

The first screenshot shows the installation of Ubuntu on Windows using WSL. The terminal displays the following steps:

```

y "wsl.exe --install
Ubuntu ya está instalado.
Iniciando Ubuntu...
Las distribuciones tam...
Installing, this may take a few minutes...
Error code: Wsl/WSL_E...
sdPlease create a default UNIX user account. The username does not need to match your Windows username.
For more information visit: https://aka.ms/wslusers
system32 wsl.exe
A continuación, se mue...
Enter new UNIX username: igma
Instalar con "wsl.exe New password:
NAME
Ubuntu
Debian
kali-linux
Ubuntu-18.04
Ubuntu-20.04
Ubuntu-22.04
OracleLinux_7_9
OracleLinux_8_7
OracleLinux_9_1
openSUSE-Leap-15.5
SUSE-Linux-Enterprise-
SUSE-Linux-Enterprise-
openSUSE-Tumbleweed
system32 wsl.exe -
Instalando: Debian GNU
se ha instalado Debian
Iniciando Debian GNU/L
Installing, this may t
please create a defaul
For more information v
Enter new UNIX usernam
New password:
Retype new password:
password: password updated successfully
Installation successful!
igma@josefernandez: $

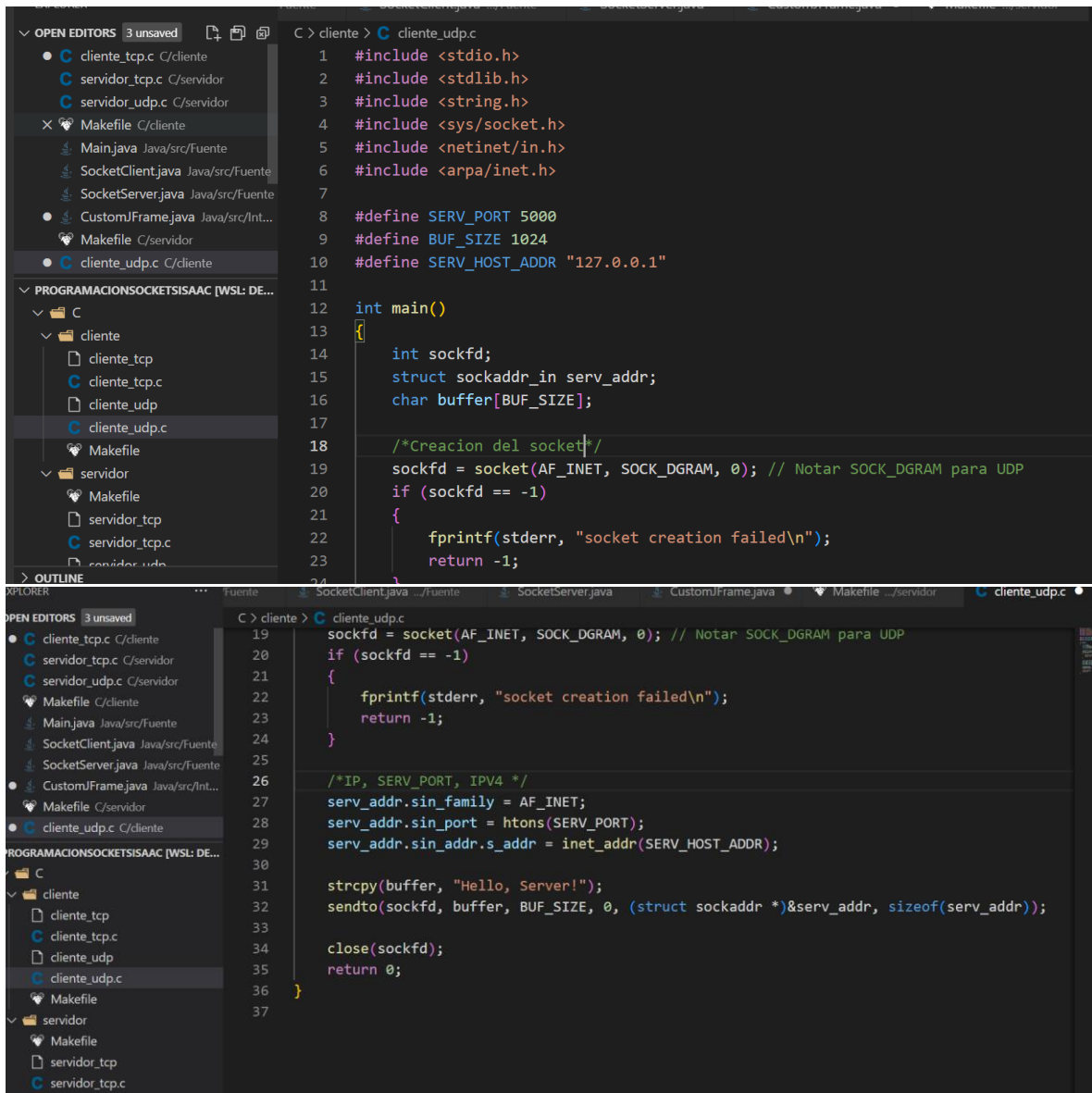
```

