

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;
    }

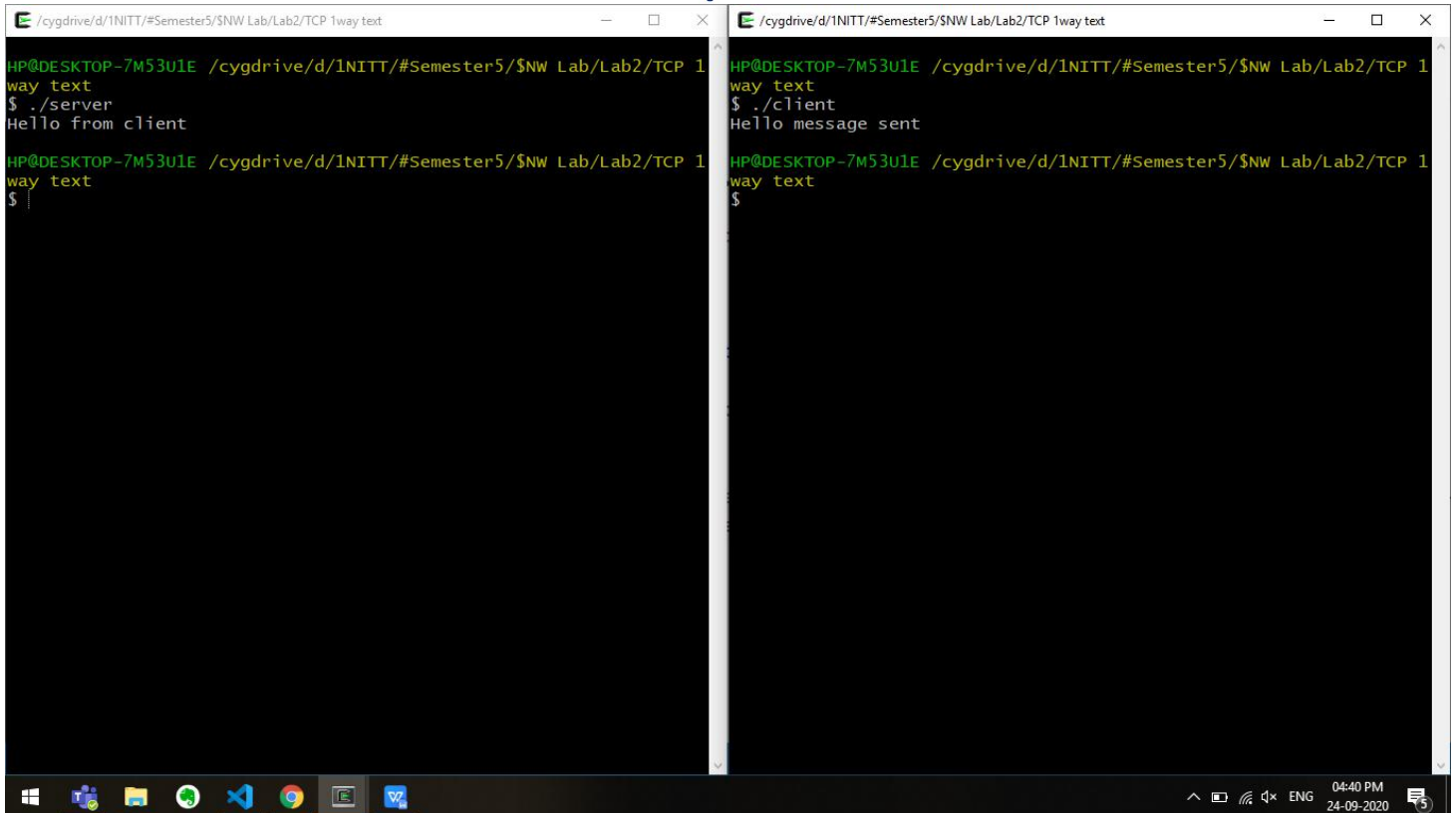
    serv_addr.sin_family = AF_INET;
    serv_addr.sin_port = htons(PORT);

    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



```
HP@DESKTOP-7M53U1E /cygdrive/d/1NITT/#Semester5/$NW Lab/Lab2/TCP 1way text
$ ./server
Hello from client

HP@DESKTOP-7M53U1E /cygdrive/d/1NITT/#Semester5/$NW Lab/Lab2/TCP 1way text
$
```

```
HP@DESKTOP-7M53U1E /cygdrive/d/1NITT/#Semester5/$NW Lab/Lab2/TCP 1way text
$ ./client
Hello message sent

HP@DESKTOP-7M53U1E /cygdrive/d/1NITT/#Semester5/$NW Lab/Lab2/TCP 1way text
$
```

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);
}

```

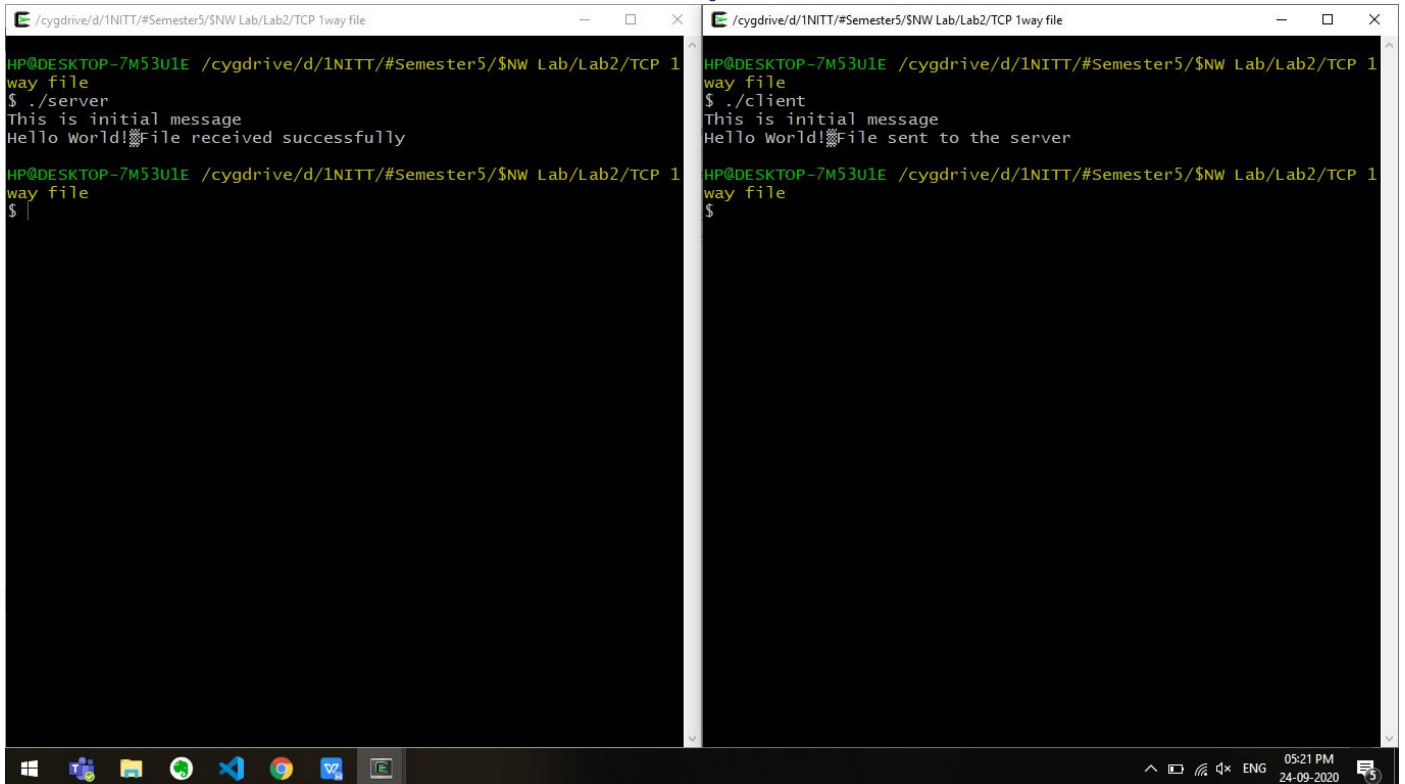
client.c

