NDP LAB

Reg No: 180953016

# Lab 1:

# Question 1:

Server Side:

#include <string.h>

#include<strings.h>

#include <unistd.h>

#include<sys/socket.h>

#include<sys/types.h>

#include<netinet/in.h>

#include<stdlib.h>

#include<stdio.h>

#define port 1234

void insertionSort(int arr[], int n)

{

int i, key, j;

for (i = 1; i < n; i++)

{

key = arr[i];

j = i - 1;

/\* Move elements of arr[0..i-1], that are

greater than key, to one position ahead

of their current position \*/

while (j >= 0 && arr[j] > key)

{

arr[j + 1] = arr[j];

j = j - 1;

}

arr[j + 1] = key;

}

}

int main()

{

int s,r;

socklen\_t len;

struct sockaddr\_in server,client;

int buff[50];

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

if(s!=-1)

printf("Socket Created \n");

server.sin\_family = AF\_INET;

server.sin\_port = htons(port);

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

r = bind(s,(struct sockaddr\*)&server,sizeof server);

if (r!=-1)

printf("Socket Binded \n");

r = listen(s,5);

if (r!=-1)

printf("Socket Listening \n");

len = sizeof client;

int ns = accept(s,(struct sockaddr\*)&client,&len);

if(ns!=-1)

printf("Client Accepted\nConnection Establised\n");

else if (ns==-1)

{

printf("Error");

exit(0);

}

/\*recv(ns,buff,sizeof(buff),0);

printf("Message From Client: %s",buff);

printf("Enter Message: ");

//fflush(stdln);

scanf("%s",buff);

send(ns,buff,sizeof(buff),0);\*/

while(1)

{

int choice = 0,i,num,x=-1;

int odd[50],even[50],n1=0,n2=0;

int n,j;

read(ns,&choice,sizeof(choice));

printf("Choice Selected %d\n",choice);

if (choice==0)

{

break;

}

read(ns,&n,sizeof(n));

recv(ns,buff,sizeof(buff),0);

printf("Recieved Array: ");

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

{

printf("%d ",buff[i]);

}

printf("\n");

switch(choice)

{

case 1:

//int j;

/\*for(i=0;i<n-1;i++)

{

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

{

if(buff[j]>buff[j+1])

{

int temp = buff[i];

buff[i] = buff[j];

buff[j] = temp;

}

}

}

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

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

{

printf("%d ",buff[i]);

}\*/

insertionSort(buff,n);

write(ns,buff,sizeof buff);

printf("Array Sorted and sent to client\n");

break;

case 2:

read(ns,&num,sizeof(int));

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

{

if(buff[i] == num)

{

x=1;

write(ns,&(i),sizeof(int));

break;

}

}

if(x!=1)

{

i=-1;

write(ns,&i,sizeof(int));

}

break;

case 3:

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

{

if(buff[i]%2==0)

even[n1++] = buff[i];

else

odd[n2++] = buff[i];

}

write(ns,&n1,sizeof(int));

write(ns,&n2,sizeof(int));

write(ns,even,sizeof(even));

write(ns,odd,sizeof(odd));

printf("Odd and Even Array Sent\n");

break;

}

}

close(ns);

close(s);

}

# Client Side:

#include <string.h>

#include<strings.h>

#include <unistd.h>

#include<sys/socket.h>

#include<sys/types.h>

#include<netinet/in.h>

#include<stdlib.h>

#include<stdio.h>

#define port 1234

int main()

{

int sfd;

struct sockaddr\_in server;

socklen\_t len;

int buff[50];

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

if(sfd!=-1)

printf("Socket Created \n");

server.sin\_family = AF\_INET;

server.sin\_port = htons(port);

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

int r = connect(sfd,(struct sockaddr\*)&server,sizeof server);

if(r!=-1)

{

printf("Connection Established with Server\n");

}

/\*send(sfd,buff,sizeof(buff),0);

recv(sfd,buff,sizeof(buff),0);.

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

close(sfd);\*/

while(1)

{

int n,choice,i,x=-1,num;

int odd[50],even[50],n1,n2;

if(x==0)

break;

printf("Enter Array Length: ");

scanf("%d",&n);

printf("Enter Elements of Array\n");

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

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

printf("Enter Choice: \n1-Sort | 2-Find | 3-Split to odd and even: \n");

scanf("%d",&choice);

write(sfd,&choice,sizeof(choice));

write(sfd,&n,sizeof n);

send(sfd,buff,sizeof buff,0);

switch(choice)

{

case 1:

read(sfd,buff,sizeof buff);

printf("Sorted Array: ");

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

{

printf("%d ",buff[i]);

}

break;

case 2:

printf("Enter Number To Find: \n");

scanf("%d",&num);

write(sfd,&num,sizeof num);

read(sfd,&num,sizeof num);

if(num!=-1)

printf("Found At Position: %d\n",num+1);

else

printf("Element not found\n");

break;

case 3:

read(sfd,&n1,sizeof(int));

read(sfd,&n2,sizeof(int));

read(sfd,even,sizeof(even));

read(sfd,odd,sizeof(odd));

printf("Even Array: ");

for(i=0;i<n1;i++)

printf("%d ",even[i]);

printf("\n");

printf("Odd Array: ");

for(i=0;i<n2;i++)

printf("%d ",odd[i]);

printf("\n");

}

printf("Do u want to continue: 1/0:\n");

int ch;

fflush(stdin);

scanf("%d",&ch);

if(ch!=1)

{

choice=0;

write(sfd,&choice,sizeof(choice));

break;