The second screenshot shows the development of a TCP client in C. The code is as follows:

```

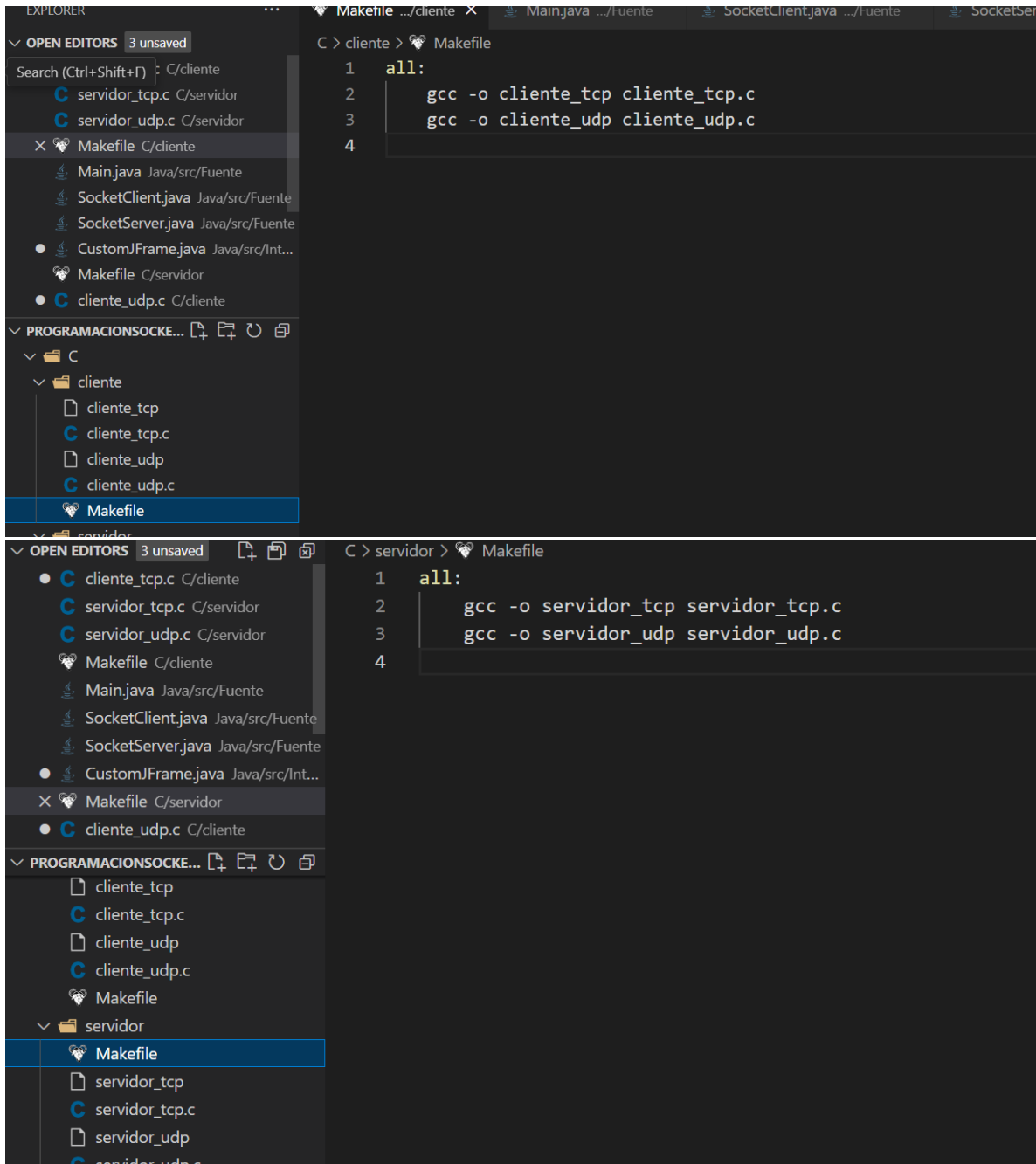
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #include <unistd.h>
5 #include <sys/socket.h>
6 #include <netinet/in.h>
7 #include <arpa/inet.h>
8
9 #define SERV_PORT 5000
10 #define BUF_SIZE 1024
11 #define SERV_HOST_ADDR "127.0.0.1"
12
13 int main()
14 {
15     int sockfd;
16     struct sockaddr_in serv_addr;
17     char buffer[BUF_SIZE];
18
19     /*creacion de socket*/
20     sockfd = socket(AF_INET, SOCK_STREAM, 0); // Notar SOCK_STREAM para TCP
21     if (sockfd == -1)
22     {
