

1) TCP

client.c

```
#include<sys/socket.h>
#include<stdio.h>
#include<string.h>
#include<unistd.h>
#include<arpa/inet.h>

int main()
{
    char buf[100];
    int k, sock_desc;
    struct sockaddr_in client;
    memset(&client, 0, sizeof(client));

    sock_desc = socket(AF_INET, SOCK_STREAM, 0);
    if(sock_desc == -1)
        printf("Error in socket creation\n");

    client.sin_family = AF_INET;
    client.sin_port = 5500;
    client.sin_addr.s_addr = inet_addr("127.0.0.1");

    k = connect(sock_desc, (struct sockaddr*)&client, sizeof(client));
    if(k == -1)
        printf("Error in socket connection\n");

    while(1)
    {
        printf("Enter data to be send to server: ");
        fgets(buf,100,stdin);

        send(sock_desc, buf, 100, 0);
        if(strncmp(buf,"end",3)==0)
            break;

        recv(sock_desc, buf, 100, 0);
        printf("Message got from Server is : %s",buf);
        if(strncmp(buf,"end",3)==0)
            break;
    }

    close(sock_desc);
    return 0;
}
```

server.c

```
#include<sys/socket.h>
#include<stdio.h>
#include<string.h>
#include<unistd.h>
#include<arpa/inet.h>

int main()
{
    char buf[100];
    int k, sock_desc, temp_sock_desc;
    socklen_t len;
    struct sockaddr_in server, client;
    memset(&server, 0, sizeof(server));
    memset(&client, 0, sizeof(client));

    sock_desc = socket(AF_INET, SOCK_STREAM, 0);

    server.sin_family = AF_INET;
    server.sin_port = 5500;
    server.sin_addr.s_addr = inet_addr("127.0.0.1");

    bind(sock_desc, (struct sockaddr*)&server, sizeof(server));
    listen(sock_desc, 20);
    len = sizeof(client);

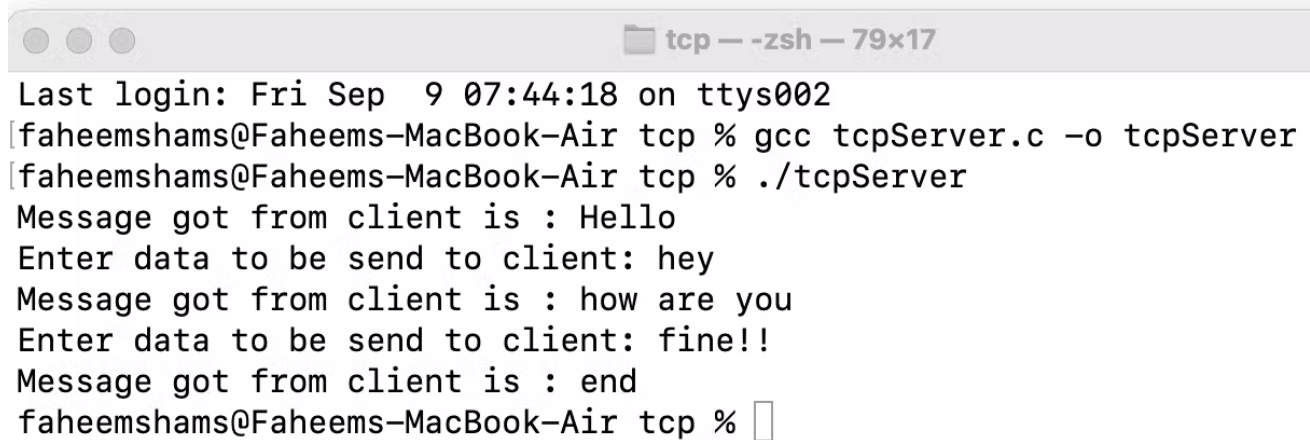
    temp_sock_desc = accept(sock_desc, (struct sockaddr*)&client, &len);

    while(1)
    {
        recv(temp_sock_desc, buf, 100, 0);
        printf("Message got from client is : %s",buf);
        if(strncmp(buf,"end",3)==0)
            break;

        printf("Enter data to be send to client: ");
        fgets(buf, 100, stdin);
        send(temp_sock_desc, buf, 100, 0);
        if(strncmp(buf,"end",3)==0)
            break;
    }
    close(temp_sock_desc);
    return 0;
}
```