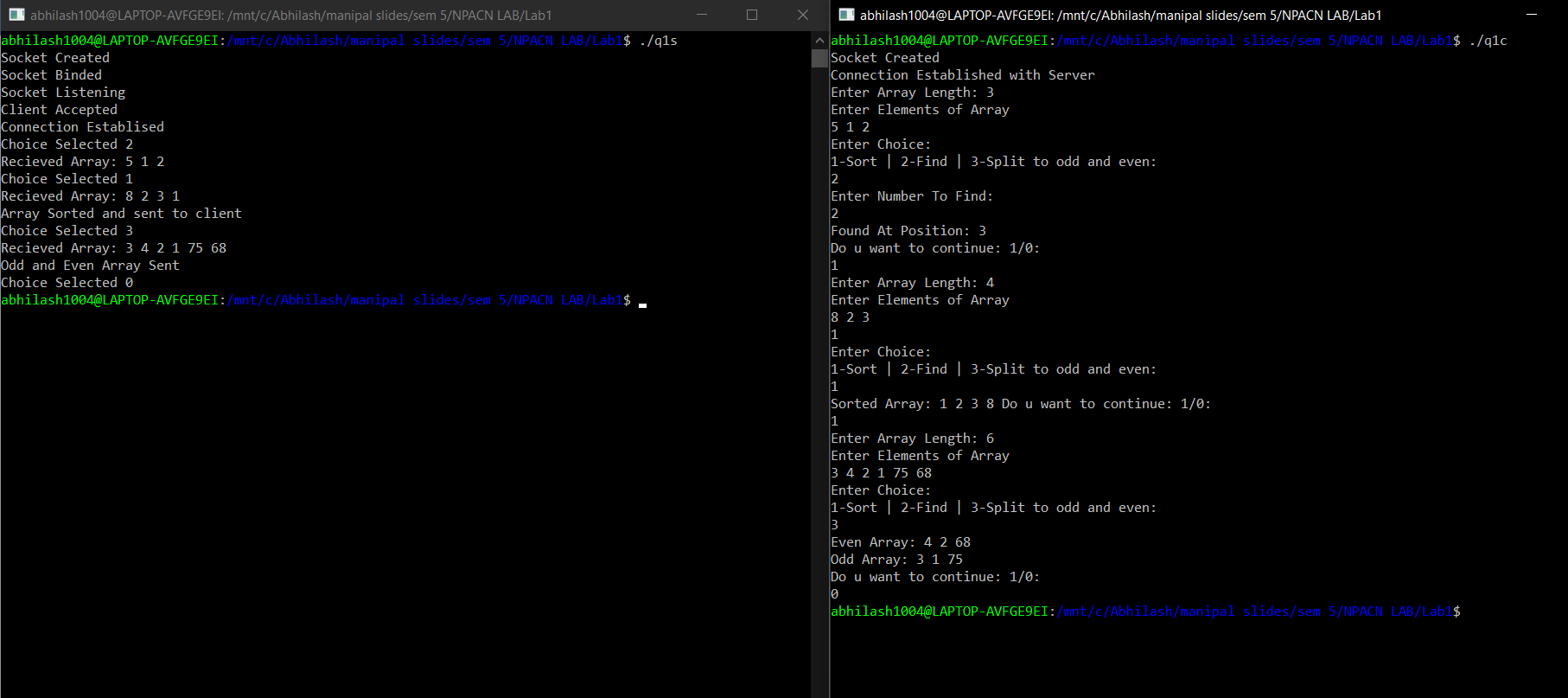
}

}

close(sfd);

}

Output:



Question 2:

Server Side:

#include <string.h>

#include<strings.h>

#include <unistd.h>

#include<sys/socket.h>

#include<sys/types.h>

#include<netinet/in.h>

#include<stdlib.h>

#include<stdio.h>

#define port 1234

int checkPalindrome(char \*str)

{

int n = strlen(str);

int i;

for(i=0;i<n/2;i++)

{

if(str[i]!=str[n-i-1])

return 0;

}

return 1;

}

void countVowels(int \*vow,char \*str)

{

printf("Copunted\n");

int n = strlen(str);

int i;

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

{

switch(str[i])

{

case 'a':

vow[0]++;

break;

case 'e':

vow[1]++;

break;

case 'i':

vow[2]++;

break;

case 'o':

vow[3]++;

break;

case 'u':

vow[4]++;

break;

}

}

printf("adnjhf\n");

}

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

{

int s,r,len;

struct sockaddr\_in server,client;

char buff[100];

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

if(s!=-1)

printf("Socket Created \n");

server.sin\_family = AF\_INET;

server.sin\_port = htons(port);

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

len = sizeof(client);

r = bind(s,(struct sockaddr\*)&server,sizeof server);

if (r!=-1)

printf("Socket Binded \n");

/\*recvfrom(s,buff,sizeof(buff),0,(struct sockaddr\*)&client,&len);

printf("Message Recieved From Client: %s\n",buff);\*/

while(1)

{

recvfrom(s,buff,sizeof(buff),0,(struct sockaddr\*)&client,&len);

printf("String Recieved\n");

if(!strcmp(buff,"halt"))

{

break;

}

int ans=checkPalindrome(buff);

int strleng = strlen(buff);

if(ans)

{

sendto(s,"It is a palindrome",sizeof("It is a palindrome"),0,(struct sockaddr\*)&client,len);

}

else

{

sendto(s,"It is not a palindrome",sizeof("It is not a palindrome"),0,(struct sockaddr\*)&client,len);

}

sendto(s,&strleng,sizeof(strleng),0,(struct sockaddr\*)&client,len);

int counti[5]={0};

countVowels(counti,buff);

sendto(s,counti,sizeof(counti),0,(struct sockaddr\*)&client,len);

}

close(s);

return 0;

}

Client Side:

#include <string.h>

#include<strings.h>

#include <unistd.h>

#include<sys/socket.h>

#include<sys/types.h>

#include<netinet/in.h>

#include<stdlib.h>

#include<stdio.h>

#define port 1234

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

{

int sfd;

struct sockaddr\_in server;

socklen\_t len;

char buff[100];

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

len = sizeof(server);

if(sfd!=-1)

printf("Socket Created \n");

server.sin\_family = AF\_INET;

server.sin\_port = htons(port);

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

//sendto(sfd,"Hello",sizeof("Hello"),0,(struct sockaddr\*)&server,len);

while(1)

{

printf("Enter String\n");

scanf("%s",buff);

sendto(sfd,buff,sizeof(buff),0,(struct sockaddr\*)&server,len);

if(!strcmp(buff,"halt"))

{

break;

}

recvfrom(sfd,buff,sizeof(buff),0,(struct sockaddr\*)&server,&len);

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

int strleng;

recvfrom(sfd,&strleng,sizeof(strleng),0,(struct sockaddr\*)&server,&len);

printf("String Length: %d\n",strleng);

int counti[5];

recvfrom(sfd,counti,sizeof(counti),0,(struct sockaddr\*)&server,&len);

printf("No of a: %d\n",counti[0]);

printf("No of e: %d\n",counti[1]);

printf("No of i: %d\n",counti[2]);

printf("No of o: %d\n",counti[3]);

printf("No of u: %d\n",counti[4]);

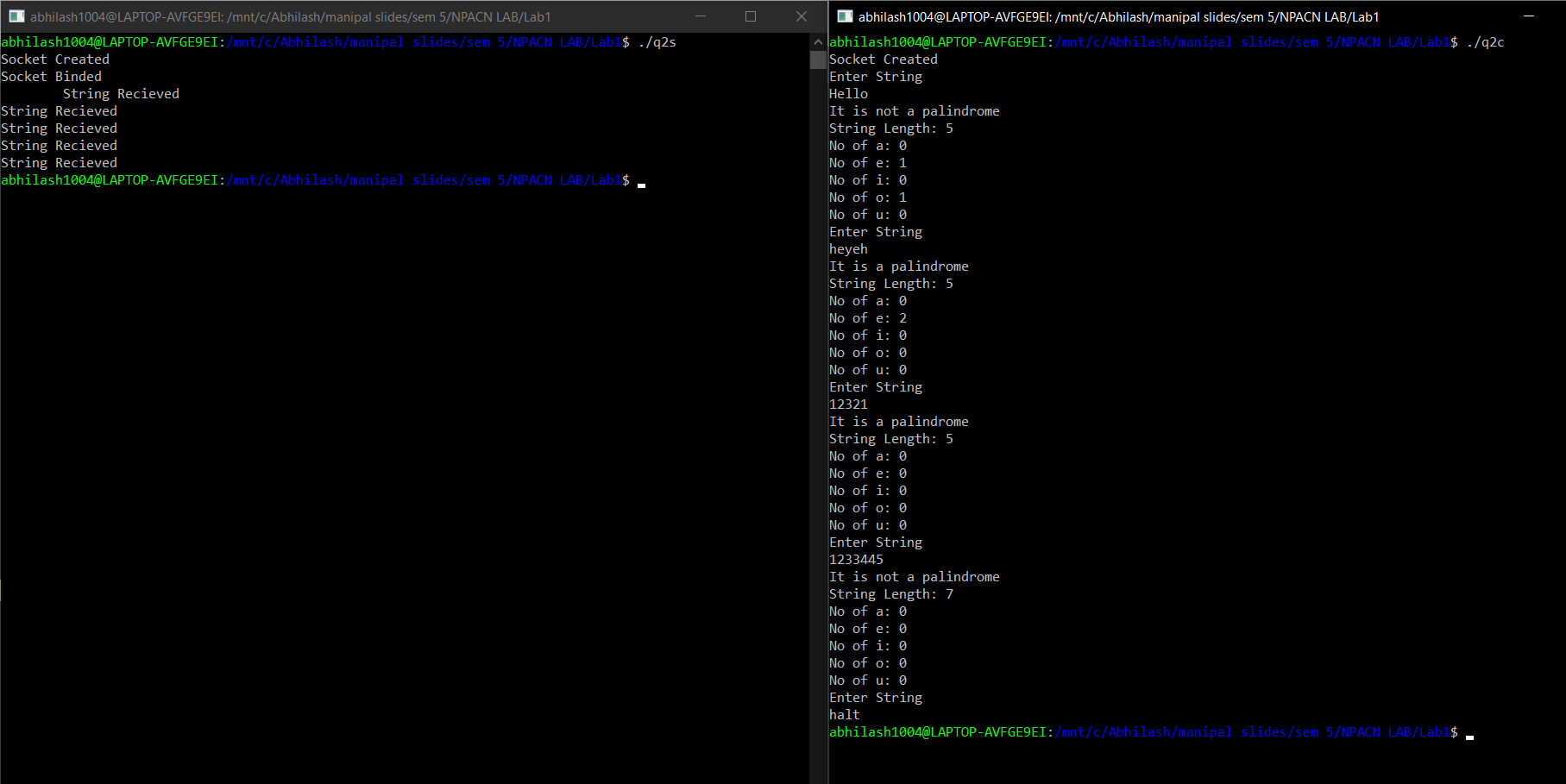
}

close(sfd);

return 0;

}

Output:



# Lab 2:

Question 1:

Client Side:

#include <string.h>

#include<strings.h>

#include<arpa/inet.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<stdio.h>

#include<unistd.h>

#include<stdlib.h>

#define port 12345

int main()

{

int s;

struct sockaddr\_in server;

socklen\_t len;

char buff[100];

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

server.sin\_port = htons(port);

server.sin\_family = AF\_INET;

inet\_aton("127.0.0.1",&server.sin\_addr);

if(s!=-1)

printf("Socket Created \n");

len = sizeof(server);

int r = connect(s,(struct sockaddr\*)&server,len);

if(r!=-1)

{

printf("Connection Established with Server\n");

}

while(1)

{

printf("Enter The File Name\n");

scanf("%s",buff);

write(s,buff,sizeof buff);

read(s,buff,sizeof buff);

if(!strcmp(buff,"File not present"))

{

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

break;

}

//read(s,buff,sizeof buff);

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

int choice;

scanf("%d",&choice);

write(s,&choice,sizeof(choice));

int ans;

char buf1[100],buf2[100];

switch(choice)

{

case 1:

printf("Enter String to be searched: \n");

scanf("%s",buff);

write(s,buff,sizeof buff);

read(s,&ans,sizeof ans);

if(ans==0)

printf("No occurences of %s\n",buff);

else

printf("%d occurences of %s\n",ans,buff);

break;

case 2:

printf("Enter String to be replaced: ");

scanf("%s",buf1);

//fgets(buf1);

printf("Enter String to be placed: ");

scanf("%s",buf2);

//fgets("%s",buf2);

write(s,buf1,sizeof buf1);

write(s,buf2,sizeof buf2);

read(s,buf1,sizeof buf1);

printf("%s\n",buf1);

break;

case 4:

close(s);

exit(0);

break;

}

}

close(s);

}

Server Side:

#include <string.h>

#include<strings.h>

#include<arpa/inet.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<stdio.h>

#include<unistd.h>

#include<stdlib.h>

#define port 12345

int countOccurrences(char \*filecontents,char \*buff)

{

int l = strlen(buff);

int ans=0,x=0,i;

int tl = strlen(filecontents);

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

{

if(filecontents[i]==buff[x])

{

x++;

}

else

{

i = i-x;

x=0;

}

if(x==l)

{

x=0;

ans+=1;

}

}

return ans;

}

int replaceAll(char \*str,char \*old,char \*new)

{

char \*pos,temp[500];

int index=0;

int oldlen = strlen(old);

//printf("%s\n",str);

int x=0;

while((pos = strstr(str,old))!=NULL)

{

x++;

index = pos - str;

strcpy(temp,str);

//printf("%s\n",str+oldlen );

str[index] = '\0';

strcat(str,new);

strcat(str,temp+index+oldlen);

}

if(x!=0)

return 1;

return 0;

}

void bubbleSort(char strData[100][100],int noOfLines)

{

int i,j;

char strTempData[100];

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

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

if(strcmp(strData[j], strData[j+1]) > 0) {

strcpy(strTempData, strData[j]);

strcpy(strData[j], strData[j+1]);

strcpy(strData[j+1], strTempData);

}

}

}

}

int main()

{

int s,ns,r;

char buff[100];

struct sockaddr\_in server,client;

socklen\_t len;

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

if(s!=-1)

printf("Socket Created \n");

server.sin\_port = htons(port);

server.sin\_family = AF\_INET;

