IES MAR DE ALBORÁN – MÓDULO PROGRAMACIÓN – 1º DAW

EVALUACIÓN UNIDADES 1-7 - 6 MAYO 2021

NOMBRE Y APELLIDOS.-

Realiza el primer commit, con el texto "commit Inicial", confirmando todos los archivos del proyecto. Debes ir confirmando los cambios en tu repositorio cada vez que el profesor así lo requiera.

Ejercicio 1 – Lectura de datos CSV (1 puntos)

Crea una clase, llamada **LecturaCSV**, que contenga un método que permita la lectura del fichero CSV de la raíz de tu proyecto. Este método recibe el nombre y ruta del fichero a leer y se encarga de devolver una estructura de tipo Map ordenada de forma que las iniciales de los módulos sean las claves y el nombre completo de cada módulo sean los valores asociados a dichas claves.

Realiza una prueba de lectura en la clase Programa y muestra por consola el resultado obtenido. Presta atención al control de excepciones.

La definición de la clase LecturaCSV, la firma del método de lectura y el tipo devuelto son correctos. No existen atributos innecesarios	0,15	
La lectura del fichero se realiza correctamente: se utilizan las clases apropiadas, se recorren todos los registros y se liberan los recursos una vez finalizada la lectura. El método es eficiente	0,45	
La creación de la estructura Map es correcta: se crea una estructura TreeMap y se mapean correctamente los campos del fichero en los pares clave-valor de la estructura	0,4	

Ejercicio 2 – Lectura de datos JSON (2 puntos)

Crea una clase, llamada **LecturaJSON**, que contenga un método que permita la lectura del fichero JSON que hay en la raíz de tu proyecto. Este método recibe el nombre y ruta del fichero a leer y se encarga de devolver una estructura de tipo ArrayList de objetos **RegistroJSON**, que es una clase POJO para almacenar la información del fichero. Ten en cuenta que algunas calificaciones son "APRO" o "NE" y otras son de tipo entero entre 0 y 10. Es recomendable guardarlas todas como String. No te preocupes por la codificación de caracteres del archivo.

Realiza una prueba de lectura en la clase Programa y muestra por consola el resultado obtenido. Presta atención al control de excepciones.

La definición de la clase LecturaJSON, la firma del método de lectura y el tipo devuelto son correctos. No existen atributos innecesarios	0,15	
La definición de atributos y métodos de la clase RegistroJSON es correcta y contiene las anotaciones JSON necesarias para su tratamiento con las librerías estudiadas	0,75	
El proyecto incluye en sus dependencias las referencias a las librerías que posibilitan el tratamiento de ficheros JSON	0,1	
La lectura del fichero se realiza correctamente, se utilizan las clases apropiadas y se guardan todos los registros en una lista. Se liberan los recursos una vez finalizada la lectura	1	

Ejercicio 3 – Mapeo de datos (2,5 puntos)

En una nueva clase llamada **RegistrosToAlumnado**, implementa un método que reciba la lista de objetos de tipo RegistroJSON y devuelva una nueva lista de objetos de tipo **Alumnado**. La clase Alumnado contiene como atributos: el nombre del alumno/a en cuestión y una estructura Map ordenada donde cada inicial de módulo (key) se mapea con su calificación (value). Implementa la interfaz <<Comparable>> usando el atributo nombre del alumno/a. La clase Alumnado es de tipo POJO y su método toString devuelve, para cada alumno/a la siguiente información, en dos líneas:

[Nombre de alumno/a]

[InicialModulo: calificación] [[InicialModulo: calificación] [InicialModulo: calificación]...

Realiza una prueba de conversión de la lista de registros JSON en una lista de objetos Alumnado ORDENADA, en la clase Programa y muestra por consola el resultado obtenido.

La definición de la clase RegistroToAlumnado, la firma del método de conversión y el tipo devuelto son correctos. No existen atributos innecesarios	0,15	
La definición de la clase Alumnado, junto con la declaración de atributos y métodos es correcta. No existen atributos innecesarios.	0,15	
La clase Alumnado implementa correctamente la interfaz y el método correspondiente	0,2	
El método toString de la clase Alumnado devuelve la información con el formato solicitado	0,25	
El método que convierte objetos RegistroJSON en objetos Alumnado realiza correctamente su función y devuelve la lista de objetos. El método es legible y eficiente	1,25	

Ejercicio 4 - Escritura de datos (2 puntos)

En una nueva clase llamada **AlumnadoToCSV**, implementa un método que reciba un objeto de tipo **Alumnado** y la estructura de datos tipo Map del ejercicio 1 de forma que el método genere en la carpeta "./alumnado" un **archivo TSV** siendo el nombre del fichero el ***nombredelalumno(sin espacios).tsv***. Este archivo contendrá tantas líneas como módulos tenga el alumno y en cada línea debe aparecer el nombre del módulo y a continuación su calificación.

Prueba el método anterior, en la clase que contiene el método main(), creando el fichero de la alumna que se llama "Delia".

La definición de la clase AlumnadoToCSV y la firma del método de escritura son correctos. No existen atributos innecesarios	0,15	
El método genera correctamente la carpeta "./alumnado"	0,15	
El nombre del fichero y el formato tsv generado es el requerido	0,2	
El contenido del fichero es el esperado	0,75	
El método que realiza la escritura cumple su función. Es legible y eficiente	1,25	

Ejercicio 4 – API Stream (1,5 puntos)

Crea la clase **Estadística** que contenga un ArrayList de Integer para guardar las calificaciones obtenidas en un módulo cualquiera.

A. Incluye métodos get y set.

B. En un método privado implementa la funcionalidad necesaria para que al recibir como parámetros las iniciales de un módulo y la lista de registros JSON, el método devuelva una lista de cadenas de texto con las calificaciones de ese módulo en cuestión. Utiliza API Stream.

C. Un método privado que recibe un String y devuelve su representación numérica o -1 si el String no es un número.

D. El constructor de la clase recibe una lista de objetos RegistroJSON y la inicial del módulo del que se quieren guardar las calificaciones. Este constructor rellena la lista de Integer sólo con las calificaciones numéricas iguales o mayores que cero que haya en ese módulo. Utiliza API Stream y los métodos de ArrayList.

En el main, crea un objeto Estadística para el módulo "OACV" e imprime las calificaciones usando el método forEach de las estructuras List.

El método indicado en B está correctamente declarado y realiza su función usando API Stream	0,5	
El método indicado en C realiza su función de forma legible y eficiente	0,25	
El constructor realiza su función usando API Stream de forma legible y eficiente	0,75	

Ejercicio 5 – Gestión de excepciones y control de versiones (1 puntos)

Realiza un control de excepciones de los métodos de tus clases así como de las posibles excepciones que se puedan generar en el programa principal.

Realiza los commits solicitados por el profesor, confirmando los cambios en los que te encuentres trabajando en cada momento en tu repositorio.

Las excepciones están correctamente gestionadas: no hay excepciones sin controlar en el programa main o en los métodos	0,5	
Todos los commits del repositorio coinciden en fecha y hora con los solicitados por el profesor	0,5	

ATENCIÓN:

- Es indispensable que los ejercicios 1 - 4 estén probados en el método main(). **Si algún ejercicio no tiene prueba, no se corrige**. Las pruebas deben hacerse independientemente de si sus resultados son correctos o no.
- Si en algún ejercicio no se siguen los principios básicos de la programación orientada a objetos **la calificación de la prueba será cero**.
- Si no se siguen las nomenclaturas de paquetes, clases, métodos, instancias,etc. según las normas comentadas en clase la nota se penalizará con **2 puntos**.
- Se deben seguir siempre **criterios de legibilidad y eficiencia**.