Socket programación

Recursos

- Libro

  https://drive.google.com/open?id=0B7jvyYoYJWvSSHNmM2E0OEYySlk
- Presentacion de sockets

  https://drive.google.com/open?id=0B7jvyYoYJWvSTlhlU1BTdkN0U0U
- Archivos

  Cliente.c

  Servidor.c
- Plataforma de trabajo

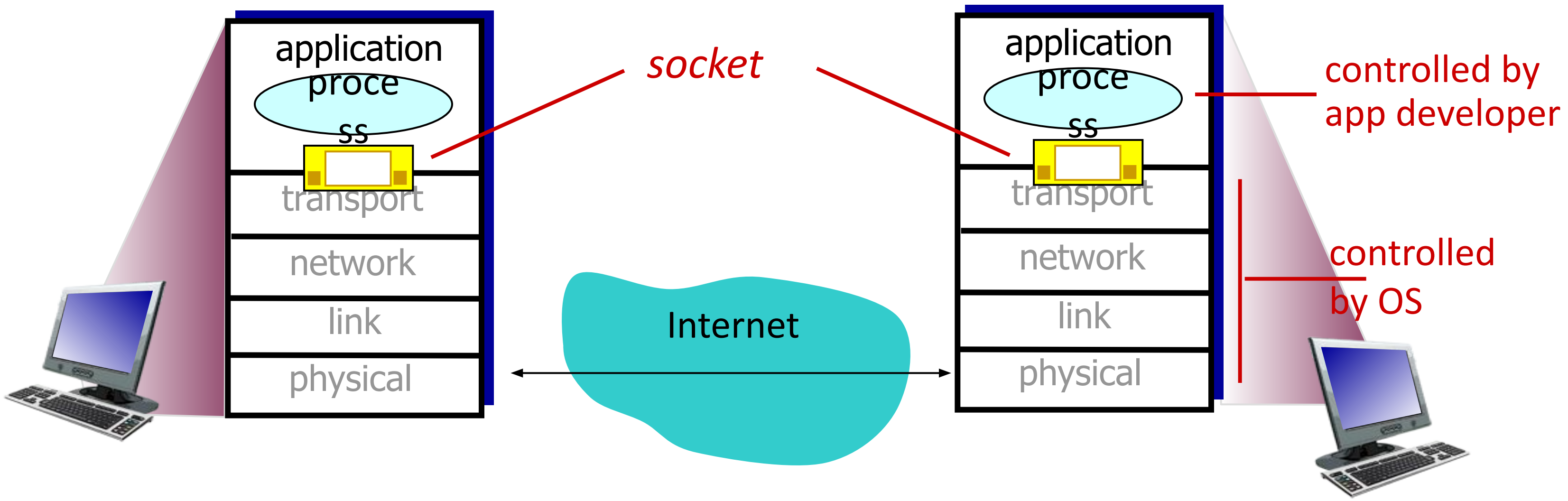
  gcc/g++

  linux

Socket programming

goal: learn how to build client/server applications that communicate using sockets

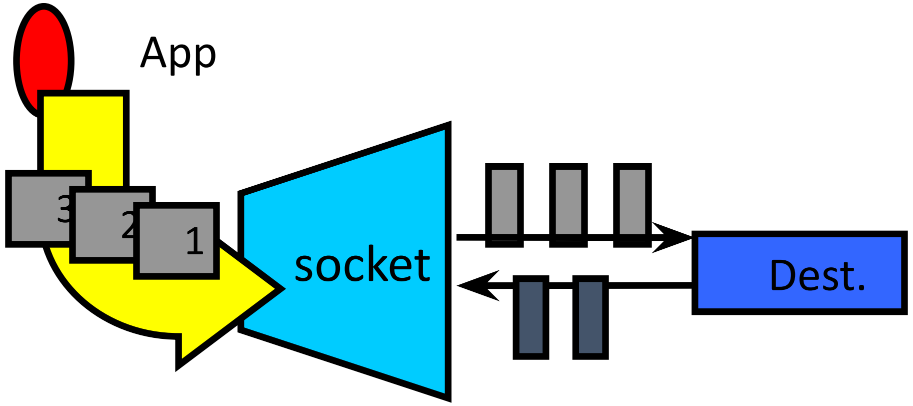
socket: door between application process and end-end-transport protocol

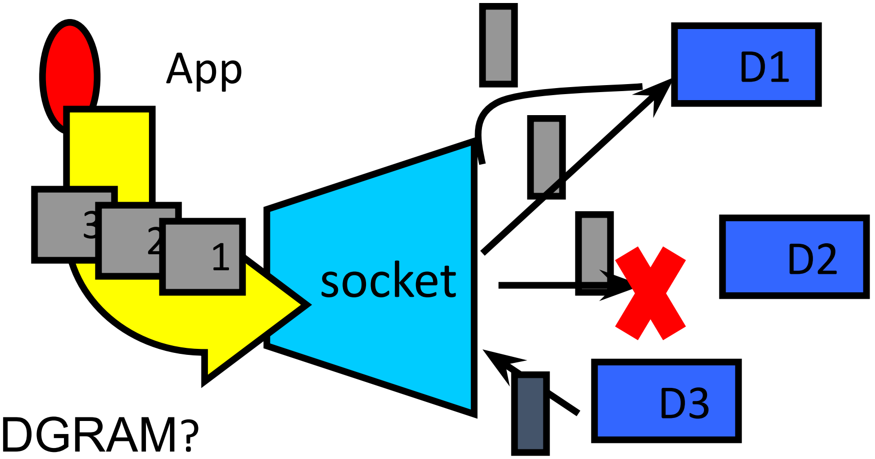


Two essential types of sockets

- SOCK_STREAM
  - a.k.a. TCP
  - reliable delivery
  - in-order guaranteed
  - connection-oriented
  - bidirectional

- SOCK_DGRAM
  - a.k.a. UDP
  - unreliable delivery
  - no order guarantees
  - no notion of “connection” – app indicates dest. for each packet
  - can send or receive

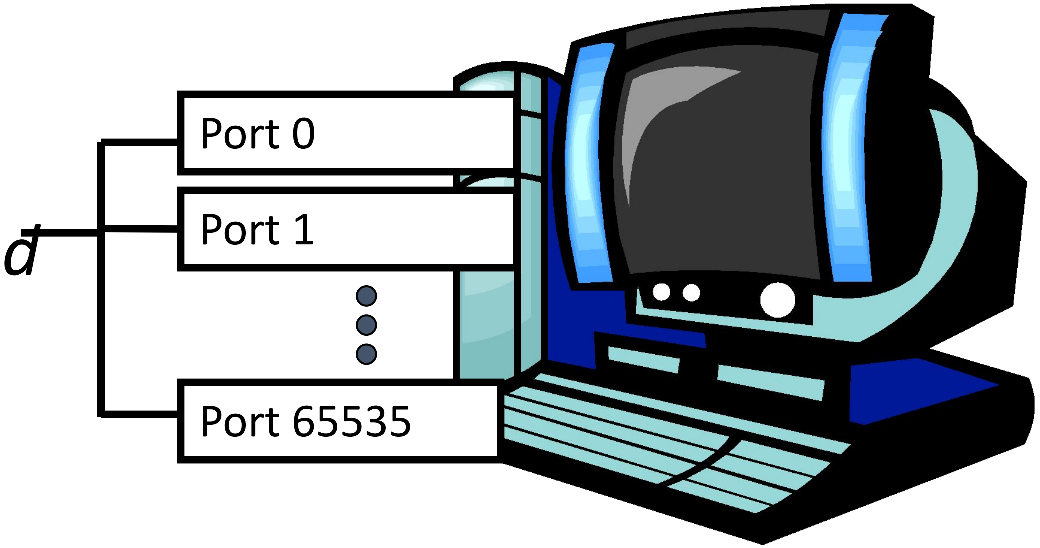




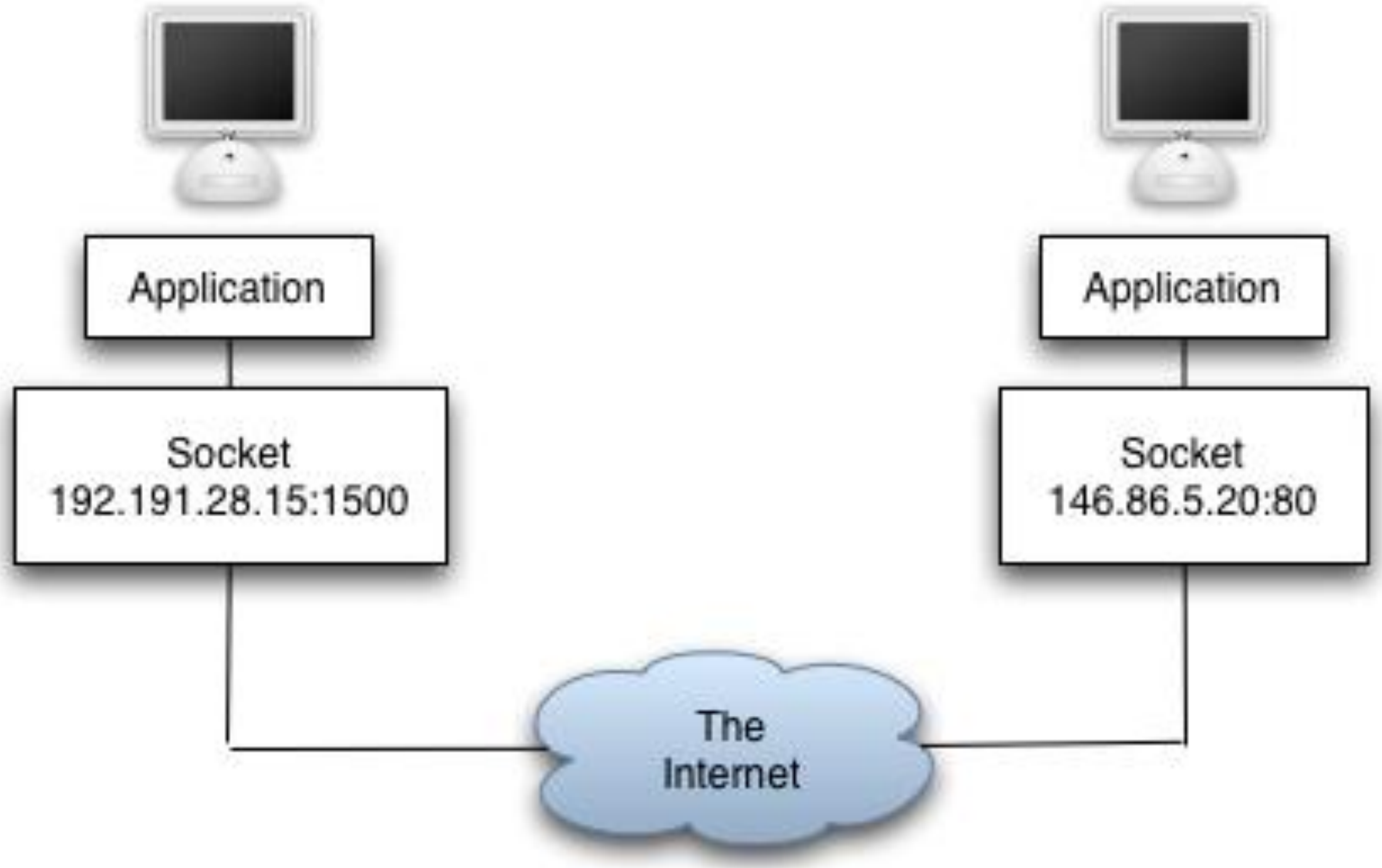
Q: why have type SOCK_DGRAM?

Ports

- Each host has 65,536 ports
- Some ports are *reserved for specific apps*
  - 20,21: FTP
  - 23: Telnet
  - 80: HTTP
  - see RFC 1700 (about 2000 ports are reserved)



r A socket provides an interface to send data to/from the network through a port



Application

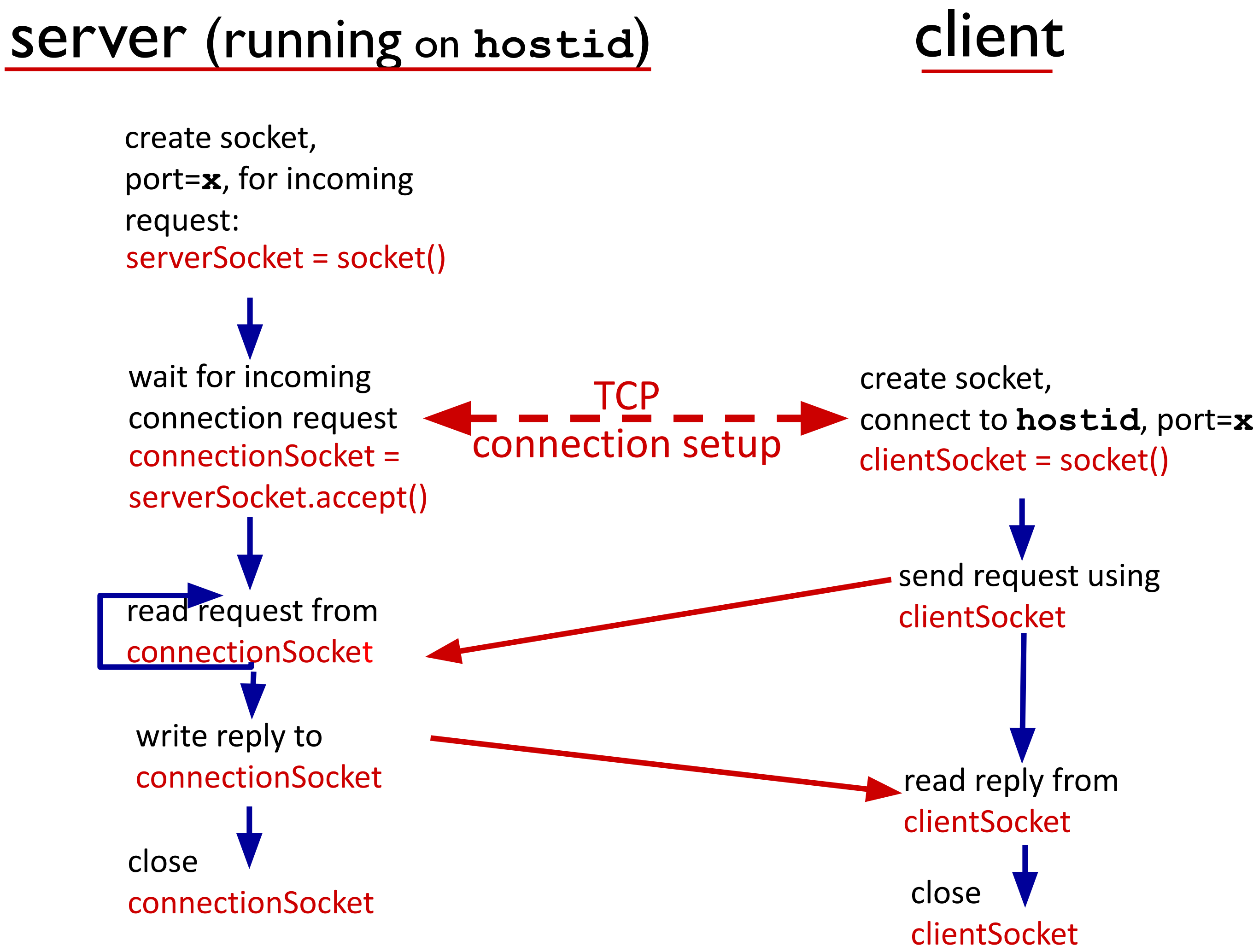
Socket
192.191.28.15:1500

Application

Socket
146.86.5.20:80

The Internet

Client/server socket interaction: TCP



server (running on **hostid**)

