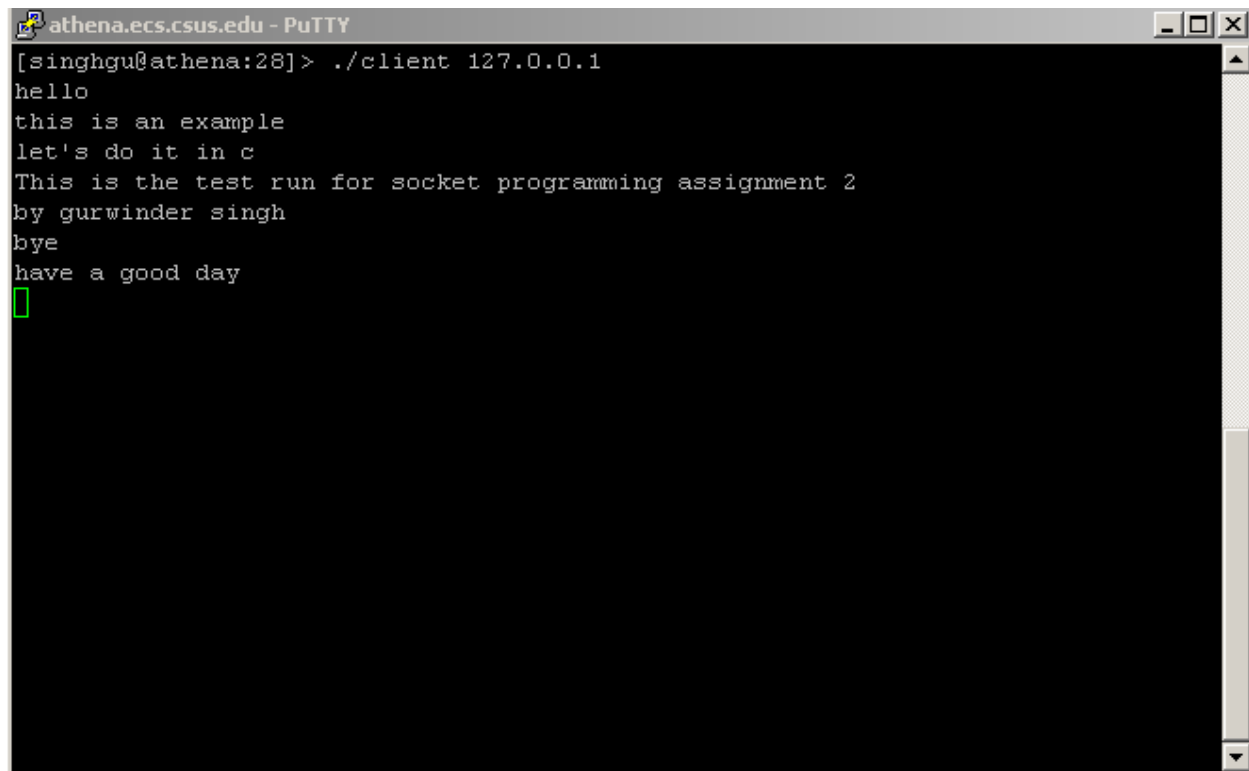


## Report for Socket Prg Project 4-C

### Client

A screenshot of a PuTTY terminal window titled 'athena.ecs.csus.edu - PuTTY'. The terminal shows the output of a client program. The prompt is '[singhgu@athena:28]> ./client 127.0.0.1'. The output consists of several lines: 'hello', 'this is an example', 'let's do it in c', 'This is the test run for socket programming assignment 2', 'by gurwinder singh', 'bye', and 'have a good day'. A green cursor is visible on the line following 'have a good day'.