23         fprintf(stderr, "socket creation failed\n");
24         return -1;
25     }
26
27     /*asignar IP, SERV_PORT, IPV4*/
28     serv_addr.sin_family = AF_INET;
29     serv_addr.sin_port = htons(SERV_PORT);
30     serv_addr.sin_addr.s_addr = inet_addr(SERV_HOST_ADDR);
31
32     /*Conectar al servidor*/
33     if (connect(sockfd, (struct sockaddr *)&serv_addr, sizeof(serv_addr)) != 0)
34     {
35         fprintf(stderr, "connection with the server failed\n");
36         return -1;
37     }
38
39     strcpy(buffer, "Hello, Server!");
40     write(sockfd, buffer, sizeof(buffer));
41
42     close(sockfd);
43     return 0;
44 }

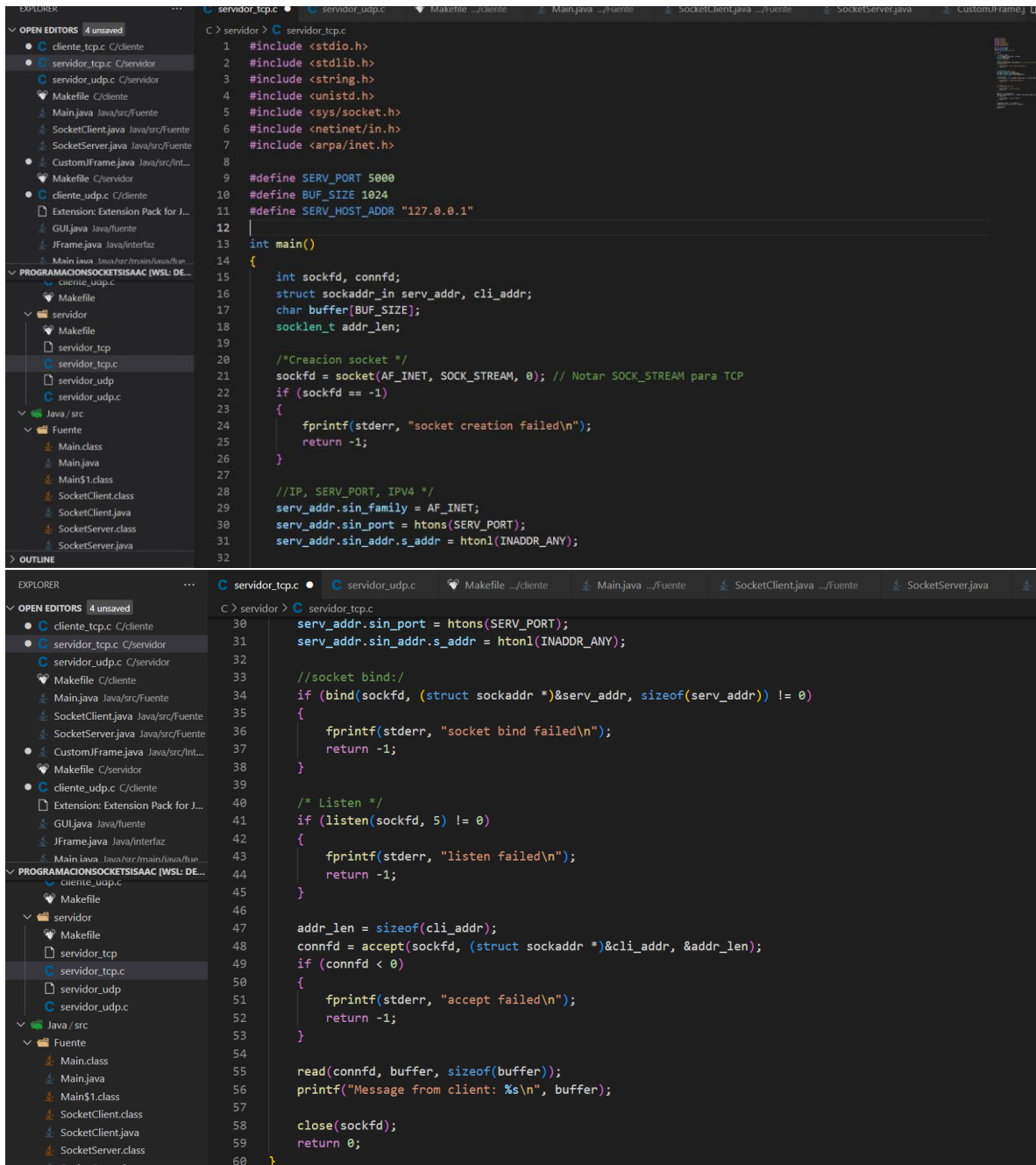
```



The image shows a code editor with a sidebar on the left containing a file explorer and an outline. The main editor area displays the code for 'cliente_udp.c' in the 'cliente' directory. The code is in C and implements a UDP client. It includes headers for `stdio.h`, `stdlib.h`, `string.h`, `sys/socket.h`, `netinet/in.h`, and `arpa/inet.h`. It defines `SERV_PORT` as 5000, `BUF_SIZE` as 1024, and `SERV_HOST_ADDR` as "127.0.0.1". The `main` function creates a socket using `socket(AF_INET, SOCK_DGRAM, 0)`, checks for errors, and if successful, sends a "Hello, Server!" message to the server using `sendto`. Finally, it closes the socket and returns 0.

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4  #include <sys/socket.h>
5  #include <netinet/in.h>
6  #include <arpa/inet.h>
7
8  #define SERV_PORT 5000
9  #define BUF_SIZE 1024
10 #define SERV_HOST_ADDR "127.0.0.1"
11
12 int main()
13 {
14     int sockfd;
15     struct sockaddr_in serv_addr;
16     char buffer[BUF_SIZE];
17
18     /*Creacion del socket*/
19     sockfd = socket(AF_INET, SOCK_DGRAM, 0); // Notar SOCK_DGRAM para UDP
20     if (sockfd == -1)
21     {
22         fprintf(stderr, "socket creation failed\n");
23         return -1;
24     }
25
26     /*IP, SERV_PORT, IPV4 */
27     serv_addr.sin_family = AF_INET;
28     serv_addr.sin_port = htons(SERV_PORT);
29     serv_addr.sin_addr.s_addr = inet_addr(SERV_HOST_ADDR);
30
31     strcpy(buffer, "Hello, Server!");
32     sendto(sockfd, buffer, BUF_SIZE, 0, (struct sockaddr *)&serv_addr, sizeof(serv_addr));
33
34     close(sockfd);
35     return 0;
36 }
37
```





```
serveridor_tcp.c
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4  #include <unistd.h>
5  #include <sys/socket.h>
6  #include <netinet/in.h>
7  #include <arpa/inet.h>
8
9  #define SERV_PORT 5000
10 #define BUF_SIZE 1024
11 #define SERV_HOST_ADDR "127.0.0.1"
12
13 int main()
14 {
15     int sockfd, connfd;
16     struct sockaddr_in serv_addr, cli_addr;
17     char buffer[BUF_SIZE];
18     socklen_t addr_len;
19
20     /*Creacion socket */
21     sockfd = socket(AF_INET, SOCK_STREAM, 0); // Notar SOCK_STREAM para TCP
22     if (sockfd == -1)
23     {
24         fprintf(stderr, "socket creation failed\n");
25         return -1;
26     }
27
28     //IP, SERV_PORT, IPV4 */
29     serv_addr.sin_family = AF_INET;
30     serv_addr.sin_port = htons(SERV_PORT);
31     serv_addr.sin_addr.s_addr = htonl(INADDR_ANY);
32
33     serv_addr.sin_port = htons(SERV_PORT);
34     serv_addr.sin_addr.s_addr = htonl(INADDR_ANY);
35
36     //socket bind:/
37     if (bind(sockfd, (struct sockaddr *)&serv_addr, sizeof(serv_addr)) != 0)
38     {
39         fprintf(stderr, "socket bind failed\n");
40         return -1;
41     }
42
43     /* Listen */
44     if (listen(sockfd, 5) != 0)
45     {
46         fprintf(stderr, "listen failed\n");
47         return -1;
48     }
49
50     addr_len = sizeof(cli_addr);
51     connfd = accept(sockfd, (struct sockaddr *)&cli_addr, &addr_len);
52     if (connfd < 0)
53     {
54         fprintf(stderr, "accept failed\n");
55         return -1;
56     }
57
58     read(connfd, buffer, sizeof(buffer));
59     printf("Message from client: %s\n", buffer);
60
61     close(sockfd);
62     return 0;
63 }
```

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4  #include <unistd.h>
5  #include <sys/socket.h>
6  #include <netinet/in.h>
7
8  #define SERV_PORT 5000
9  #define BUF_SIZE 1024
10 #define SERV_HOST_ADDR "127.0.0.1"
11
12 int main()
13 {
14     int sockfd;
15     struct sockaddr_in serv_addr, cli_addr;
16     char buffer[BUF_SIZE];
17     socklen_t addr_len;
18
19     /*Creacion */
20     sockfd = socket(AF_INET, SOCK_DGRAM, 0); // Notar SOCK_DGRAM para UDP
21     if (sockfd == -1)
22     {
23         fprintf(stderr, "socket creation failed\n");
24         return -1;
25     }
26
27     /*Asignacion*/
28     serv_addr.sin_family = AF_INET;
29     serv_addr.sin_port = htons(SERV_PORT);
30     serv_addr.sin_addr.s_addr = htonl(INADDR_ANY);
31
32     /*bind*/
33     if (bind(sockfd, (struct sockaddr *)&serv_addr, sizeof(serv_addr)) != 0)
34     {
35         fprintf(stderr, "socket bind failed\n");
36         return -1;
37     }
38
39     addr_len = sizeof(cli_addr);
40     recvfrom(sockfd, buffer, BUF_SIZE, 0, (struct sockaddr *)&cli_addr, &addr_len);
41     printf("Message from client: %s\n", buffer);
42
43     close(sockfd);
44     return 0;
45 }
46

```

```

C > servidor > Makefile
1  all:
2      gcc -o servidor_tcp servidor_tcp.c
3      gcc -o servidor_udp servidor_udp.c
4

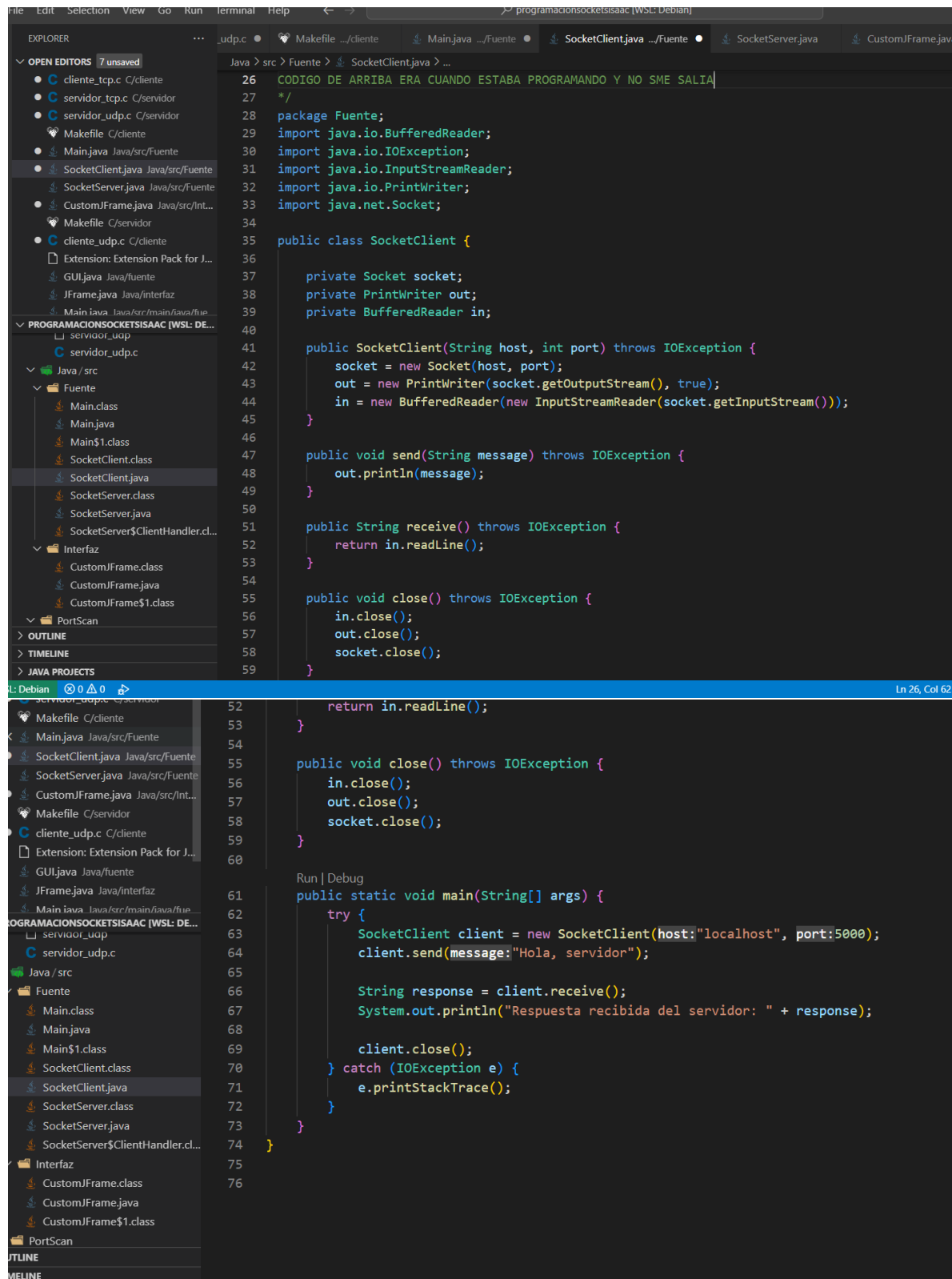
```

```

1  package Fuente;
2  import java.awt.EventQueue;
3  import Interfaz.CustomJFrame;
4
5  public class Main {
6      Run | Debug
7      public static void main(String[] args) {
8          EventQueue.invokeLater(new Runnable() {
9              public void run() {
10                 try {
11                     CustomJFrame frame = new CustomJFrame();
12                     frame.setVisible(true);
13                 } catch (Exception e) {
14                     e.printStackTrace();
15                 }
16             }
17         });
18     }
19     //CODIGO MAL PORQUE NO USE GUI USE UN CUSTOM:
20     /*
21     package Fuente;
22     import java.awt.EventQueue;
23
24     public class Main {
25         public static void main(String[] args) {
26             EventQueue.invokeLater(new Runnable() {
27                 public void run() {
28                     try {
29                         GUI frame = new GUI();
30                         frame.setVisible(true);

