**Networks Lab : Basic Programs**

**-V Venkataraman 106118106**

**TCP 1WAY TEXT**

**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 );

    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;

    }

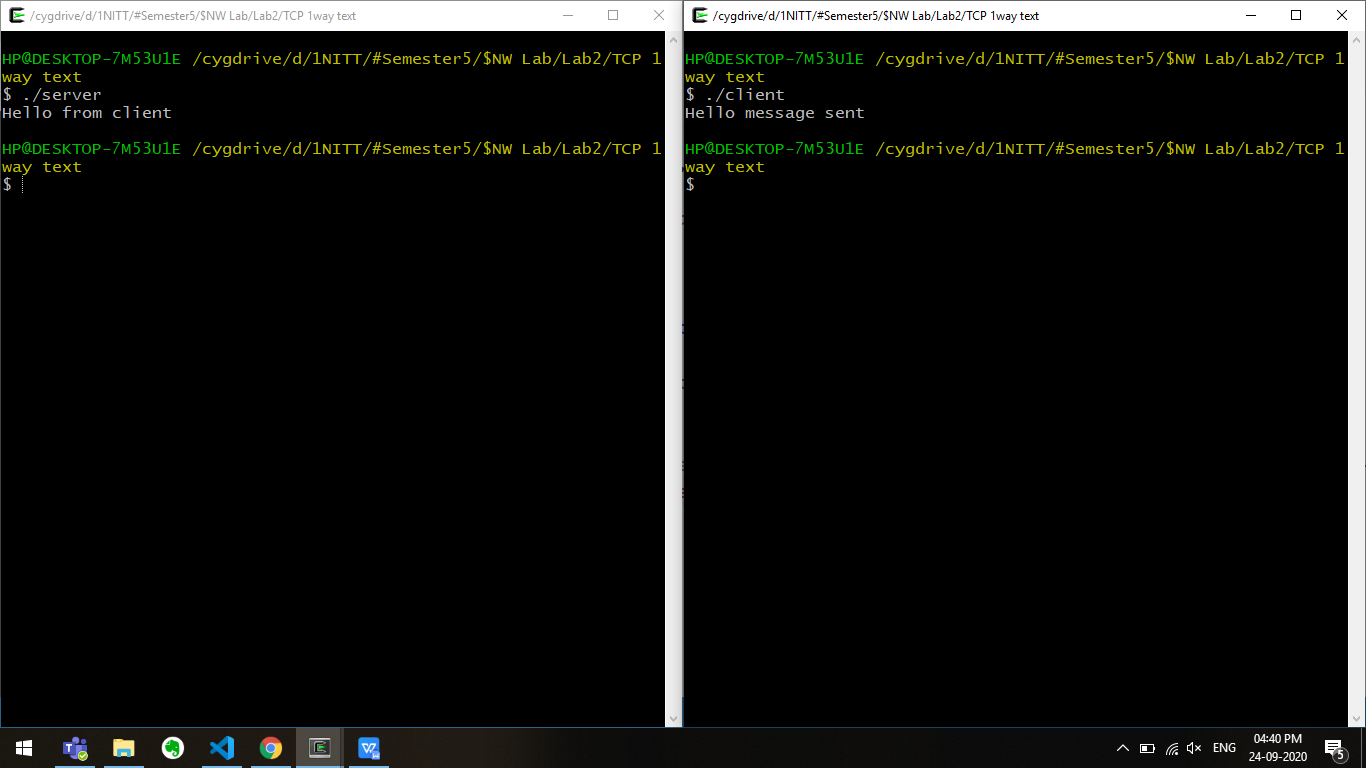
    send(sock , hello , strlen(hello) , 0 );

    printf("Hello message sent\n");

    return 0;

}

**Output**



**TCP 1WAY FILE**

**server.c**

#include<stdio.h>

#include<stdlib.h>

#include<unistd.h>

#include<string.h>

#include<sys/socket.h>

#include<sys/types.h>

#include<arpa/inet.h>

#include<netinet/in.h>

#include<sys/ioctl.h>

#include<netdb.h>

#define PORT 8080

#define MAXLINE 2048

int main()