OUTPUT

```
[faheemshams@Faheems-MacBook-Air tcp % gcc tcpClient.c -o tcpClient
[faheemshams@Faheems-MacBook-Air tcp % ./tcpClient
Enter data to be send to server: Hello
Message got from Server is : hey
Enter data to be send to server: how are you
Message got from Server is : fine!!
Enter data to be send to server: end
faheemshams@Faheems-MacBook-Air tcp %
```

A screenshot of a macOS terminal window. The title bar shows three window control buttons (red, yellow, green) on the left and a folder icon followed by the text "tcp - zsh - 79x17" on the right. The terminal content shows the execution of a TCP server program, including compilation and running, with messages received from a client.

```
tcp - zsh - 79x17
Last login: Fri Sep  9 07:44:18 on ttys002
[faheemshams@Faheems-MacBook-Air tcp % gcc tcpServer.c -o tcpServer
[faheemshams@Faheems-MacBook-Air tcp % ./tcpServer
Message got from client is : Hello
Enter data to be send to client: hey
Message got from client is : how are you
Enter data to be send to client: fine!!
Message got from client is : end
faheemshams@Faheems-MacBook-Air tcp %
```

2) UDP

client.c

```
#include<stdio.h>
#include<sys/socket.h>
#include<string.h>
#include<arpa/inet.h>
#include<unistd.h>

int main()
{
    char buf[100];
    int sock_desc;
    struct sockaddr_in client;
    socklen_t len;

    sock_desc = socket(AF_INET, SOCK_DGRAM, 0);
    bzero(&client, sizeof(client));

    client.sin_family = AF_INET;
    client.sin_port = 5656;
    client.sin_addr.s_addr = inet_addr("127.0.0.1");

    len = sizeof(client);

    while(1)
    {
        printf("Enter data to be send to Server : ");
        fgets(buf,100,stdin);

        sendto(sock_desc, buf, 100, 0, (struct sockaddr*)&client, len);
        if(strncmp(buf,"end",3) == 0)
            break;
        recvfrom(sock_desc, buf, 100, 0, (struct sockaddr*)&client, &len);
        printf("Message got from Server : %s",buf);
        if(strncmp(buf,"end",3) == 0)
            break;
    }

    close(sock_desc);
    return 0;
}
```

server.c

```

#include<stdio.h>
#include<string.h>
#include<sys/socket.h>
#include<arpa/inet.h>
#include<unistd.h>

int main()
{
    char buf[100];
    int sock_desc;
    struct sockaddr_in server, client;
    socklen_t len;

    sock_desc = socket(AF_INET, SOCK_DGRAM, 0);
    bzero(&server, sizeof(server));           //instead of memset, it fills with 0 default

    server.sin_family = AF_INET;
    server.sin_port = 5656;
    server.sin_addr.s_addr = inet_addr("127.0.0.1");

    bind(sock_desc, (struct sockaddr*)&server, sizeof(server));
    len = sizeof(client);

    while(1)
    {
        recvfrom(sock_desc, buf, sizeof(buf), 0, (struct sockaddr*)&client, &len);
        printf("Message got from Client : %s", buf);
        if(strncmp(buf, "end", 3) == 0)
            break;
        printf("Enter data to be send to Client : ");
        fgets(buf, 100, stdin);

        sendto(sock_desc, buf, 100, 0, (struct sockaddr*)&client, len);
        if(strncmp(buf, "end", 3) == 0)
            break;
    }

    close(sock_desc);
}

```

OUTPUT

```
[faheemshams@Faheems-MacBook-Air udp % gcc udpClient.c -o udpClient
[faheemshams@Faheems-MacBook-Air udp % ./udpClient
Enter data to be send to Server : hello
Message got from Server : hey
Enter data to be send to Server : good
Message got from Server : end
faheemshams@Faheems-MacBook-Air udp % █
```

```
[faheemshams@Faheems-MacBook-Air udp % gcc udpServer.c -o udpServer
[faheemshams@Faheems-MacBook-Air udp % ./udpServer
Message got from Client : hello
Enter data to be send to Client : hey
Message got from Client : good
Enter data to be send to Client : end
faheemshams@Faheems-MacBook-Air udp % █
```

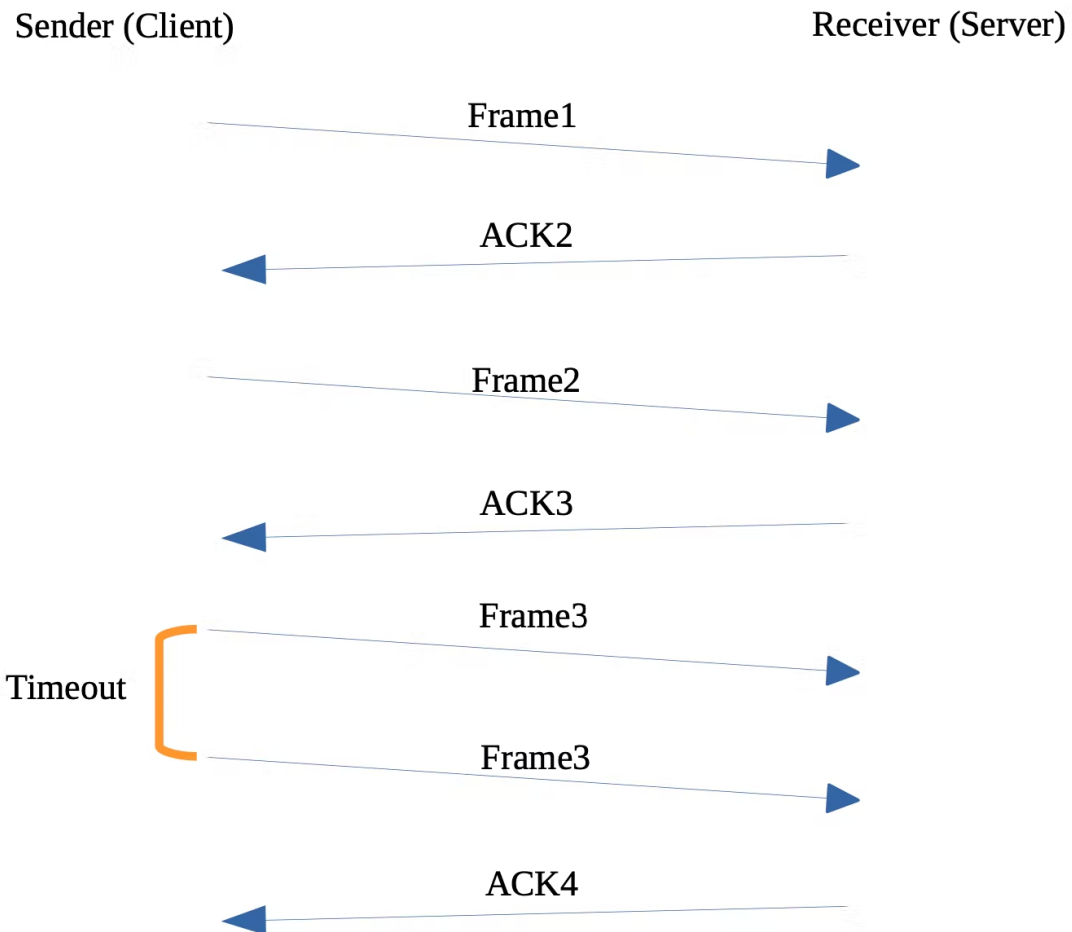
3) STOP & WAIT

Server

Input: Number of frames: 10
Number of lost frames: 2
Lost frames: 3 5

Client

Input: Number of frames: 10



client.c (sender)

```
#include<stdio.h>
#include<sys/socket.h>
#include<string.h>
#include<arpa/inet.h>
```

```

#include<stdlib.h>
#include<unistd.h>

int main()
{
char buf[10],frameNum[10];
int sock_desc;
struct sockaddr_in client;
socklen_t len;

sock_desc = socket(AF_INET, SOCK_DGRAM, 0);
bzero(&client, sizeof(client));

client.sin_family = AF_INET;
client.sin_port = 5656;
client.sin_addr.s_addr = inet_addr("127.0.0.1");
len = sizeof(client);

int n, i=1 ; //i --> frame number
printf("Enter the number of frames\n");
scanf("%d",&n);
char delim[] = "ACK";

while(1)
{
strcpy(buf,"Frame");
sprintf(frameNum,"%d",i); //frame number to string
strcat(buf,frameNum);
sendto(sock_desc, buf, 10, 0, (struct sockaddr*)&client, len);
printf("%s send!!\n\n",buf);
if(i==n)
break;

recvfrom(sock_desc, buf, 10, 0, (struct sockaddr*)&client, &len);
char *ptr = strtok(buf, delim); //For eg, ptr points to 2 in ACK2
int check = atoi(ptr);
if(check == i+1)
{
printf("Acknowledgement got from Server : %s\n",buf);
printf("Acknowledgement success, next packet ready to sent\n");
++i;
}
else if(check == 0)
{
printf("Frame is Missing, resending frame%d\n",i);
continue;
}
}
}

```



```
close(sock_desc);
return 0;
}
```

server.c (receiver)

```
#include<stdio.h>
#include<string.h>
#include<sys/socket.h>
#include<arpa/inet.h>
#include<unistd.h>
#include<stdlib.h>

int main()
{
    char buf[10], ackNum[10], delim[] = "Frame";
    int sock_desc;
    struct sockaddr_in server, client;
    socklen_t len;

    sock_desc = socket(AF_INET, SOCK_DGRAM, 0);
    bzero(&server, sizeof(server)); //instead of memset, it fills with 0

    server.sin_family = AF_INET;
    server.sin_port = 5656;
    server.sin_addr.s_addr = inet_addr("127.0.0.1");

    bind(sock_desc, (struct sockaddr*)&server, sizeof(server));
    len = sizeof(client);

    int n, lostFrames;
    printf("Enter the number of frames\n");
    scanf("%d", &n);
    printf("Enter number of lost frames\n");
    scanf("%d", &lostFrames);

    int lost[lostFrames];
    printf("Enter lost frames\n");
    for(int i=0; i<lostFrames; ++i)
        scanf("%d", &lost[i]);

    int i = 2;
    while(i <= n)
    {
        int flag = 0;
        recvfrom(sock_desc, buf, sizeof(buf), 0, (struct sockaddr*)&client, &len);
        char *ptr = strtok(buf, delim);
        for(int j=0; j<lostFrames; ++j)
```

```

{
    int check = atoi(ptr);
    if(lost[j] == check) //checking frame is in lostframe or not
    {
        strcpy(buf,"ACK0");
        lost[j] = 0;
        ++flag;
        break;
    }
}

if(flag == 0) //frame not in lost frame, so send ack
{
    strcpy(buf,"ACK");
    sprintf(ackNum,"%d",i++); //ack number to string
    strcat(buf,ackNum);
}
sendto(sock_desc, buf, 10, 0, (struct sockaddr*)&client, len);
printf("Acknowledgment sent : %s\n\n",buf);
}

close(sock_desc);
}

```

OUTPUT

```
[faheemshams@Faheems-MacBook-Air FlowControl % gcc client.c -o client
[faheemshams@Faheems-MacBook-Air FlowControl % ./client
Enter the number of frames
6
Frame1 send!!
```

```
Acknowledgement got from Server : ACK2
Acknowledgement success, next packet ready to sent
Frame2 send!!
```

```
Acknowledgement got from Server : ACK3
Acknowledgement success, next packet ready to sent
Frame3 send!!
```

```
Frame is Missing, resending frame3
Frame3 send!!
```

```
Acknowledgement got from Server : ACK4
Acknowledgement success, next packet ready to sent
Frame4 send!!
```

```
Acknowledgement got from Server : ACK5
Acknowledgement success, next packet ready to sent
Frame5 send!!
```

```
Frame is Missing, resending frame5
Frame5 send!!
```

```
Acknowledgement got from Server : ACK6
Acknowledgement success, next packet ready to sent
Frame6 send!!
```

```
[faheemshams@Faheems-MacBook-Air FlowControl % gcc server.c -o server
[faheemshams@Faheems-MacBook-Air FlowControl % ./server
Enter the number of frames
6
Enter number of lost frames
2
Enter lost frames
3 5
Acknowledgment sent : ACK2

Acknowledgment sent : ACK3

Acknowledgment sent : ACK0

Acknowledgment sent : ACK4

Acknowledgment sent : ACK5

Acknowledgment sent : ACK0

Acknowledgment sent : ACK6

faheemshams@Faheems-MacBook-Air FlowControl %
```

4) FTP

client

```
#include<sys/socket.h>
#include<stdio.h>
#include<string.h>
#include<unistd.h>
#include<arpa/inet.h>
#include<sys/stat.h>
#include<stdlib.h>
#include<fcntl.h>

int main()
{
    char buf[100], temp[100], fname[20];
    FILE *fp;
    int k, sock_desc;
    struct sockaddr_in client;

    sock_desc = socket(AF_INET, SOCK_STREAM, 0);
    if(sock_desc == -1)
        printf("Error in socket creation\n");

    bzero(&client, sizeof(client));
    client.sin_family = AF_INET;
    client.sin_port = 5050;
    client.sin_addr.s_addr = inet_addr("127.0.0.1");

    k = connect(sock_desc, (struct sockaddr*)&client, sizeof(client));
    if(k == -1)
        printf("Error in socket connection\n");

    while(1)
    {
        printf("Enter the command\n");
        fgets(buf,100,stdin);

        if(strncmp(buf,"bye",3) == 0)
        {
            send(sock_desc, buf, sizeof(buf),0);
            break;
        }
        char *token = strtok(buf, " ");
        token = strtok(NULL, "\n");
        strcpy(fname, token);
        send(sock_desc, buf, sizeof(buf),0);           //if input is get hello.txt, now buf = get
```

```

if(strncmp(buf,"get",3)==0)
{
send(sock_desc, fname, sizeof(fname), 0);
recv(sock_desc, buf, sizeof(buf), 0);           //checks if file is present or not

if(strncmp(buf, "notfound", 8) != 0)
{
fp = fopen(fname, "w");
while(1)
{
recv(sock_desc, buf, sizeof(buf), 0);
if(strncmp(buf,"completed",9) == 0)
break;
fputs(buf, fp);
}
printf("file %s downloaded from server\n\n",fname);
fclose(fp);
}
}
else if(strncmp(buf,"put",3)==0)
{
send(sock_desc, fname, sizeof(fname), 0);
fp = fopen(fname, "r");
if(fp == NULL)
{
printf("Filename %s not found\n\n",fname);
send(sock_desc, "notfound", 8, 0);
}
else
{
send(sock_desc, "found", 5, 0);
while(fgets(buf, sizeof(buf), fp))
{
send(sock_desc, buf, sizeof(buf), 0);
}
send(sock_desc, "completed", 9, 0);
printf("File %s uploaded successfully\n\n",fname);
}
fclose(fp);
}
else
printf("Invalid input\n\n");
close(sock_desc);
close(k);
return 0;
}

```

server

```
#include<stdio.h>
#include<sys/socket.h>
#include<unistd.h>
#include<arpa/inet.h>
#include<string.h>
#include<fcntl.h>

int main()
{
    FILE *fp;
    char buf[100], fname[20];
    int k, sock_desc, temp_sock_desc;

    struct sockaddr_in server, client;
    socklen_t len;

    sock_desc = socket(AF_INET, SOCK_STREAM, 0);
    bzero(&server, sizeof(server));

    server.sin_family = AF_INET;
    server.sin_port = 5050;
    server.sin_addr.s_addr = inet_addr("127.0.0.1");

    bind(sock_desc, (struct sockaddr*)&server, sizeof(server));
    listen(sock_desc, 5);
    len = sizeof(client);
    temp_sock_desc = accept(sock_desc, (struct sockaddr*)&client, &len);
    while(1)
    {
        recv(temp_sock_desc, buf, 100, 0); //recieves command : get, put or bye
        if(strncmp(buf, "get", 3) == 0)
        {
            recv(temp_sock_desc, fname, 20, 0);
            fp = fopen(fname, "r");
            if(fp == NULL)
            {
                printf("Filename %s not found\n\n", fname);
                send(temp_sock_desc, "notfound", 8, 0);
            }
            else
            {
                send(temp_sock_desc, "found", 5, 0);
                while(fgets(buf, sizeof(buf), fp))
                {
                    send(temp_sock_desc, buf, sizeof(buf), 0);
                }
                printf("File %s transferred to client\n\n", fname);
            }
        }
    }