### Source Code:

```
#include <stdio.h>
#include <sys/types.h>
#include <stdlib.h>
#include <strings.h>
#include <string.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <unistd.h>
#include <netdb.h>

#define SERVER_PORT 5432
#define MAX_LINE 256

int main(int argc, char * argv[])
{
```

```

FILE *fp;
struct hostent *hp;
struct sockaddr_in sin;
char *host;
char buf[MAX_LINE];
int s;
int len;

host = argv[1];

/* translate host name into peer's IP address */
hp = gethostbyname(host);

/* build address data structure */
bzero((char *)&sin, sizeof(sin));
sin.sin_family = AF_INET; /* Internet Address*/
bcopy(hp->h_addr, (char *)&sin.sin_addr, hp->h_length);
sin.sin_port = htons(SERVER_PORT);

/* active open PF_INET is protocol family*/

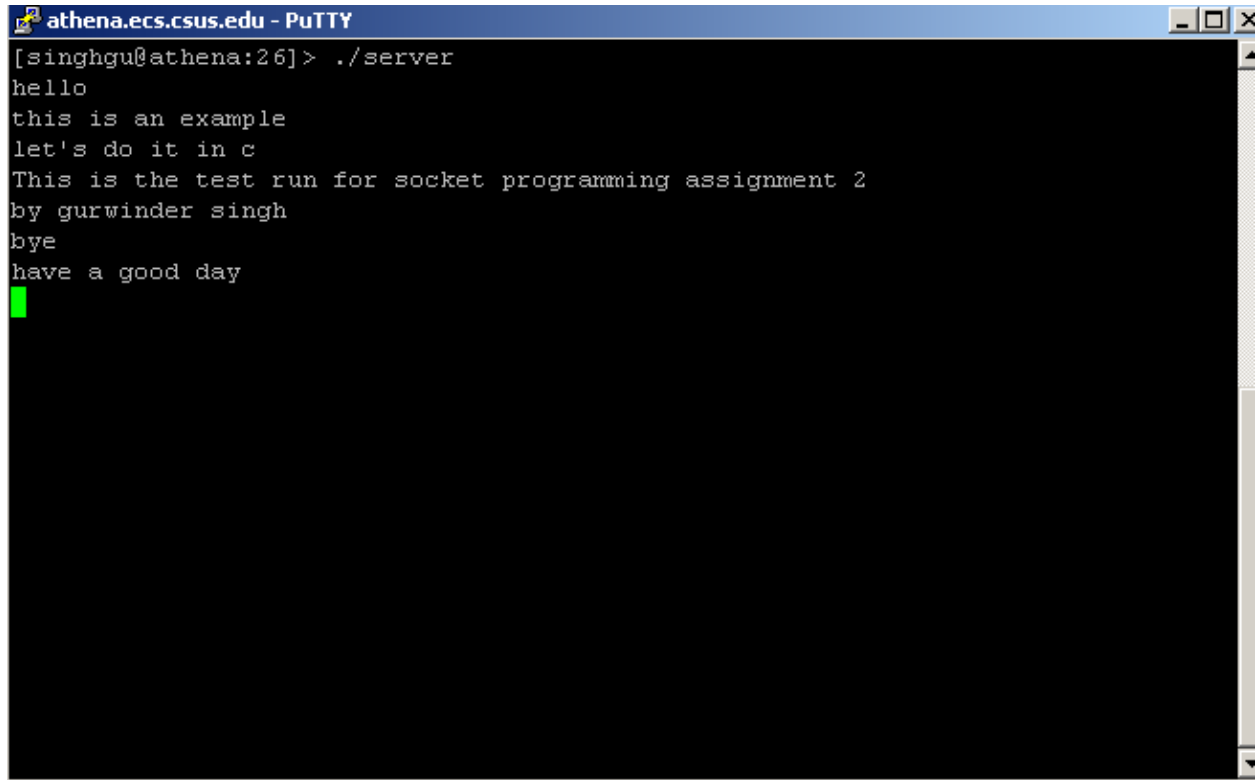
int sockfd = socket (AF_INET, SOCK_STREAM, 0); // creating a socket
if(connect(sockfd, (struct sockaddr *) &sin, sizeof(sin)) < 0)
{
    printf("Socket connection failed");//connection with server
    return EXIT_FAILURE;
}

/* main loop: get and send lines of text */
while (1){
    bzero (buf, MAX_LINE);
    fgets(buf, sizeof(buf), stdin);

    buf[MAX_LINE-1] = '\0';
    len = strlen(buf) + 1;
    send(sockfd, buf, len, 0);
}
close (sockfd);
return 0;
}

```

# Server

A screenshot of a PuTTY terminal window titled 'athena.ecs.csus.edu - PuTTY'. The terminal shows the command './server' being executed, which outputs a series of messages: 'hello', 'this is an example', 'let's do it in c', 'This is the test run for socket programming assignment 2', 'by gurwinder singh', 'bye', and 'have a good day'. A green cursor is visible on the line following the last message.

```
athena.ecs.csus.edu - PuTTY
[singhgu@athena:26]> ./server
hello
this is an example
let's do it in c
This is the test run for socket programming assignment 2
by gurwinder singh
bye
have a good day
█
```