{

    int sockfd,connfd,len;

    char buffer[MAXLINE];

    struct sockaddr\_in servaddr,cliaddr;

    bzero(&servaddr,sizeof(servaddr));

    bzero(&cliaddr,sizeof(cliaddr));

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

    servaddr.sin\_family=AF\_INET;

    servaddr.sin\_port=htons(PORT);

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

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

    listen(sockfd,2);

    len=sizeof(cliaddr);

    connfd=accept(sockfd,(struct sockaddr\*)&cliaddr,&len);

    read(connfd,buffer,MAXLINE);

    printf("%s",buffer);

    FILE \*fp;

    fp=fopen("text2.txt","w");

    fprintf(fp,"%s",buffer);

    printf("File received successfully\n");

    close(sockfd);

}

**client.c**

#include<stdio.h>

#include<stdlib.h>

#include<unistd.h>

#include<string.h>

#include<sys/socket.h>

#include<sys/types.h>

#include<arpa/inet.h>

#include<netinet/in.h>

#include<sys/ioctl.h>

#include<netdb.h>

#define PORT 8080

#define MAXLINE 2048

int main()

{

    int sockfd;

    char buffer[MAXLINE];

    struct sockaddr\_in servaddr;

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

    bzero(&servaddr,sizeof(servaddr));

    servaddr.sin\_family=AF\_INET;

    servaddr.sin\_port=htons(PORT);

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

    connect(sockfd,(struct sockaddr\*)&servaddr,sizeof(servaddr));

    bzero(buffer,sizeof(buffer));

    FILE \*f;

    f=fopen("text1.txt","r");

    for(int i=0;i<MAXLINE;i++)

    {

        buffer[i]=fgetc(f);

        if(buffer[i]==EOF)

            break;

    }

    write(sockfd,buffer,sizeof(buffer));

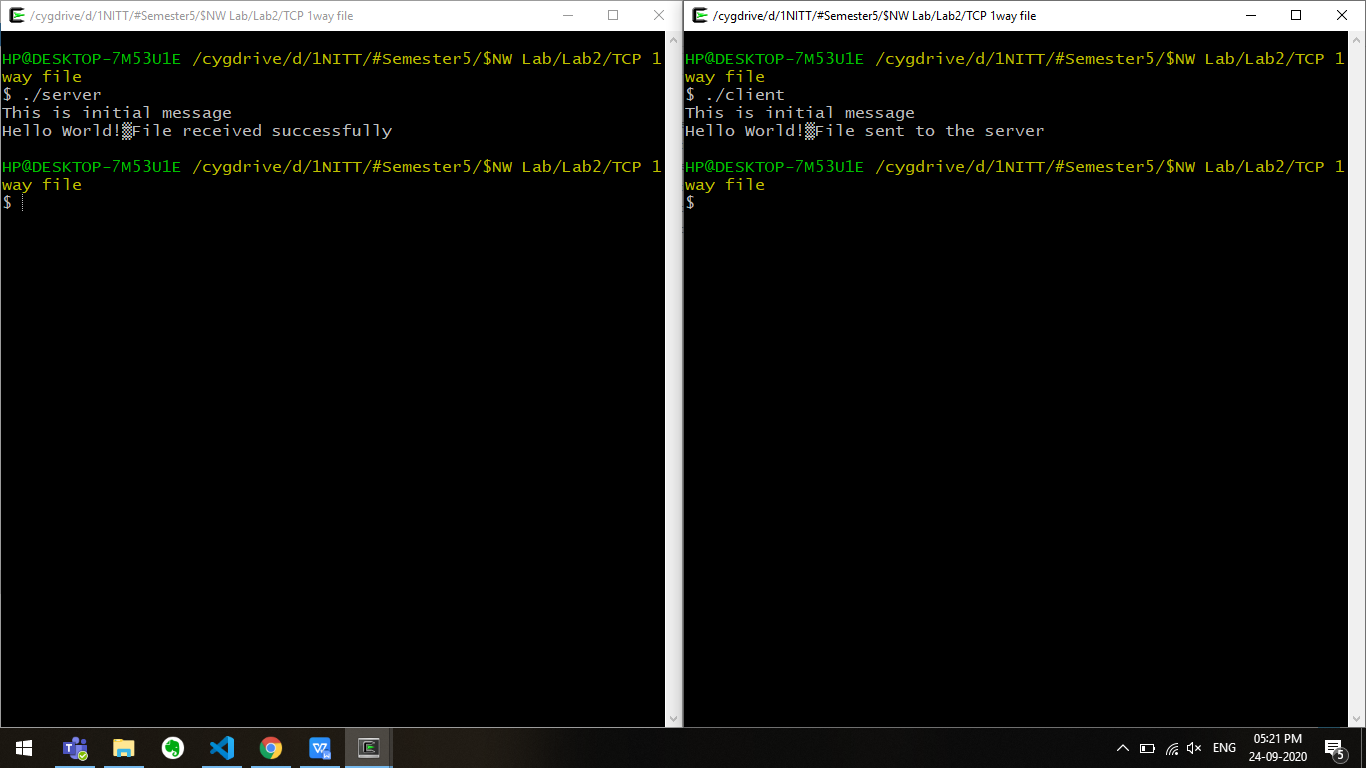
    printf("%s",buffer);

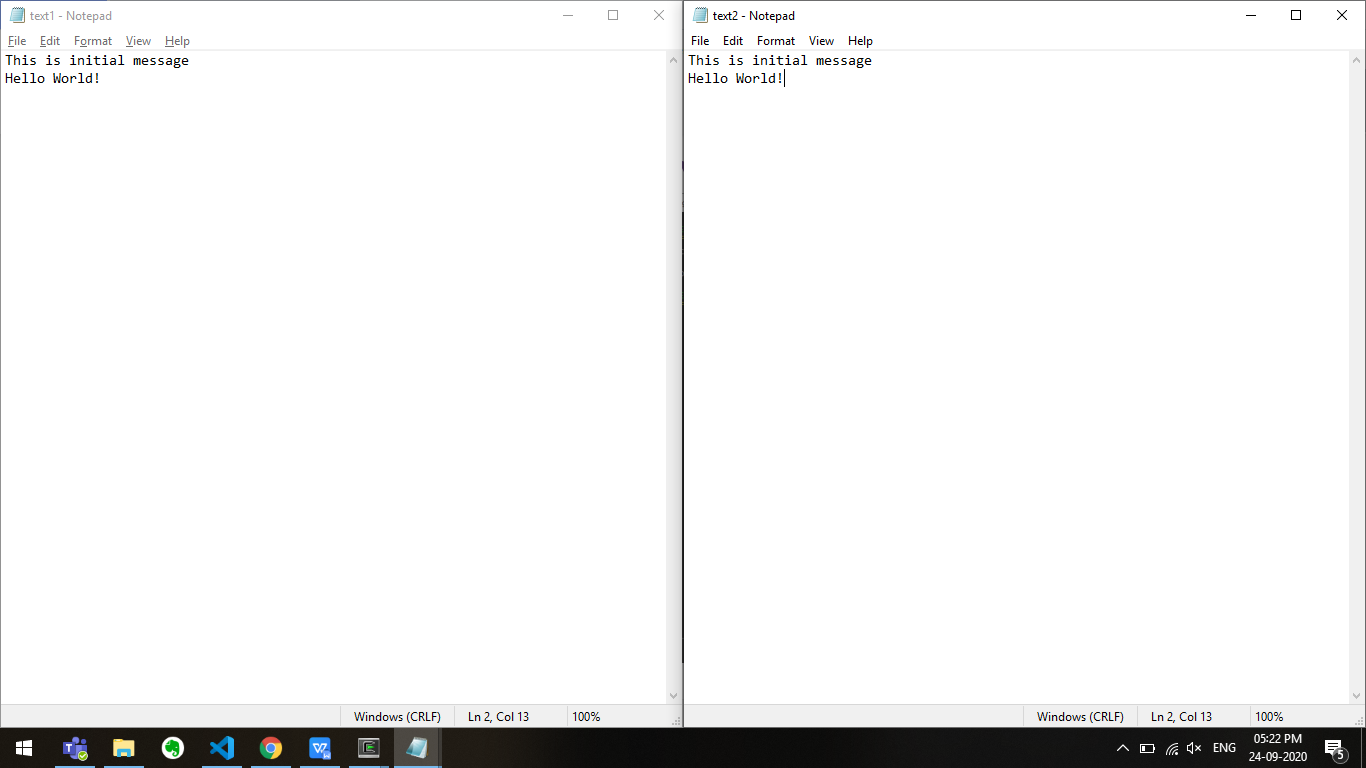
    printf("File sent to the server\n");

    close(sockfd);

