Bloque 5. Paradigmas de la Programación

Tarea 4: Primos con sockets UDP

Enunciado:

4.- Crear una aplicación distribuida con sockets UDP, a la que se le envíe un número y nos devuelva como resultado si el número es primo o no.

A tener en cuenta:

- Tendremos un Cliente y un Servidor, ambos clase Main.
- Se implementará con Socket Datagrama

IMPORTANTE: INTRODUCIR EL VALOR LA MANERA CORRECTA: n**-** con un guión final.

Código Servidor:

```
import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.net.SocketException;

public class Servidor {

    public static void main(String[] args) throws SocketException {
        DatagramSocket socket = new DatagramSocket(4445); //Creamos socket UDP
        String n;
        int esPrimo = 0;
        String resp = null;
        try {
            System.out.println("Servidor preparado. . . . ");
            byte[] buf = new byte[128];
            //Preparamos la respuesta
            DatagramPacket paquete = new DatagramPacket(buf, buf.length);
            //Recibimos el datagrama
            socket.receive(paquete);
            n = new String(paquete.getData()); //Pasamos datos a String

            //Método para pasar de String a Int
            //int valor =Integer.parseInt(n);
            String[] values = n.split("-");
            int valor = Integer.parseInt(values[0]);
            System.out.println("Contenido del mensaje recibido . . . Número a evaluar: " + n);
            //Método para comprobar si es primo o no
            if (valor <= 1) { // Si es 1 o menor, directamente no será primo
                esPrimo = 0;
                resp = " NO es PRIMO";
            } else { // Si es mayor, realizo la comprobación
                for (int i = 2; i <= valor / 2; i++) {
                    if ((valor % i) == 0) {
                        esPrimo = 0;
                        resp = " NO es PRIMO";

                    } else {
                        esPrimo = 1;
                        resp = " Es PRIMO";
```

```
            }
          }
        }

        System.out.println("*****Contenido del mensaje a enviar****\nEl número: " + valor + resp);
        String mensaje = "El número: " + valor + resp;

        buf = mensaje.getBytes(); //Para enviarlo necesitamos pasarlo a array de bytes
        InetAddress destino = paquete.getAddress(); //El destino lo sacamos del paquete recibido
        int puerto = paquete.getPort(); //Ídem con el puerto
        paquete = new DatagramPacket(buf, buf.length, destino, puerto);
        socket.send(paquete);
      } catch (IOException e) {
      }
      socket.close();
   }
}
```

Código Cliente:

```
import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.util.Scanner;

public class Cliente {

   public static void main(String[] args) throws IOException {

      DatagramSocket socket = new DatagramSocket(); //Creamos el socket UDP
      InetAddress destino = InetAddress.getByName("localhost"); //Obtenemos la direccion de localhost
      String n = null; //Inicializamos el mensaje a null
      Scanner sc = new Scanner(System.in);
      System.out.println("Cliente preparado . . . .");
      System.out.println("**Introduce un número para evaluar SEGUIDO DE UN - AL TERMINAR** ");
      n = sc.nextLine();
      System.out.println("Vamos a enviar el mensaje . . . ." + n);
      //Definimos el numero de bytes del bufer
      //El buffer servirá para almacenar la info que envisaremos de manera temporal
      //en caso de que haya demoras
      byte[] buf = n.getBytes(); //Para enviar tiene que ser byte[]
      //Paquete con el datagrama a enviar, contine: mensaje, longitud, puerto de destino y origen
      DatagramPacket paquete = new DatagramPacket(buf, buf.length, destino, 4445);
      socket.send(paquete); //enviamos
      buf = new byte[128];

      String inp = sc.nextLine();
```

```
        // convert the String input into the byte array.
        buf = inp.getBytes();

        paquete = new DatagramPacket(buf, buf.length);
        socket.receive(paquete);
        String recibido = new String(paquete.getData()); //Pasamos datos a String
        System.out.println(recibido);
        socket.close();
    }
}
```

Resultado

Desde Cliente...	Desde Servidor...
run: Cliente preparado . . . . **Introduce un número para evaluar SEGUIDO DE UN - AL TERMINAR** 911- Vamos a enviar el mensaje . . . .911- BUILD STOPPED (total time: 53 seconds)	run: Servidor preparado. . . . Contenido del mensaje recibido . . . Número a evaluar: 911-