}
```

```

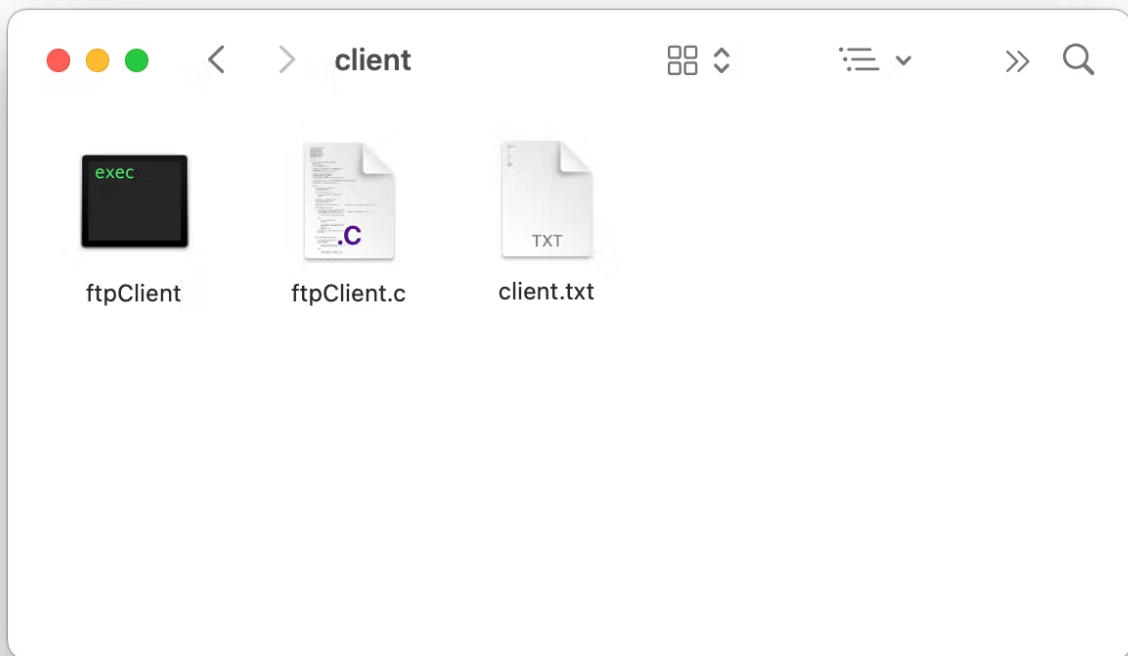
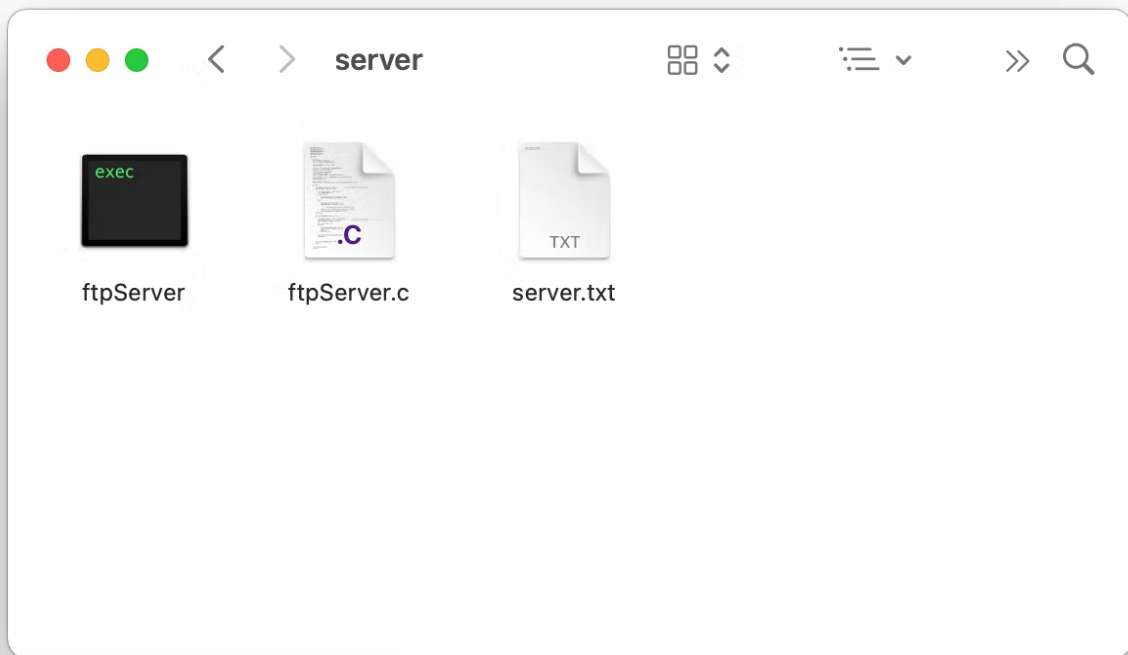
send(temp_sock_desc, "completed", 9, 0);
}
fclose(fp);
}

else if(strncmp(buf, "put", 3) == 0)
{
recv(temp_sock_desc, fname, sizeof(fname), 0);
recv(temp_sock_desc, buf, sizeof(buf), 0);           //recieves file not found or not
if(strncmp(buf, "notfound", 8) != 0)
{
fp = fopen(fname, "w");
while(1)
{
recv(temp_sock_desc, buf, sizeof(buf), 0);
if(strncmp(buf, "completed", 9) == 0)
break;
fputs(buf, fp);
}
printf("File %s recieved from client\n\n", fname);
fclose(fp);
}}

else if(strncmp(buf, "bye", 3) == 0)
break;
}
close(temp_sock_desc);
return 0;
}

```

FOLDERS BEFORE PROGRAM RUN



OUTPUT

```
• faheemshams@Faheems-MacBook-Air server % gcc ftpServer.c -o ftpServer
• faheemshams@Faheems-MacBook-Air server % ./ftpServer
File server.txt transferred to client

File client.txt recieved from client

Filename hey not found
◦ faheemshams@Faheems-MacBook-Air server % █

• faheemshams@Faheems-MacBook-Air client % gcc ftpClient.c -o ftpClient
• faheemshams@Faheems-MacBook-Air client % ./ftpClient
Enter the command
get server.txt
file server.txt downloaded from server

Enter the command
put client.txt
File client.txt uploaded successfully

Enter the command
get hey
Enter the command
put hey
Filename hey not found

Enter the command
bye
◦ faheemshams@Faheems-MacBook-Air client % █
```

AFTER RUNNING PROGRAM