create socket, port=**x**, for incoming request:
serverSocket = socket()

wait for incoming connection request
connectionSocket = serverSocket.accept()

read request from connectionSocket

write reply to connectionSocket

close connectionSocket

TCP connection setup

client

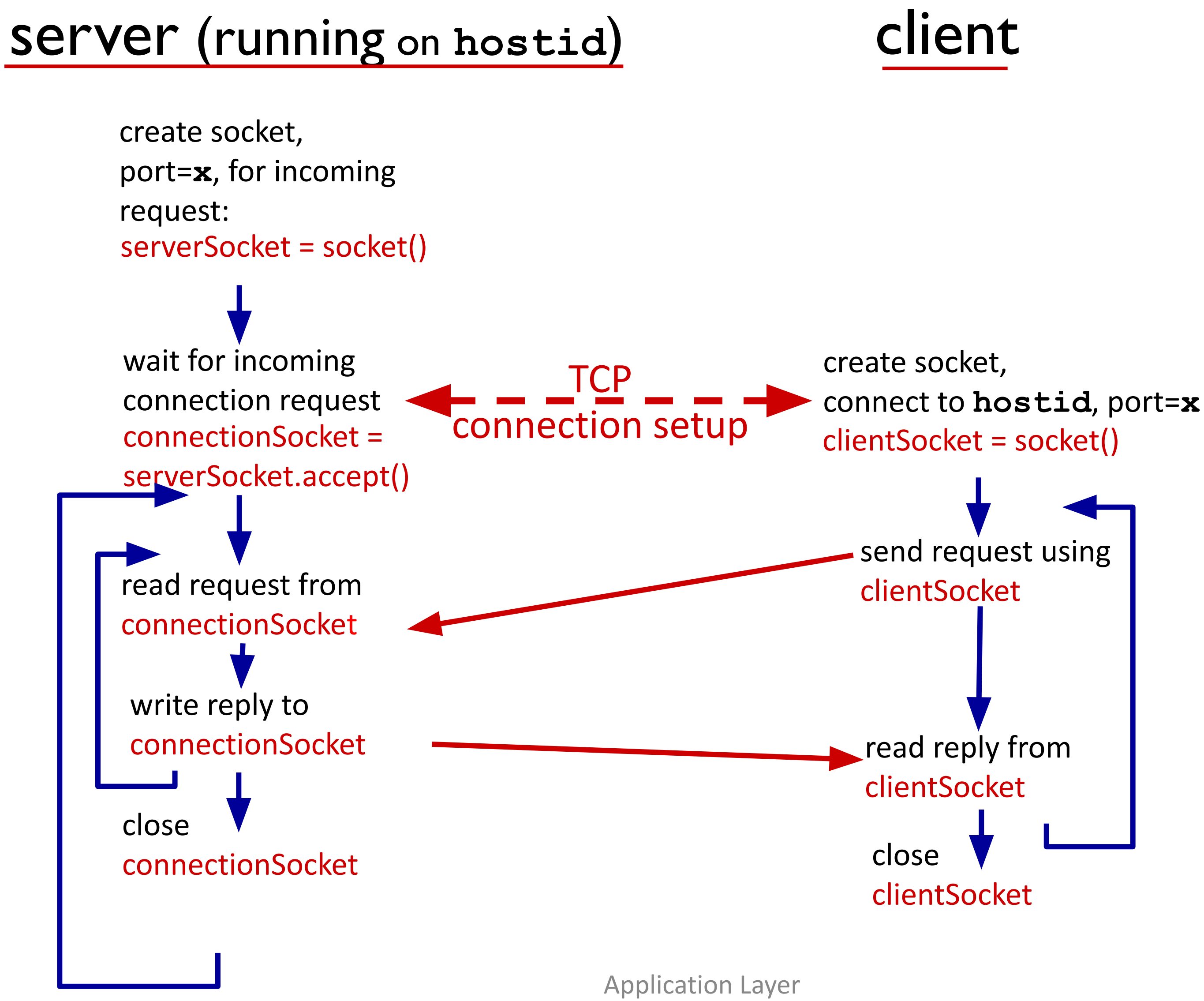
create socket, connect to **hostid**, port=**x**
clientSocket = socket()

send request using clientSocket

read reply from clientSocket

close clientSocket

Client/server socket interaction: Chat



server (running on **hostid**)

create socket,
port=**x**, for incoming
request:
serverSocket = socket()

wait for incoming
connection request
connectionSocket =
serverSocket.accept()

read request from
connectionSocket

write reply to
connectionSocket

close
connectionSocket

TCP
connection setup

client

create socket,
connect to **hostid**, port=**x**
clientSocket = socket()

send request using
clientSocket

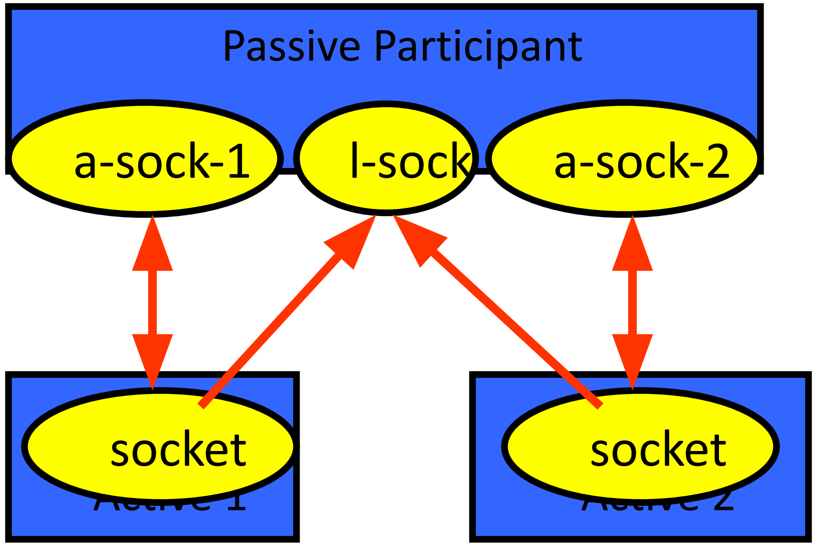
read reply from
clientSocket

close
clientSocket

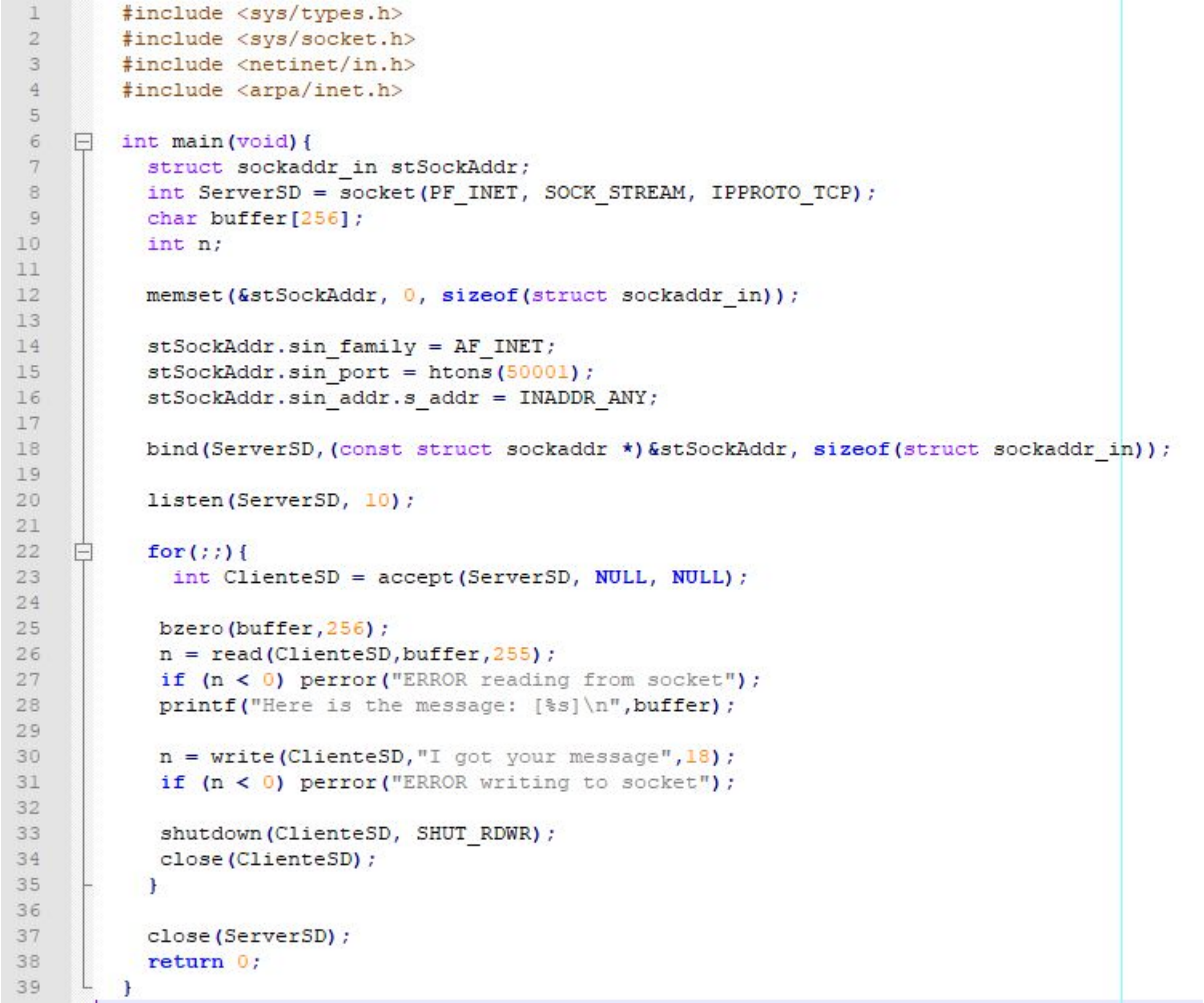
Connection setup multiple users

- Passive participant
  - step 1: listen (for incoming requests)
  - step 3: accept (a request)
  - step 4: data transfer
- The accepted connection is on a new socket
- The old socket continues to listen for other active participants
- Why?

- Active participant
  - step 2: request & establish connection
  - step 4: data transfer

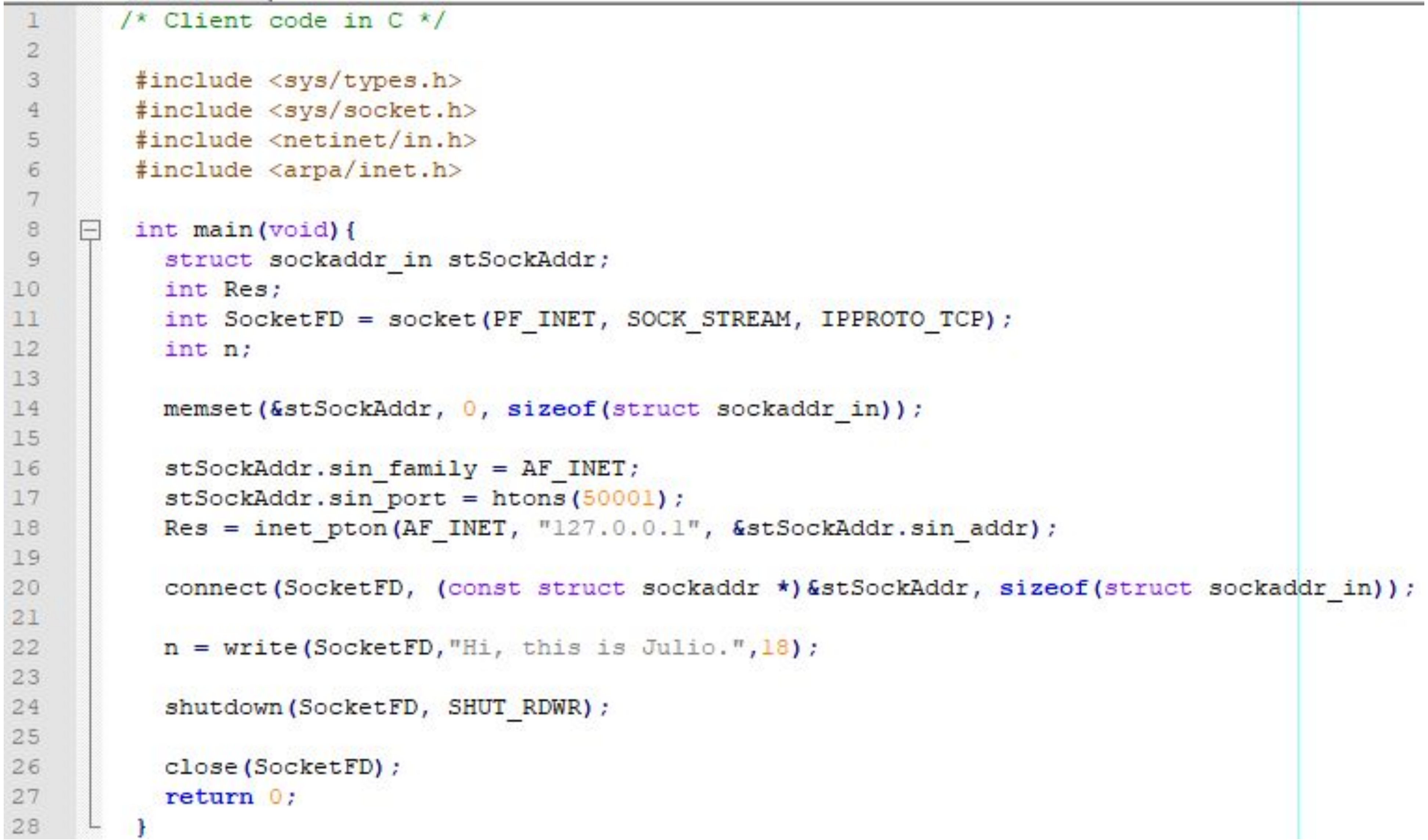


Server.c



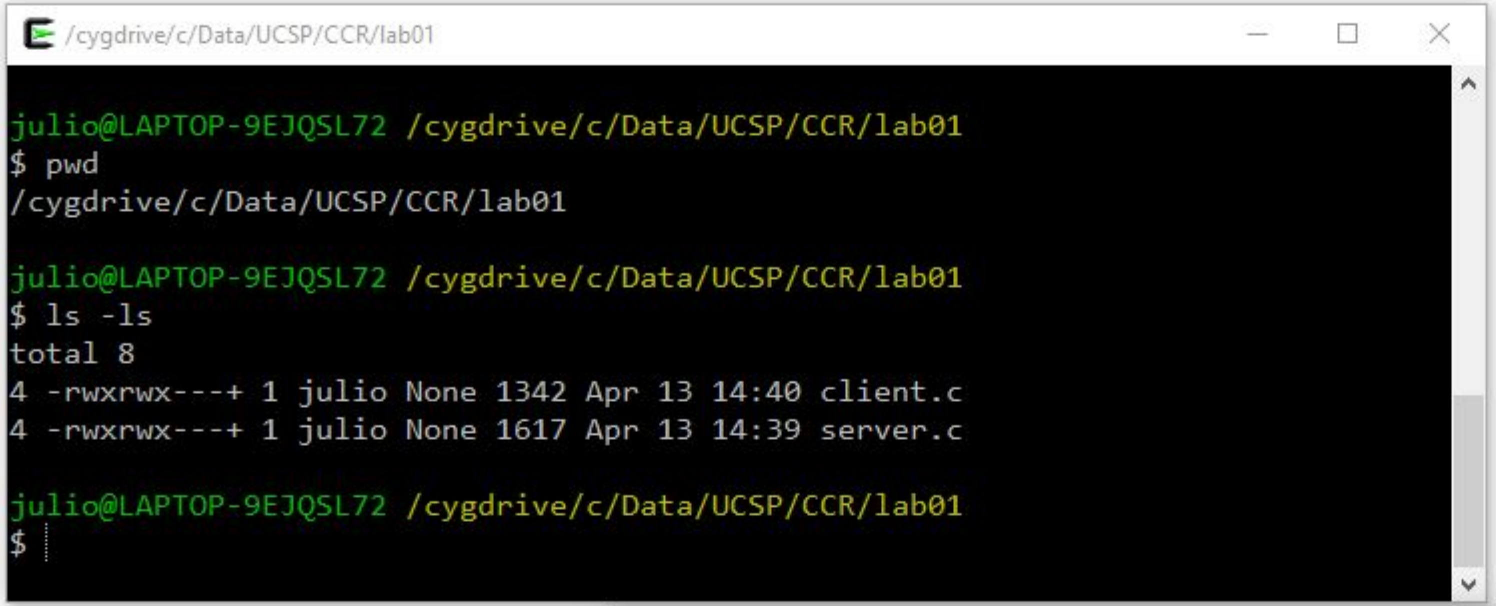
```c
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>

int main(void){
  struct sockaddr_in stSockAddr;
  int ServerSD = socket(PF_INET, SOCK_STREAM, IPPROTO_TCP);
  char buffer[256];
  int n;

  memset(&stSockAddr, 0, sizeof(struct sockaddr_in));

  stSockAddr.sin_family = AF_INET;
  stSockAddr.sin_port = htons(50001);
  stSockAddr.sin_addr.s_addr = INADDR_ANY;

  bind(ServerSD,(const struct sockaddr *)&stSockAddr, sizeof(struct sockaddr_in));

  listen(ServerSD, 10);

  for(;;){
    int ClienteSD = accept(ServerSD, NULL, NULL);

   bzero(buffer,256);
   n = read(ClienteSD,buffer,255);
   if (n < 0) perror("ERROR reading from socket");
   printf("Here is the message: [%s]\n",buffer);

   n = write(ClienteSD,"I got your message",18);
   if (n < 0) perror("ERROR writing to socket");

   shutdown(ClienteSD, SHUT_RDWR);
   close(ClienteSD);
  }

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

Client.c

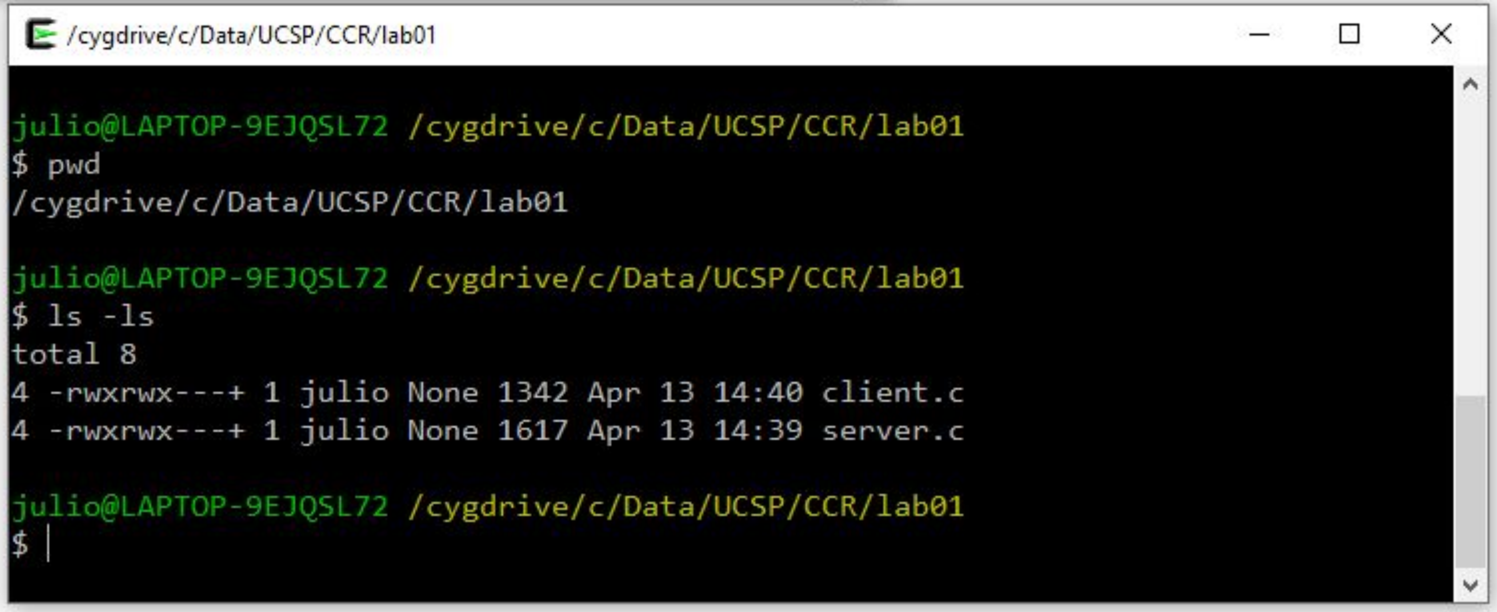


```
/* Client code in C */

 #include <sys/types.h>
 #include <sys/socket.h>
 #include <netinet/in.h>
 #include <arpa/inet.h>

 int main(void){
   struct sockaddr_in stSockAddr;
   int Res;
   int SocketFD = socket(PF_INET, SOCK_STREAM, IPPROTO_TCP);
   int n;

   memset(&stSockAddr, 0, sizeof(struct sockaddr_in));

   stSockAddr.sin_family = AF_INET;
   stSockAddr.sin_port = htons(50001);
   Res = inet_pton(AF_INET, "127.0.0.1", &stSockAddr.sin_addr);

   connect(SocketFD, (const struct sockaddr *)&stSockAddr, sizeof(struct sockaddr_in));

   n = write(SocketFD,"Hi, this is Julio.",18);

   shutdown(SocketFD, SHUT_RDWR);

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

Lab01.1

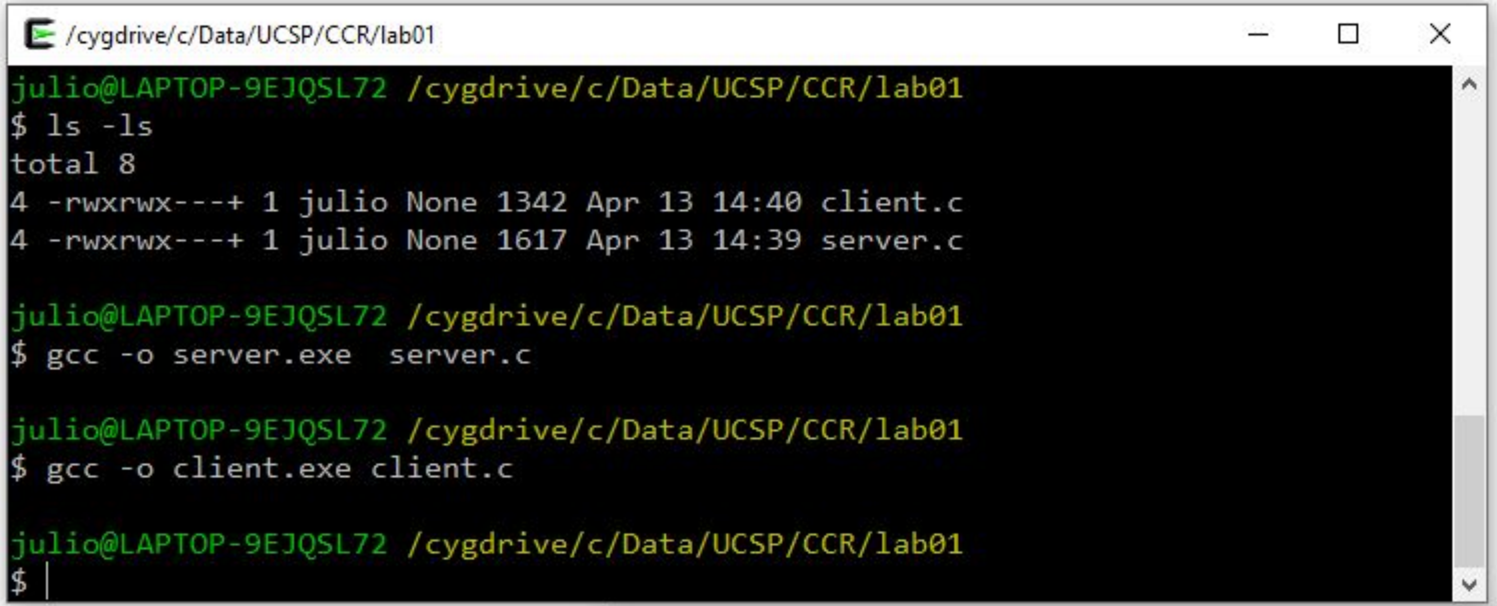


```
julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$ pwd
/cygdrive/c/Data/UCSP/CCR/lab01

julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$ ls -ls
total 8
4 -rwxrwx---+ 1 julio None 1342 Apr 13 14:40 client.c
4 -rwxrwx---+ 1 julio None 1617 Apr 13 14:39 server.c

julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$
```

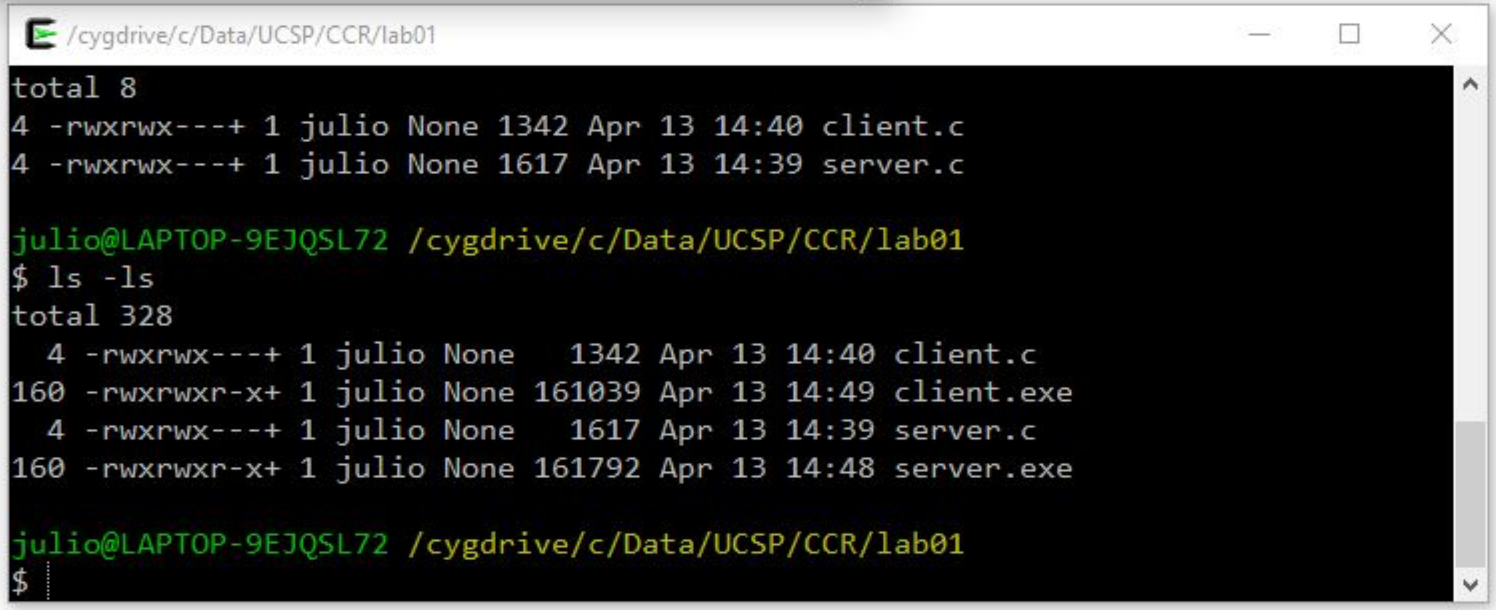


```
julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$ pwd
/cygdrive/c/Data/UCSP/CCR/lab01

julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$ ls -ls
total 8
4 -rwxrwx---+ 1 julio None 1342 Apr 13 14:40 client.c
4 -rwxrwx---+ 1 julio None 1617 Apr 13 14:39 server.c

julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$
```

