Archivos

Se pide resolver cada uno de los ejercicios utilizando diagramas de clases UML y el lenguaje de programación Java.

[L.01] - Primeros pasos.

Crear un nuevo proyecto de tipo aplicación llamado Persistencia e implementar las siguientes clases:

- La clase Parseadora tendrá todos métodos estáticos.
- El método privado verificarSiExisteDirectorio deberá verificar si existe el directorio y si no existe deberá crearlo.
  - ➢ Si existe o no existía pero lo pudo crear, retornar ***true***.
  - ➢ Si no existía o no se pudo crear, retornar ***false***.
- El método privado verificarSiExisteArchivo deberá verificar si existe el directorio y si no existe deberá crearlo.
  - ➢ Si existe o no existía pero lo pudo crear, retornar ***true***.
  - ➢ Si no existía o no se pudo crear, retornar ***false***.

Una vez diseñado el diagrama de clases UML, crear, implementar y probar las entidades en el 'main'.

[L.02] - CRUD Alumnos.

Crear un proyecto de biblioteca de clases con una clase llamada Alumno, la cuál contendrá los siguientes miembros:

- legajo, de tipo entero.
- apellido, de tipo cadena
- nombre, de tipo cadena

  Todos los atributos son privados y poseen su respectivo getter.

- Constructor público, que recibe tres parámetros (para inicializar los atributos).
- Sobrescritura del método **toString**, la cuál retornará todos los valores de los atributos, separados por un guión medio. Ejemplo: "91218 - Quintero - Juanfer"

Agregar al proyecto, la clase Archivo, que contendrá los siguientes métodos estáticos:

- guardar(T dato, String path, boolean agregar). Este método genérico, retornará ***true***, si se pudo guardar el dato en el archivo que se indica en *path*. Caso contrario, retorna ***false***.

- leer(String path). Este método retorna una cadena de caracteres que contiene el contenido del archivo que se indica en *path*.
- buscarAlumno(int legajo, String path). Este método retorna ***true***, si el legajo del alumno se encuentra en el archivo que se indica en *path*. Caso contrario, retorna ***false***.
- modificarAlumno(Alumno alumno, String path). Este método retornará ***true***, si se pudo modificar el registro del alumno en el archivo que se indica en *path*. Caso contrario, retorna ***false***.
- eliminarAlumno(Alumno alumno, String path). Este método retornará ***true***, si se pudo eliminar el registro del alumno en el archivo que se indica en *path*. Caso contrario, retorna ***false***.

Una vez diseñado el diagrama de clases UML, crear, implementar y probar las entidades en un proyecto de consola.

[L.03] - Burbujeo de excepciones (registradas en archivos)

A partir del ejercicio [K.03] - Burbujeo de excepciones, se pide crear la clase ArrchivoTexto que deberá contener:

1. Un método guardar que agrega información al archivo de texto ubicado en la ruta pasada cómo parámetro. También recibirá un String con la información a guardar.
2. Un método leer que retorna el contenido del archivo ubicado en la ruta pasada como parámetro. En caso de no existir, lanzará la excepción relacionada.
3. Tomar las líneas del 'main' y modificarlas donde se capturan las excepciones. Quitar los System.out.println y en su lugar guardar todos los datos del error en un archivo de texto, cuyo nombre será la fecha y hora actual, con el formato **[año][mes][día]-[hora][minuto].txt**
   Ejemplo: ***20181209-1902.txt***
4. Fuera del bloque catch, utilizar el método leer para mostrar por consola los mensajes de error.