}

**Output**





**TCP 2WAY TEXT**

**server.c**

#include <stdio.h>

#include <netdb.h>

#include <unistd.h>

#include <netinet/in.h>

#include <stdlib.h>

#include <string.h>

#include <sys/socket.h>

#include <sys/types.h>

#define MAX 80

#define PORT 8081

#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 acccept failed...\n");

exit(0);

}

else

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

// Function for chatting between client and server

func(connfd);

// After chatting close the socket

close(sockfd);

}

**client.c**

#include <stdio.h>

#include <netdb.h>

#include <unistd.h>

#include <netinet/in.h>

#include <stdlib.h>

#include <string.h>

#include <sys/socket.h>

#include <sys/types.h>

#include <arpa/inet.h>

#define MAX 80

#define PORT 8081

#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;

inet\_pton(AF\_INET, "127.0.0.1", &servaddr.sin\_addr);

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

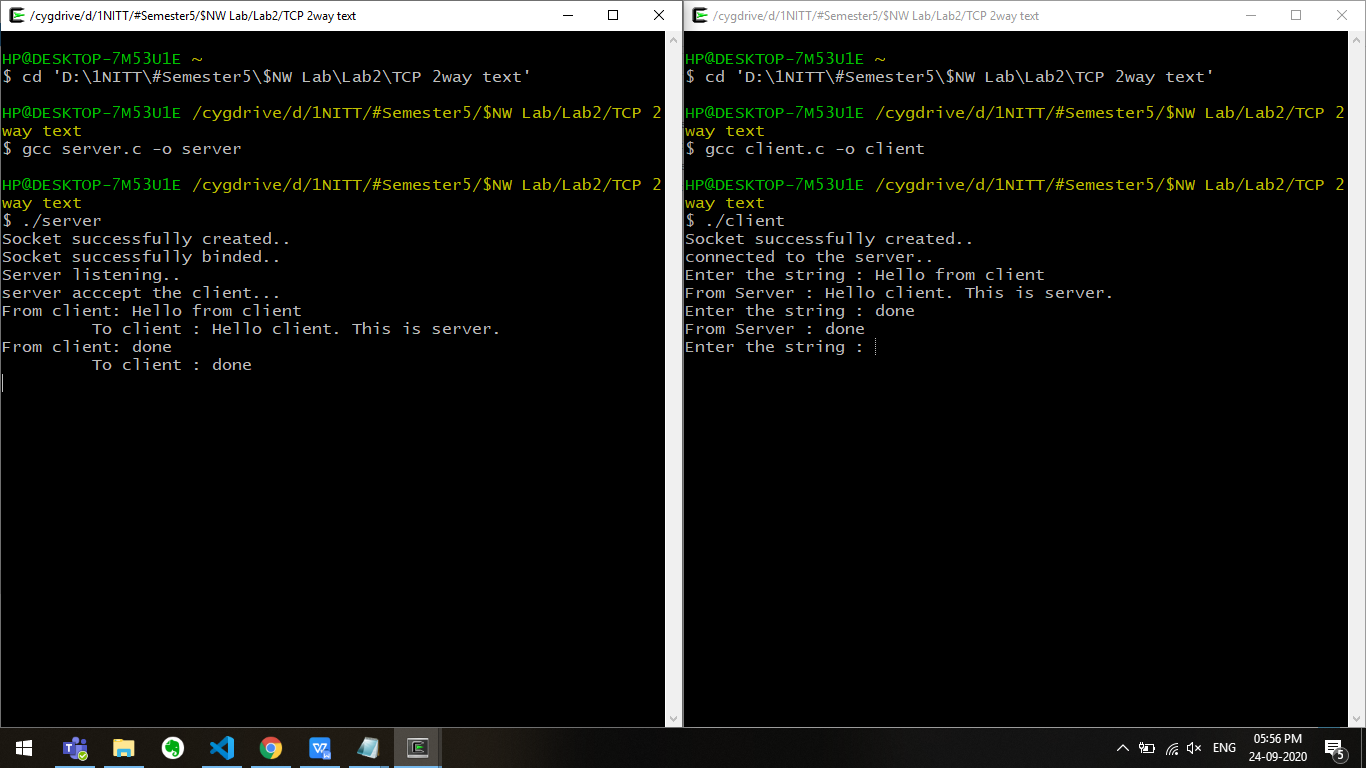
func(sockfd);

// close the socket

close(sockfd);

}

**Output**



**UDP 1WAY TEXT**

**server.c**

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <string.h>

#include <sys/types.h>

#include <sys/socket.h>

#include <arpa/inet.h>

#include <netinet/in.h>

#define PORT 8080

#define MAXLINE 1024

// Driver code

int main() {

int sockfd;

char buffer[MAXLINE];

char \*hello = "Hello from server";

struct sockaddr\_in servaddr, cliaddr;

// Creating socket file descriptor

if ( (sockfd = socket(AF\_INET, SOCK\_DGRAM, 0)) < 0 ) {

perror("socket creation failed");

exit(EXIT\_FAILURE);

}

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

memset(&cliaddr, 0, sizeof(cliaddr));

// Filling server information

servaddr.sin\_family = AF\_INET; // IPv4

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

servaddr.sin\_port = htons(PORT);

// Bind the socket with the server address

if ( bind(sockfd, (const struct sockaddr \*)&servaddr,sizeof(servaddr)) < 0 )

{

perror("bind failed");

exit(EXIT\_FAILURE);

}

int len, n;

len = sizeof(cliaddr); //len is value/resuslt

n = recvfrom(sockfd, (char \*)buffer, MAXLINE, 0, ( struct sockaddr \*) &cliaddr,&len);

buffer[n] = '\0';

printf("Client : %s\n", buffer);

sendto(sockfd, (const char \*)hello, strlen(hello),0, (const struct sockaddr \*) &cliaddr,len);

printf("Hello message sent.\n");

return 0;

}

**client.c**

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <string.h>

#include <sys/types.h>

#include <sys/socket.h>

#include <arpa/inet.h>

#include <netinet/in.h>

#define MSG\_CONFIRM  0x800

#define PORT 8080

#define MAXLINE 1024

// Driver code

int main() {

int sockfd;

char buffer[MAXLINE];

char \*hello = "Hello from client";

struct sockaddr\_in servaddr;

// Creating socket file descriptor

if ( (sockfd = socket(AF\_INET, SOCK\_DGRAM, 0)) < 0 ) {

perror("socket creation failed");

exit(EXIT\_FAILURE);

}

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

// Filling server information

servaddr.sin\_family = AF\_INET;

servaddr.sin\_port = htons(PORT);

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

int n, len=sizeof(servaddr);

sendto(sockfd, (const char \*)hello, strlen(hello), 0, (const struct sockaddr \*) &servaddr,len);

printf("Hello message sent.\n");

n = recvfrom(sockfd, (char \*)buffer, MAXLINE,0, (struct sockaddr \*) &servaddr,&len);

buffer[n] = '\0';