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

inet\_aton("127.0.0.1",&server.sin\_addr);

r = bind(s,(struct sockaddr\*) &server,sizeof(server));

if (r!=-1)

printf("Socket Binded \n");

r = listen(s,5);

if (r!=-1)

printf("Socket Listening \n");

len = sizeof(client);

ns = accept(s,(struct sockaddr \*)&client,&len);

if(ns!=-1)

printf("Client Accepted\nConnection Establised\n");

while(1)

{

read(ns,buff,sizeof buff);

printf("File Name Recieved: %s\n",buff);

FILE \*f;

f = fopen(buff,"r");

if(f==NULL)

{

char temp[]="File not present";

write(ns,temp,sizeof temp);

break;

}

char temp2[] = "Enter 1. Search 2. Replace 3. Reorder 4. Exit: ";

write(ns,temp2,sizeof temp2);

int x=0,i;

char c;

int l;

char filecontents[5000];

// while(c=getc(f)!=EOF)

// {

// filecontents[x++] = c;

// printf("%c",c);

// }

//filecontents[x] = '\n';

//printf("File Contents: %s\n",filecontents);

int choice;

read(ns,&choice,sizeof choice);

printf("Selected Choce: %d\n",choice);

char buf1[100],buf2[100];

int tl = strlen(filecontents),ans=0,ans1=0;

switch(choice)

{

case 1:

read(ns,buff,sizeof buff);

while(fgets(filecontents,5000,f)!=NULL)

{

ans+=countOccurrences(filecontents,buff);

//char \*c1;

// while((c1 = strstr(filecontents,buff))!=NULL)

// ans1+=1;

}

/\*char \*c1=strstr(filecontents,buff);

if(c1!=NULL)

{

printf("FOUND!!! %c\n",\*c1);

}\*/

//printf("%d\n",ans1);

write(ns,&ans,sizeof ans);

break;

case 2:

i=0;

while((c = (char)getc(f))!=EOF)

{

filecontents[i++] = c;

}

filecontents[i]='\0';

//printf("%s\n",filecontents);

fclose(f);

read(ns,buf1,sizeof buf1);

read(ns,buf2,sizeof buf2);

//printf("%s\n",filecontents );

i = replaceAll(filecontents,buf1,buf2);

//printf("%s\n",filecontents );

if(i)

{

FILE \*f2;

f2 = fopen(buff,"w");

fputs(filecontents,f2);

char temp3[] = "String Replaced";

write(ns,temp3,sizeof temp3);

}

else

{

char temp4[] = "String not found";

write(ns,temp4,sizeof temp4);

}

break;

case 3:

i=0;

while((c = (char)getc(f))!=EOF)

{

filecontents[i++] = c;

}

filecontents[i]='\0';

char ch5[100][100];

char \*token=strtok(filecontents," ");

i=0;

while(token!=NULL)

{

//printf("%s\n",token );

strcpy(ch5[i++],token);

token=strtok(NULL," ");

}

bubbleSort(ch5,i);

int z;

char buff9[100];

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

{

char b[1000];

sprintf(b,"%s ",ch5[z]);

strcat(buff9,b);

}

fclose(f);

FILE \*f3;

f3 = fopen(buff,"w");

fputs(buff9,f3);

break;

case 4:

close(ns);

close(s);

exit(0);

}

}

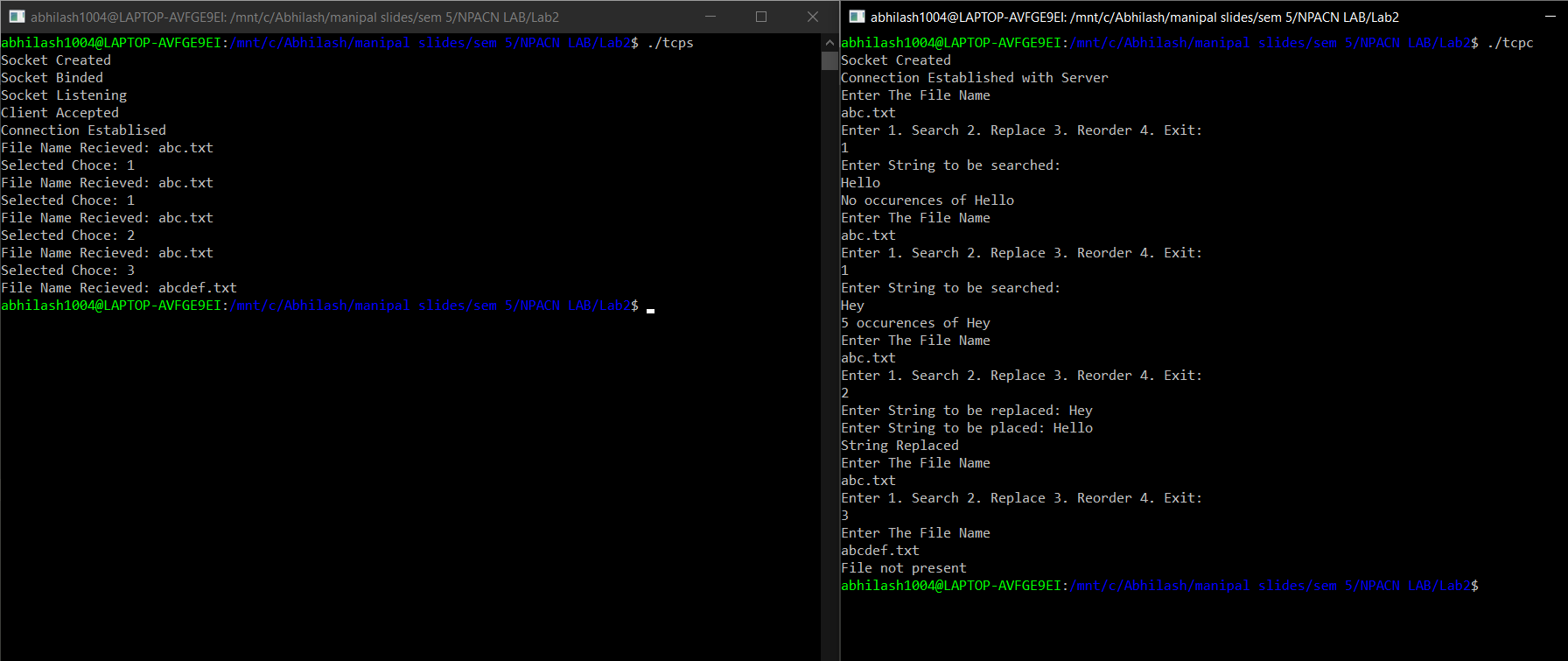
close(ns);

close(s);

return 0;

}

Output:



Question 2:

Client Side:

#include <string.h>

#include<strings.h>

#include<arpa/inet.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<stdio.h>

