**ACTIVIDAD 1**

**Enunciado**

Realizar un programa que obtenga todas las direcciones IP asociadas con el dominio [www.spotify.com](http://www.spotify.com), utilizando la clase Java InetAddress.

El nombre de la clase principal será Actividad1.

El resultado esperado será equivalente al siguiente, teniendo en cuenta que las direcciones IP pueden variar:

Direcciones asociadas a Spotify:

www.spotify.com/194.132.198.118

www.spotify.com/194.132.198.149

[www.spotify.com/194.132.197.198](http://www.spotify.com/194.132.197.198)

**Solución**

**// Actividad1.java**

package actividad1\_INETADDRESS;

import java.net.InetAddress;

import java.net.UnknownHostException;

public class Actividad1 {

public static void main(String[] args) {

String dominio = "www.spotify.com";

try {

InetAddress[] direccionesIP = InetAddress.*getAllByName*(dominio);

System.***out***.println("Direcciones asociadas a Spotify :");

for (InetAddress direccion : direccionesIP) {

System.***out***.println(dominio + "/" + direccion.getHostAddress());

}

} catch (UnknownHostException e) {

System.***err***.println("No se pudo obtener la dirección IP del dominio: " + dominio);

e.printStackTrace();

}

}

}

**Resolución de problemas**

**ACTIVIDAD 2**

**Enunciado**

Realizar un programa que pase como parámetro una dirección y obtenga todas las direcciones asociadas a ese parámetro, utilizando la clase Java InetAddress.

El nombre de la clase principal será Actividad2.

El resultado esperado cuando se haya pasado como parámetro una dirección existente será equivalente al siguiente, teniendo en cuenta que las direcciones IP pueden variar:

Las direcciones asociadas a web.gencat.cat son:

web.gencat.cat/84.53.132.41

web.gencat.cat/84.53.132.18

El resultado esperado cuando no se haya pasado ningún parámetro es el siguiente:

Se necesita una URL para obtener su dirección

**Solución**

**// Actividad2.java**

package actividad1\_INETADDRESS;

import java.net.InetAddress;

import java.net.UnknownHostException;

public class Actividad2 {

public static void main(String[] args) {

if (args.length == 0) {

System.***out***.println("Se necesita una URL para obtener su dirección.");

} else {

String url = args[0];

*obtenerDirecciones*(url);

}

}

public static void obtenerDirecciones(String url) {

try {

InetAddress[] direccionesIP = InetAddress.*getAllByName*(url);

System.***out***.println("Las direcciones asociadas a " + url + " son:");

for (InetAddress direccion : direccionesIP) {

System.***out***.println(url + "/" + direccion.getHostAddress());

}

} catch (UnknownHostException e) {

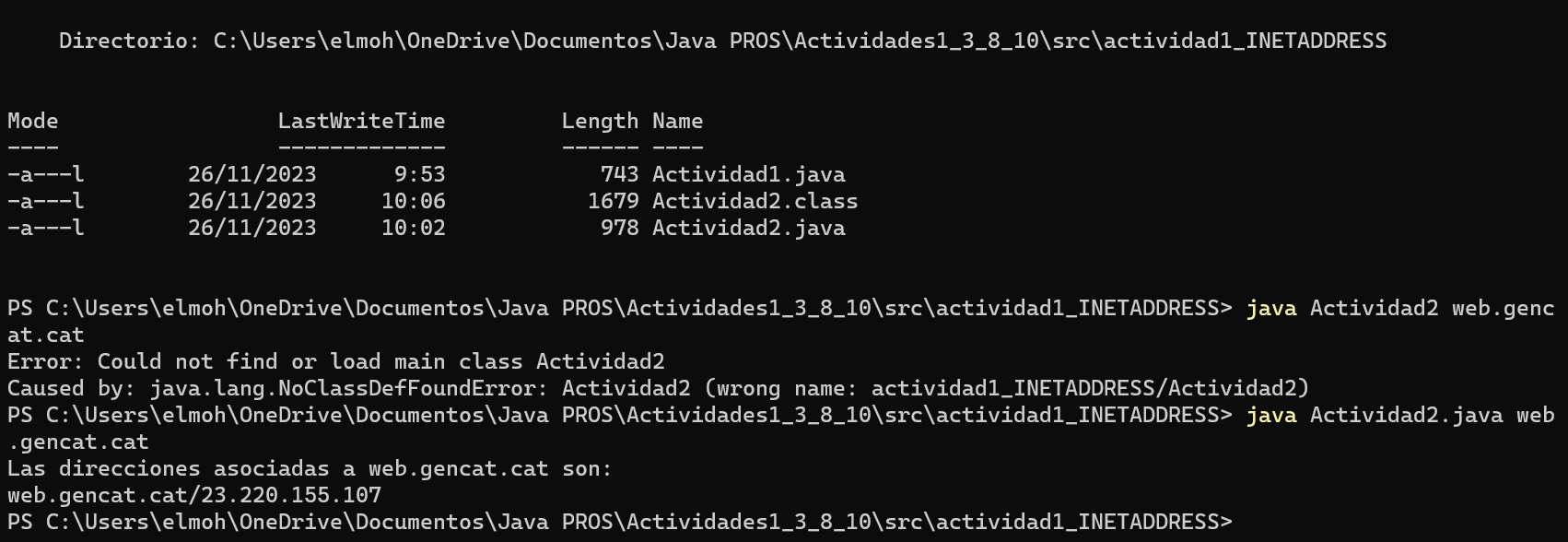
System.***err***.println("No se pudo obtener la dirección IP de la URL: " + url);

e.printStackTrace();

}

}

}

****

**Resolución de problemas**

**ACTIVIDAD 3**

**Enunciado**

Realiza un programa que reciba como parámetro un nombre de dominio y te devuelva su dirección IP. Si no se le pasa ningún parámetro válido, devolverá las direcciones IP locales.

La clase Java a utilizar es InetAddress. El nombre de la clase principal será Actividad3.

El resultado esperado cuando no se pase ningún parámetro, será equivalente al siguiente:

Dirección IP:172.20.105.120

Nombre:DM2-PROF

DM2-PROF/172.20.105.120

172.20.105.120.

DM2-PROF/172.20.105.120

DM2-PROF/192.168.56.1

DM2-PROF/192.168.191.1

DM2-PROF/192.168.150.1

DM2-PROF/fe80:0:0:0:80f7:f52e:1842:bb04%9

DM2-PROF/fe80:0:0:0:680c:9d5e:2450:df7f%5

DM2-PROF/fe80:0:0:0:71f6:e794:453c:5895%12

DM2-PROF/fe80:0:0:0:64cb:7ced:4f85:4e81%4

El resultado esperado cuando se pase como parámetro una dirección existente será el siguiente:

Dirección IP:204.93.177.191

Nombre:www.ciudadjardin.com

www.ciudadjardin.com/204.93.177.191

204.93.177.191.

www.ciudadjardin.com/204.93.177.191

**Solución**

**// Actividad3.java**

package actividad1\_INETADDRESS;

import java.net.InetAddress;

import java.net.UnknownHostException;

public class Actividad3 {

public static void main(String[] args) {

if (args.length == 0) {

*mostrarDireccionesLocales*();

} else {

String dominio = args[0];

*obtenerYMostrarDireccionIP*(dominio);

}

}

private static void mostrarDireccionesLocales() {

try {

InetAddress localHost = InetAddress.*getLocalHost*();

System.***out***.println("Dirección IP: " + localHost.getHostAddress());

System.***out***.println("Nombre: " + localHost.getHostName());

System.***out***.println(localHost.getHostName() + "/" + localHost.getHostAddress() + ".");

InetAddress[] localAddresses = InetAddress.*getAllByName*(localHost.getHostName());

for (InetAddress localAddress : localAddresses) {

System.***out***.println(localHost.getHostName() + "/" + localAddress.getHostAddress());

}

} catch (UnknownHostException e) {

System.***err***.println("Error al obtener la dirección IP local");

e.printStackTrace();

}

}

private static void obtenerYMostrarDireccionIP(String dominio) {

try {

InetAddress[] direccionesIP = InetAddress.*getAllByName*(dominio);

System.***out***.println("Dirección IP: " + direccionesIP[0].getHostAddress());

System.***out***.println("Nombre: " + direccionesIP[0].getHostName());

System.***out***.println(dominio + "/" + direccionesIP[0].getHostAddress() + ".");

for (InetAddress direccion : direccionesIP) {

System.***out***.println(dominio + "/" + direccion.getHostAddress());

}

} catch (UnknownHostException e) {

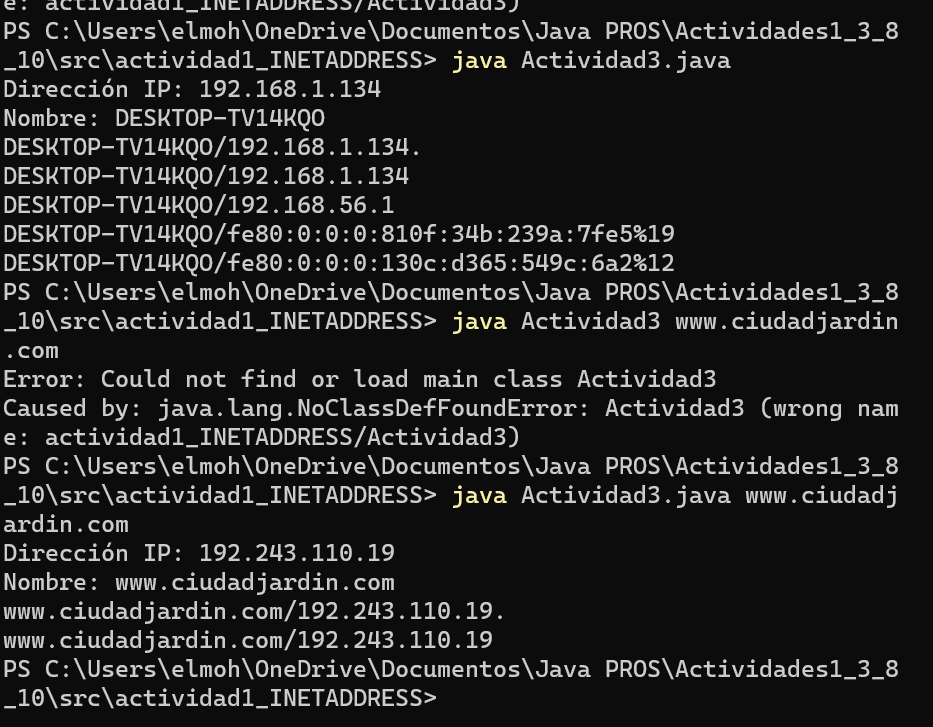
System.***err***.println("No se pudo obtener la dirección IP del dominio: " + dominio);

e.printStackTrace();

}

}

}

****

**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%).