Regular Eval-1

Client

#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,i,j;

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.");

int ch=1;

int a[50];

char arr[50],IP[50];

int n;

strcpy(arr,"Hello from client");

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

if(sntb==-1)

{

close(s);

printf("\nMessage Sending Failed");

exit(0);

}

strcpy(IP,"127.0.0.1");

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

if(sntb==-1)

{

close(s);

printf("\nMessage Sending Failed");

exit(0);

}

char buff2[50];

recb=recv(s,buff2,sizeof(buff2),0);

if(recb==-1)

{

close(s);

printf("\nMessage Recieving Failed");

exit(0);

}

printf("\n");

puts(buff2);

close(s);

}

Server:

#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];

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");

char arr[50];

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

if(recb==-1)

{

printf("\nMessage Recieving Failed");

close(s);

close(ns);

exit(0);

}

puts(arr);

//printf("\n");

//printf("%i ", htonl(client.sin\_addr.s\_addr));

//printf("\n %i", htons(client.sin\_port));

// printf("\n %i", htonl("127.0.0.1"));

printf("\n");

char IP[50];

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

strcat(IP,":");

strcat(IP,"3000");

puts(IP);

FILE \* fTemp;

char fil[50];

if( access( "temp.txt", F\_OK ) != -1 ) {

// file exists

fTemp = fopen("temp.txt", "w");

fputs(IP, fTemp);

} else {

// file doesn't exist

strcpy(buff,"File does not exist!");

}

char changedBuff[50];

int i;

for(i = 0; i<strlen(IP);i++)

{

if(IP[i]=='.'){

strcat(changedBuff,"dot");

}

else if(IP[i]==':'){

strcat(changedBuff,"colon");

}

else{

char x[2];

x[0] =IP[i];

x[i] ='\0';

//printf("%s",x);

printf("\n");

strcat(changedBuff,x);

}

}

puts(changedBuff);

/\*

sntb=send(ns,changedBuff,sizeof(changedBuff),0);

{

close(s);

close(ns);

printf("\nMessage Sending Failed");

exit(0);

}

\*/

close(ns);

close(s);

}

Output:

