

exit(0);

}

//parent process sorts characters in descending order

if(pid!=0){

wait(NULL);

for(int i=0;i<a-1;i++){

for(int j=0;j<a-1-i;j++){

if(alph[j]<alph[j+1]){temp=alph[j];alph[j]=alph[j+1];alph[j+1]=temp;}

}

}

sbyte=send(clientfd,alph,sizeof(alph),0);

if(sbyte==-1){printf("\nSend error");close(clientfd);close(listenfd);exit(0);}

}

close(clientfd);

close(listenfd);

return 0;

}

Client

#include<stdio.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<string.h>

#include<arpa/inet.h>

#include<unistd.h>

#include<sys/types.h>

#include<stdlib.h>

int main()

{

int clientfd,retval,rbyte,sbyte;

struct sockaddr\_in servaddr;

char buff[1024],num[1024],alph[1024];

//socket creation

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

if(clientfd==-1){printf("\nSocket error");exit(0);}

printf("\nSocket Success");

bzero(&servaddr,sizeof(servaddr));

servaddr.sin\_family=AF\_INET;

servaddr.sin\_port=htons(7891);

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

//sending connection request

retval=connect(clientfd,(struct sockaddr\*) &servaddr,sizeof(servaddr));

if(retval==-1){printf("\nConnect error");exit(0);}

printf("\nConnect Success");

//sending string

bzero(buff,sizeof(buff));

printf("\n Enter string to send=");

gets(buff);

sbyte=send(clientfd,buff,strlen(buff),0);

if(sbyte==-1){printf("\nsend Error");close(clientfd);exit(0);}

//receiving sorted number string

bzero(buff,sizeof(buff));

rbyte=recv(clientfd,num,sizeof(num),0);

if(rbyte==-1){printf("\nRecv Error");close(clientfd);exit(0);}

printf("\nNum received=%s\n",num);

//receiving sorted alpha string

rbyte=recv(clientfd,alph,sizeof(alph),0);

printf("Alph received=%s\n",alph);

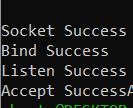
close(clientfd);

return 0;

}

OUTPUT

Server:



Client:

