

Hello

Input :

hell.c //server file

```
#include<stdio.h>
#include<sys/socket.h>
#include <arpa/inet.h>
#include<string.h>
#include<stdlib.h>
#include <unistd.h> // for close
#define DATA "Hello from server"
int main(int argc, char* argv[])
{
    /*Variables*/
    int sock;
    struct sockaddr_in server;
    int mysock;
    char buffer[1024];
    int rval;
    /*Create Sockets*/
    sock = socket(AF_INET, SOCK_STREAM, 0);
    if(sock<0)
    {
        perror("Failed to create Socket");
        exit(1);
    }
    server.sin_family = AF_INET;
    server.sin_addr.s_addr = INADDR_ANY;
    server.sin_port = htons(5050);

    /*Call Bind*/
    if(bind(sock, (struct sockaddr *)&server, sizeof(server)))
    {
        perror("Bind Failed");
        exit(1);
    }

    /*Listen*/
    listen(sock, 5);

    /*Accept*/
    do
    {
        mysock = accept(sock, (struct sockaddr *) 0, 0);
```

```

        if(mysock == -1)
        {
            perror("Accept Failed");
        }
        else
        {
            memset(buffer, 0, sizeof(buffer));

            //Receive data from the client
            if(recv(mysock, buffer, sizeof(buffer), 0)<0)
            {
                perror("Receive Failed");
                exit(1);
            }
            printf("Received Message: %s\n",buffer);
            printf("Sending Message: %s\n",DATA);

            //Sendinf data to the client
            if(send(mysock,DATA,sizeof(DATA), 0)<0)
            {
                perror("Send Failed");
                close(sock);
                exit(1);
            }
            close(mysock);
        }
    }while(1);
    return 0;
}

```

helloclient.c

```

#include<stdio.h>
#include<sys/socket.h>
#include<netdb.h>
#include<string.h>
#include<stdlib.h>
#include<unistd.h>
#define DATA "Hello from client"
int main(int ssargc, char *argv[])
{
    int sock;
    struct sockaddr_in server;
    struct hostent *hp;
    char buffer[1024];

```

```

//Defining a socket
sock= socket(AF_INET, SOCK_STREAM, 0);
if(sock<0)
{
perror("Socket Failed");
close(sock);
exit(1);
}
server.sin_family = AF_INET;

//Accept command line argument as IP address
hp = gethostbyname(argv[1]);
if(hp==0)
{
perror("gethostbynme Failed");
close(sock);
exit(1);
}

memcpy(&server.sin_addr, hp->h_addr, hp->h_length);
server.sin_port = htons(5050);

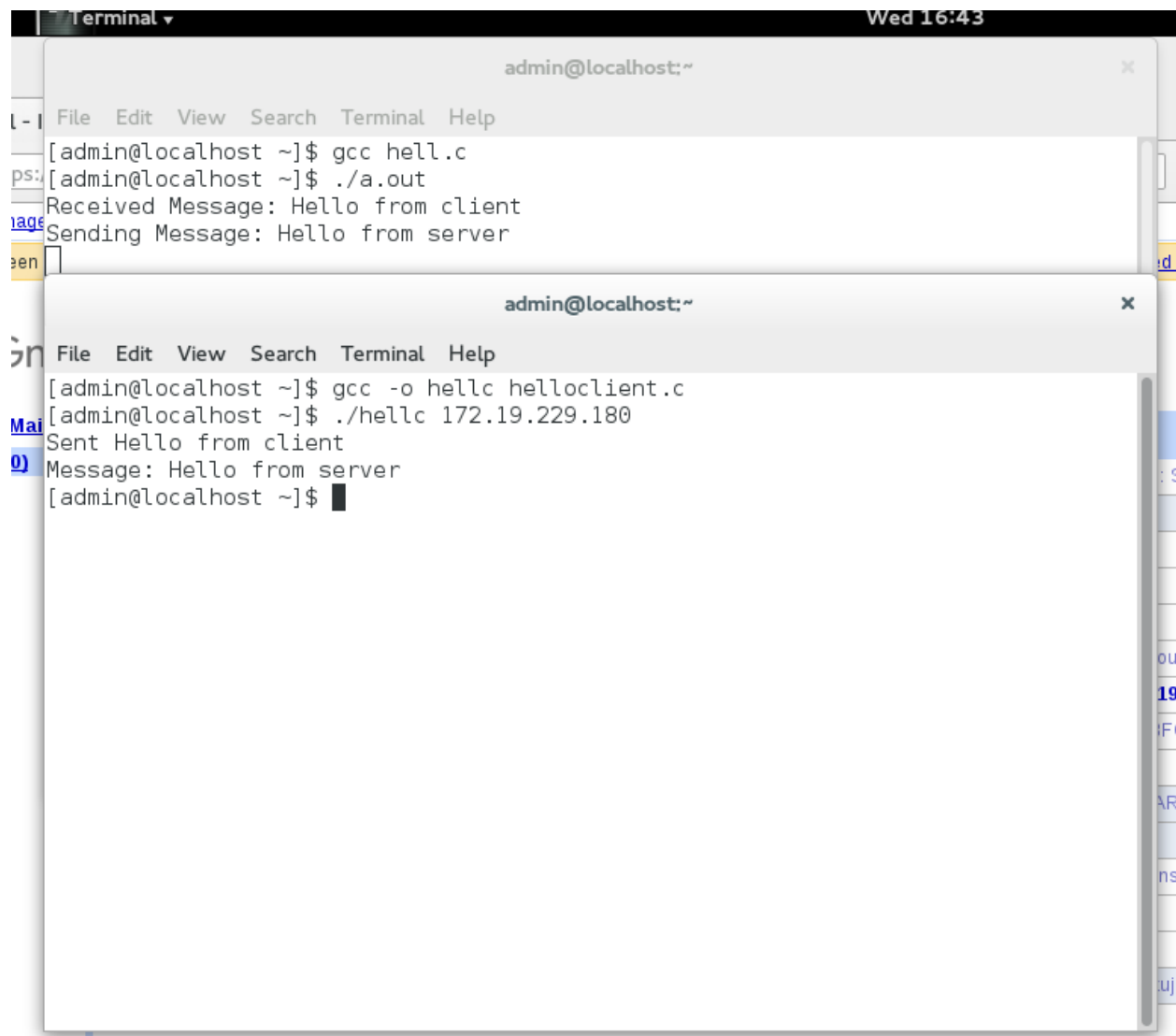
//Connect to the server
if(connect(sock, (struct sockaddr *) &server, sizeof(server))<0)
{
perror("connect failed");
close(sock);
exit(1);
}

//Send data to the server
if(send(sock,DATA,sizeof(DATA), 0)<0)
{
perror("Send Failed");
close(sock);
exit(1);
}
printf("Sent %s\n",DATA);

//Receive data from the server
recv(sock, buffer, sizeof(buffer), 0);
printf("Message: %s\n",buffer);
close(sock);
return 0;
}

```

Output:



The image shows two overlapping terminal windows from a macOS environment. The top window, titled 'Terminal' with a dropdown arrow and 'Wed 16:43' in the top right, has a title bar 'admin@localhost:~'. It contains the following text: a menu bar with 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'; a prompt '[admin@localhost ~]\$ gcc hell.c'; a second prompt '[admin@localhost ~]\$./a.out'; and two lines of output: 'Received Message: Hello from client' and 'Sending Message: Hello from server'. The bottom window, also titled 'admin@localhost:~', contains: a menu bar with 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'; a prompt '[admin@localhost ~]\$ gcc -o hellc helloclient.c'; a second prompt '[admin@localhost ~]\$./hellc 172.19.229.180'; and three lines of output: 'Sent Hello from client', 'Message: Hello from server', and a final prompt '[admin@localhost ~]\$' with a cursor. The background of the bottom window shows a blurred sidebar with various application icons.

```
Terminal ▾ Wed 16:43
admin@localhost:~
File Edit View Search Terminal Help
[admin@localhost ~]$ gcc hell.c
[admin@localhost ~]$ ./a.out
Received Message: Hello from client
Sending Message: Hello from server

admin@localhost:~
File Edit View Search Terminal Help
[admin@localhost ~]$ gcc -o hellc helloclient.c
[admin@localhost ~]$ ./hellc 172.19.229.180
Sent Hello from client
Message: Hello from server
[admin@localhost ~]$
```