```

#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

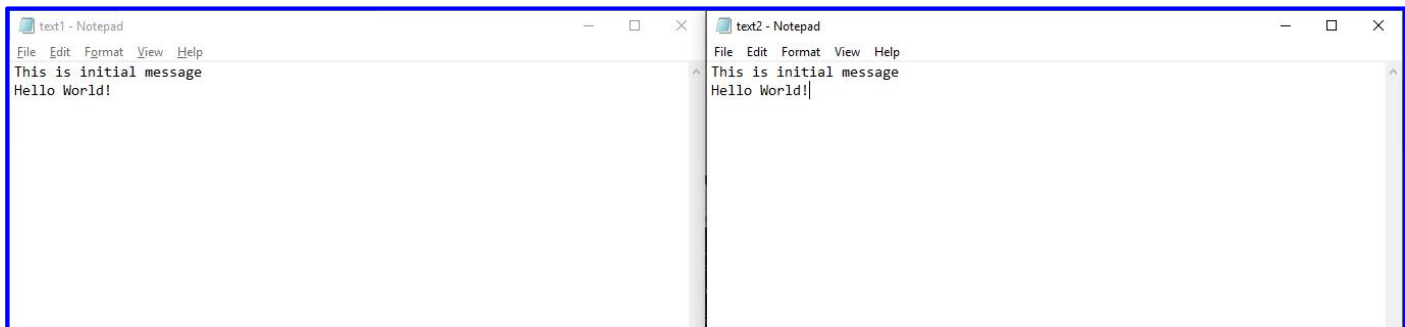


```
HP@DESKTOP-7M53U1E /cygdrive/d/1NITT/#Semester5/$NW Lab/Lab2/TCP 1 way file
$ ./server
This is initial message
Hello World!File received successfully

HP@DESKTOP-7M53U1E /cygdrive/d/1NITT/#Semester5/$NW Lab/Lab2/TCP 1 way file
$ |

HP@DESKTOP-7M53U1E /cygdrive/d/1NITT/#Semester5/$NW Lab/Lab2/TCP 1 way file
$ ./client
This is initial message
Hello World!File sent to the server

HP@DESKTOP-7M53U1E /cygdrive/d/1NITT/#Semester5/$NW Lab/Lab2/TCP 1 way file
$ |
```



```
text1 - Notepad
File Edit Format View Help
This is initial message
Hello World!

text2 - Notepad
File Edit Format View Help
This is initial message
Hello World!
```

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 accept failed...\n");
exit(0);
}
else
printf("server accpet 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 ((strcmp(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

The image displays two terminal windows side-by-side, showing the execution of a TCP 2-way communication program. The left window is the server's terminal, and the right window is the client's terminal.

Server Terminal Output:

```

HP@DESKTOP-7M53U1E ~
$ cd 'D:\INITT\#Semester5\SNW Lab\Lab2\TCP 2way text'
HP@DESKTOP-7M53U1E /cygdrive/d/INITT/#Semester5/SNW Lab/Lab2/TCP 2
way text
$ gcc server.c -o server
HP@DESKTOP-7M53U1E /cygdrive/d/INITT/#Semester5/SNW Lab/Lab2/TCP 2
way text
$ ./server
Socket successfully created..
Socket successfully binded..
Server listening..
server accept the client..
From client: Hello from client
To client : Hello client. This is server.
From client: done
To client : done

```

Client Terminal Output:

```

HP@DESKTOP-7M53U1E ~
$ cd 'D:\INITT\#Semester5\SNW Lab\Lab2\TCP 2way text'
HP@DESKTOP-7M53U1E /cygdrive/d/INITT/#Semester5/SNW Lab/Lab2/TCP 2
way text
$ gcc client.c -o client
HP@DESKTOP-7M53U1E /cygdrive/d/INITT/#Semester5/SNW Lab/Lab2/TCP 2
way text
$ ./client
Socket successfully created..
connected to the server..
Enter the string : Hello from client
From Server : Hello client. This is server.
Enter the string : done
From Server : done
Enter the string : |