Lab01.1

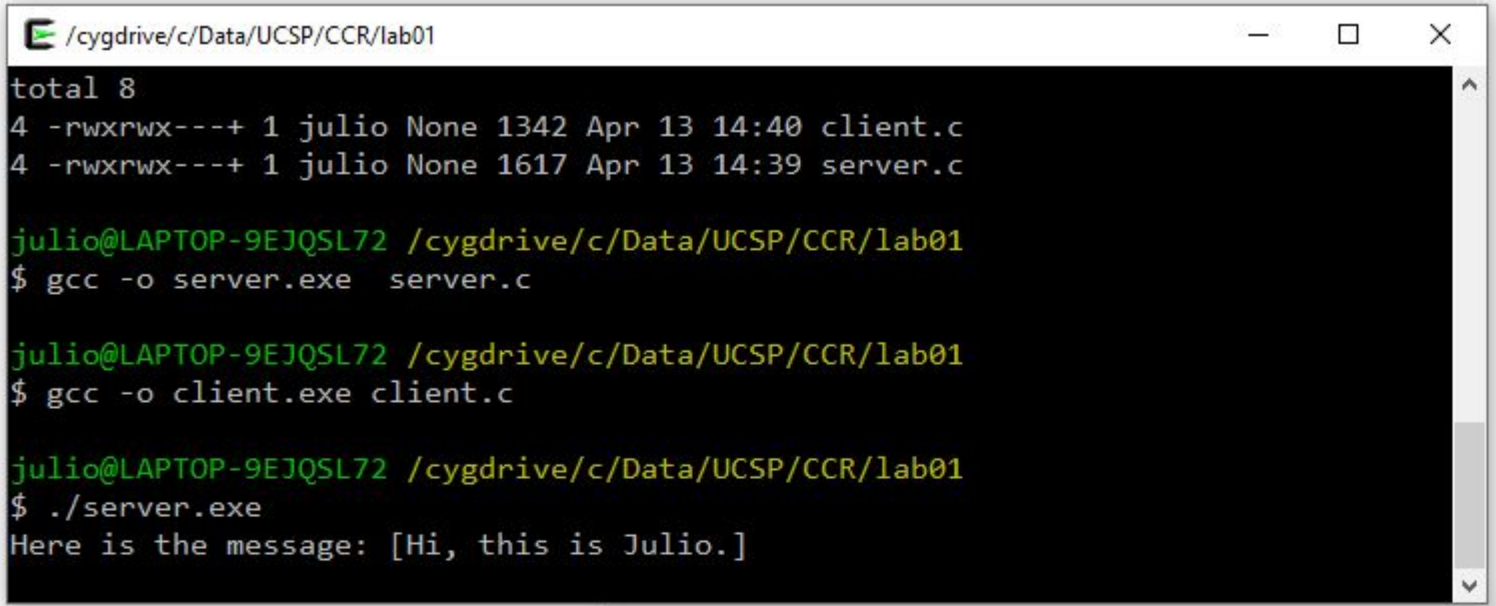


```
julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$ ls -ls
total 8
4 -rwxrwx---+ 1 julio None 1342 Apr 13 14:40 client.c
4 -rwxrwx---+ 1 julio None 1617 Apr 13 14:39 server.c

julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$ gcc -o server.exe  server.c

julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$ gcc -o client.exe client.c

julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$ 
```

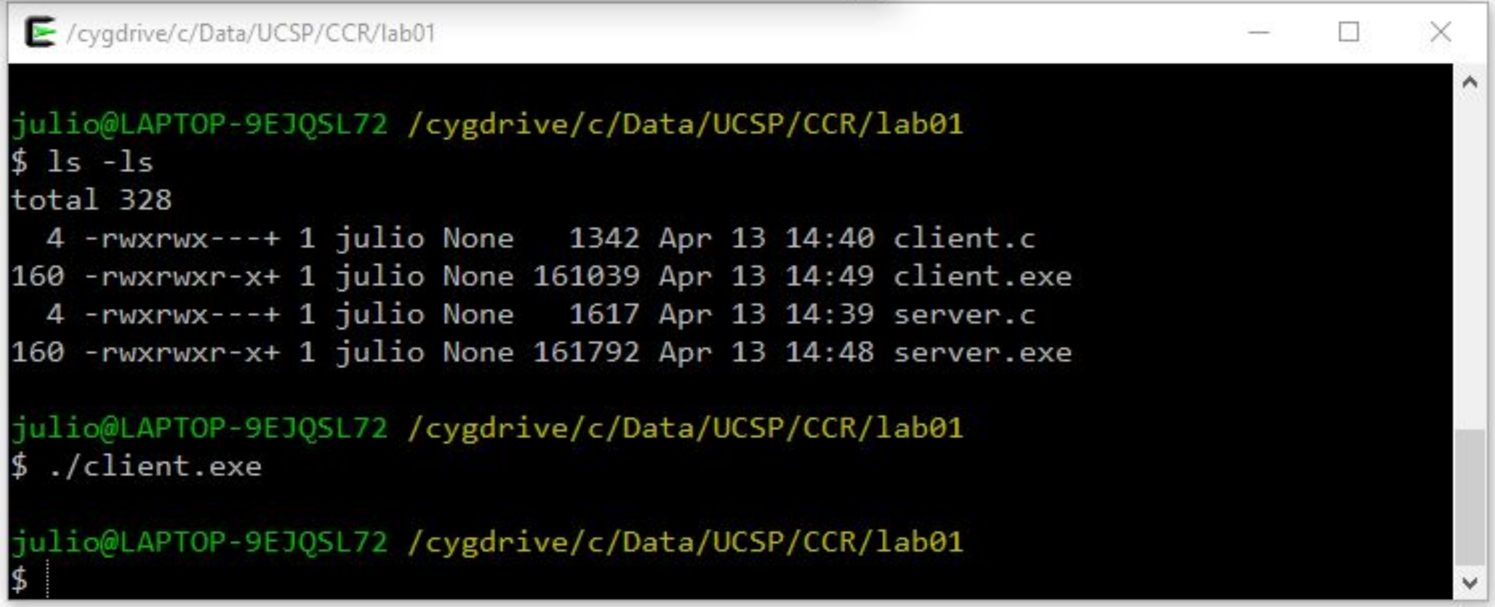


```
total 8
4 -rwxrwx---+ 1 julio None 1342 Apr 13 14:40 client.c
4 -rwxrwx---+ 1 julio None 1617 Apr 13 14:39 server.c

julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$ ls -ls
total 328
  4 -rwxrwx---+ 1 julio None   1342 Apr 13 14:40 client.c
160 -rwxrwxr-x+ 1 julio None 161039 Apr 13 14:49 client.exe
  4 -rwxrwx---+ 1 julio None   1617 Apr 13 14:39 server.c
160 -rwxrwxr-x+ 1 julio None 161792 Apr 13 14:48 server.exe

julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$ 
```

Lab01.1

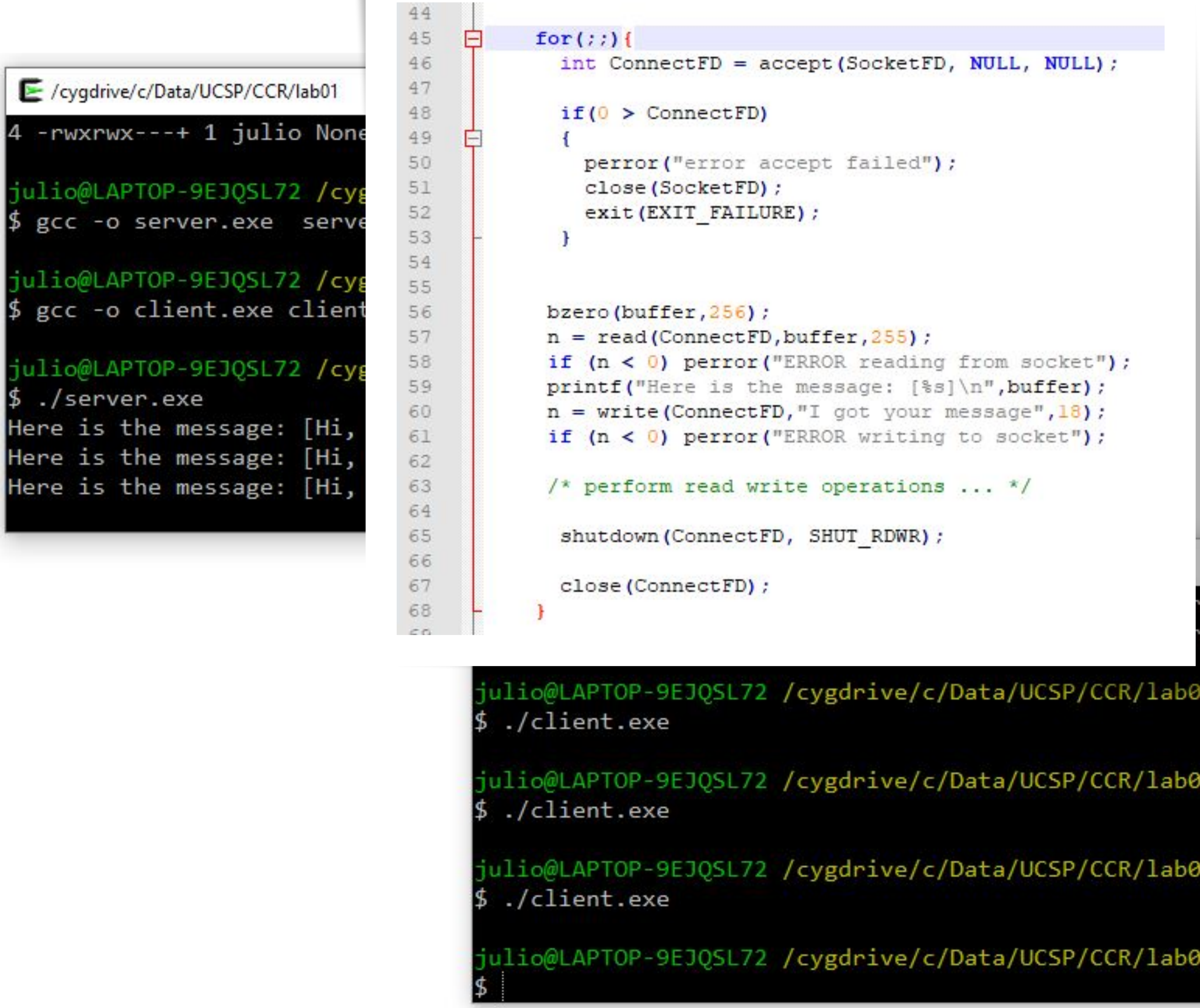


```
total 8
4 -rwxrwx---+ 1 julio None 1342 Apr 13 14:40 client.c
4 -rwxrwx---+ 1 julio None 1617 Apr 13 14:39 server.c

julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$ gcc -o server.exe  server.c

julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$ gcc -o client.exe client.c

julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$ ./server.exe
Here is the message: [Hi, this is Julio.]
```



```

julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$ ls -ls
total 328
  4 -rwxrwx---+ 1 julio None   1342 Apr 13 14:40 client.c
160 -rwxrwxr-x+ 1 julio None 161039 Apr 13 14:49 client.exe
  4 -rwxrwx---+ 1 julio None   1617 Apr 13 14:39 server.c
160 -rwxrwxr-x+ 1 julio None 161792 Apr 13 14:48 server.exe

julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$ ./client.exe

julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$
```

Lab01.2

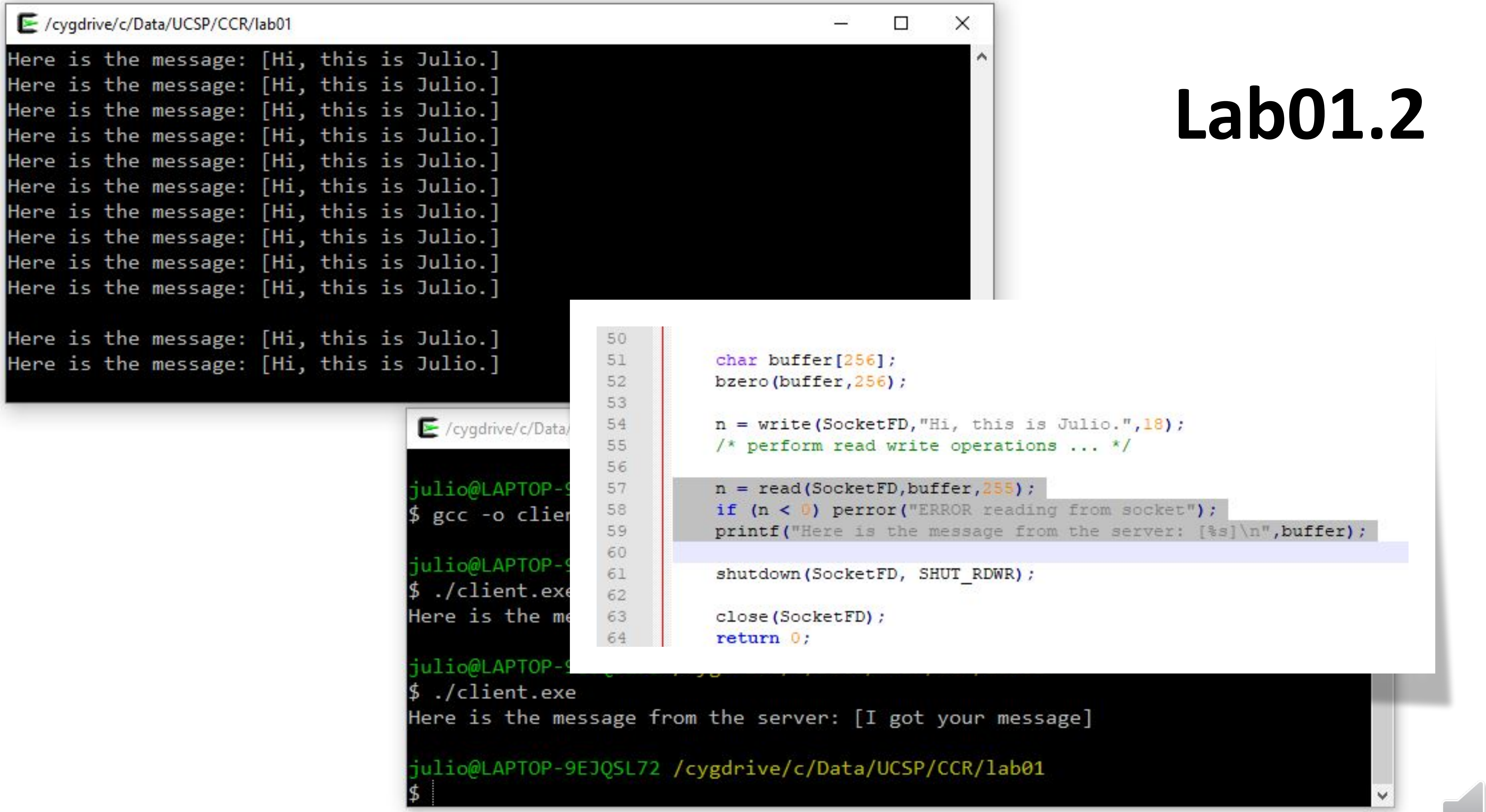


```
/cygdrive/c/Data/UCSP/CCR/lab01
4 -rwxrwx---+ 1 julio None

julio@LAPTOP-9EJQSL72 /cyg
$ gcc -o server.exe  serve

julio@LAPTOP-9EJQSL72 /cyg
$ gcc -o client.exe client

julio@LAPTOP-9EJQSL72 /cyg
$ ./server.exe
Here is the message: [Hi,
Here is the message: [Hi,
Here is the message: [Hi,
```

```c
    for(;;){
      int ConnectFD = accept(SocketFD, NULL, NULL);

      if(0 > ConnectFD)
      {
        perror("error accept failed");
        close(SocketFD);
        exit(EXIT_FAILURE);
      }


     bzero(buffer,256);
     n = read(ConnectFD,buffer,255);
     if (n < 0) perror("ERROR reading from socket");
     printf("Here is the message: [%s]\n",buffer);
     n = write(ConnectFD,"I got your message",18);
     if (n < 0) perror("ERROR writing to socket");

     /* perform read write operations ... */

      shutdown(ConnectFD, SHUT_RDWR);

      close(ConnectFD);
    }
```

```
ver.c
ver.exe

julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$ ./client.exe

julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$ ./client.exe

julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$ ./client.exe

julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$
```

