106118106\_Socket\_Programming\_Assignment

One Way text transfer TCP

server.c

#include <unistd.h>

#include <stdio.h>

#include <sys/socket.h>

#include <stdlib.h>

#include <netinet/in.h>

#include <string.h>

#define PORT 8080

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

{

    int server\_fd, new\_socket, valread;

    struct sockaddr\_in address;

    int opt = 1;

    int addrlen = sizeof(address);

    char buffer[1024] = {0};

    char \*hello = "Hello from server";

    if ((server\_fd = socket(AF\_INET, SOCK\_STREAM, 0)) == 0)

    {

        perror("socket failed");

        exit(EXIT\_FAILURE);

    }

    if (setsockopt(server\_fd, SOL\_SOCKET, SO\_REUSEADDR,&opt, sizeof(opt)))

    {

        perror("setsockopt");

        exit(EXIT\_FAILURE);

    }

    address.sin\_family = AF\_INET;

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

    address.sin\_port = htons( PORT );

    if (bind(server\_fd, (struct sockaddr \*)&address,

                                 sizeof(address))<0)

    {

        perror("bind failed");

        exit(EXIT\_FAILURE);

    }

    if (listen(server\_fd, 3) < 0)

    {

        perror("listen");

        exit(EXIT\_FAILURE);

    }

    if ((new\_socket=accept(server\_fd, (struct sockaddr \*)&address,socklen\_t\*)&addrlen))<0)

    {

        perror("accept");

        exit(EXIT\_FAILURE);

    }

    valread = read( new\_socket , buffer, 1024);

    printf("%s\n",buffer );

    send(new\_socket , hello , strlen(hello) , 0 );

    printf("Hello message sent\n");

return 0; }

client.c

#include <stdio.h>

#include <sys/socket.h>

#include <arpa/inet.h>

#include <unistd.h>

#include <string.h>

#define PORT 8080

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

{

    int sock = 0, valread;

    struct sockaddr\_in serv\_addr;

    char \*hello = "Hello from client";

    char buffer[1024] = {0};

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

    {

        printf("\n Socket creation error \n");

        return -1;

    }

    serv\_addr.sin\_family = AF\_INET;

    serv\_addr.sin\_port = htons(PORT);

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

    {

        printf("\nInvalid address/ Address not supported \n");

        return -1;

    }

    if (connect(sock, (struct sockaddr \*)&serv\_addr, sizeof(serv\_addr)) < 0)

    {

        printf("\nConnection Failed \n");

        return -1;

    }

    send(sock , hello , strlen(hello) , 0 );

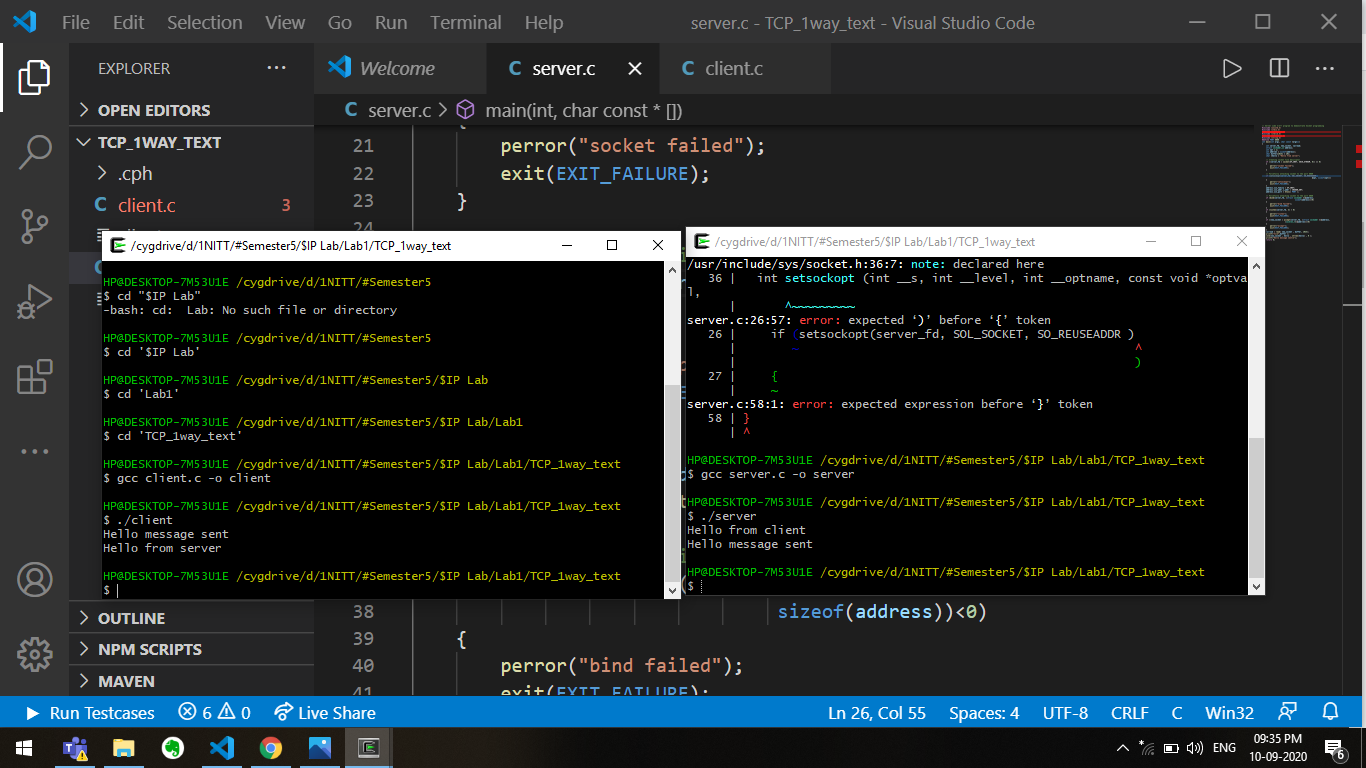
    printf("Hello message sent\n");

    valread = read( sock , buffer, 1024);

    printf("%s\n",buffer );

    return 0;