```


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

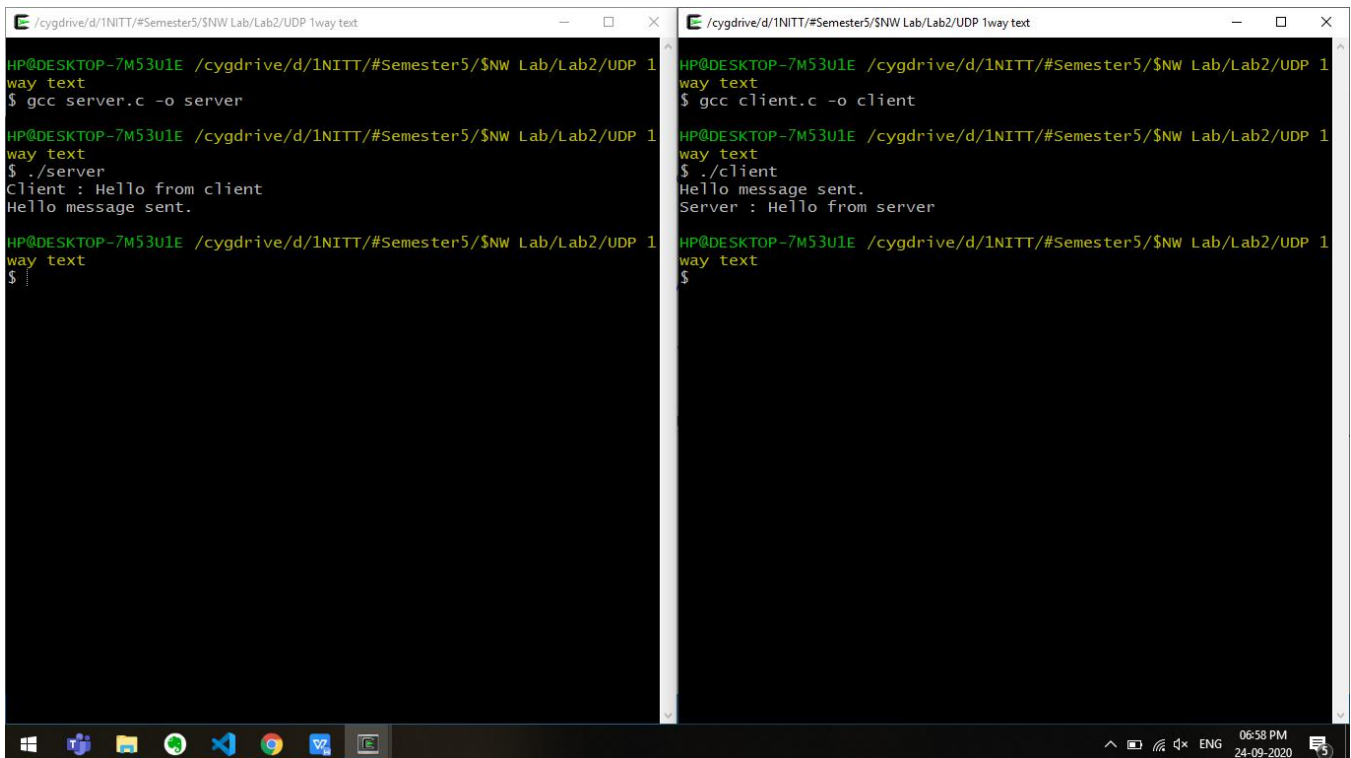
    n = recvfrom(sockfd, (char *)buffer, MAXLINE, 0, ( struct sockaddr *) &cliaddr,&len);
    buffer[n] = '\0';
    printf("Client : %s\n", buffer);
    sendto(sockfd, (const char *)hello, strlen(hello),0, (const struct sockaddr *) &cliaddr,le
n);
    printf("Hello message sent.\n");
    return 0;
}
```

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