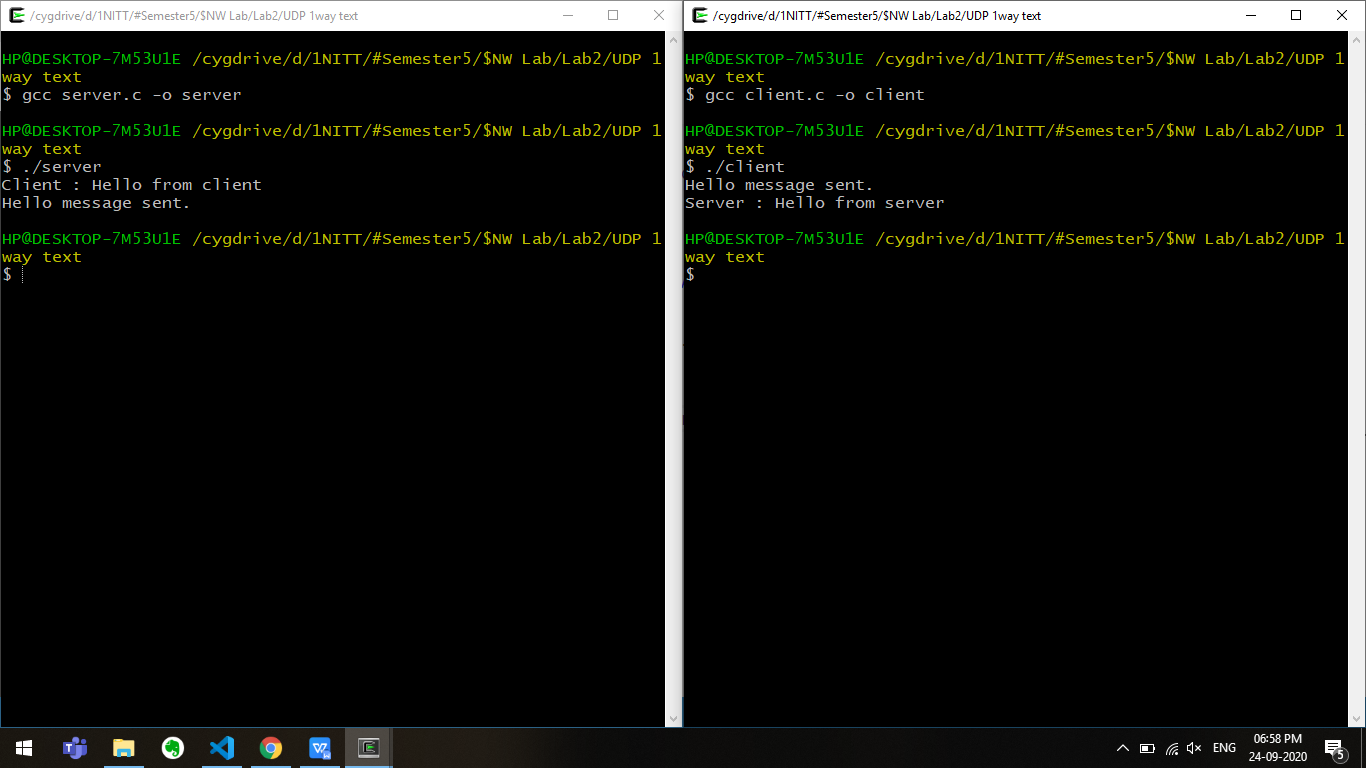
printf("Server : %s\n", buffer);

close(sockfd);

return 0;

}

**Output**



**UDP 1WAY FILE**

**server.c**

#include <stdio.h>

#include <string.h>

#include <fcntl.h>

#include <arpa/inet.h>

#include<unistd.h>

#include <sys/socket.h>

#define SERV\_PORT 49312

#define MAXLINE 1024

char \*END\_FLAG = "================END";

void run(int sockfd, struct sockaddr \*cliaddr, socklen\_t clilen)

{

int n, fd;

socklen\_t len;

char buf[MAXLINE];

len = clilen;

fd = open("server.txt", O\_RDWR | O\_CREAT, 0666);

while ((n = recvfrom(sockfd, buf, MAXLINE, 0, cliaddr, &len))) {

buf[n] = 0;

if (!(strcmp(buf, END\_FLAG))) {

break;

}

write(fd, buf, n);

}

close(fd);

}

int main(int argc, char \*\*argv)

{

int sockfd;

struct sockaddr\_in servaddr, cliaddr;

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

bzero(&servaddr, sizeof(servaddr));

servaddr.sin\_family = AF\_INET;

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

servaddr.sin\_port = htons(SERV\_PORT);

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

run(sockfd, (struct sockaddr \*) &cliaddr, sizeof(cliaddr));

printf("File received successfully\n");

return 0;

}

**client.c**

#include<stdio.h>

#include<stdlib.h>

#include <unistd.h>

#include <string.h>

#include <arpa/inet.h>

#include <sys/socket.h>

#include <fcntl.h>

#define SERV\_PORT 49312

#define MAXLINE 1024

char \*END\_FLAG = "================END";

int main(int argc, char \*\*argv)

{

int sockfd, n, fd;

struct sockaddr\_in servaddr;

char buf[MAXLINE];

char \*target, \*path;

bzero(&servaddr, sizeof(servaddr));

servaddr.sin\_family = AF\_INET;

servaddr.sin\_port = htons(SERV\_PORT);

inet\_pton(AF\_INET, "127.0.0.1", &servaddr.sin\_addr);

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

fd = open("client.txt", O\_RDONLY);

while ((n = read(fd, buf, MAXLINE)) > 0) {

sendto(sockfd, buf, n, 0, (struct sockaddr \*) &servaddr,

sizeof(servaddr));

}

sendto(sockfd, END\_FLAG, strlen(END\_FLAG), 0, (struct sockaddr \*)

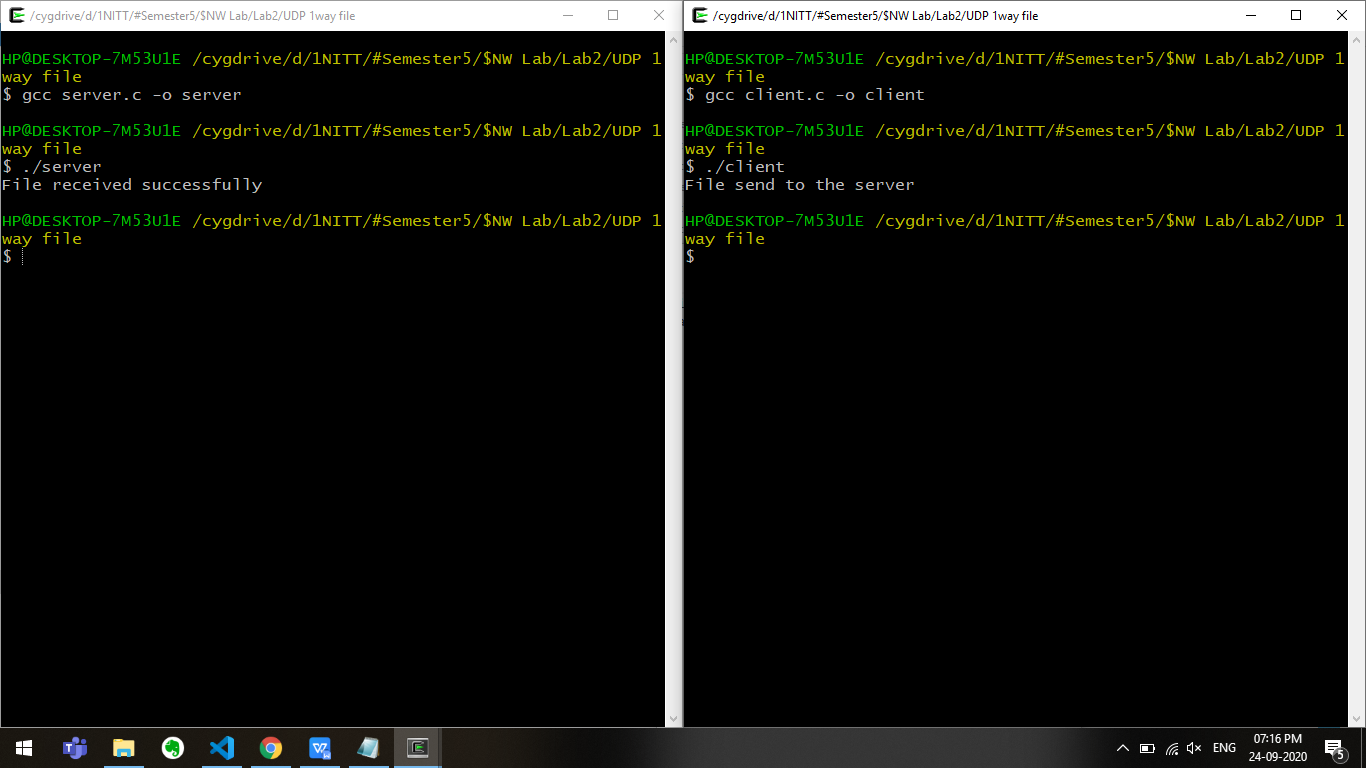
&servaddr, sizeof(servaddr));

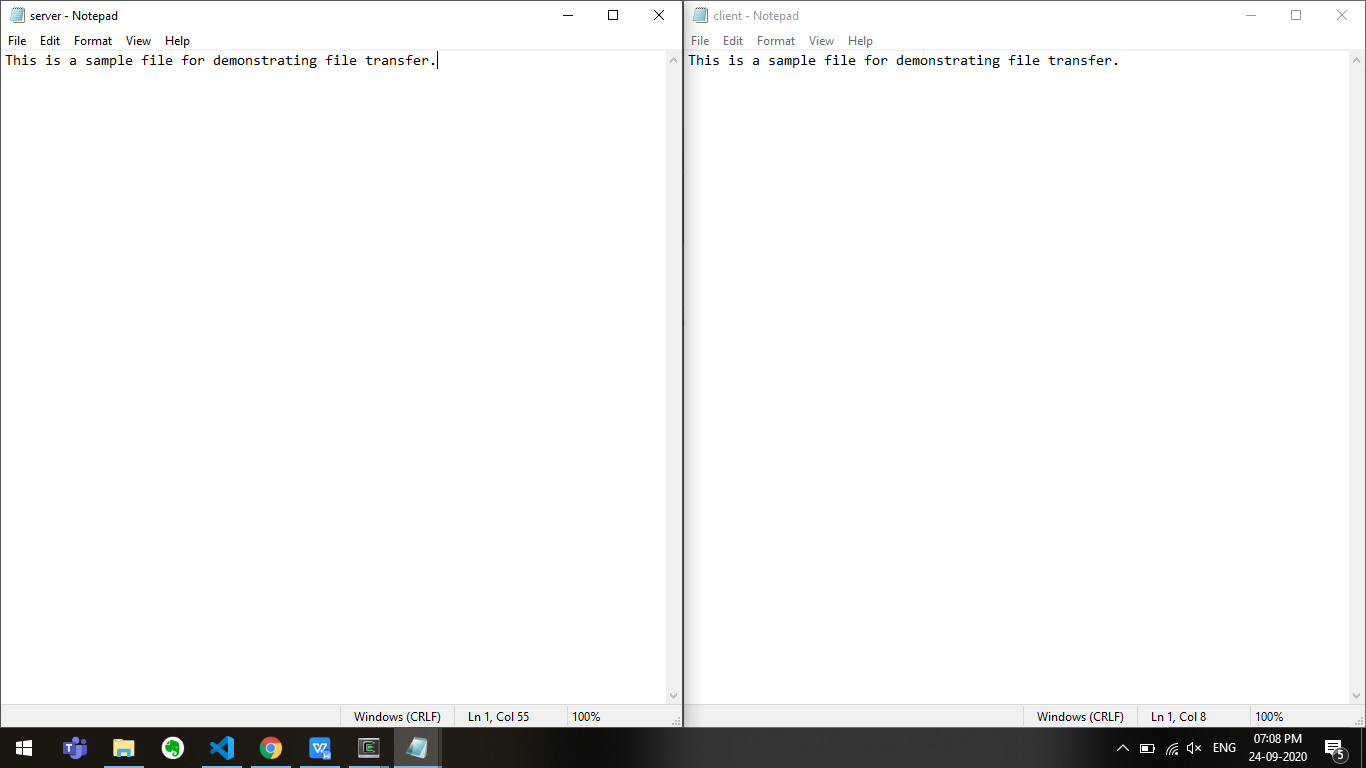
printf("File send to the server\n");

return 0;

}

**Output**





**UDP 2WAY TEXT**

**server.c**

#include<stdio.h>

#include<netinet/in.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<netdb.h>

#include<string.h>

#include<stdlib.h>

#include<unistd.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 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.c**

#include<sys/socket.h>

#include<netdb.h>

#include<string.h>

#include<stdlib.h>

#include<stdio.h>

#include<unistd.h>

#include <sys/types.h>

#include <arpa/inet.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;

inet\_pton(AF\_INET, "127.0.0.1", &servaddr.sin\_addr);

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);

}

**Output**

