**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**

**public class Ejercicio1 {**

**public static void main(String[] args) {**

**InetAddress address[];**

**try**

**{**

**address = InetAddress.getAllByName("www.spotify.com");**

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

**for (int i = 0; i < address.length; i++)**

**System.out.println(address[i]);**

**}**

**catch (UnknownHostException e)**

**{**

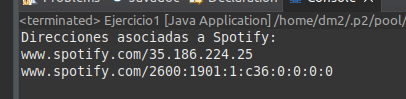
**System.out.println(e.getMessage());**

**}**

**}**

**}**

**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**

**public class Ejercicio2 {**

**public static void main( String[] args ) {**

**InetAddress address[];**

**if(args[0].equals("")) {**

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

**}else {**

**try {**

**System.out.println( "Direccion asociadas a "+args[0].substring(args[0].indexOf(".")+1,args[0].lastIndexOf(".")) );**

**address = InetAddress.getAllByName( args[0] );**

**for (int i = 0; i < address.length; i++) {**

**System.out.println( address[i] );**

**}**

**} catch (UnknownHostException e) {**

**// TODO Auto-generated catch block**

**e.printStackTrace();**

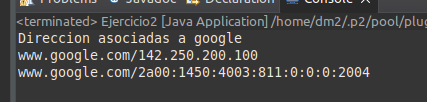
**}**

**}**

**}**

**}**

**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**

**public class Ejercicio3 {**

**public static void main(String[] args){**

**String direcciones="";**

**try {**

**String dir = direcciones;**

**InetAddress address[] = InetAddress.getAllByName(direcciones);**

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

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

**for (int i = 0; i < address.length; i++)**

**System.out.println(address[i]);**

**}**

**catch (UnknownHostException e) {**

**InetAddress local;**

**InetAddress localAddresses[];**

**try**

**{**

**local = InetAddress.getLocalHost();**

**localAddresses = InetAddress.getAllByName(local.getHostName());**

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

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

**for (int i = 0; i < localAddresses.length; i++)**

**System.out.println(localAddresses[i]);**

**}**

**catch (UnknownHostException f){**

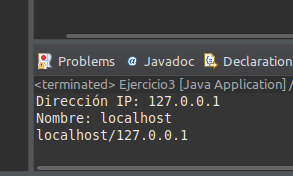
**f.printStackTrace();**

**}**

**}**

**}**

**Resolución de problemas**

****

**REQUISITOS**

1. Entregar las soluciones de las actividades en un único archivo comprimido cuyo nombre es: el nombre original de la actividad, seguido de un guión bajo y del primer apellido del alumno. Donde tendremos:
   1. Un archivo de extensión .odt o .docx que contendrá
      1. Una copia del código del programa
      2. Una captura de pantalla del resultado de la ejecución del programa
   2. Los archivos con el código fuente de extensión .java

**CRITERIOS DE CORRECCIÓN**

1. Cumplimiento de los requisitos (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%).