Proyecto Cápsulas

En este proyecto se trabajará con datos sobre cápsulas espaciales de Space X, lanzadas entre los años 2006 y 2020. En estos datos encontramos la siguiente identidad:

-**Cápsula**: contiene la información relativa al lanzamiento y estado de una cápsula en concreto, dando los datos siguientes, disponibles en formato CSV.

capsule_id	serial	status	reuse_count	water_landings	land_landings	Launch_date
5e9e2c5bf35918ed873b2664	C101	retired	true	1	3	26/11/2014
5e9e2c5bf3591882af3b2665	C102	retired	true	1	4	25/12/2018
5e9e2c5bf3591835983b2666	C103	unknown	true	1	3	20/07/2017
5e9e2c5bf359189ef23b2667	C104	unknown	true	1	2	11/02/2007
5e9e2c5bf3591859a63b2668	C105	unknown	true	1	3	18/02/2006
5e9e2c5bf3591880643b2669	C106	active	true	3	2	22/07/2009
5e9e2c5bf35918165f3b266a	C107	unknown	true	1	3	30/08/2018
5e9e2c5cf359188bfb3b266b	C108	active	true	3	2	01/06/2011
5e9e2c5cf35918407d3b266c	C109	destroyed	true	1	0	16/06/2013

Los elementos que se implementarán en este proyecto se incluirán en el paquete **fp.capsula**. Los aspectos más destacables son:

-**Capsula**: clase para implementar el tipo básico.

-**Archivo**: tipo contenedor que incluye, además, algunos métodos de consulta basados en tratamientos secuenciales.

-**FactoriaCapsula**: Clase para dar soporte a la creación de objetos tipo **Capsula** y **Archivo** a partir de los datos en el fichero CSV.

1) Crear la clase **Capsula** con los siguientes atributos:

·) **Id**: atributo *String* con la id de la cápsula.

·) **Serial**: atributo *String* con el código serial de la cápsula, debe contener una “C”.

·) **status**: atributo *Enum* con el estado de la cápsula, todas deben tener uno (**ACTIVE, DESTROYED, RETIRED, UNKNOWN**).

·) **Reuse**: atributo *Boolean* que indica si la cápsula ha sido reutilizada o no.

·) **Water**: atributo I*nteger* con el número de amerizajes que ha realizado la cápsula.

·) **Land**: atributo *Integer* con el número de aterrizajes que ha realizado la cápsula.

·) **Launch**: atributo *LocalDate* con la primera fecha de lanzamiento.

2) Crear los siguientes métodos de la clase **Capsula** comprobando las restricciones de los atributos en los casos que sea necesario:

·) **Capsula**: constructor de la clase a partir de los atributos en el orden que se indica anteriormente.

·) **Capsula**: segundo constructor de la clase a partir de una cadena de caracteres con el siguiente formato:
"capsule_id;serial;status;reuse_count;water_landings;land_landings;Launch_date"

·) **Métodos getters**: para todos los atributos de la clase.

·) **Capsula::aterrizajesTotales**: propiedad derivada que indica el número de aterrizajes (tierra y agua) totales de una cápsula.

·) **Capsula::toString**: mostrando todos los atributos.

·) **Capsula::hashcode**: usando todos los atributos para determinar la igualdad.

·) **Capsula::equals**: usando la misma selección de atributos que el método *hashcode*.

·) **Capsula::compareTo**: para determinar un orden natural.

3) Crear la clase **Archivo** con los siguientes atributos y métodos:

·)**capsulas:** atributo con un conjunto de objetos **Capsula**.

·)**Archivo:** constructor vacío de la clase **Archivo**.

·)**Archivo:** constructor de la clase **Archivo**.

·)**Archivo::añadirCapsula:** método para añadir una **Capsula** al conjunto de capsulas.

·)**Archivo::toString:** mostrando todos los atributos.

·)**Archivo::CalcularStatus:** cuenta cuántas cápsulas se encuentran en el estado dado como parámetro.

·)**Archivo::Reciclar:** devuelve *true* si se han reutilizado todas las cápsulas al menos una vez.

·)**Archivo::CalcularIdComun:** devuelve *true* si todas las cápsulas comparten los mismos primeros 9 caracteres de sus Ids.

·)**Archivo::CalcularMediaLandings:** calcula la media de aterrizajes totales (tierra y agua) entre todas las cápsulas.

·)**Archivo::SalvarPlaneta:** devuelve un *map* que asocia el serial de las cápsulas con aquellas que han sido reutilizadas, al menos, cuatro veces (se utiliza como criterio para esta última condición el número de aterrizajes totales).

·)**Archivo::ordenFecha:** devuelve un *Stream* con las cápsulas ordenadas por fecha.

·)**Archivo::minimotierra:** indica el menor número de aterrizajes en tierra de entre todas las cápsulas cuyo lanzamiento fue previo a la fecha dada como parámetro.

·)**Archivo::AgruparStatus:** devuelve un map que asocia los *Status* con las cápsulas que se encuentran en dicho estado.

·)**Archivo::calcularCapsulaMayorAterrizajes:** devuelve la cápsula que ha tenido mayor número de aterrizajes totales.

·)**Archivo::lost**: Devuelve un *map* que asocia las *Ids* de las cápsulas con las mismas para las que cuyo estado es *unknown*.

·)**Archivo::waterfriendly**: Devuelve un *map* que asocia los *códigos seriales* de las cápsulas con aquellas que han aterrizado 2 o más veces en masas de agua.

4) Crear la clase **FactoriaCapsula** con los siguientes métodos estáticos:

·)**FactoriaCapsula::leerCapsulas:** método que devuelve un objeto **Archivo** a partir de la ruta del fichero en el que se encuentran los datos de las cápsulas.

·)**FactoriaCapsula::parseaCapsulas:** método para construir un objeto **Capsula** a partir de una línea del fichero CSV de entrada.