**ACTIVIDAD 8**

**Enunciado**

Realizar un programa con sockets UDP que establezca un intercambio de información entre un servidor y un único cliente. El intercambio consistirá en que el cliente le enviará al servidor un número y el servidor le devolverá al cliente dicho número elevado al cubo.

Con respecto al **cliente**:

* El nombre de su clase principal será Cliente
* Suponiendo que el número enviado al servidor sea el 4, el resultado esperado de su ejecución será el siguiente:

Esperando respuesta...

Esperando respuesta...: el cubo de + 4 es 64

Adiós…

* Enviará al puerto 12346 del servidor (localhost) el número del que se quiere obtener su valor elevado al cubo
* El número a enviar al servidor no se capturará por pantalla ni se pasará al programa a través un argumento, sino que se definirá dentro del programa, teniendo en cuenta que su rango de valores oscilará entre -128 y +127
* El socket usado para el envío del datagrama al servidor, usará el puerto local 34568
* La frase “Esperando respuesta…” se visualizará justo después de haber enviado el datagrama al servidor
* La frase “Esperando respuesta….: el cubo de…” se visualizará justo después de haber recibido el datagrama del servidor
* La frase “Adiós…” se visualizará justo antes de realizar el cierre del socket

Con respecto al **servidor**:

* El nombre de su clase principal será Servidor
* El resultado esperado de su ejecución será el siguiente, teniendo en cuenta que la dirección IP variará en función del equipo desde el que se ejecute el programa servidor:

Esperando datagrama...

Vamos a calcular el cubo de: 4

IP de origen: /192.168.56.1

Puerto de origen: 34568

Enviamos el resultado...64

Adiósss

* El puerto por el que escuchará las peticiones del cliente será el 12346
* La frase “Esperando datagrama…” se visualizará justo antes de instanciar un objeto de la clase DatagramPacket
* La frase “Vamos a calcular el cubo…” y las dos siguientes se visualizarán después de haber recibido el datagrama desde el cliente
* La frase “Enviamos el resultado…” se visualizará después de haber enviado al cliente el datagrama a través del correspondiente socket
* La frase “Adiósss” se visualizará justo antes de realizar el cierre del socket

Recomendaciones a la hora de realizar las **pruebas** de esta actividad:

* Iniciar la ejecución del servidor antes que la del cliente
* Variar de puerto si surgen problemas

**Solución**

**//Cliente**

package code;

import java.net.DatagramPacket;

import java.net.DatagramSocket;

import java.net.InetAddress;

public class Cliente {

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

// Puerto

int puerto = 12346;

InetAddress destino = InetAddress.getLocalHost();

// Matriz de bytes

byte[] mensaje= new byte[1024];

//Mensaje a enviar al servidor

String numero = "4";

// Codificar el string en bytes

mensaje = numero.getBytes();

// Construir el datagrama a enviar al servidor

DatagramPacket envio = new DatagramPacket (mensaje, mensaje.length, destino, puerto);

System.out.println("Esperando respuesta...");

DatagramSocket socket = new DatagramSocket(34568); // Puerto local

// Enviar el datagrama al servidor

socket.send(envio);

// Recibir el datagrama del servidor

byte[] bufer = new byte[1024];

DatagramPacket recibo = new DatagramPacket (bufer, bufer.length);

socket.receive (recibo);

// Obtener el datagrama en String

String paquete = new String (recibo.getData());

// Mostrar respuesta del servidor

System.out.print("Esperando respuesta...: ");

System.out.println("el cubo de "+numero+" es "+paquete.trim());

// Cerrar el socket de cliente

System.out.println("Adiós...");

socket.close();

}

}

**//Servidor**

package code;

import java.net.DatagramPacket;

import java.net.DatagramSocket;

import java.net.InetAddress;

public class Servidor {

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

int puerto = 12346;

// Buffer para recibir los datagramas

byte[] bufer = new byte[1024];

// Asociar el socket al puerto 12346 del servidor

DatagramSocket socket = new DatagramSocket ( puerto);

// Esperar la llegada del datagrama al servidor

System.out.println("Esperando datagrama.......");

DatagramPacket recibo = new DatagramPacket (bufer, bufer.length);

// Recibir el datagrama

socket.receive (recibo);

// Obtener el número de bytes del datagrama

int bytesRec = recibo.getLength();

// Obtener el datagrama en String

String paquete = new String (recibo.getData());

// Mostrar el mensaje recibido

System.out.println("Vamos a calcular el cubo de: "+paquete.trim());

System.out.println("IP de origen: " + recibo.getAddress().getHostAddress());

System.out.println("Puerto de origen: " + recibo.getPort());

// Enviar respuesta a cliente

String numero = "64";

byte[] mensaje= new byte[1024];

mensaje = numero.getBytes();

InetAddress destino = InetAddress.getLocalHost();

int puertoCliente = 34568;

DatagramPacket envio = new DatagramPacket (mensaje, mensaje.length, destino, puertoCliente);

socket.send(envio);

// Mostrar el mensaje enviado

System.out.println("Enviamos el resultado..."+numero);

//Cerrar el socket de servidor

System.out.println("Adiósss");

socket.close();

}

}

**Resolución de problemas**

**CRITERIOS DE CORRECCIÓN**

1. Nivel de ajuste a la nomenclatura de objetos definida en el enunciado de la actividad y nivel de ajuste al almacenamiento de la información requerida en las subcarpetas definidas para la actividad y unidad didáctica correspondientes (15%).
2. Hacer lo que se indica en el enunciado (55%).
3. Claridad del código Java (15%).
4. Documentación del código Java (15%).
5. Explicación detallada de cómo se han resuelto problemas que hayan podido surgir en la realización de la actividad (hasta un 15% adicional sobre la nota obtenida).   
   Dicha explicación se incluirá al principio de los archivos de código fuente Java mediante líneas de comentarios.   
   La primera línea de comentarios sólo incluirá lo siguiente:  
    // Resolución de problemas.
6. Variables no usadas (-15%).
7. Librerías no usadas (-15%).