Proyecto de laboratorio - Java

Estructura general del proyecto

A. El proyecto debe contener tres carpetas: src, data y doc.
B. La carpeta data debe incluir el fichero CSV del dataset seleccionado.
C. La carpeta src debe incluir el código fuente, organizado en paquetes. Crear un paquete **fp**. Crear dentro de él un paquete **utiles** para las clases de utilidad, un paquete **common** para los tipos auxiliares, y un paquete para los tipos del dominio de trabajo (p.ej. música, cine…). Por último, crear un paquete **test** dentro del paquete de los tipos.
   Ejemplo:
   - fp.**utiles** (aquí estará, al menos, la clase *Checkers*)
   - fp.**common** (si es necesario)
   - fp.**clima** o fp.**expedientes** o fp.**hábitos** o fp.**nacimientos** o fp.**vuelos**
   - fp.**clima.test** o fp.**expedientes.test** o fp**.hábitos.test** o fp.**nacimientos.test** o fp.**vuelos.test**
D. El proyecto debe contener un fichero README.md donde se describirán los datos y los tipos implementados.

Entrega 1 (18-03-2020)

Para el tipo del primer cuatrimestre.

Crear la estructura básica del proyecto.

Implementar el tipo base, que debe cumplir los siguientes requisitos:

- Tener al menos **dos constructores** (uno con todas las propiedades y otro con menos propiedades y la/s que falte/n se inicializa/n a algún valor por defecto que **piense** el o la alumno/a).
- Tener **al menos una propiedad derivada**. (cada alumno/a **piense** en una propiedad derivada que se pueda obtener a partir de los datos básicos del dataset). Resaltarla/s en README
- Tener definida una **representación** como cadena (la que **piense** el o la alumno/a).
- Tener definido un criterio de **igualdad** (el que **piense** el o la alumno/a).
- Tener definido un criterio de **orden natural** (el que **piense** el o la alumno/a).
- Tener definidas **al menos dos restricciones** sobre datos de tipos diferentes (que **piense** el o la alumno/a)

Realizar un test del tipo en el que se comprueben los resultados de cada uno de los métodos del apartado anterior. En el caso de pruebas de las restricciones, una vez probadas, las correspondientes líneas se dejan comentadas.

La entrega debe incluir la implementación y prueba del tipo, así como la documentación del tipo, tanto en el fichero README.md como en el propio código mediante breves comentarios de documentación.

Entrega 2 (22-04-2022)

Crear una factoría, con sus métodos correspondientes:

- Método "deStringA***Tipo***", también se le llama "parsear***Tipo***" (donde ***Tipo*** es el tipo que esté realizando el o las alumno/a) que recibe como parámetro una cadena con el formato de las líneas del fichero CSV, y **devuelve un objeto** del tipo a partir de esa cadena.
- Método "leer***Tipo***s" (donde ***Tipo*** es el tipo que esté realizando el o las alumno/a) que recibe como parámetro una cadena que contiene el nombre y ruta del fichero CSV, y **devuelve una lista de objetos** del tipo. *Este método se apoya en el anterior*.

Crear un tipo contenedor, que debe cumplir los siguientes requisitos:

- Tener, al menos, una propiedad de tipo colección (lista, conjunto o conjunto ordenado) para almacenar los elementos del tipo base que se leerán desde el dataSet.
- Tener al menos dos constructores:
  a) Uno con todas las propiedades básicas (excepto la colección), que cree un objeto de tipo contenedor sin ningún elemento en la colección.
  b) Otro con todas las propiedades básicas y una colección de objetos del tipo base, que cree un objeto de tipo contenedor con todos los elementos de la colección.
- Tener definido un criterio de igualdad y una representación como cadena.
- Tener operaciones para:
  a) Obtener el número de elementos.
  b) Añadir un elemento.
  c) Añadir una colección de elementos.
  d) Eliminar un elemento.

Escribir en el tipo contenedor métodos que realizan los siguientes tratamientos secuenciales, implementados con bucles:

1) A escoger uno de los dos siguientes: existe / para todo.
2) A escoger uno de los tres siguientes: contador / suma / media aritmética o promedio.
3) Obtener una colección a partir de una selección con filtrado.
4) Un método de agrupación que devuelva un Map en el que las claves sean alguna propiedad del tipo base, y los valores una colección (List, Set, SortedSet) de objetos del tipo base.
5) Un método de acumulación que devuelva un Map en el que las claves sean alguna propiedad del tipo base, y los valores el conteo o la suma de los objetos del tipo base almacenados en el contenedor que tienen como valor esa propiedad.

Realizar un test creando un tipo contenedor y leyendo en él los datos contenidos en el fichero CSV. Probar todos los métodos que realizan tratamientos secuenciales.

La entrega debe incluir la implementación y prueba del tipo contenedor y la factoría, así como la documentación de ambos, tanto en el fichero README.md como en el propio código mediante comentarios de documentación.

Entrega 3 (20-05-2022)

Añadir al tipo contenedor un tercer constructor con todas las propiedades básicas (excepto la colección) y un Stream de objetos del tipo base, que cree un objeto del tipo contenedor con todos los elementos del Stream.

Añadir a la factoría, realizada en la entrega 2, un método que recibe como parámetro una cadena que contiene el nombre y ruta del fichero CSV, y devuelve un objeto del tipo contenedor creado mediante el constructor anterior.

A continuación, añadir al tipo contenedor los siguientes métodos, todos ellos implementados mediante Streams.

Bloque I: Implementar, documentar y probar CINCO métodos que trabajen sobre el dataset y respondan a preguntas interesantes.

6) A escoger uno de los dos siguientes: existe / para todo (el mismo implementado en la entrega 2, pero con streams).
7) A escoger uno de los tres siguientes: contador/suma/media (el mismo implementado en la entrega 2, pero con streams).
8) Una selección con filtrado (la misma implementada en la entrega 2, pero con streams).
9) Un máximo/mínimo con filtrado.
10) Una selección, con filtrado y ordenación.

Bloque II: Implementar, documentar y probar CINCO métodos que trabajen sobre el dataset y respondan a preguntas interesantes.

11) Los métodos (4) y (5) implementados en la entrega 2, pero con streams.
12) Un método en cuya implementación se use, o bien el Collector collectingAndThen, o bien el Collector mapping.
13) Un método que devuelva un Map en el que las claves sean un atributo o una función sobre un atributo, y los valores son máximos/mínimos de los elementos que tienen ese valor.
14) Un método que devuelva un SortedMap en el que las claves sean un atributo o una función sobre un atributo, y los valores sean listas con los n mejores o peores elementos que comparten el valor de ese atributo (o función sobre el atributo).
15) Un método que calcule un Map y devuelva la clave con el valor asociado (mayor o menor) de todo el Map.

La entrega debe incluir la implementación y prueba de los métodos, así como la documentación de todos ellos, tanto en el fichero README.md como en el propio código mediante comentarios de documentación.