#include<unistd.h>

#include<stdlib.h>

#define port 12345

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

{

/\* code \*/

int s;

struct sockaddr\_in server;

socklen\_t len;

char buff[100];

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

len = sizeof(server);

if(s!=-1)

printf("Socket Created \n");

server.sin\_family = AF\_INET;

server.sin\_port = htons(port);

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

while(1)

{

printf("Enter The File Name\n");

scanf("%s",buff);

//write(s,buff,sizeof buff);

sendto(s,buff,sizeof(buff),0,(struct sockaddr\*)&server,len);

//read(s,buff,sizeof buff);

recvfrom(s,buff,sizeof(buff),0,(struct sockaddr\*)&server,&len);

if(!strcmp(buff,"File not present"))

{

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

break;

}

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

int choice;

scanf("%d",&choice);

//write(s,&choice,sizeof(choice));

sendto(s,&choice,sizeof(choice),0,(struct sockaddr\*)&server,len);

int ans;

char buf1[100],buf2[100];

switch(choice)

{

case 1:

printf("Enter String to be searched: \n");

scanf("%s",buff);

//write(s,buff,sizeof buff);

sendto(s,buff,sizeof(buff),0,(struct sockaddr\*)&server,len);

//read(s,&ans,sizeof ans);

recvfrom(s,&ans,sizeof(ans),0,(struct sockaddr\*)&server,&len);

if(ans==0)

printf("No occurences of %s\n",buff);

else

printf("%d occurences of %s\n",ans,buff);

break;

case 2:

printf("Enter String to be replaced: ");

scanf("%s",buf1);

//fgets(buf1);

printf("Enter String to be placed: ");

scanf("%s",buf2);

//fgets("%s",buf2);

//write(s,buf1,sizeof buf1);

//write(s,buf2,sizeof buf2);

sendto(s,buf1,sizeof(buf1),0,(struct sockaddr\*)&server,len);

sendto(s,buf2,sizeof(buf2),0,(struct sockaddr\*)&server,len);

//read(s,buf1,sizeof buf1);

recvfrom(s,buf1,sizeof(buf1),0,(struct sockaddr\*)&server,&len);

printf("%s\n",buf1);

break;

case 4:

close(s);

exit(0);

break;

}

}

return 0;

}

Server Side:

#include <string.h>

#include<strings.h>

#include<arpa/inet.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<stdio.h>

#include<unistd.h>

#include<stdlib.h>

#define port 12345

int countOccurrences(char \*filecontents,char \*buff)

{

int l = strlen(buff);

int ans=0,x=0,i;

int tl = strlen(filecontents);

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

{

if(filecontents[i]==buff[x])

{

x++;

}

else

{

i = i-x;

x=0;

}

if(x==l)

{

x=0;

ans+=1;

}

}

return ans;

}

int replaceAll(char \*str,char \*old,char \*new)

{

char \*pos,temp[500];

int index=0;

int oldlen = strlen(old);

//printf("%s\n",str);

int x=0;

while((pos = strstr(str,old))!=NULL)

{

x++;

index = pos - str;

strcpy(temp,str);

//printf("%s\n",str+oldlen );

str[index] = '\0';

strcat(str,new);

strcat(str,temp+index+oldlen);

}

if(x!=0)

return 1;

return 0;

}

void bubbleSort(char strData[100][100],int noOfLines)

{

int i,j;

char strTempData[100];

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

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

if(strcmp(strData[j], strData[j+1]) > 0) {

strcpy(strTempData, strData[j]);

strcpy(strData[j], strData[j+1]);

strcpy(strData[j+1], strTempData);

}

}

}

}

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

{

/\* code \*/

int s,r,len;

struct sockaddr\_in server,client;

char buff[100];

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

if(s!=-1)

printf("Socket Created \n");

server.sin\_family = AF\_INET;

server.sin\_port = htons(port);

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

len = sizeof(client);

r = bind(s,(struct sockaddr\*)&server,sizeof server);

if (r!=-1)

printf("Socket Binded \n");

while(1)

{

recvfrom(s,buff,sizeof(buff),0,(struct sockaddr\*)&client,&len);

printf("File Name Recieved: %s\n",buff);

FILE \*f;

f = fopen(buff,"r");

if(f==NULL)

{

char temp[]="File not present";

//write(ns,temp,sizeof temp);

sendto(s,temp,sizeof temp,0,(struct sockaddr\*)&client,len);

break;

}

char temp2[] = "Enter 1. Search 2. Replace 3. Reorder 4. Exit: ";

//write(ns,temp2,sizeof temp2);

sendto(s,temp2,sizeof temp2,0,(struct sockaddr\*)&client,len);

int x=0,i;

char c;

int l;

char filecontents[5000];

int choice;

//read(ns,&choice,sizeof choice);

recvfrom(s,&choice,sizeof(choice),0,(struct sockaddr\*)&client,&len);

printf("Selected Choce: %d\n",choice);

char buf1[100],buf2[100];

int tl = strlen(filecontents),ans=0,ans1=0;

switch(choice)

{

case 1:

//read(ns,buff,sizeof buff);

recvfrom(s,buff,sizeof(buff),0,(struct sockaddr\*)&client,&len);

while(fgets(filecontents,5000,f)!=NULL)

{

ans+=countOccurrences(filecontents,buff);

//char \*c1;

// while((c1 = strstr(filecontents,buff))!=NULL)

// ans1+=1;

}

/\*char \*c1=strstr(filecontents,buff);

if(c1!=NULL)

{

printf("FOUND!!! %c\n",\*c1);

}\*/

//printf("%d\n",ans1);

//write(ns,&ans,sizeof ans);

sendto(s,&ans,sizeof ans,0,(struct sockaddr\*)&client,len);

break;

case 2:

i=0;

while((c = (char)getc(f))!=EOF)

{

filecontents[i++] = c;

}

filecontents[i]='\0';

//printf("%s\n",filecontents);

fclose(f);

// read(ns,buf1,sizeof buf1);

// read(ns,buf2,sizeof buf2);

recvfrom(s,buf1,sizeof(buf1),0,(struct sockaddr\*)&client,&len);

recvfrom(s,buf2,sizeof(buf2),0,(struct sockaddr\*)&client,&len);

//printf("%s\n",filecontents );

i = replaceAll(filecontents,buf1,buf2);

//printf("%s\n",filecontents );

if(i)

{

FILE \*f2;

f2 = fopen(buff,"w");

fputs(filecontents,f2);

char temp3[] = "String Replaced";

//write(ns,temp3,sizeof temp3);

sendto(s,temp3,sizeof temp3,0,(struct sockaddr\*)&client,len);

}

else

{

char temp4[] = "String not found";