}



One Way file transfer TCP

receive\_file.c

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <arpa/inet.h>

#include <netinet/in.h>

#include <sys/socket.h>

#include "transfer.h"

void writefile(int sockfd, FILE \*fp);

ssize\_t total=0;

int main(int argc, char \*argv[])

{

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

    if (sockfd == -1)

    {

        perror("Can't allocate sockfd");

        exit(1);

    }

    struct sockaddr\_in clientaddr, serveraddr;

    memset(&serveraddr, 0, sizeof(serveraddr));

    serveraddr.sin\_family = AF\_INET;

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

    serveraddr.sin\_port = htons(SERVERPORT);

    if (bind(sockfd, (const struct sockaddr \*) &serveraddr, sizeof(serveraddr)) == -1)

    {

        perror("Bind Error");

        exit(1);

    }

    if (listen(sockfd, LINSTENPORT) == -1)

    {

        perror("Listen Error");

        exit(1);

    }

    socklen\_t addrlen = sizeof(clientaddr);

    int connfd = accept(sockfd, (struct sockaddr \*) &clientaddr, &addrlen);

    if (connfd == -1)

    {

        perror("Connect Error");

        exit(1);

    }

    close(sockfd);

    char filename[BUFFSIZE] = {0};

    if (recv(connfd, filename, BUFFSIZE, 0) == -1)

    {

        perror("Can't receive filename");

        exit(1);

    }

    FILE \*fp = fopen(filename, "wb");

    if (fp == NULL)

    {

        perror("Can't open file");

        exit(1);

    }

    char addr[INET\_ADDRSTRLEN];

    printf("Start receive file: %s from %s\n", filename, inet\_ntop(AF\_INET, &clientaddr.sin\_addr, addr, INET\_ADDRSTRLEN));

    writefile(connfd, fp);

    printf("Receive Success, NumBytes = %ld\n", total);

    fclose(fp);

    close(connfd);

    return 0;

}

void writefile(int sockfd, FILE \*fp)

{

    ssize\_t n;

    char buff[MAX\_LINE] = {0};

    while ((n = recv(sockfd, buff, MAX\_LINE, 0)) > 0)

    {

        total+=n;

        if (n == -1)

        {

            perror("Receive File Error");

            exit(1);

        }

        if (fwrite(buff, sizeof(char), n, fp) != n)

        {

            perror("Write File Error");

            exit(1);

        }

        memset(buff, 0, MAX\_LINE);

    }

}

Send\_file.c

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <libgen.h>

#include <unistd.h>

#include <arpa/inet.h>

#include <netinet/in.h>

#include <sys/socket.h>

#include "transfer.h"

void sendfile(FILE \*fp, int sockfd);

ssize\_t total=0;

int main(int argc, char\* argv[])

{

    if (argc != 3)

    {

        perror("usage:send\_file filepath <IPaddress>");

        exit(1);

    }

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

    if (sockfd < 0)

    {

        perror("Can't allocate sockfd");

        exit(1);

    }

    struct sockaddr\_in serveraddr;

    memset(&serveraddr, 0, sizeof(serveraddr));

    serveraddr.sin\_family = AF\_INET;

    serveraddr.sin\_port = htons(SERVERPORT);

    if (inet\_pton(AF\_INET, argv[2], &serveraddr.sin\_addr) < 0)

    {

        perror("IPaddress Convert Error");

        exit(1);

    }

    if (connect(sockfd, (const struct sockaddr \*) &serveraddr, sizeof(serveraddr)) < 0)

    {

        perror("Connect Error");

        exit(1);

    }

    char \*filename = basename(argv[1]);

    if (filename == NULL)

    {

        perror("Can't get filename");

        exit(1);

    }

    char buff[BUFFSIZE] = {0};

    strncpy(buff, filename, strlen(filename));

    if (send(sockfd, buff, BUFFSIZE, 0) == -1)

    {

        perror("Can't send filename");

        exit(1);

    }

    FILE \*fp = fopen(argv[1], "rb");

    if (fp == NULL)

    {

        perror("Can't open file");

        exit(1);

    }

    sendfile(fp, sockfd);

    //puts("Send Success");

    printf("Send Success, NumBytes = %ld\n", total);

    fclose(fp);

    close(sockfd);

    return 0;

}

void sendfile(FILE \*fp, int sockfd)

{

    int n;

    char sendline[MAX\_LINE] = {0};

    while ((n = fread(sendline, sizeof(char), MAX\_LINE, fp)) > 0)

    {

        total+=n;

        if (n != MAX\_LINE && ferror(fp))

        {

            perror("Read File Error");

            exit(1);

        }

        if (send(sockfd, sendline, n, 0) == -1)

        {

            perror("Can't send file");

            exit(1);

        }

        memset(sendline, 0, MAX\_LINE);

    }

}

