

	Ciclo Formativo de Grado Superior. Desarrollo de Aplicaciones Multiplataforma / Desarrollo de Aplicaciones Web
	<b>Módulo:</b> Programación
	<b>Profesor:</b> Á. Enrique Pineda Navas
	Examen 3ª evaluación
<b>Fecha:</b> 7/06/2023	<b>Nombre:</b>

## 1. ESTRUCTURA Y CARACTERÍSTICAS

- La prueba se basa en los criterios de evaluación que establece el currículo vigente del ciclo formativo de grado superior de Técnico Superior en Desarrollo de Aplicaciones Multiplataforma en la Comunidad Autónoma Extremadura del módulo PROGRAMACIÓN (DECRETO 259/2011 - Código: 0485)
- Consta de **3 ejercicios**, cuya puntuación se detalla en el enunciado de cada uno de ellos, pudiendo alcanzar un máximo de **10** puntos en total. La nota de esta prueba representará el 70% de la nota de la evaluación.
- La duración total de la prueba es de **4 horas**, pudiendo el alumno organizar el orden de realización de los ejercicios como considere oportuno. A modo de orientación, el tiempo estimado en la realización de los ejercicios es de 45' para el ejercicio 1 y entre una hora y hora y media para los ejercicios 2 y 3.

## 2. NORMATIVA

- La prueba se realizará utilizando el lenguaje de programación **JAVA** y el entorno de desarrollo **Netbeans**.
- NO está permitido** la utilización de Internet ni tampoco del material desarrollado durante el curso (apuntes y prácticas), dado que los ejercicios planteados en esta prueba son similares a las prácticas realizadas en clase.
- NO está permitido** el uso del teléfono móvil (se recomienda que esté apagado) ni servicios de correo electrónico, chats o demás programas de mensajería, debido al carácter **INDIVIDUAL** de la prueba.
- Se valorará no solo el resultado del ejercicio sino también la eficiencia del código implementado por el alumno.
- Todos los métodos de tipo “mostrar” deben implementarse haciendo uso del recorrido de la estructura.
- La penalización por falta de claridad en el código y/o mala utilización o ausencia de sangría será del 10% de la nota del apartado..

## 3. EJERCICIOS

**IMPORTANTE: PARA QUE UN EJERCICIO SEA VALORADO ES IMPRESCINDIBLE QUE COMPILE Y SE EJECUTE.**

	Ciclo Formativo de Grado Superior. Desarrollo de Aplicaciones Multiplataforma / Desarrollo de Aplicaciones Web
	<b>Módulo:</b> Programación
	<b>Profesor:</b> Á. Enrique Pineda Navas
	Examen 3ª evaluación
<b>Fecha:</b> 7/06/2023	<b>Nombre:</b>

**EJERCICIO 1.- (3 puntos) – FICHEROS DE OBJETOS (Mínimo 1 punto)** – El proyecto de Teleasistencia que está desarrollando el IES Valle del Jerte necesita registrar las alarmas recibidas.

Para registrarlas, implementaremos la clase *Alarma* que almacenará el nombre del teleoperador, el nombre del usuario del servicio y la fecha en la que se produjo la alarma (La fecha la introducirá el usuario por pantalla y utilizaremos las clases específicas para almacenarla) **(0,25 puntos)**

El programa mostrará al sistema el siguiente menú:

1. Registrar alarma (escribirá en el fichero registros.obj la información de la nueva alarma). **(1 punto)**
2. Mostrar alarmas registradas (mostrará todas las alarmas del fichero). **(0,75 puntos)**
3. Buscar por teleoperador (Pediremos por pantalla el nombre del teleoperador y mostraremos el total de alarmas en las que ha participado dicho teleoperador). **(1 punto)**

NOTA: Si el fichero existe queremos añadir objetos respetando su contenido.

**EJERCICIO 2.- (3,5 puntos) – ESTRUCTURAS DE DATOS Y FICHEROS DE TEXTO (Mínimo 1,25 puntos)** – El conocido youtuber y podcaster Jordi Wild se ha puesto en contacto con los alumnos de Programación del IES Valle del Jerte para que éstos desarrollen un software capaz de gestionar información relevante de sus podcast y las distintas plataformas donde escucharlos.

Para gestionar la información, necesitaremos crear la clase *Plataforma*, de la que almacenaremos su nombre y el número de reproducciones. De cada *podcast* que sube Jordi Wild, necesitaremos almacenar el nombre del podcast y una lista con las plataformas donde ha sido subido dicho podcast. Para almacenar los podcast utilizaremos un HashMap, donde la clave será el propio nombre del podcast.

Ejemplo:

El podcast “*The Wild Project #187 Carles Porta*” tiene la siguiente lista de plataformas:

- Celda 0  $\Rightarrow$  iVoox / 275.000 reproducciones.
- Celda 1  $\Rightarrow$  Google Podcast / 315.000 reproducciones.
- Celda 2  $\Rightarrow$  Spotify / 80.000 reproducciones.

En total, “*The Wild Project #187 Carles Porta*” ha tenido 670.000 reproducciones (275.000 + 315.000 + 80.000).

1. Añadir podcast. Insertaremos el podcast en el mapa junto con las plataformas en las que ha sido subido (el podcast se subirá en, al menos, una plataforma). **(0,75 puntos)**
2. Mostrar toda la información de los podcast. Utiliza un for-each para recorrer el mapa y un iterador para recorrer la lista **(0,5 puntos)**
3. Buscar podcast (Pediremos la clave y mostraremos todas las plataformas donde ha sido subido. En caso de no existir, se mostrará el mensaje correspondiente). **(1 puntos)**

	Ciclo Formativo de Grado Superior. Desarrollo de Aplicaciones Multiplataforma / Desarrollo de Aplicaciones Web
	<b>Módulo:</b> Programación
	<b>Profesor:</b> Á. Enrique Pineda Navas
	Examen 3ª evaluación
<b>Fecha:</b> 7/06/2023	<b>Nombre:</b>

4. Generar informe según plataforma. Pediremos el nombre de la plataforma y se generará un fichero txt (nombre de la plataforma.txt) con el nombre de cada podcast y las reproducciones que ha tenido en dicha plataforma (En caso de que algún podcast no se haya subido a la plataforma, no escribiremos nada en el fichero acerca de él) **(1 puntos)**
5. Salir del programa.

NOTAS: Controlaremos, mediante la excepción correspondiente, que la opción del menú no sea una letra. La pediremos una y otra vez mientras no sea un número. **(0,25 puntos)**

Ejemplo salida ficheros:

fichero <i>Google Podcast.txt</i>	fichero <i>Spotify.txt</i>	fichero <i>iVoox.txt</i>
<i>Jon Sistiaga#280</i> <i>Carles Portas#315</i> <i>Perez Reverte#400</i>	<i>Jon Sistiaga#195</i> <i>Carles Portas#80</i> <i>Perez Reverte#380</i>	<i>Carles Portas#275</i>


**EJERCICIO 3.- (3,5 puntos) – VECTORES DE OBJETOS (Mínimo 1,25 puntos)** – Con motivo de las futuras vacunaciones en las residencias de ancianos de la Comunidad, el Consejero de Sanidad se ha puesto en contacto con nosotros para desarrollar un software que gestione las dosis administradas en las distintas residencias.

Nuestro programa debe gestionar las residencias donde se realicen las vacunaciones y, de cada residencia, almacenar la información de los usuarios vacunados.

Para guardar la información utilizaremos un vector, donde cada celda va a contener objetos de la clase *Residencia*. De cada residencia almacenaremos su localidad (String) y un arrayList con objetos de la clase *Usuario*, donde cada usuario dispondrá de un nombre (String) y un dni (String).

El menú que ofrezcamos a la Consejería tendrá las siguientes opciones:

1. Rellenar una única residencia junto con los usuarios que reciben la vacuna. (Pediremos la posición –controlando que está dentro del rango de celdas del vector- donde insertar la residencia y, si la posición está ocupada, mostraremos un mensaje y no pediremos la información de la misma. Si la posición está libre, crearemos la residencia y pediremos usuarios mientras así se desee). **(1,5 puntos)**
2. Mostrar. Donde la información a mostrar de cada residencia será la localidad y el número de usuarios vacunados en esa residencia. (Si la celda está vacía no se mostrará información ninguna de la celda en cuestión) **(0,5 puntos)**

	Ciclo Formativo de Grado Superior. Desarrollo de Aplicaciones Multiplataforma / Desarrollo de Aplicaciones Web
	<b>Módulo:</b> Programación
	<b>Profesor:</b> Á. Enrique Pineda Navas
	Examen 3ª evaluación
<b>Fecha:</b> 7/06/2023	<b>Nombre:</b>

3. Buscar por localidad. Método de tipo Residencia que recibe tanto el vector de residencias como la localidad. Si existe la localidad, retornamos la residencia y mostramos su localidad y sus usuarios. Si no, mostraremos un mensaje indicando que la localidad no está registrada.  
**(1,5 puntos)**
4. Salir del programa.

Hasta que no pulsemos 4 no saldremos del programa y se volverá a mostrar el menú.