//write(ns,temp4,sizeof temp4);

sendto(s,temp4,sizeof temp4,0,(struct sockaddr\*)&client,len);

}

break;

case 3:

i=0;

while((c = (char)getc(f))!=EOF)

{

filecontents[i++] = c;

}

filecontents[i]='\0';

char ch5[100][100];

char \*token=strtok(filecontents," ");

i=0;

while(token!=NULL)

{

//printf("%s\n",token );

strcpy(ch5[i++],token);

token=strtok(NULL," ");

}

bubbleSort(ch5,i);

int z;

char buff9[100];

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

{

char b[1000];

sprintf(b,"%s ",ch5[z]);

strcat(buff9,b);

}

fclose(f);

FILE \*f3;

f3 = fopen(buff,"w");

fputs(buff9,f3);

break;

case 4:

close(s);

exit(0);

}

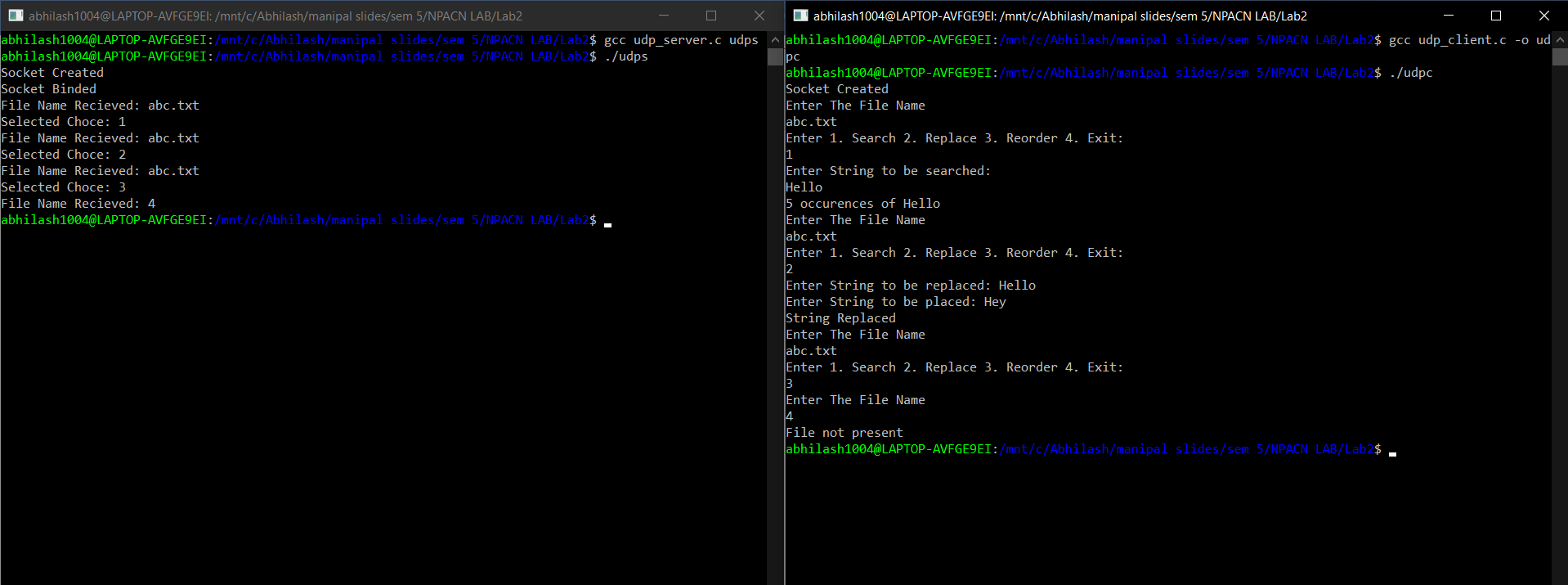
}

close(s);

return 0;

}

Output:



# Lab 3:

Question 1:

Client Side:

#include<string.h>

#include<arpa/inet.h>

#include<stdlib.h>

#include<stdio.h>

#include<unistd.h>

#include<sys/socket.h>

#include<sys/types.h>

#include<netinet/in.h>

#include<fcntl.h>

#include<sys/stat.h>

int main()

{

int s,r,recb,sntb,x;

printf("INPUT port number: ");

scanf("%d", &x);

struct sockaddr\_in server;

char buff[50],buff2[50];

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

if(s==-1)

{

printf("\nSocket creation error.");

exit(0);

}

printf("\nSocket created.");

server.sin\_family=AF\_INET;

server.sin\_port=htons(x);

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

r=connect(s,(struct sockaddr\*)&server,sizeof(server));

if(r==-1)

{

printf("\nConnection error.");

exit(0);

}

printf("\nSocket connected.");

printf("\n\n");

int pid;

pid=fork();

while(1){

if(pid>0)

{

//parent

recb=recv(s,buff,sizeof(buff),0);

if(recb==-1)

{

printf("\nMessage Recieving Failed");

close(s);

exit(0);

}

if(strcmp(buff,"BYE")==0)

break;

printf("\nParent - Message Recieved: ");

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

}

else

{

//child

printf("\nChild - Type Message: ");

scanf("%s", buff2);

sntb=send(s,buff2,sizeof(buff2),0);

if(sntb==-1)

{

close(s);

printf("\nMessage Sending Failed");

exit(0);

}

if(strcmp(buff2,"BYE")==0)

break;

}

}

close(s);

}

Server Side:

#include<string.h>

#include<arpa/inet.h>

#include<stdlib.h>

#include<stdio.h>

#include<unistd.h>

#include<sys/socket.h>

#include<sys/types.h>

#include<netinet/in.h>

#include<fcntl.h>

#include<sys/stat.h>

int main()

{

int s,r,recb,sntb,x;

printf("INPUT port number: ");

scanf("%d", &x);

struct sockaddr\_in server;

char buff[50],buff2[50];

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

if(s==-1)

{

printf("\nSocket creation error.");

exit(0);

}

printf("\nSocket created.");

server.sin\_family=AF\_INET;

server.sin\_port=htons(x);

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

r=connect(s,(struct sockaddr\*)&server,sizeof(server));

if(r==-1)

{

printf("\nConnection error.");

exit(0);

}

printf("\nSocket connected.");

printf("\n\n");

int pid;

pid=fork();

while(1){

if(pid>0)

{

//parent

recb=recv(s,buff,sizeof(buff),0);

if(recb==-1)

{

printf("\nMessage Recieving Failed");

close(s);

exit(0);

}

if(strcmp(buff,"BYE")==0)

break;

printf("\nParent - Message Recieved: ");

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

}

else

{

//child

printf("\nChild - Type Message: ");

scanf("%s", buff2);

sntb=send(s,buff2,sizeof(buff2),0);

if(sntb==-1)

{

close(s);

printf("\nMessage Sending Failed");

exit(0);

}

if(strcmp(buff2,"BYE")==0)

break;

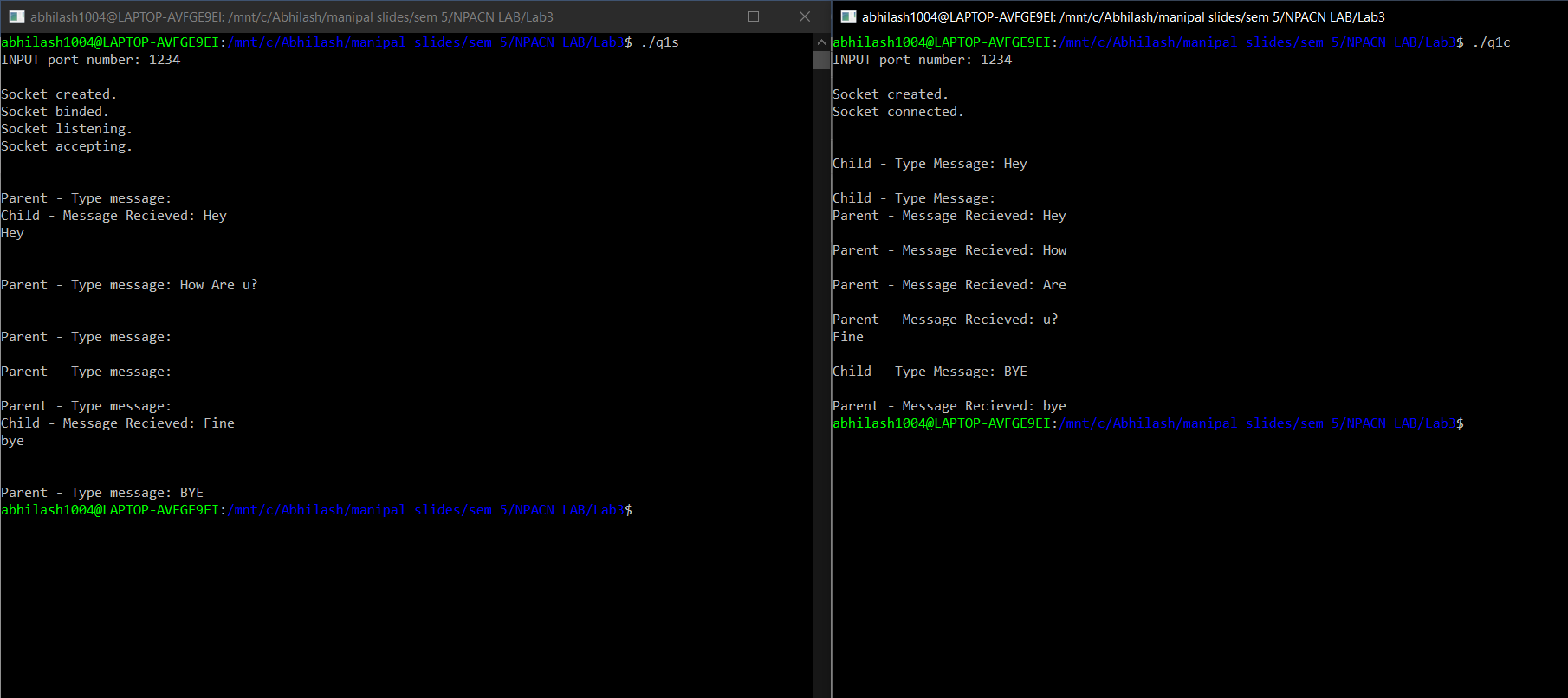
}

}

close(s);

}

Output:



Question 2:

Client Side:

#include<string.h>

#include<arpa/inet.h>

#include<stdlib.h>

#include<stdio.h>

#include<unistd.h>

#include<sys/socket.h>

#include<sys/types.h>

#include<netinet/in.h>

#include<fcntl.h>

#include<sys/stat.h>

int main()

{

int s,r,recb,sntb,x;

int sa;

socklen\_t len;

printf("INPUT port number: ");

scanf("%d", &x);

struct sockaddr\_in server,client;

char buff[50];

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

if(s==-1)

{

printf("\nSocket creation error.");

exit(0);

}

printf("\nSocket created.");

server.sin\_family=AF\_INET;

server.sin\_port=htons(x);

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

sa=sizeof(server);

len=sizeof(server);

while(1){

printf("\n\n");

printf("Type Message (enter 'stop' to exit): ");

scanf("%s", buff);

sntb=sendto(s,buff,sizeof(buff),0,(struct sockaddr \*)&server, len);

if(sntb==-1)

{

close(s);

printf("\nMessage sending Failed");

exit(0);

}

if(!strcmp(buff,"stop"))

break;

}

close(s);

}

Server Side:

#include<string.h>

#include<unistd.h>

#include<sys/socket.h>

#include<sys/types.h>

#include<netinet/in.h>

#include<stdlib.h>

#include<stdio.h>

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

{

char temp;

temp = \*x;

\*x = \*y;

\*y = temp;

}

void permute(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));

permute(a, l+1, r);

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

}

}

}

int main()

{

int s,r,recb,sntb,x;

int ca;

printf("INPUT port number: ");

scanf("%d", &x);

socklen\_t len;

struct sockaddr\_in server,client;

char buff[50];

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

if(s==-1)

{

printf("\nSocket creation error.");

exit(0);

}

printf("\nSocket created.\n");

server.sin\_family=AF\_INET;

server.sin\_port=htons(x);

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

len=sizeof(client);

ca=sizeof(client);

r=bind(s,(struct sockaddr\*)&server,sizeof(server));

if(r==-1)

{

printf("\nBinding error.");

exit(0);

}

printf("\nSocket binded.\n");

while(1){

recb=recvfrom(s,buff,sizeof(buff),0,(struct sockaddr\*)&client,&ca);

if(recb==-1)

{

printf("\nMessage Recieving Failed");

close(s);

exit(0);

}

printf("\nMessage Recieved: ");

printf("%s", buff);

if(!strcmp(buff,"stop"))

break;

printf("\nPermutations of the string are: \n");

int n=strlen(buff);

