**Indicaciones de entrega**

Una vez realizada la tarea elaborarás un único fichero **.zip** donde figuren las respuestas correspondientes. El envío se realizará a través de la plataforma de la forma establecida para ello, y el archivo se nombrará siguiendo las siguientes pautas:

**apellido1\_apellido2\_nombre\_SIGxx\_Tarea**

Asegúrate que el nombre no contenga la letra ñ, tildes ni caracteres especiales extraños. Así por ejemplo la alumna **Begoña Sánchez Mañas para la primera unidad del MP de DAW**, debería nombrar esta tarea como...

**sanchez\_manas\_begona\_PSP\_Tarea01**

**Cada ejercicio ha de desarrollarse en un documento distinto, siguiendo los nombres de ej1, ej2,..**

**Ejercicios**

1. **(0.5 pts)** Realiza un programa que admita desde la línea de comandos un nombre de máquina o una dirección IP y visualice información sobre ella.
2. **(0.6 pts)** Realiza un programa que obtenga en un objeto InetAddress con el Host local, y muestre la información relativa al **nombre del Host**, su **dirección IP** y el método **toString** del objeto creado:



* Crea un objeto de la clase InetAddress
* Asígnale la información relativa al host local (recuerda que puede haber excepción)
* Usa los métodos necesarios
* import java.net.InetAddress

1. **(0.7 pts)** Modifica el programa anterior para que obtenga la información a través del nombre del host y añade una línea con la forma canónica del nombre

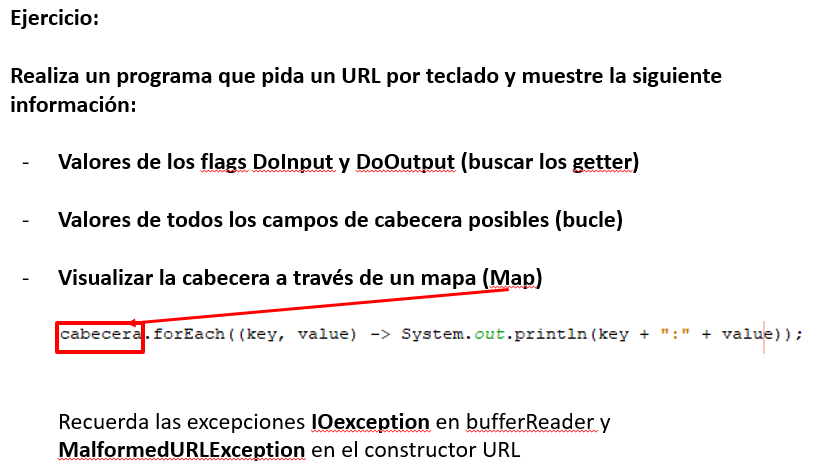
* Modifícalo para que obtenga la IP de google visualiza las diferencias.
* Modifícalo para que obtenga la IP de un nombre de Host o dirección de internet introducido por teclado

\*\*[**getAllByName**](https://docs.oracle.com/javase/7/docs/api/java/net/InetAddress.html)**([String](https://docs.oracle.com/javase/7/docs/api/java/lang/String.html) host)** Devuelve un array con las IPs de todas las interfaces de red

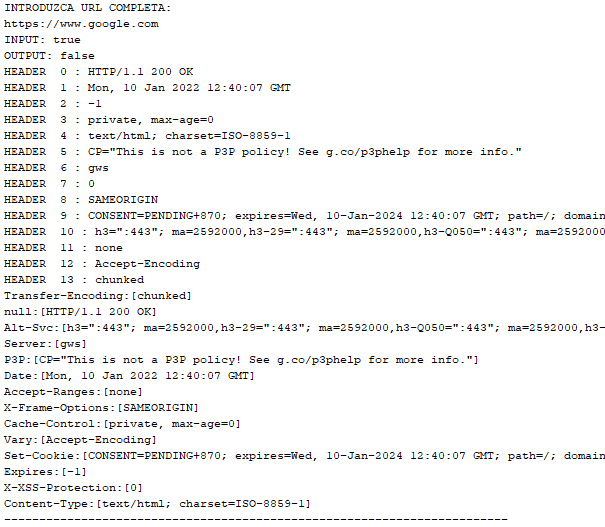
1. **(1.5 pts) Realiza un programa que pida un URL por teclado y muestre su contenido (en html):**

* Crea un objeto de la clase URL a partir de los datos de entrada
* Utiliza el método openStream()
* Para leer el **inputStream** devuelto por openStream(), puedes hacer los siguiente:
  + Crear un bufferReader al cual le pasaremos un InputStreamReader sobre el inputStream obtenido
  + Leer el bucle y mostrar por pantalla el bufferReader mientras su readLine() no sea nulo
  + Cerrar el bufferReader con .close();
* Recuerda las excepciones **IOexception** en bufferReader y **MalformedURLException** en el constructor URL

1. **(1.5 pts)** Realiza un programa que pida una URL por teclado y muestre la siguiente información:
   * Valores de los flags DoInput y DoOutput (buscar los getter)
   * Valores de todos los campos de la cabecera posibles (bucle)
   * Visualiza la cabecera a través de un mapa (Map)



Recuerda las excepciones **IOexception** en bufferReader y **MalformedURLException** en el constructor URL.



1. **(0.2 pts) Ejercicio Formulario PHP**

* Verificar instalación XAMPP o similar y localizar carpeta htdocs
* Crear una carpeta nueva llamada php y dentro los archivos index.html y ejemplo.php
* **Copiar y entender** el html y el php siguientes.

<html>

<body>

<form action="ejemplo.php" method="post">

<p>Introduce Nombre:

<input name="nombre" type="txt" size="15"></p>

<p>Introduce Apellidos:

<input name="apellidos" type="txt" size="15"></p>

<input type="submit" name="ver" value="ver">

<input type="reset" value="reset"></p>

</form>

</body>

</html>

<?php

$nom=$\_POST["nombre"];

$ape=$\_POST["apellidos"];

echo "Bienvenido $nom de apellido: &ape";

?>

* **Abrir en el navegador** la ubicación en el host local del servidor index.html y probar el ejemplo:
  + localhost/htdocs/psp/index.html Interfaz de usuario gráfica, Texto, Aplicación

    Descripción generada automáticamente

1. **(2 pts)** Realiza un programa que usando la clase **URLConnection** envíe los datos requeridos por el php (*nombre* y *apellidos*).

Para ello:

* Crear un **objeto URL** al script que queremos conectar “http://localhost/php/ejemplo.php”
* **Establecer la comunicacion** con la URL obteniendo el objeto URLConnection correspondiente.
* Configurar la conexión para permitir el **envío de datos** (setDoOutput)
* Una forma de **enviar los datos** es a través de un objeto PrintWriter pasándole en el constructor el Stream de salida obtenido por getOutputStream() del objeto URLConnection (Los parámetros se pasan con &: &nombre=”nombre”)
* Para **recibir los datos del php** de vuelta, utilizar el inputStreamReader como en el ejemplo anterior
* Recuerda las excepciones y cerrar los flujos de entrada y salida una vez usados.

1. **(3 pts)** Realizar un programa que se conecte al html creado anteriormente, utilizando los métodos de URLConnection. Además, en la salida del programa deben aparecer los siguientes elementos:

* Dirección + [método]: valor
* Fecha de última modificación + [método]: valor
* Tipo de contenido + [método]: valor
* ========================================
* Todos los campos de la cabecera + [método]: valores cada uno en una línea y tipo: valor
* ========================================
* Los campos de la cabecera 1 y 4: + [método]: valor cada uno en una línea
* ========================================
* Contenido de + [método]: valor

**NOTA**: Para recorrer una estructura Map podemos usar una estructura Iterator. Para obtener un iterador sobre el map se invoca a los métodos *entrySetO* e *iteratorO.* Para mover el iterador utilizaremos el método *nextO* y para comprobar si ha llegado al final usamos el método *hasNextO.* De la estructura recuperaremos los valores mediante

* *getKeyO,* para la clave y *getValueO,* para el valor.