## Source Code:

```
#include <stdio.h>
#include <sys/types.h>
#include <stdlib.h>
#include <strings.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <unistd.h>
#include <netdb.h>
#define SERVER_PORT 5432
#define MAX_PENDING 5
#define MAX_LINE 256
int main()
{
    struct sockaddr_in sin;
    char buf[MAX_LINE];
    int len;
    int s, new_s;
    /* build address data structure */
    bzero((char *)&sin, sizeof(sin));
    sin.sin_family = AF_INET;
    sin.sin_addr.s_addr = INADDR_ANY;
    sin.sin_port = htons(SERVER_PORT);
    /* setup passive open */
    //fill your code here to create the socket
```

```

    int sockfd = socket (AF_INET, SOCK_STREAM, 0);
    if(sockfd < 0){
        printf("Error creating socket.");
        return EXIT_FAILURE;
    }
    //fill your code to bind the socket to the address data structure
    if ((bind (sockfd, (struct sockaddr *)&sin, sizeof(sin))) < 0){
        error("Socket Bind failed \n");
    }

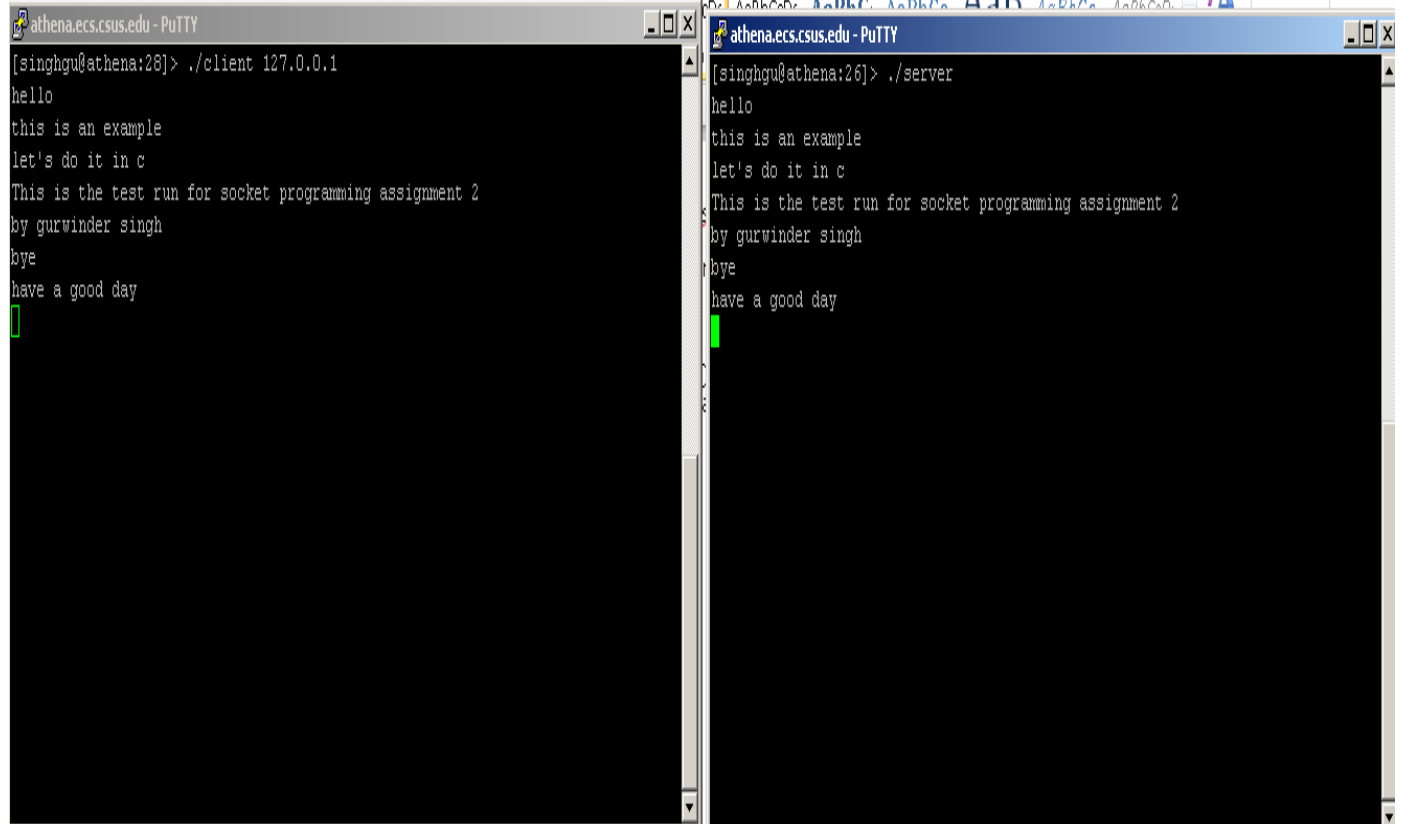
    //fill your code to make the socket to listen
    listen (sockfd, 2);

    int sockfd2= accept (sockfd, (struct sockaddr *) &sin, &len);
    if(sockfd2 < 0){
        printf("Error socket connecting with client. \n");
        return EXIT_FAILURE;
    }

    /* wait for connection, then receive and print text */
    while(1)
    {
        //fill your code to accept connections
        bzero(buf,MAX_LINE);
        int userinput = read( connection2,buf,MAX_LINE);
        printf("%s", buf);

    }
    close(sockfd);
    close(sockfd2);
    return 0
}

```



The image shows two side-by-side terminal windows from a PuTTY session on a system named 'athena.eecs.csus.edu'. The left window shows the execution of a client program, and the right window shows the execution of a server program. Both windows display the same sequence of text, indicating a successful communication between the two.

```
[singhgu@athena:28]> ./client 127.0.0.1
hello
this is an example
let's do it in c
This is the test run for socket programming assignment 2
by gurwinder singh
bye
have a good day
█
```

```
[singhgu@athena:26]> ./server
hello
this is an example
let's do it in c
This is the test run for socket programming assignment 2
by gurwinder singh
bye
have a good day
█
```