Lab01.2



/cygdrive/c/Data/UCSP/CCR/lab01

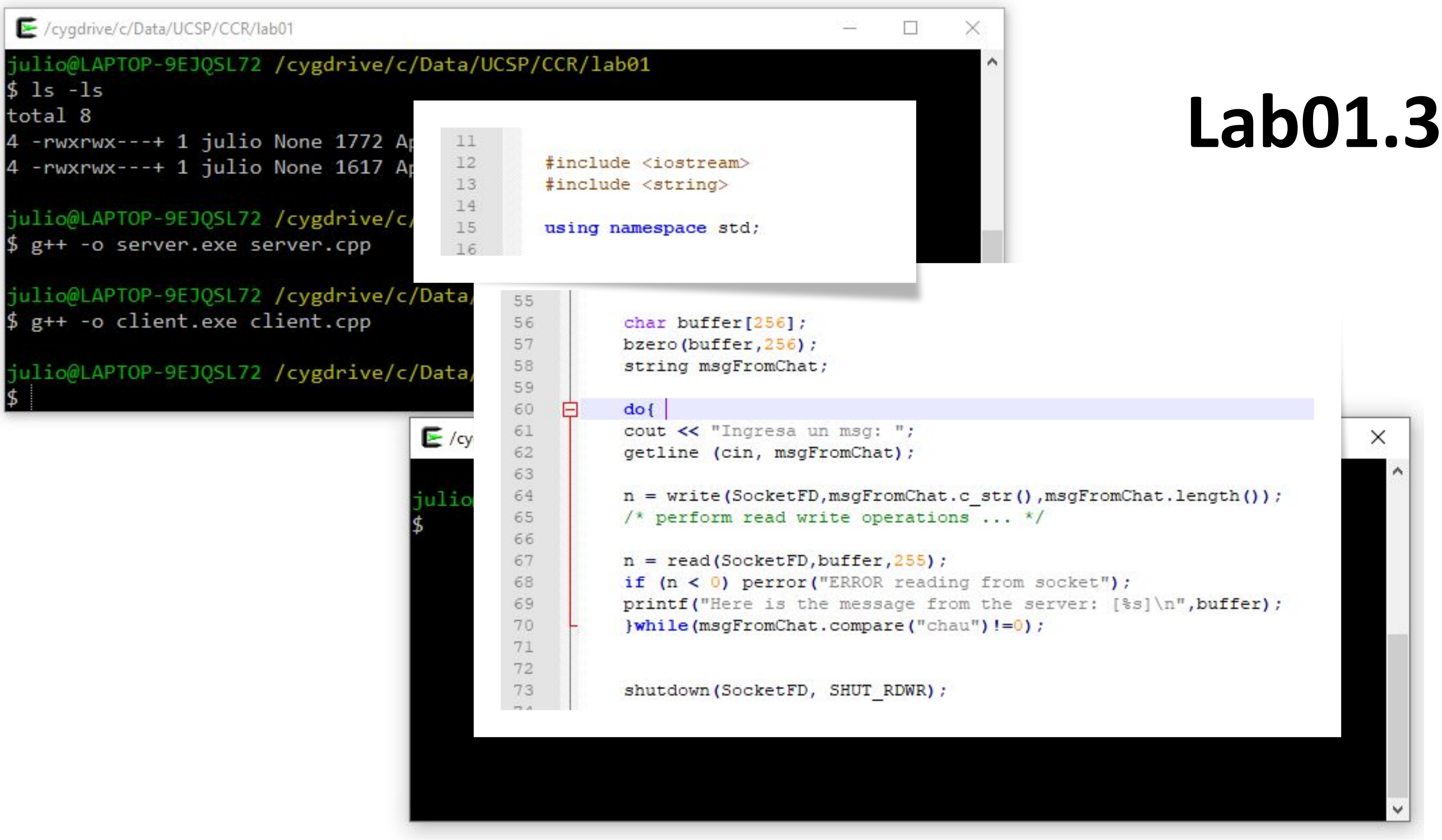
```
Here is the message: [Hi, this is Julio.]
Here is the message: [Hi, this is Julio.]
Here is the message: [Hi, this is Julio.]
Here is the message: [Hi, this is Julio.]
Here is the message: [Hi, this is Julio.]
Here is the message: [Hi, this is Julio.]
Here is the message: [Hi, this is Julio.]
Here is the message: [Hi, this is Julio.]
Here is the message: [Hi, this is Julio.]
Here is the message: [Hi, this is Julio.]

Here is the message: [Hi, this is Julio.]
Here is the message: [Hi, this is Julio.]
```

```c

    char buffer[256];
    bzero(buffer,256);

    n = write(SocketFD,"Hi, this is Julio.",18);
    /* perform read write operations ... */

    n = read(SocketFD,buffer,255);
    if (n < 0) perror("ERROR reading from socket");
    printf("Here is the message from the server: [%s]\n",buffer);

    shutdown(SocketFD, SHUT_RDWR);

    close(SocketFD);
    return 0;
```

/cygdrive/c/Data/

```
julio@LAPTOP-9
$ gcc -o clie

julio@LAPTOP-9
$ ./client.exe
Here is the me

julio@LAPTOP-9
$ ./client.exe
Here is the message from the server: [I got your message]

julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$
```

Lab01.3



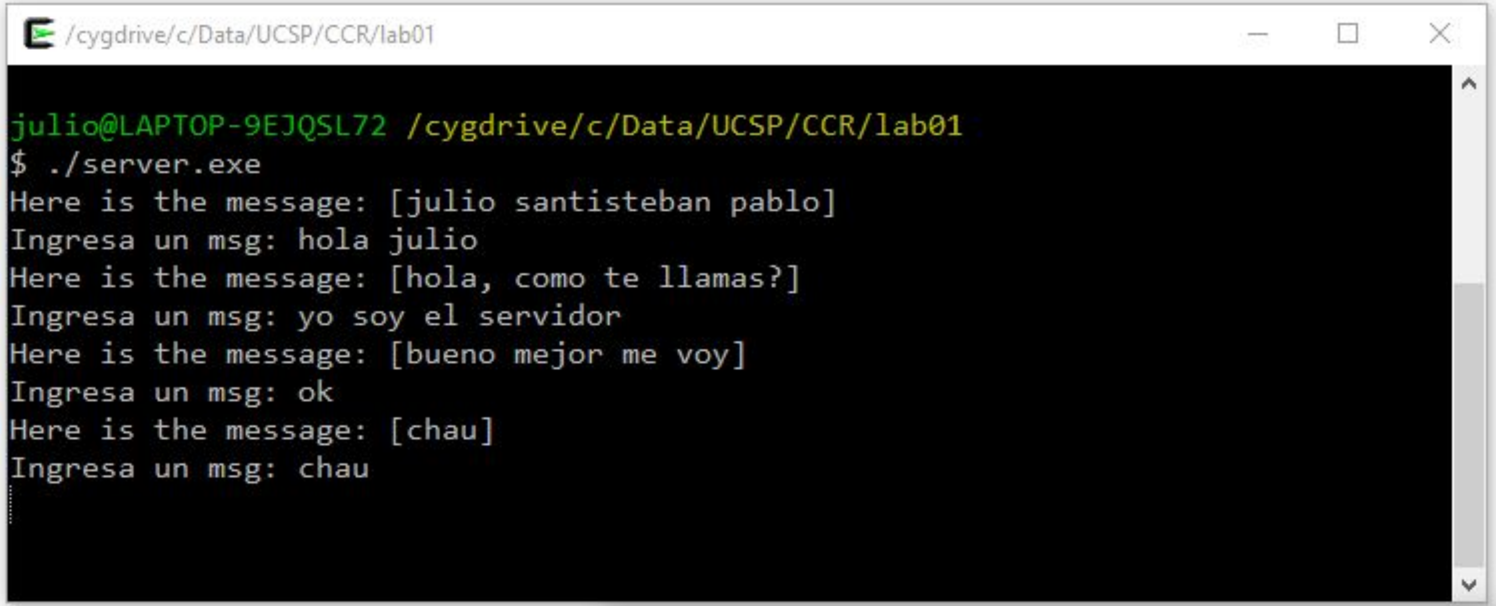
```
julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$ ls -ls
total 8
4 -rwxrwx---+ 1 julio None 1772 Ap
4 -rwxrwx---+ 1 julio None 1617 Ap

julio@LAPTOP-9EJQSL72 /cygdrive/c/
$ g++ -o server.exe server.cpp

julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/
$ g++ -o client.exe client.cpp

julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/
$
```

```cpp

#include <iostream>
#include <string>

using namespace std;

```

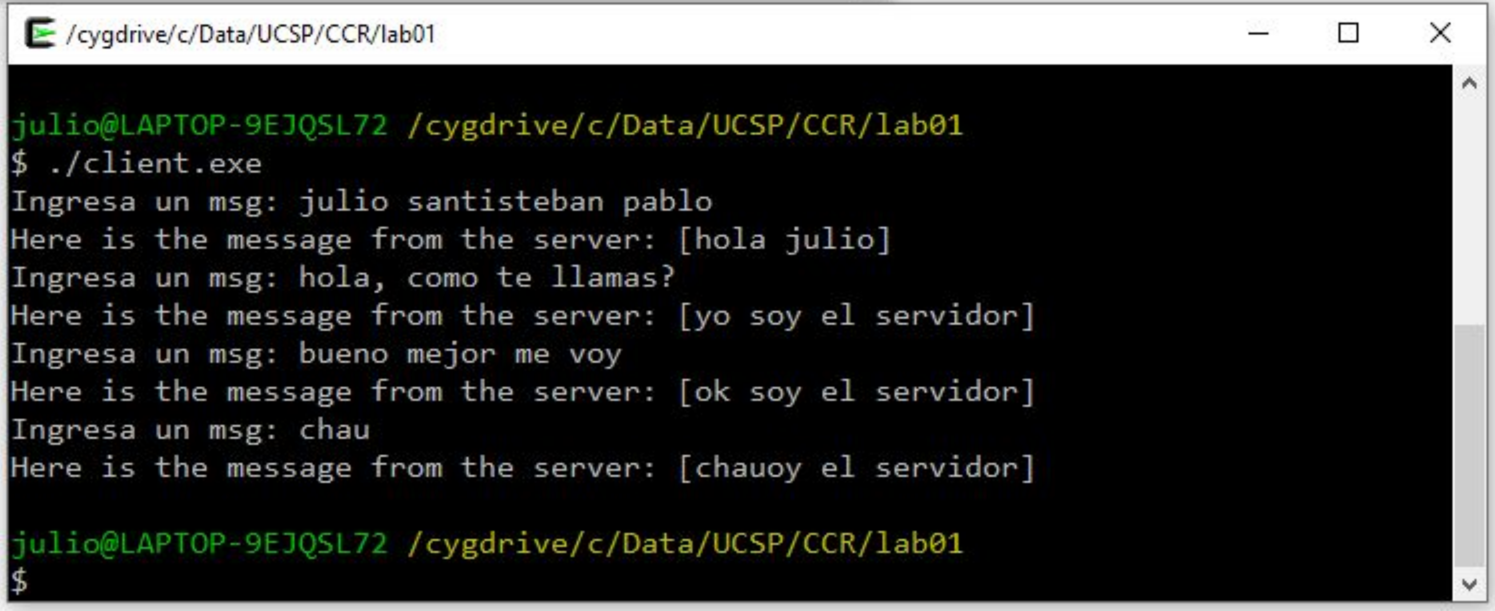
```cpp

    char buffer[256];
    bzero(buffer,256);
    string msgFromChat;

    do{
    cout << "Ingresa un msg: ";
    getline (cin, msgFromChat);

    n = write(SocketFD,msgFromChat.c_str(),msgFromChat.length());
    /* perform read write operations ... */

    n = read(SocketFD,buffer,255);
    if (n < 0) perror("ERROR reading from socket");
    printf("Here is the message from the server: [%s]\n",buffer);
    }while(msgFromChat.compare("chau")!=0);


    shutdown(SocketFD, SHUT_RDWR);
```

```
julio
$
```