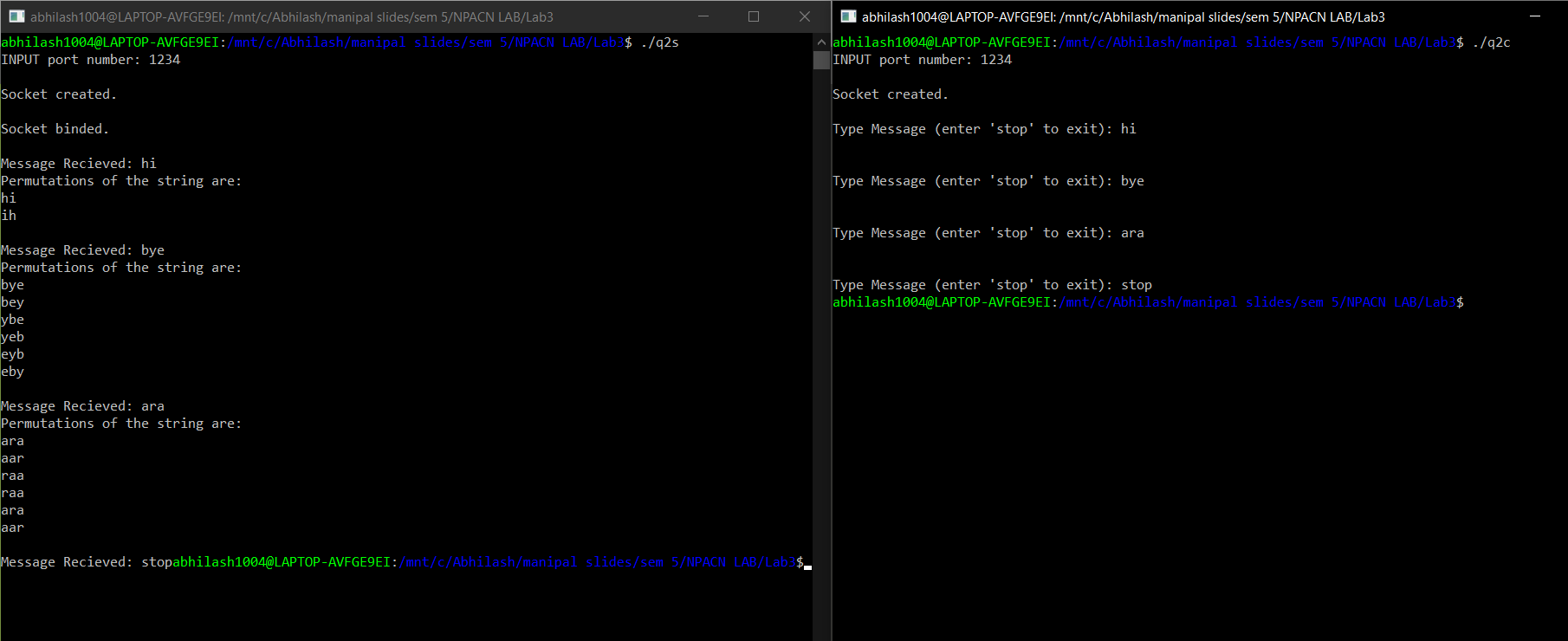
permute(buff, 0, n-1);

}

close(s);

}

Ouput:



Question 3:

Client Side:

#include<string.h>

#include<arpa/inet.h>

#include<stdlib.h>

#include<stdio.h>

#include<unistd.h>

#include<sys/socket.h>

#include<sys/types.h>

#include<netinet/in.h>

#include<fcntl.h>

#include<sys/stat.h>

int main()

{

int s,r,recb,sntb,x;

printf("INPUT port number: ");

scanf("%d", &x);

struct sockaddr\_in server;

char buff[50];

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

if(s==-1)

{

printf("\nSocket creation error.");

exit(0);

}

printf("\nSocket created.");

server.sin\_family=AF\_INET;

server.sin\_port=htons(x);

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

r=connect(s,(struct sockaddr\*)&server,sizeof(server));

if(r==-1)

{

printf("\nConnection error.");

exit(0);

}

printf("\nSocket connected.\n");

printf("\n\n");

strcpy(buff,"");

printf("Enter alphanumeric string: ");

scanf("%s", buff);

sntb=send(s,buff,sizeof(buff),0);

if(sntb==-1)

{

close(s);

printf("\nMessage Sending Failed");

exit(0);

}

close(s);

}

Server Side:

#include<string.h>

#include<unistd.h>

#include<sys/socket.h>

#include<sys/types.h>

#include<netinet/in.h>

#include<stdlib.h>

#include<stdio.h>

int main()

{

int s,r,recb,sntb,x,ns,a=0;

printf("INPUT port number: ");

scanf("%d", &x);

socklen\_t len;

struct sockaddr\_in server,client;

char buff[50],buff2[50];

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

if(s==-1)

{

printf("\nSocket creation error.");

exit(0);

}

printf("\nSocket created.");

server.sin\_family=AF\_INET;

server.sin\_port=htons(x);

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

r=bind(s,(struct sockaddr\*)&server,sizeof(server));

if(r==-1)

{

printf("\nBinding error.");

exit(0);

}

printf("\nSocket binded.");

r=listen(s,1);

if(r==-1)

{

close(s);

exit(0);

}

printf("\nSocket listening.");

len=sizeof(client);

ns=accept(s,(struct sockaddr\*)&client, &len);

if(ns==-1)

{

close(s);

exit(0);

}

printf("\nSocket accepting.\n");

recb=recv(ns,buff,sizeof(buff),0);

if(recb==-1)

{

printf("\nMessage Recieving Failed");

close(s);

close(ns);

exit(0);

}

printf("\nMessage Recieved: ");

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

int pid=fork();

if(pid>0){

//parent

char num[50]={0},alpha[50]={0};

int i,j,k;

i=0;

j=0;

k=0;

int n=strlen(buff);

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

{

if((buff[i]>=65 && buff[i]<=91)||(buff[i]>=97 && buff[i]<=123))

{

alpha[j]=buff[i];

j++;

}

else

{

num[k]=buff[i];

k++;

}

}

alpha[j]='\0';

num[k]='\0';

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

{

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

{

if(alpha[x]<alpha[x+1])

{

char temp=alpha[x];

alpha[x]=alpha[x+1];

alpha[x+1]=temp;

}

}

}

printf("\nParent process - Sorted alphabetical string is: \n");

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

printf("\n\n");

}

else{

//child

char num[50]={0},alpha[50]={0};

int i,j,k;

i=0;

j=0;

k=0;

int n=strlen(buff);

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

{

if((buff[i]>=65 && buff[i]<=91)||(buff[i]>=97 && buff[i]<=123))

{

alpha[j]=buff[i];

j++;

}

else

{

num[k]=buff[i];

k++;

}

}

alpha[j]='\0';

num[k]='\0';

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

{

for(int x=0;x<k-i-1;x++)

{

if(num[x]>num[x+1])

{

char temp=num[x];

num[x]=num[x+1];

num[x+1]=temp;

}

}

}

printf("\nChild process - Sorted numerical string is: \n");

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

printf("\n\n");

}

close(ns);

close(s);

}

Output:

