



**Instituto Politécnico Nacional**  
**Escuela Superior De Computo**



**Aplicaciones para Comunicaciones en Red**

## **Ejercicio 2**

### **Ejercicio de sockets (envío de múltiples archivos)**

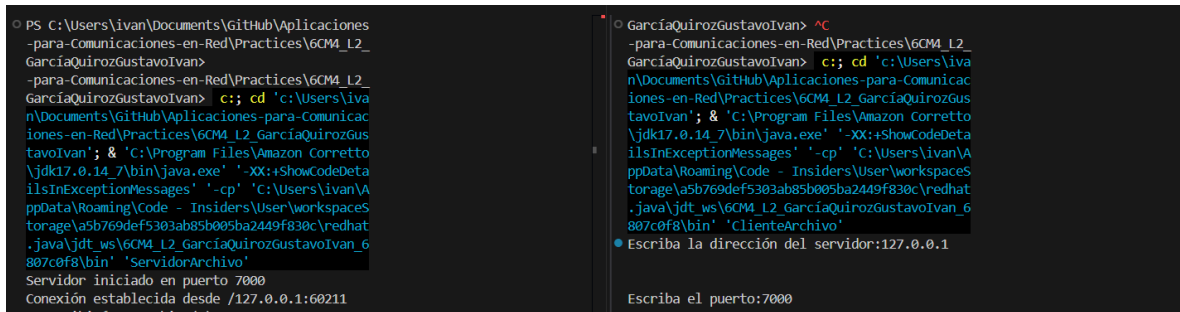
Nombre Del Alumno: García Quiroz Gustavo Ivan

Grupo: 6CM4

Nombre de la Profesora: Sandra Ivette Bautista Rosales

## Capturas de pantalla:

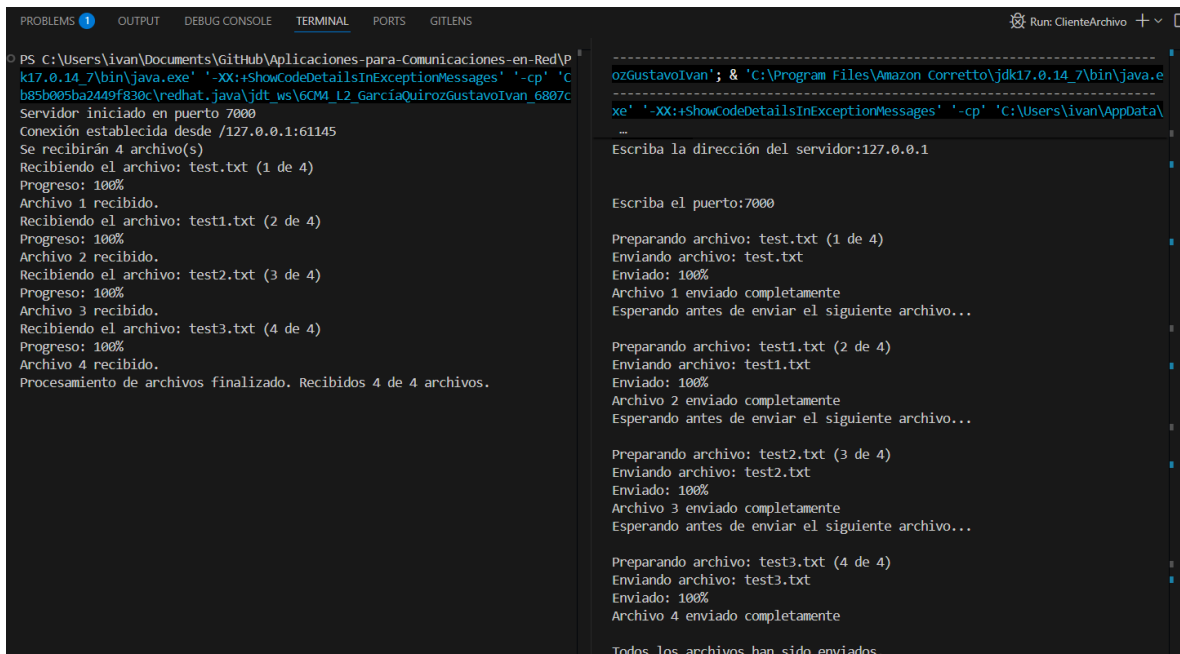
- La ejecución del programa
  - conexión entre cliente y servidor



```
PS C:\Users\ivan\Documents\GitHub\Aplicaciones-para-Comunicaciones-en-Red\Practices\6CM4_L2_GarcíaQuirozGustavoIvan>
-para-Comunicaciones-en-Red\Practices\6CM4_L2_GarcíaQuirozGustavoIvan> c;; cd 'c:\Users\ivan\Documents\GitHub\Aplicaciones-para-Comunicaciones-en-Red\Practices\6CM4_L2_GarcíaQuirozGustavoIvan'; & 'C:\Program Files\Amazon Corretto\jdk17.0.14_7\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\ivan\AppData\Roaming\Code - Insiders\User\workspaceStorage\5b769def5303ab85b005ba2449f830c\redhat.java\jdt_ws\6CM4_L2_GarcíaQuirozGustavoIvan_6807c0f8\bin' 'ServidorArchivo'
Servidor iniciado en puerto 7000
Conexión establecida desde /127.0.0.1:60211

GarcíaQuirozGustavoIvan> ^C
-para-Comunicaciones-en-Red\Practices\6CM4_L2_GarcíaQuirozGustavoIvan> c;; cd 'c:\Users\ivan\Documents\GitHub\Aplicaciones-para-Comunicaciones-en-Red\Practices\6CM4_L2_GarcíaQuirozGustavoIvan'; & 'C:\Program Files\Amazon Corretto\jdk17.0.14_7\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\ivan\AppData\Roaming\Code - Insiders\User\workspaceStorage\5b769def5303ab85b005ba2449f830c\redhat.java\jdt_ws\6CM4_L2_GarcíaQuirozGustavoIvan_6807c0f8\bin' 'ClienteArchivo'
Escriba la dirección del servidor:127.0.0.1
Escriba el puerto:7000
```

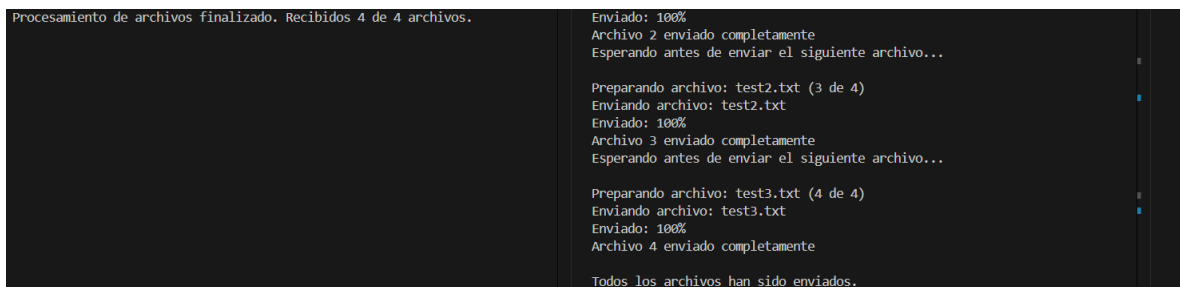
- progreso del envío de varios archivos



```
PS C:\Users\ivan\Documents\GitHub\Aplicaciones-para-Comunicaciones-en-Red\Practices\6CM4_L2_GarcíaQuirozGustavoIvan_6807c0f8>
k17.0.14_7\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\ivan\AppData\Roaming\Code - Insiders\User\workspaceStorage\5b769def5303ab85b005ba2449f830c\redhat.java\jdt_ws\6CM4_L2_GarcíaQuirozGustavoIvan_6807c0f8\bin' 'ServidorArchivo'
Servidor iniciado en puerto 7000
Conexión establecida desde /127.0.0.1:61145
Se recibirán 4 archivo(s)
Recibiendo el archivo: test.txt (1 de 4)
Progreso: 100%
Archivo 1 recibido.
Recibiendo el archivo: test1.txt (2 de 4)
Progreso: 100%
Archivo 2 recibido.
Recibiendo el archivo: test2.txt (3 de 4)
Progreso: 100%
Archivo 3 recibido.
Recibiendo el archivo: test3.txt (4 de 4)
Progreso: 100%
Archivo 4 recibido.
Procesamiento de archivos finalizado. Recibidos 4 de 4 archivos.

Run: ClienteArchivo
-----
Escriba la dirección del servidor:127.0.0.1
Escriba el puerto:7000
Preparando archivo: test.txt (1 de 4)
Enviando archivo: test.txt
Enviado: 100%
Archivo 1 enviado completamente
Esperando antes de enviar el siguiente archivo...
Preparando archivo: test1.txt (2 de 4)
Enviando archivo: test1.txt
Enviado: 100%
Archivo 2 enviado completamente
Esperando antes de enviar el siguiente archivo...
Preparando archivo: test2.txt (3 de 4)
Enviando archivo: test2.txt
Enviado: 100%
Archivo 3 enviado completamente
Esperando antes de enviar el siguiente archivo...
Preparando archivo: test3.txt (4 de 4)
Enviando archivo: test3.txt
Enviado: 100%
Archivo 4 enviado completamente
Todos los archivos han sido enviados.
```

- confirmación de los archivos recibidos



```
Procesamiento de archivos finalizado. Recibidos 4 de 4 archivos.
Enviado: 100%
Archivo 2 enviado completamente
Esperando antes de enviar el siguiente archivo...
Preparando archivo: test2.txt (3 de 4)
Enviando archivo: test2.txt
Enviado: 100%
Archivo 3 enviado completamente
Esperando antes de enviar el siguiente archivo...
Preparando archivo: test3.txt (4 de 4)
Enviando archivo: test3.txt
Enviado: 100%
Archivo 4 enviado completamente
Todos los archivos han sido enviados.
```

- The screenshot shows an IDE with the following components:

  - Explorer Panel:** Displays the project structure. The 'src' folder contains 'ServidorArchivo.java' and 'ClienteArchivo.java'.
  - Run Console:** Shows the output of the application. It indicates that the server is running on port 7000 and has received 4 files (test1.txt, test2.txt, test3.txt, and test4.txt).
  - Run Configuration:** The configuration is set to 'Run: ClienteArchivo'.

- 
- The screenshot shows an IDE with the following content:
- Explorer:**
    - 6CM4\_L2\_GARCIAQUIROZGUSTAV...
    - x
    - ClienteArchivo.java
    - ServidorArchivo.java
  - Editor:**
    - ClienteArchivo.java**

```

1  package com.garciaquirozgustavo;
2  import java.io.*;
3  import java.net.*;
4
5  public class ClienteArchivo {
6      public static void main(String[] args){
7
8      }
9
10     // Leemos y enviamos el contenido del archivo en pequeños bloques
11     fis = new FileInputStream(archivo);
12     byte[] b = new byte[1024];
13     long enviados = 0;
14
15     while (fis.available() > 0) {
16         b = fis.read(b);
17         enviar(b);
18     }
19
20     }
21
22     private void enviar(byte[] b) {
23         try {
24             Socket socket = new Socket("127.0.0.1", 7000);
25             OutputStream outputStream = socket.getOutputStream();
26             outputStream.write(b);
27             outputStream.flush();
28         } catch (IOException e) {
29             e.printStackTrace();
30         }
31     }
32
33     private void recibir() {
34         try {
35             Socket socket = new Socket("127.0.0.1", 7000);
36             InputStream inputStream = socket.getInputStream();
37             byte[] b = new byte[1024];
38             while (inputStream.available() > 0) {
39                 b = inputStream.read(b);
40                 System.out.println("Recibido: " + new String(b));
41             }
42         } catch (IOException e) {
43             e.printStackTrace();
44         }
45     }
46
47     private void cerrarConexion() {
48         try {
49             Socket socket = new Socket("127.0.0.1", 7000);
50             socket.close();
51         } catch (IOException e) {
52             e.printStackTrace();
53         }
54     }
55
56     private void mostrarMenu() {
57         System.out.println("Seleccione una opción:");
58         System.out.println("1. Enviar archivo");
59         System.out.println("2. Recibir archivo");
60         System.out.println("3. Cerrar conexión");
61         System.out.println("4. Salir");
62     }
63
64     private void ejecutarMenu() {
65         while (true) {
66             mostrarMenu();
67             int opcion = Integer.parseInt(System.out.nextLine());
68             switch (opcion) {
69                 case 1:
70                     enviar(archivo);
71                     break;
72                 case 2:
73                     recibir();
74                     break;
75                 case 3:
76                     cerrarConexion();
77                     break;
78                 case 4:
79                     System.exit(0);
80                     break;
81                 default:
82                     System.out.println("Opción no válida");
83             }
84         }
85     }
86
87     private void inicializar() {
88         archivo = System.out.println("Ingrese la dirección del archivo:");
89         System.out.println(System.out.nextLine());
90     }
91
92     private void inicializarServidor() {
93         System.out.println("Ingrese la dirección del servidor:");
94         System.out.println(System.out.nextLine());
95     }
96
97     private void inicializarPuerto() {
98         System.out.println("Ingrese el puerto:");
99         System.out.println(System.out.nextLine());
100    }
101
102    private void inicializarTodos() {
103        inicializar();
104        inicializarServidor();
105        inicializarPuerto();
106    }
107
108    public static void main(String[] args) {
109        ClienteArchivo cliente = new ClienteArchivo();
110        cliente.inicializarTodos();
111        cliente.ejecutarMenu();
112    }
113

```
    - ServidorArchivo.java**

```

1  package com.garciaquirozgustavo;
2  import java.io.*;
3  import java.net.*;
4
5  public class ServidorArchivo {
6      public static void main(String[] args){
7
8      }
9
10     // Leemos y enviamos el contenido del archivo en pequeños bloques
11     fis = new FileInputStream(archivo);
12     byte[] b = new byte[1024];
13     long recibidos = 0;
14
15     while (fis.available() > 0) {
16         b = fis.read(b);
17         recibir(b);
18     }
19
20     }
21
22     private void recibir(byte[] b) {
23         try {
24             Socket socket = new Socket("127.0.0.1", 7000);
25             InputStream inputStream = socket.getInputStream();
26             inputStream.write(b);
27             inputStream.flush();
28         } catch (IOException e) {
29             e.printStackTrace();
30         }
31     }
32
33     private void enviar() {
34         try {
35             Socket socket = new Socket("127.0.0.1", 7000);
36             OutputStream outputStream = socket.getOutputStream();
37             outputStream.write(b);
38             outputStream.flush();
39         } catch (IOException e) {
40             e.printStackTrace();
41         }
42     }
43
44     private void cerrarConexion() {
45         try {
46             Socket socket = new Socket("127.0.0.1", 7000);
47             socket.close();
48         } catch (IOException e) {
49             e.printStackTrace();
50         }
51     }
52
53     private void mostrarMenu() {
54         System.out.println("Seleccione una opción:");
55         System.out.println("1. Enviar archivo");
56         System.out.println("2. Recibir archivo");
57         System.out.println("3. Cerrar conexión");
58         System.out.println("4. Salir");
59     }
60
61     private void ejecutarMenu() {
62         while (true) {
63             mostrarMenu();
64             int opcion = Integer.parseInt(System.out.nextLine());
65             switch (opcion) {
66                 case 1:
67                     enviar();
68                     break;
69                 case 2:
70                     recibir();
71                     break;
72                 case 3:
73                     cerrarConexion();
74                     break;
75                 case 4:
76                     System.exit(0);
77                     break;
78                 default:
79                     System.out.println("Opción no válida");
80             }
81         }
82     }
83
84     private void inicializar() {
85         archivo = System.out.println("Ingrese la dirección del archivo:");
86         System.out.println(System.out.nextLine());
87     }
88
89     private void inicializarServidor() {
90         System.out.println("Ingrese la dirección del servidor:");
91         System.out.println(System.out.nextLine());
92     }
93
94     private void inicializarPuerto() {
95         System.out.println("Ingrese el puerto:");
96         System.out.println(System.out.nextLine());
97     }
98
99     private void inicializarTodos() {
100        inicializar();
101        inicializarServidor();
102        inicializarPuerto();
103    }
104
105    public static void main(String[] args) {
106        ServidorArchivo servidor = new ServidorArchivo();
107        servidor.inicializarTodos();
108        servidor.ejecutarMenu();
109    }
110

```
  - File Explorer Dialog:**
    - Buscar en: x
    - test.txt
    - test1.txt
    - test2.txt
    - test3.txt
    - Nombre de archivo: test.txt test1.txt test2.txt test3.txt
    - Archivos de tipo: Todos los Archivos
    - Abrir
    - Cancelar