```
HP@DESKTOP-7M53U1E /cygdrive/d/1NITT/#Semester5/$NW Lab/Lab2/UDP 1way text
$ gcc server.c -o server
HP@DESKTOP-7M53U1E /cygdrive/d/1NITT/#Semester5/$NW Lab/Lab2/UDP 1way text
$ ./server
Client : Hello from client
Hello message sent.
HP@DESKTOP-7M53U1E /cygdrive/d/1NITT/#Semester5/$NW Lab/Lab2/UDP 1way text
$

HP@DESKTOP-7M53U1E /cygdrive/d/1NITT/#Semester5/$NW Lab/Lab2/UDP 1way text
$ gcc client.c -o client
HP@DESKTOP-7M53U1E /cygdrive/d/1NITT/#Semester5/$NW Lab/Lab2/UDP 1way text
$ ./client
Hello message sent.
Server : Hello from server
HP@DESKTOP-7M53U1E /cygdrive/d/1NITT/#Semester5/$NW Lab/Lab2/UDP 1way text
$
```

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 clien)
{
    int n, fd;
    socklen_t len;
    char buf[MAXLINE];
    len = clien;
    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;
}

```

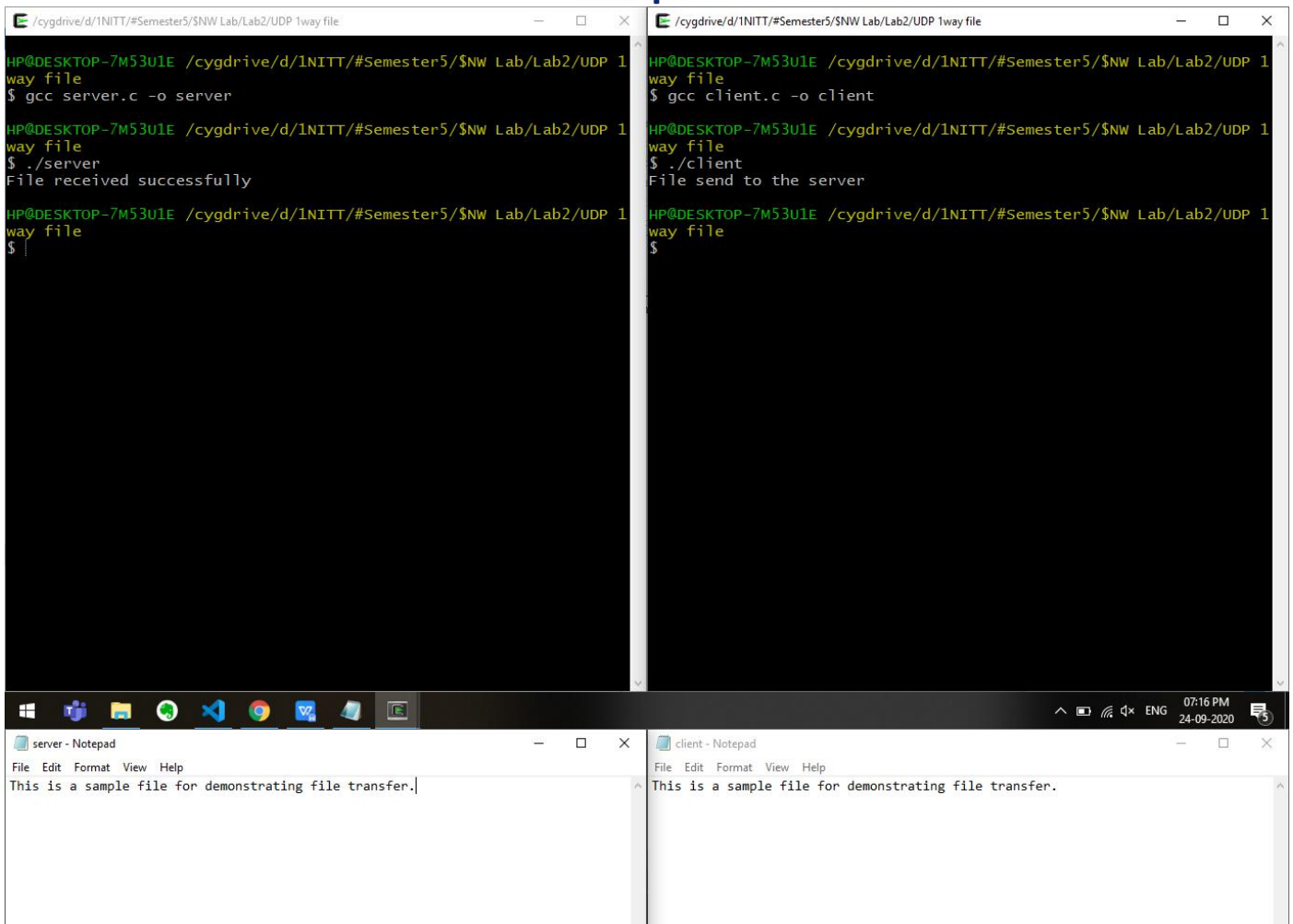
client.c

```

#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



The screenshot displays a Windows desktop environment. At the top, there are two terminal windows side-by-side. The left terminal window shows the following commands and output:

```
HP@DESKTOP-7M53U1E /cygdrive/d/1NITT/#Semester5/$NW Lab/Lab2/UDP 1way file
$ gcc server.c -o server
HP@DESKTOP-7M53U1E /cygdrive/d/1NITT/#Semester5/$NW Lab/Lab2/UDP 1way file
$ ./server
File received successfully
HP@DESKTOP-7M53U1E /cygdrive/d/1NITT/#Semester5/$NW Lab/Lab2/UDP 1way file
$
```

The right terminal window shows the following commands and output:

```
HP@DESKTOP-7M53U1E /cygdrive/d/1NITT/#Semester5/$NW Lab/Lab2/UDP 1way file
$ gcc client.c -o client
HP@DESKTOP-7M53U1E /cygdrive/d/1NITT/#Semester5/$NW Lab/Lab2/UDP 1way file
$ ./client
File send to the server
HP@DESKTOP-7M53U1E /cygdrive/d/1NITT/#Semester5/$NW Lab/Lab2/UDP 1way file
$
```

Below the terminal windows, there are two Notepad windows. The left Notepad window is titled 'server - Notepad' and contains the text: 'This is a sample file for demonstrating file transfer.' The right Notepad window is titled 'client - Notepad' and also contains the text: 'This is a sample file for demonstrating file transfer.'

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

The image displays two terminal windows side-by-side, showing the execution of a UDP 2-way text communication program. The left window shows the server's output, and the right window shows the client's output.

Left Terminal (Server):

```

HP@DESKTOP-7M53U1E ~
$ cd 'D:\1NITT\#Semester5\SNW Lab\Lab2\UDP 2way text'
HP@DESKTOP-7M53U1E /cygdrive/d/1NITT/#Semester5/$NW Lab/Lab2/UDP 2
$ gcc server.c -o server
HP@DESKTOP-7M53U1E /cygdrive/d/1NITT/#Semester5/$NW Lab/Lab2/UDP 2
way text
$ ./server
Socket successfully created..
Socket successfully binded..
From client Hello World!
To client How are you?
From client I am fine, and you?
To client Good
HP@DESKTOP-7M53U1E /cygdrive/d/1NITT/#Semester5/$NW Lab/Lab2/UDP 2
way text
$

```

Right Terminal (Client):

```

HP@DESKTOP-7M53U1E ~
$ cd 'D:\1NITT\#Semester5\SNW Lab\Lab2\UDP 2way text'
HP@DESKTOP-7M53U1E /cygdrive/d/1NITT/#Semester5/$NW Lab/Lab2/UDP 2
way text
$ gcc client.c -o client
HP@DESKTOP-7M53U1E /cygdrive/d/1NITT/#Semester5/$NW Lab/Lab2/UDP 2
way text
$ ./client
Socket successfully created..
Enter string : Hello World!
From Server : How are you?
Enter string : I am fine, and you?
From Server : Good
Enter string :
HP@DESKTOP-7M53U1E /cygdrive/d/1NITT/#Semester5/$NW Lab/Lab2/UDP 2
way text
$

```