```

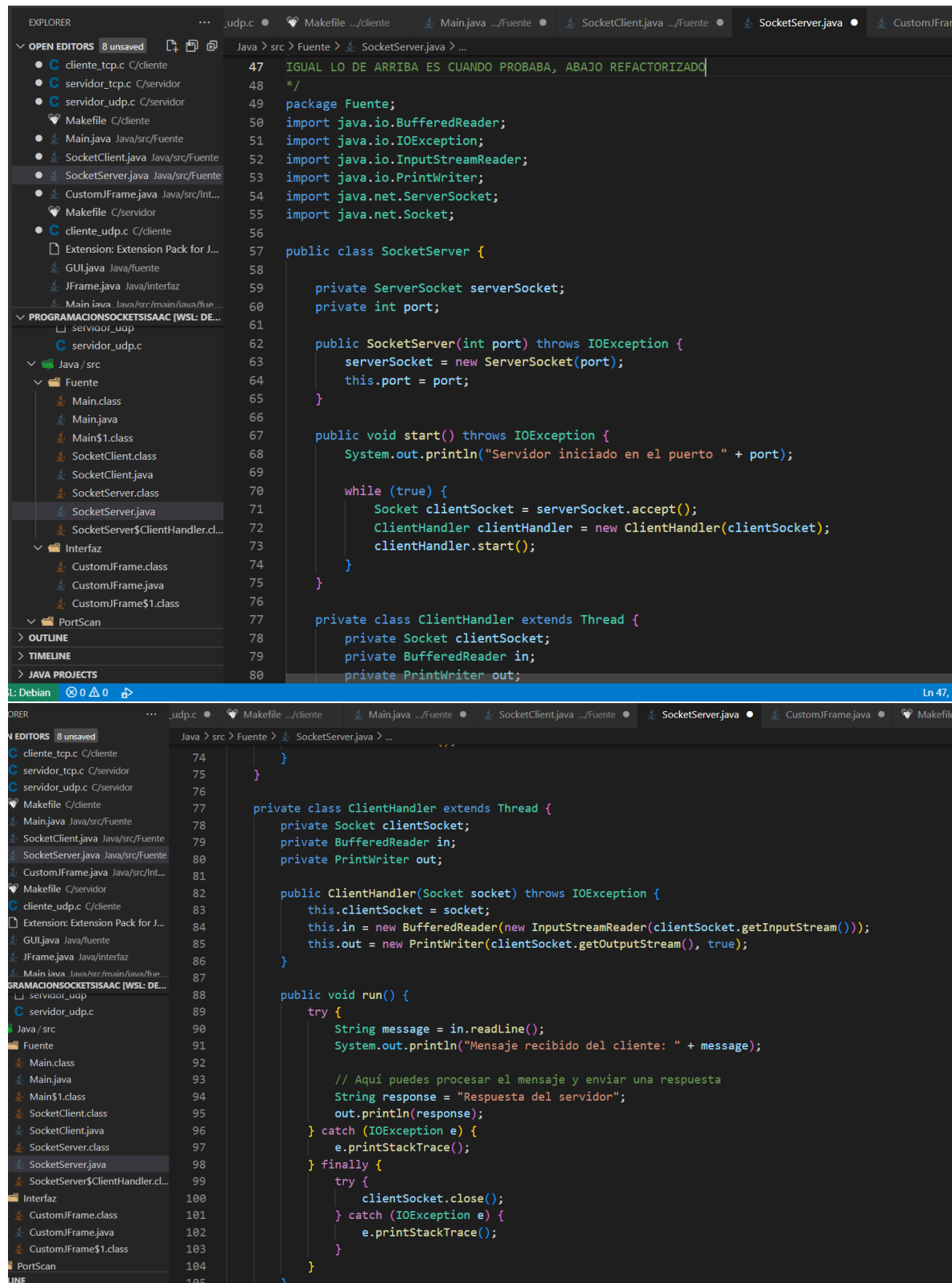
Isaac Reyes



```
file Edit Selection View Go Run Terminal Help
programacionsocketsisaac [WSL: Debian]

EXPLORER
OPEN EDITORS 7 unsaved
  cliente_tcp.c C/cliente
  servidor_tcp.c C/servidor
  servidor_udp.c C/servidor
  Makefile C/cliente
  Main.java Java/src/Fuente
  SocketClient.java Java/src/Fuente
  SocketServer.java Java/src/Fuente
  CustomJFrame.java Java/src/Interfaz
  Makefile C/servidor
  cliente_udp.c C/cliente
  Extension: Extension Pack for J...
  GUI.java Java/fuente
  JFrame.java Java/interfaz
  Main.java Java/src/main/java/fu...
  PROGRAMACIONSOCKETSSISAAC [WSL: DE...
    servidor_udp
    C servidor_udp.c
  Java / src
    Fuente
      Main.class
      Main.java
      Main$1.class
      SocketClient.class
      SocketClient.java
      SocketServer.class
      SocketServer.java
      SocketServer$ClientHandler.cl...
    Interfaz
      CustomJFrame.class
      CustomJFrame.java
      CustomJFrame$1.class
    PortScan
  OUTLINE
  TIMELINE
  JAVA PROJECTS

Java > src > Fuente > SocketClient.java > ...
26 CODIGO DE ARRIBA ERA CUANDO ESTABA PROGRAMANDO Y NO SME SALIA
27 */
28 package Fuente;
29 import java.io.BufferedReader;
30 import java.io.IOException;
31 import java.io.InputStreamReader;
32 import java.io.PrintWriter;
33 import java.net.Socket;
34
35 public class SocketClient {
36
37     private Socket socket;
38     private PrintWriter out;
39     private BufferedReader in;
40
41     public SocketClient(String host, int port) throws IOException {
42         socket = new Socket(host, port);
43         out = new PrintWriter(socket.getOutputStream(), true);
44         in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
45     }
46
47     public void send(String message) throws IOException {
48         out.println(message);
49     }
50
51     public String receive() throws IOException {
52         return in.readLine();
53     }
54
55     public void close() throws IOException {
56         in.close();
57         out.close();
58         socket.close();
59     }
60
61     Run | Debug
62     public static void main(String[] args) {
63         try {
64             SocketClient client = new SocketClient(host:"localhost", port:5000);
65             client.send(message:"Hola, servidor");
66
67             String response = client.receive();
68             System.out.println("Respuesta recibida del servidor: " + response);
69
70             client.close();
71         } catch (IOException e) {
72             e.printStackTrace();
73         }
74     }
75
76 }
```



```
47 IGUAL LO DE ARRIBA ES CUANDO PROBABA, ABAJO REFACTORIZADO
48 */
49 package Fuente;
50 import java.io.BufferedReader;
51 import java.io.IOException;
52 import java.io.InputStreamReader;
53 import java.io.PrintWriter;
54 import java.net.ServerSocket;
55 import java.net.Socket;
56
57 public class SocketServer {
58
59     private ServerSocket serverSocket;
60     private int port;
61
62     public SocketServer(int port) throws IOException {
63         serverSocket = new ServerSocket(port);
64         this.port = port;
65     }
66
67     public void start() throws IOException {
68         System.out.println("Servidor iniciado en el puerto " + port);
69
70         while (true) {
71             Socket clientSocket = serverSocket.accept();
72             ClientHandler clientHandler = new ClientHandler(clientSocket);
73             clientHandler.start();
74         }
75     }
76
77     private class ClientHandler extends Thread {
78         private Socket clientSocket;
79         private BufferedReader in;
80         private PrintWriter out;
81
82         public ClientHandler(Socket socket) throws IOException {
83             this.clientSocket = socket;
84             this.in = new BufferedReader(new InputStreamReader(clientSocket.getInputStream()));
85             this.out = new PrintWriter(clientSocket.getOutputStream(), true);
86         }
87
88         public void run() {
89             try {
90                 String message = in.readLine();
91                 System.out.println("Mensaje recibido del cliente: " + message);
92
93                 // Aquí puedes procesar el mensaje y enviar una respuesta
94                 String response = "Respuesta del servidor";
95                 out.println(response);
96             } catch (IOException e) {
97                 e.printStackTrace();
98             } finally {
99                 try {
100                     clientSocket.close();
101                 } catch (IOException e) {
102                     e.printStackTrace();
103                 }
104             }
105         }
106     }
107 }
```


The image shows a screenshot of an IDE with two open Java files. The top file is `SocketServer.java` and the bottom file is `CustomJFrame.java`.

SocketServer.java

```

98         } finally {
99             try {
100                 clientSocket.close();
101             } catch (IOException e) {
102                 e.printStackTrace();
103             }
104         }
105     }
106 }
107
108 Run | Debug
109 public static void main(String[] args) {
110     try {
111         SocketServer server = new SocketServer(port:5000);
112         server.start();
113     } catch (IOException e) {
114         e.printStackTrace();
115     }
116 }
117

```

CustomJFrame.java

```

1 package Interfaz;
2
3 import java.io.IOException;
4 import javax.swing.JButton;
5 import javax.swing.JFrame;
6 import javax.swing.JTextField;
7 import java.awt.event.ActionListener;
8 import java.awt.event.ActionEvent;
9 import Fuente.SocketClient;
10
11 public class CustomJFrame extends JFrame {
12     private JTextField textField;
13     private SocketClient client;
14
15     public CustomJFrame() {
16         setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
17         setBounds(100, 100, 450, 300);
18         getContentPane().setLayout(null);
19
20         JButton btnSend = new JButton("Enviar");
21         btnSend.addActionListener(new ActionListener() {
22             public void actionPerformed(ActionEvent e) {
23                 String message = textField.getText();
24                 try {
25                     if (client == null) {
26                         client = new SocketClient(host:"localhost", port:5000);
27                     }
28                     client.send(message);
29                 } catch (IOException ex) {
30                     ex.printStackTrace();
31                 }
32             }
33         });
34         btnSend.setBounds(335, 227, 89, 23);
35

```

The screenshot displays an IDE with two main panels. The top panel shows the Java source code for `CustomJFrame.java`, which is part of a GUI application. The bottom panel shows the C source code for `portscan.c`, which implements the port scanning logic. A terminal window at the bottom shows the compilation and execution commands.

Java Code (CustomJFrame.java):

```
31     }
32 }
33
34 btnSend.setBounds(335, 227, 89, 23);
35 getContentPane().add(btnSend);
36
37 textField = new JTextField();
38 textField.setBounds(10, 228, 315, 20);
39 getContentPane().add(textField);
40 textField.setColumns(10);
41 }
42 }
```

C Code (portscan.c):

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <sys/socket.h>
4 #include <arpa/inet.h>
5 #include <string.h>
6
7 int main(int argc, char *argv[]) {
8     int sock, port;
9     struct sockaddr_in target;
10
11     // Socket
12     if ((sock = socket(AF_INET, SOCK_STREAM, 0)) < 0) {
13         perror("socket");
14         return 1;
15     }
16
17     //Establezco parametros del server
18     target.sin_family = AF_INET;
19     target.sin_addr.s_addr = inet_addr("127.0.0.1");
20
21     //65535 ports
22     for (port = 1; port <= 65535; port++) {
23         target.sin_port = htons(port);
24         if (connect(sock, (struct sockaddr *)&target, sizeof(target)) == 0) {
25             printf("Puerto %d abierto\n", port);
26         }
27     }
28
29     close(sock);
30
31     return 0;
32 }
```

Terminal Output:

```
PortScan > Makefile
1 all:
2 gcc -o portscan portscan.c
3
4 clean:
5 rm -f portscan
6
```

```

PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE

• igra@josefernandez:~/programacionsocketsisaac$ cd C/cliente
• igra@josefernandez:~/programacionsocketsisaac/C/cliente$ make
gcc -o cliente_tcp cliente_tcp.c
gcc -o cliente_udp cliente_udp.c
cliente_udp.c: In function 'main':
cliente_udp.c:34:5: warning: implicit declaration of function 'close'; did you mean 'pclose'? [-Wimplicit-function-declaration]
    34 |     close(sockfd);
        |     ~~~~~
        |     pclose
• igra@josefernandez:~/programacionsocketsisaac/C/cliente$ cd ..
• igra@josefernandez:~/programacionsocketsisaac/C$ servidor
bash: servidor: command not found
• igra@josefernandez:~/programacionsocketsisaac/C$ cd servidor
• igra@josefernandez:~/programacionsocketsisaac/C/servidor$ make
gcc -o servidor_tcp servidor_tcp.c
gcc -o servidor_udp servidor_udp.c
• igra@josefernandez:~/programacionsocketsisaac/C/servidor$ cd ..
• igra@josefernandez:~/programacionsocketsisaac/C$ cd ..
• igra@josefernandez:~/programacionsocketsisaac$ cd PortScan
• igra@josefernandez:~/programacionsocketsisaac/PortScan$ make
gcc -o portscan portscan.c
portscan.c: In function 'main':
portscan.c:29:5: warning: implicit declaration of function 'close'; did you mean 'pclose'? [-Wimplicit-function-declaration]
    29 |     close(sock);
        |     ~~~~~
        |     pclose
• igra@josefernandez:~/programacionsocketsisaac/PortScan$ cd ..
• igra@josefernandez:~/programacionsocketsisaac$ cd Java/src
• igra@josefernandez:~/programacionsocketsisaac/Java/src$ javac Fuente/*.java Interfaz/*.java
• igra@josefernandez:~/programacionsocketsisaac/Java/src$

PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE

• igra@josefernandez:~/programacionsocketsisaac$ cd C/servidor
• igra@josefernandez:~/programacionsocketsisaac/C/servidor$ ./servidor_tcp
Message from client: Hello, Server!
• igra@josefernandez:~/programacionsocketsisaac/C/servidor$

bash src
bash...
bash servidor
bash cliente
java src
bash PortScan

```

CONEXIÓN

```

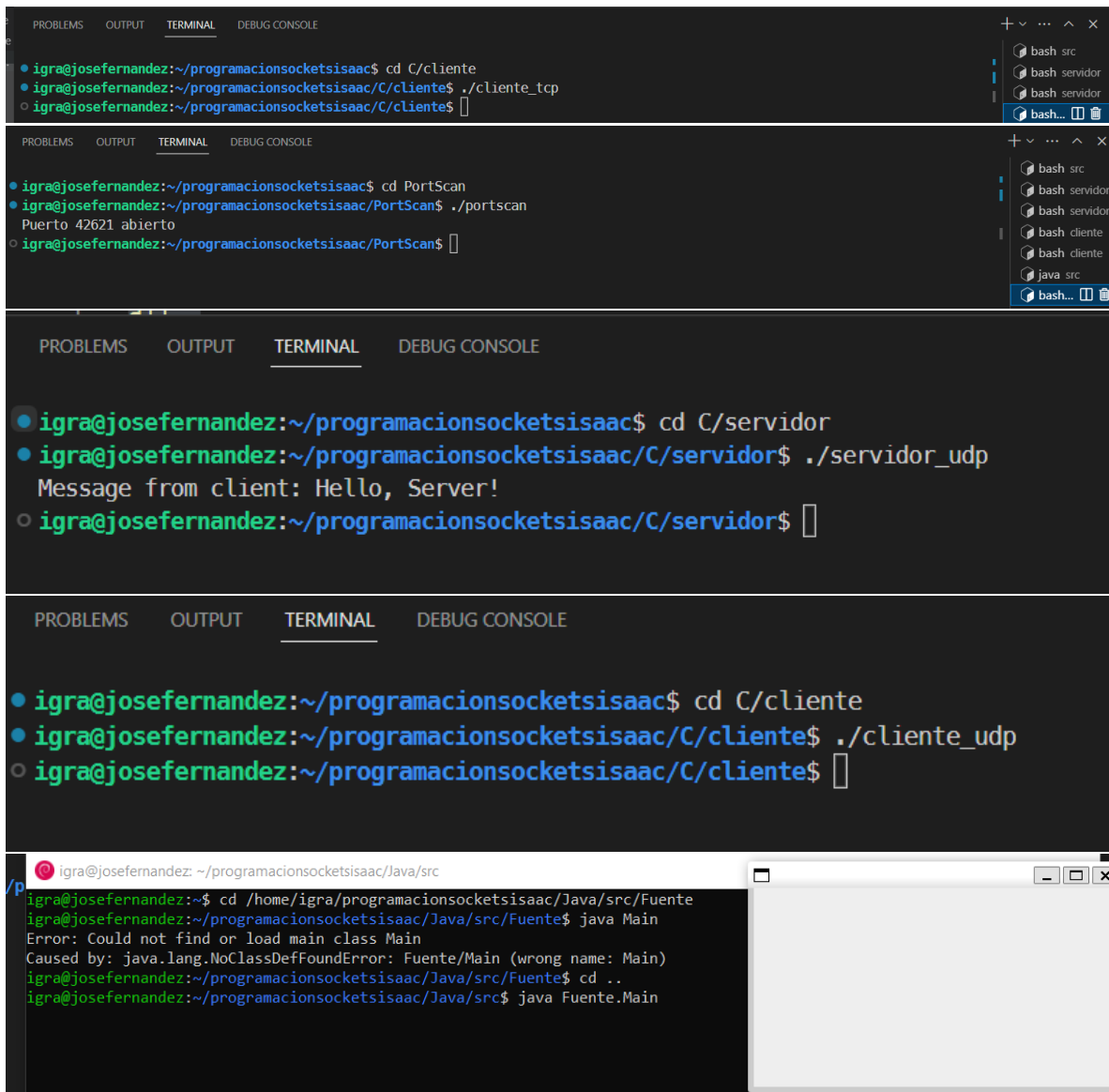
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE

• igra@josefernandez:~/programacionsocketsisaac$ cd C/servidor
• igra@josefernandez:~/programacionsocketsisaac/C/servidor$ ./servidor_udp
Message from client: Hello, Server!
• igra@josefernandez:~/programacionsocketsisaac/C/servidor$

PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE

• igra@josefernandez:~/programacionsocketsisaac$ cd C/servidor
• igra@josefernandez:~/programacionsocketsisaac/C/servidor$ ./servidor_udp

```



```
igra@josefernandez:~/programacionsocketsisaac$ cd C/cliente
igra@josefernandez:~/programacionsocketsisaac/C/cliente$ ./cliente_tcp
igra@josefernandez:~/programacionsocketsisaac/C/cliente$

igra@josefernandez:~/programacionsocketsisaac$ cd PortScan
igra@josefernandez:~/programacionsocketsisaac/PortScan$ ./portscan
Puerto 42621 abierto
igra@josefernandez:~/programacionsocketsisaac/PortScan$

igra@josefernandez:~/programacionsocketsisaac$ cd C/servidor
igra@josefernandez:~/programacionsocketsisaac/C/servidor$ ./servidor_udp
Message from client: Hello, Server!
igra@josefernandez:~/programacionsocketsisaac/C/servidor$

igra@josefernandez:~/programacionsocketsisaac$ cd C/cliente
igra@josefernandez:~/programacionsocketsisaac/C/cliente$ ./cliente_udp
igra@josefernandez:~/programacionsocketsisaac/C/cliente$

igra@josefernandez: ~/programacionsocketsisaac/Java/src
igra@josefernandez:~$ cd /home/igra/programacionsocketsisaac/Java/src/Fuente
igra@josefernandez:~/programacionsocketsisaac/Java/src/Fuente$ java Main
Error: Could not find or load main class Main
Caused by: java.lang.NoClassDefFoundError: Fuente/Main (wrong name: Main)
igra@josefernandez:~/programacionsocketsisaac/Java/src/Fuente$ cd ..
igra@josefernandez:~/programacionsocketsisaac/Java/src$ java Fuente.Main
```

No se ve porque WSL no aguanta la gráfica.