Lab01.3

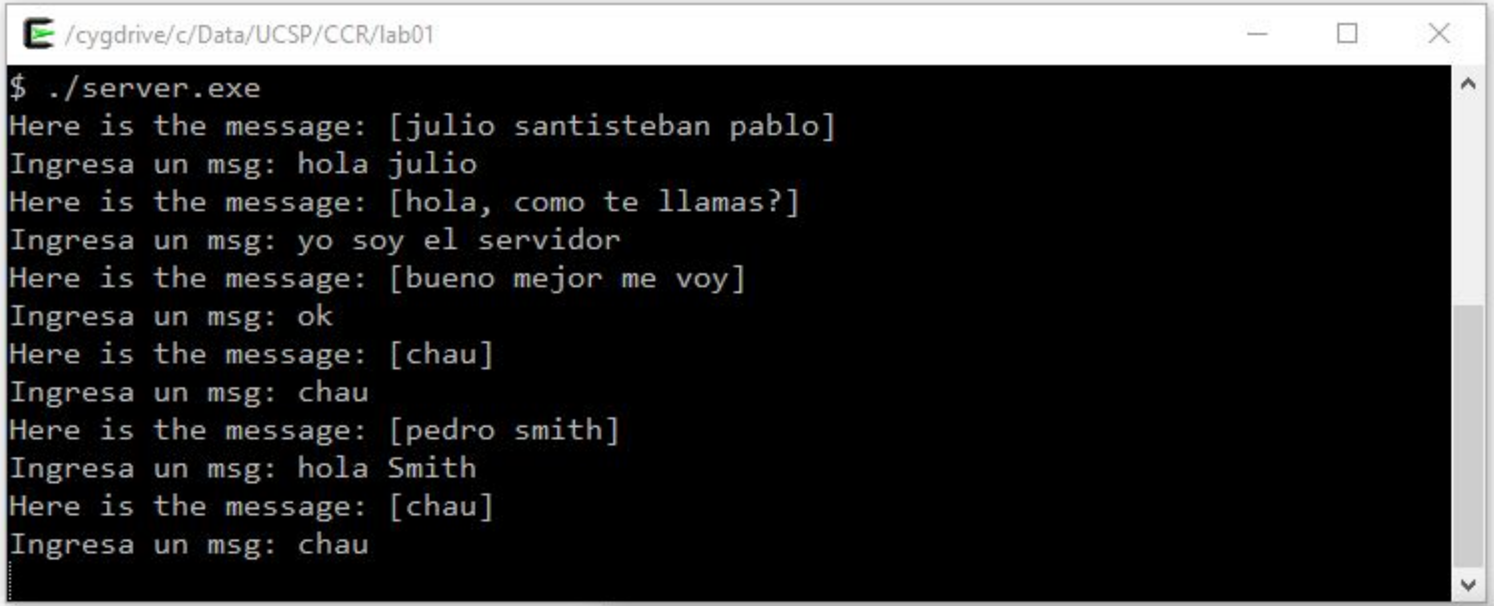


```
julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$ ./server.exe
Here is the message: [julio santisteban pablo]
Ingresa un msg: hola julio
Here is the message: [hola, como te llamas?]
Ingresa un msg: yo soy el servidor
Here is the message: [bueno mejor me voy]
Ingresa un msg: ok
Here is the message: [chau]
Ingresa un msg: chau
```

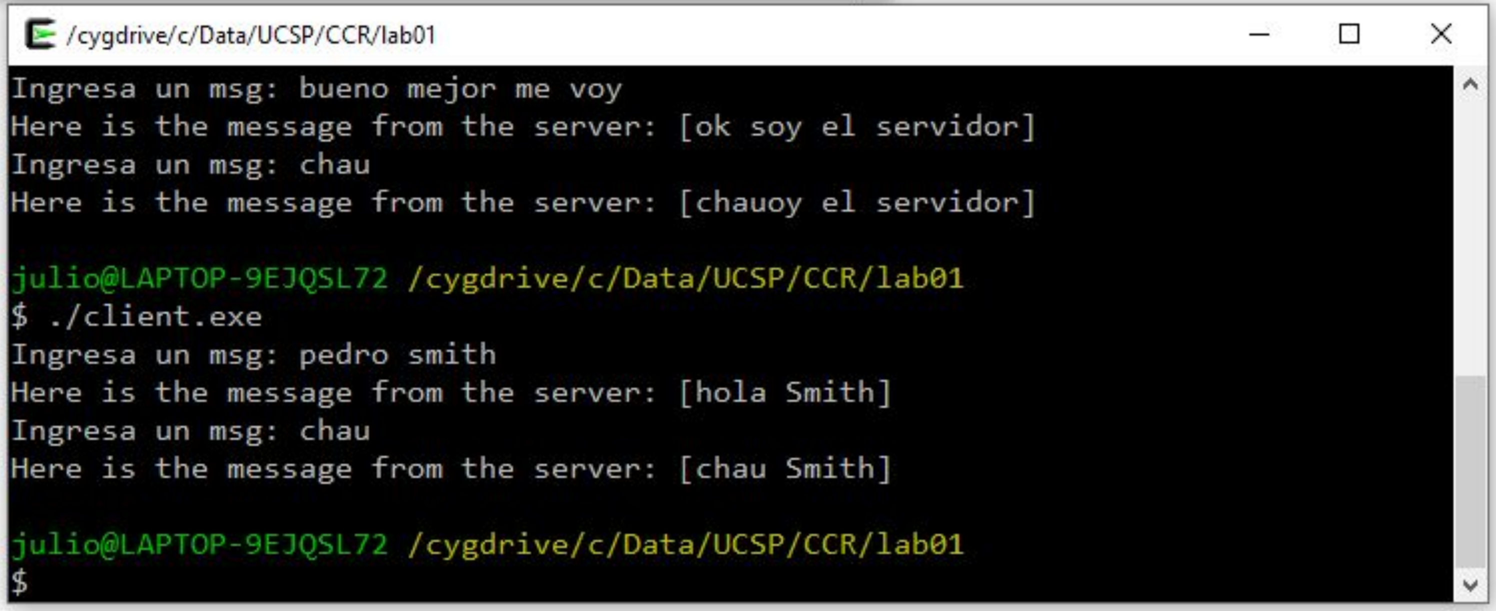


```
julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$ ./client.exe
Ingresa un msg: julio santisteban pablo
Here is the message from the server: [hola julio]
Ingresa un msg: hola, como te llamas?
Here is the message from the server: [yo soy el servidor]
Ingresa un msg: bueno mejor me voy
Here is the message from the server: [ok soy el servidor]
Ingresa un msg: chau
Here is the message from the server: [chauoy el servidor]

julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$
```

Lab01.3



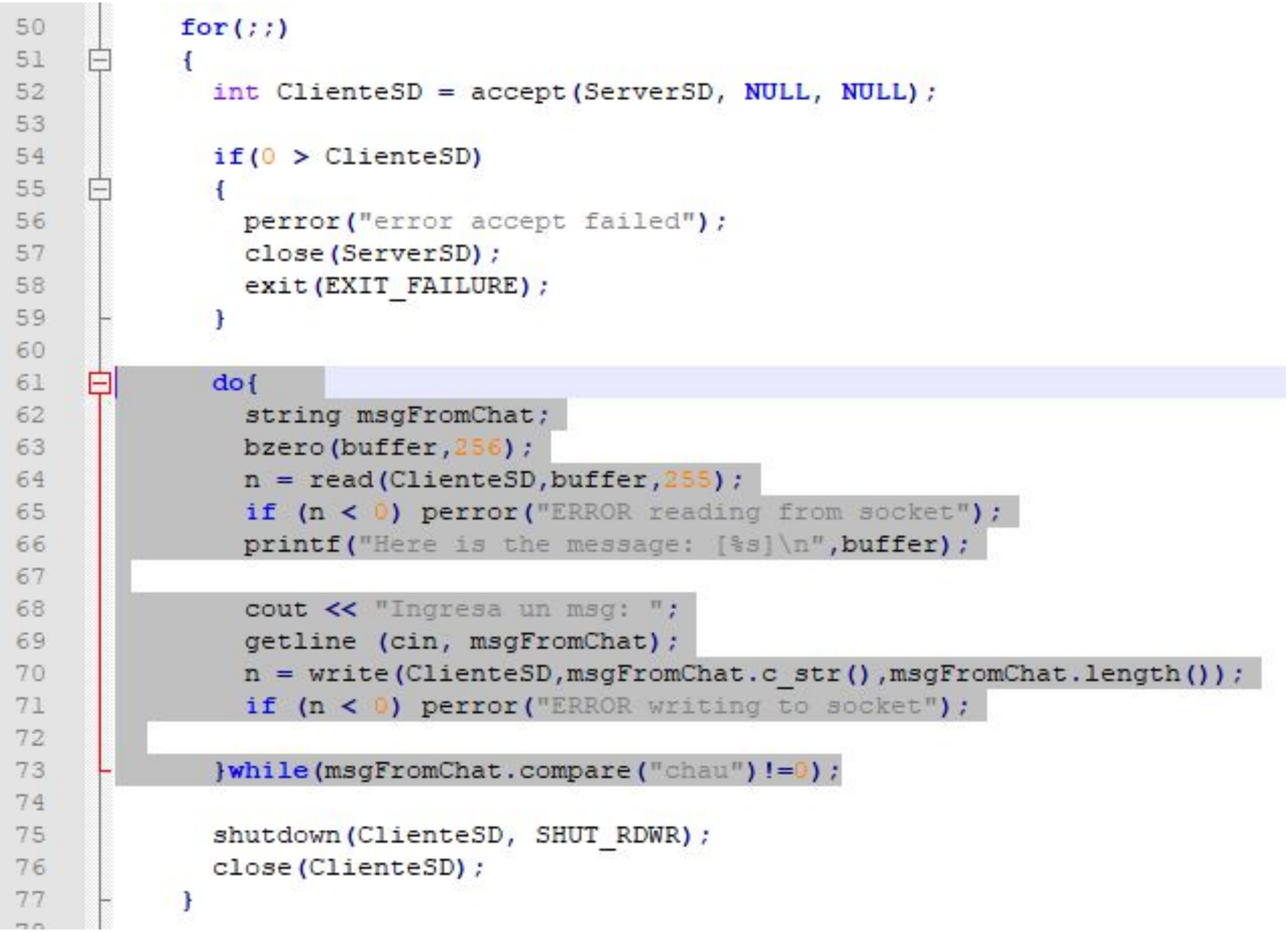
```
$ ./server.exe
Here is the message: [julio santisteban pablo]
Ingresa un msg: hola julio
Here is the message: [hola, como te llamas?]
Ingresa un msg: yo soy el servidor
Here is the message: [bueno mejor me voy]
Ingresa un msg: ok
Here is the message: [chau]
Ingresa un msg: chau
Here is the message: [pedro smith]
Ingresa un msg: hola Smith
Here is the message: [chau]
Ingresa un msg: chau
```



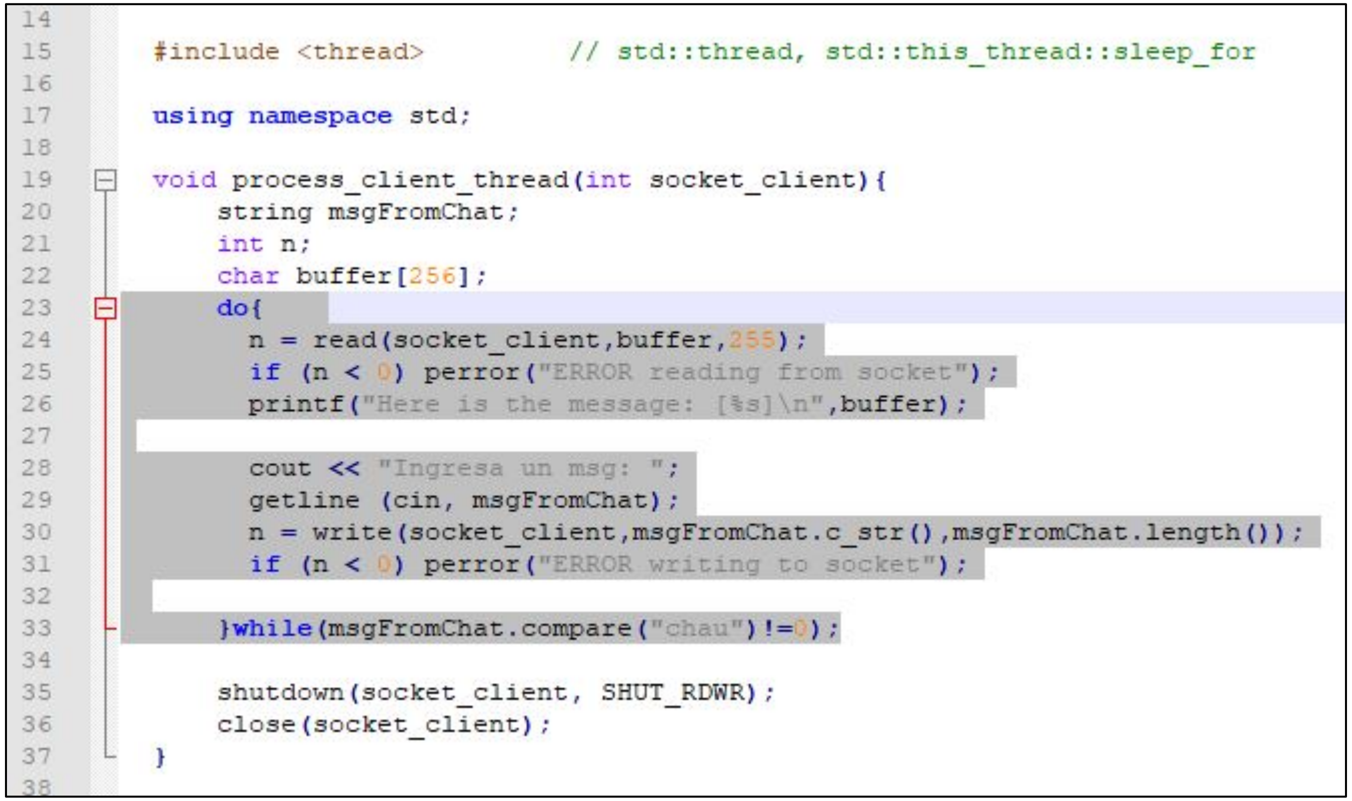
```
Ingresa un msg: bueno mejor me voy
Here is the message from the server: [ok soy el servidor]
Ingresa un msg: chau
Here is the message from the server: [chauoy el servidor]

julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$ ./client.exe
Ingresa un msg: pedro smith
Here is the message from the server: [hola Smith]
Ingresa un msg: chau
Here is the message from the server: [chau Smith]

julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$
```

Uso de threads

Lab01.4

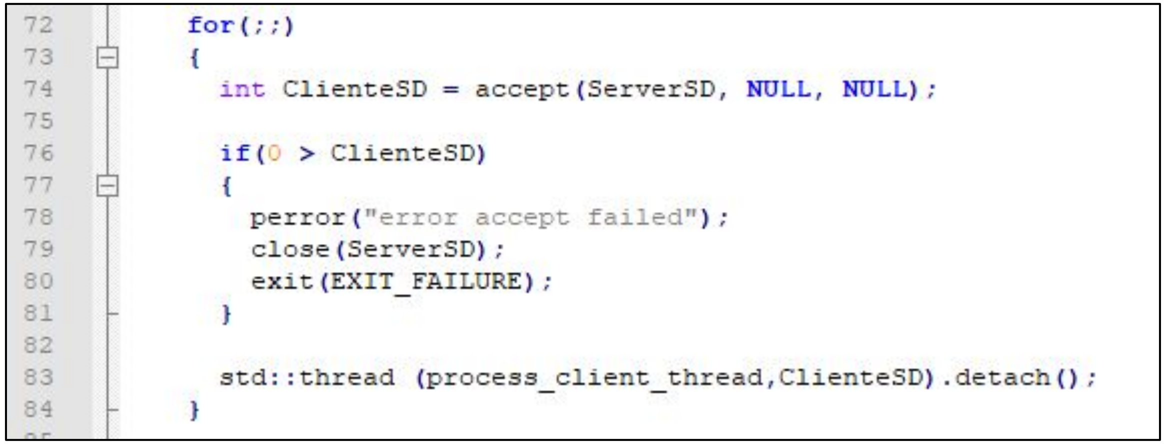


```
    for(;;)
    {
      int ClienteSD = accept(ServerSD, NULL, NULL);

      if(0 > ClienteSD)
      {
        perror("error accept failed");
        close(ServerSD);
        exit(EXIT_FAILURE);
      }

      do{
        string msgFromChat;
        bzero(buffer,256);
        n = read(ClienteSD,buffer,255);
        if (n < 0) perror("ERROR reading from socket");
        printf("Here is the message: [%s]\n",buffer);

        cout << "Ingresa un msg: ";
        getline (cin, msgFromChat);
        n = write(ClienteSD,msgFromChat.c_str(),msgFromChat.length());
        if (n < 0) perror("ERROR writing to socket");

      }while(msgFromChat.compare("chau")!=0);

      shutdown(ClienteSD, SHUT_RDWR);
      close(ClienteSD);
    }
```



```

#include <thread>          // std::thread, std::this_thread::sleep_for

using namespace std;

void process_client_thread(int socket_client){
    string msgFromChat;
    int n;
    char buffer[256];
    do{
      n = read(socket_client,buffer,255);
      if (n < 0) perror("ERROR reading from socket");
      printf("Here is the message: [%s]\n",buffer);

      cout << "Ingresa un msg: ";
      getline (cin, msgFromChat);
      n = write(socket_client,msgFromChat.c_str(),msgFromChat.length());
      if (n < 0) perror("ERROR writing to socket");

    }while(msgFromChat.compare("chau")!=0);

    shutdown(socket_client, SHUT_RDWR);
    close(socket_client);
}
```

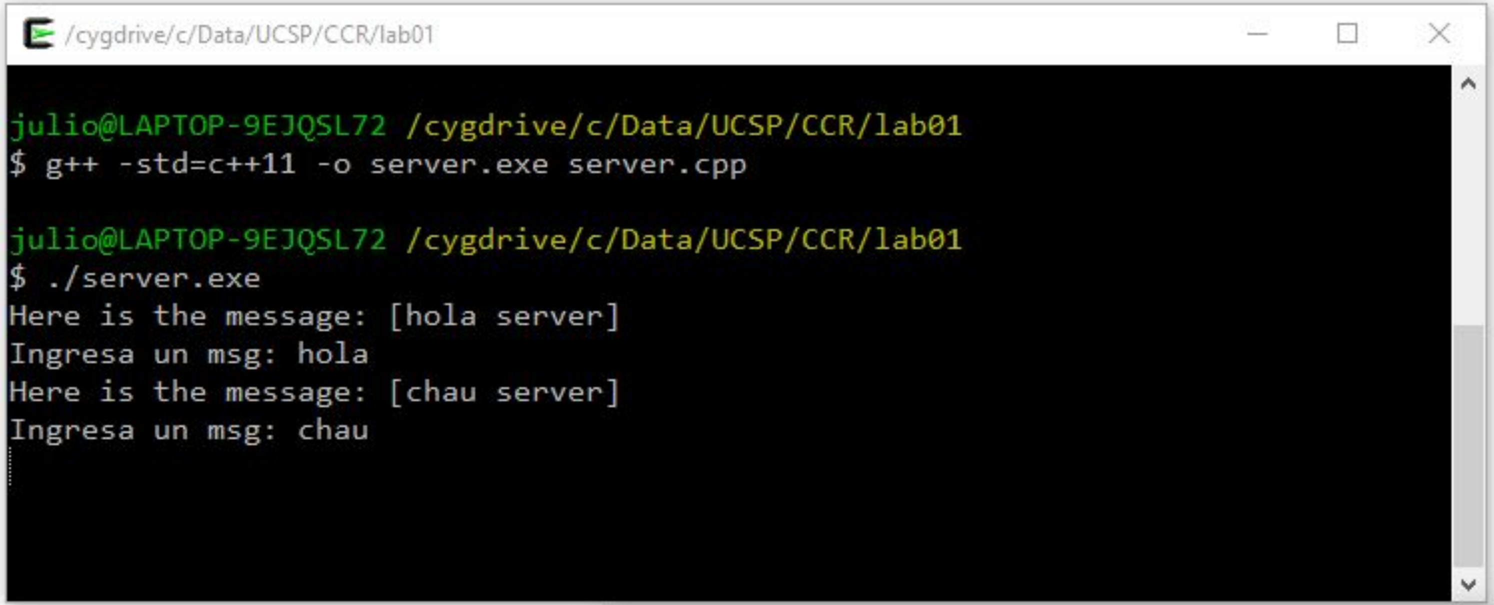
Lab01.4



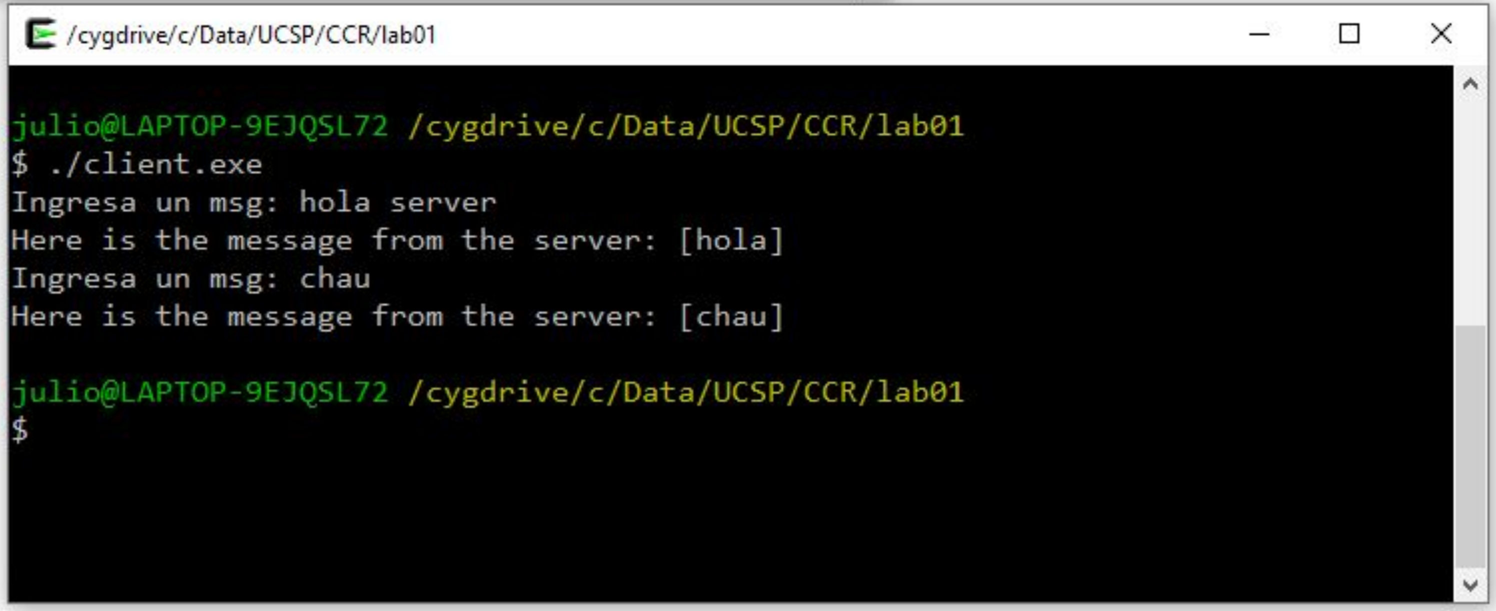
```
    for(;;)
    {
      int ClienteSD = accept(ServerSD, NULL, NULL);

      if(0 > ClienteSD)
      {
        perror("error accept failed");
        close(ServerSD);
        exit(EXIT_FAILURE);
      }

      std::thread (process_client_thread,ClienteSD).detach();
    }
```

http://www.cplusplus.com/reference/thread/thread/detach/

Lab01.4

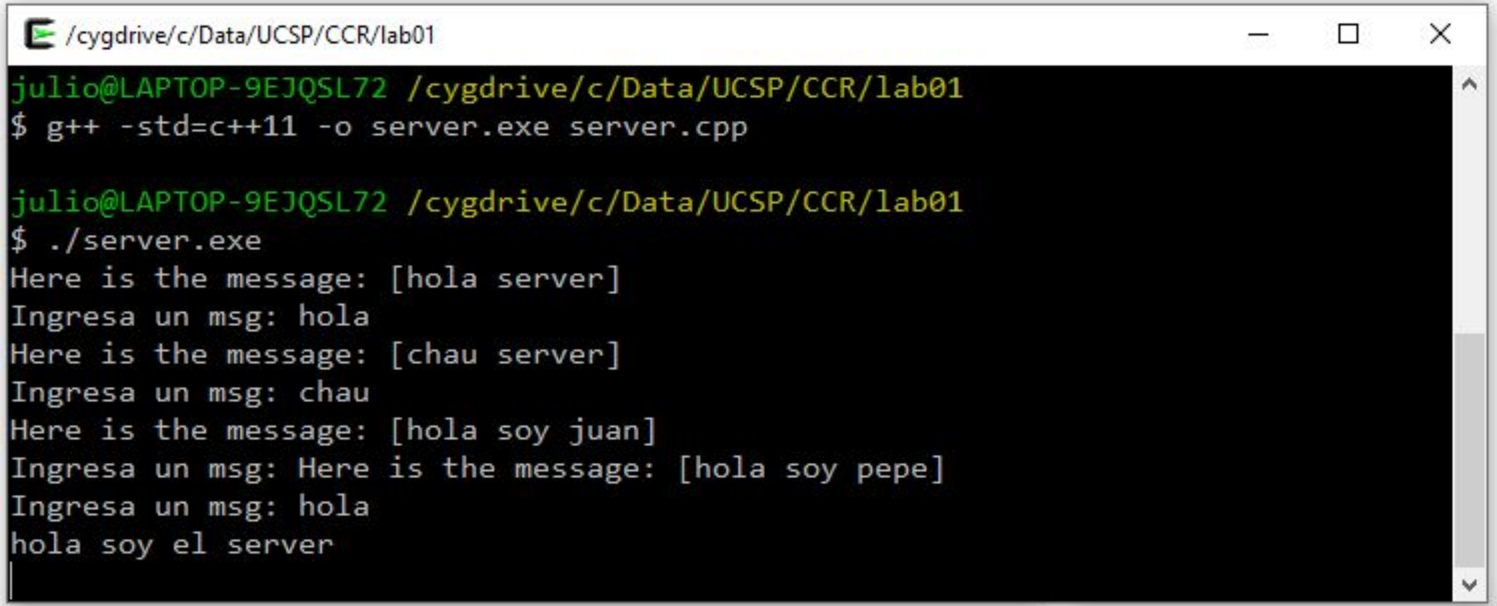


```
julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$ g++ -std=c++11 -o server.exe server.cpp

julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$ ./server.exe
Here is the message: [hola server]
Ingresa un msg: hola
Here is the message: [chau server]
Ingresa un msg: chau
```

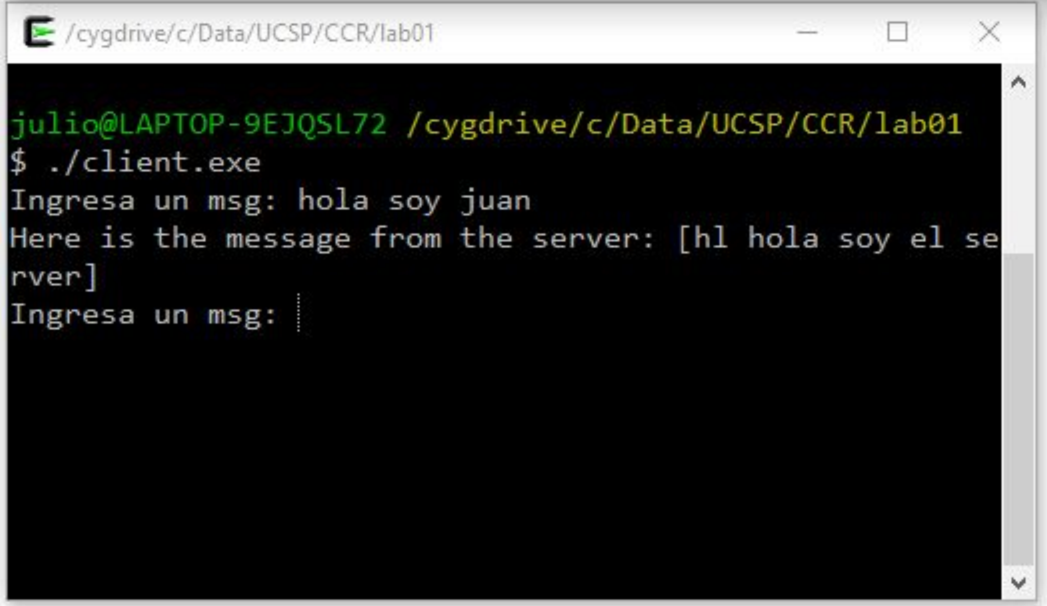


```
julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$ ./client.exe
Ingresa un msg: hola server
Here is the message from the server: [hola]
Ingresa un msg: chau
Here is the message from the server: [chau]

julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$
```

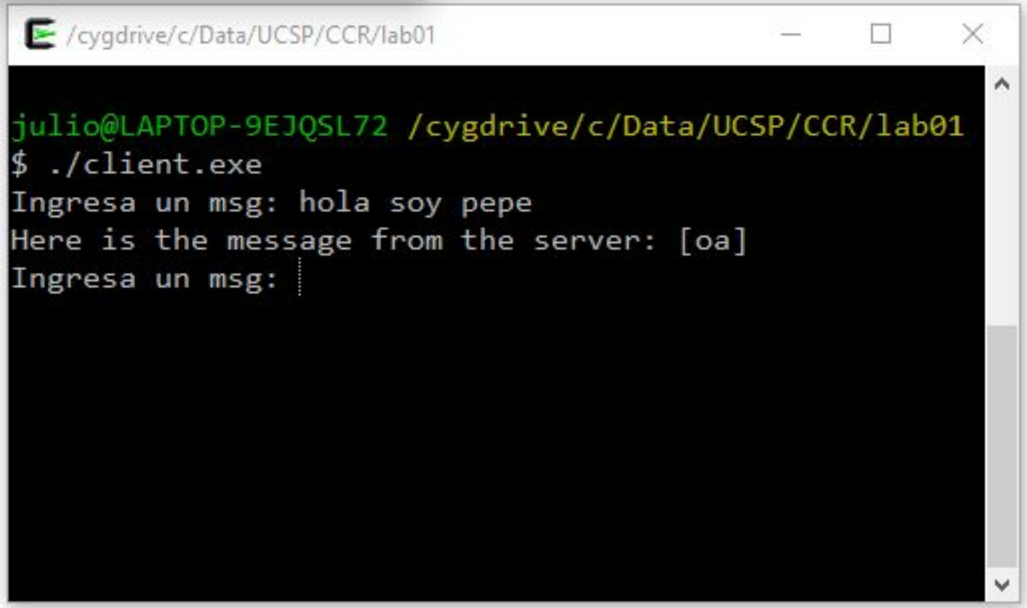
Lab01.4



```
julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$ g++ -std=c++11 -o server.exe server.cpp

julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$ ./server.exe
Here is the message: [hola server]
Ingresa un msg: hola
Here is the message: [chau server]
Ingresa un msg: chau
Here is the message: [hola soy juan]
Ingresa un msg: Here is the message: [hola soy pepe]
Ingresa un msg: hola
hola soy el server
```



```
julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$ ./client.exe
Ingresa un msg: hola soy juan
Here is the message from the server: [hl hola soy el se
rver]
Ingresa un msg:
```

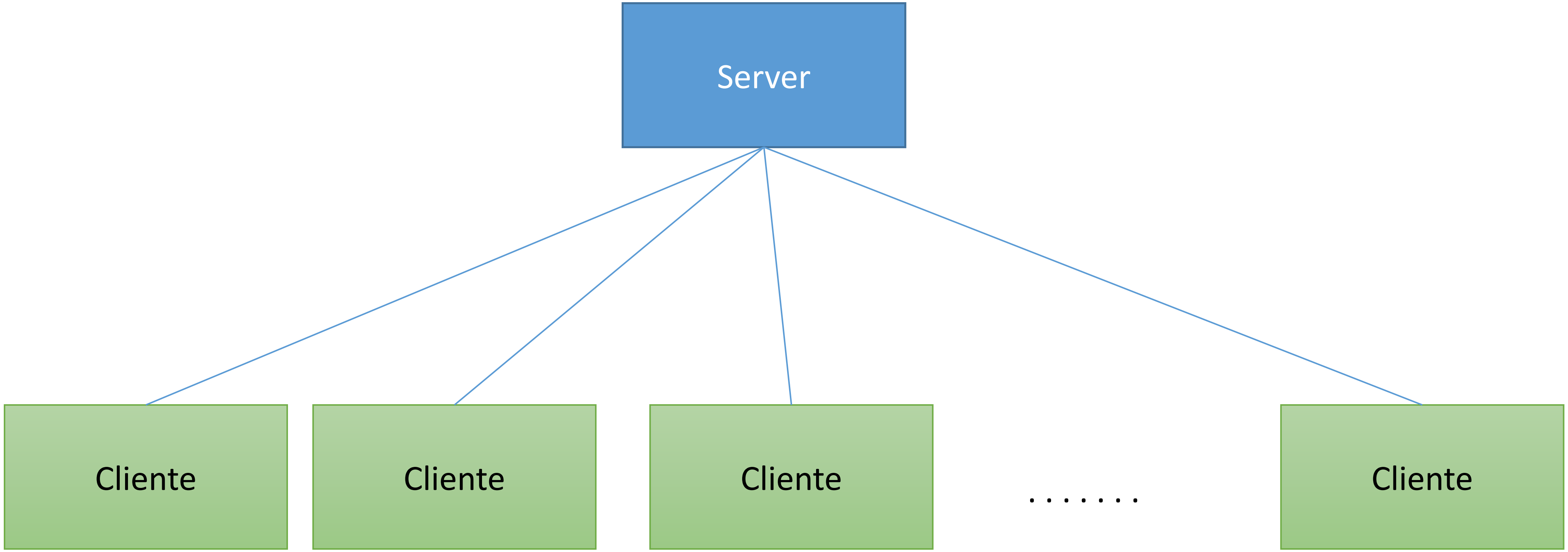


```
julio@LAPTOP-9EJQSL72 /cygdrive/c/Data/UCSP/CCR/lab01
$ ./client.exe
Ingresa un msg: hola soy pepe
Here is the message from the server: [oa]
Ingresa un msg:
```

Tarea de laboratorio 01

- Hacer una calculadora
  - Se le envía 2 valores y el tipo de operación
  - el servidor debe devolver la respuesta indicando el tipo de operación
- El servidor debe implementar threads
- Cliente:
- Ingrese su operación: 562 326 +
- El servidor responde: La suma es 888

Tarea de laboratorio 02



Tarea de laboratorio 02

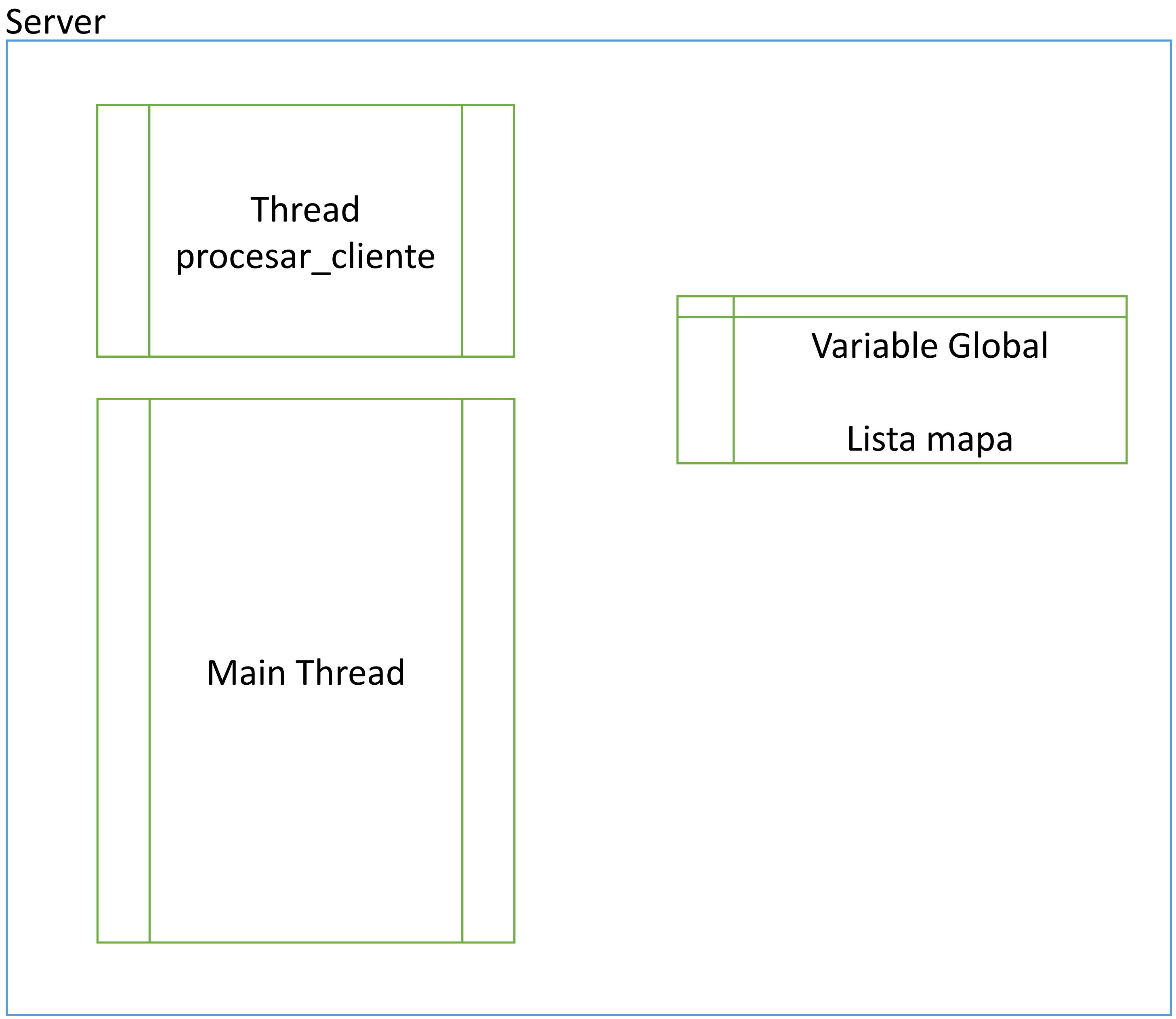
- Implementar un chat entre clientes
- El servidor deberá mantener una lista de clientes con su nickname y socket descriptor
  - Utilizar un mapa, que sea una variable global

    std::map<string,int> lista de clientes;
- El cliente deberá enviar
  - Nickname a con quien comunicarse
  - Segundo el mensaje a enviar
  - Si solo se envía el nickname, este será considerado como el nickname del cliente
  - Si el Nick name es igual a “lista”, el servidor enviara la lista de nicknames al cliente
- El Thread en el servidor
  - Debe buscar el nickname en el mapa y extraer el socket descriptor
  - si no existe debe de enviar ‘nickname no existe’

Tarea de laboratorio 02

- El cliente deberá de implementar 02 threads
  - Uno para leer del socket e imprimir en el terminal
    - por el momento la lectura es una especia de pulling o busywait
  - Otro para leer los mensajes del terminal y enviarlo a través del socket
- El servidor deberá implementar solo un thread, pero con una instancias por cliente.
  - Implementa un servicio de pase de mensajes
  - Registro de nickname
  - Responder a nickname = “lista”, este será un comando

http://www.cplusplus.com/reference/map/map/operator[]/



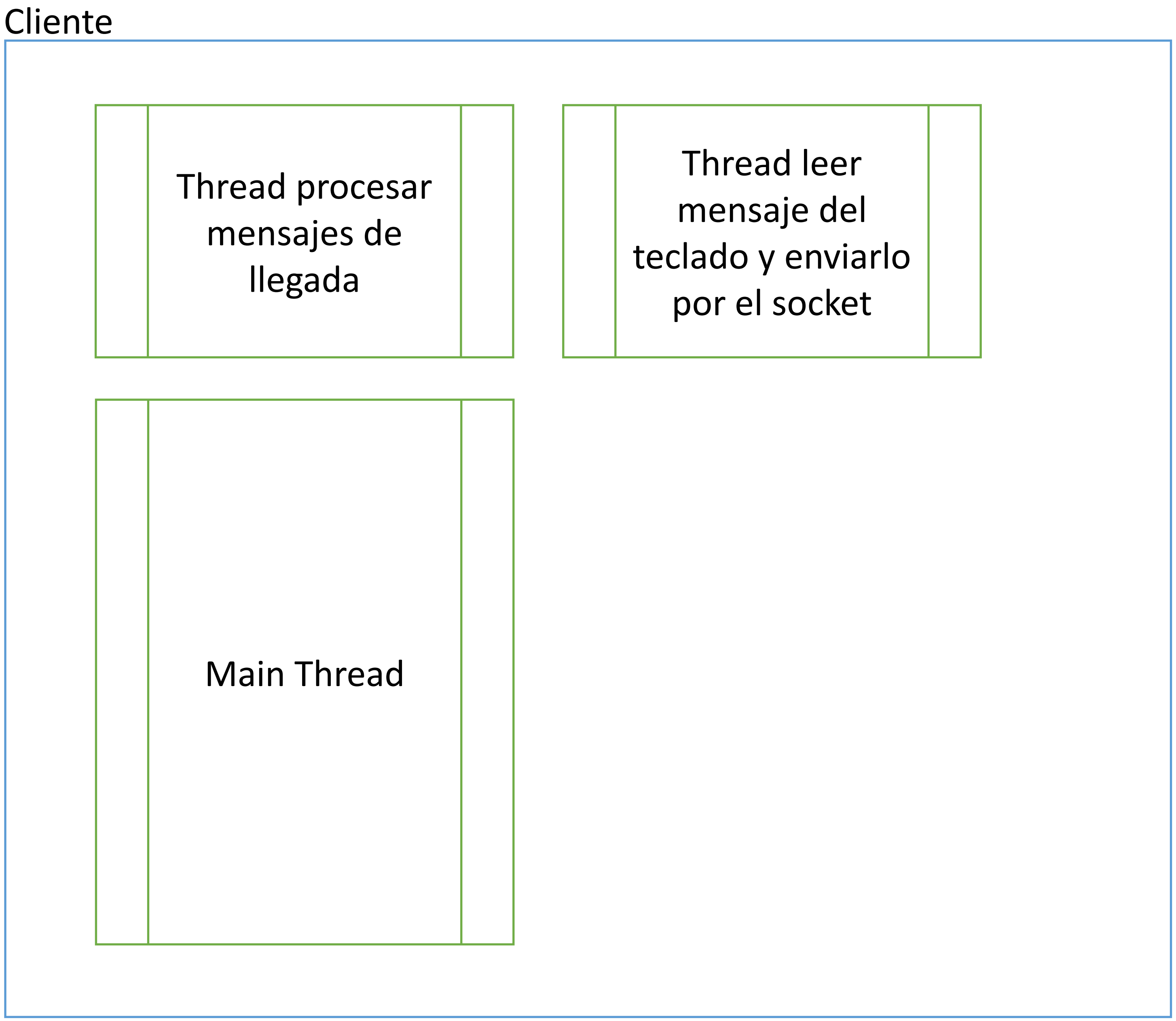
Server

Thread procesar_cliente

Main Thread

Variable Global

Lista mapa



Cliente

Thread procesar mensajes de llegada

Thread leer mensaje del teclado y enviarlo por el socket

Main Thread