Práctica examen Extraordinario

TAREA 1: Ficheros y generación de datos

TAREA 1D) [1.25 puntos] (ficheros: DeustoSpace, Principal)

Programa un método **cargarPlanetasCSV()** en la clase **DeustoSpace** que actualice la lista de los planetas añadiéndole los datos del fichero "planetas.csv".

El fichero planetas.csv contiene los siguientes datos (separados por ";"):

Planeta;Colores;Masa (10^24kg);Longitud del Día (horas);Distancia del Sol (10^6 km);Temperatura Media (°C);¿Anillos?;Composición

Por ejemplo:

Marte;Rojo,Marrón,Beige;0.642;247;228;-65;No;Roca,Hielo

Los campos "Colores" y "Composición" pueden estar compuestos por más de un elemento. Si es así, están separados por comas. En el csv el campo Anillos contiene los valores Sí/No, pero deberás guardar el dato booleano.

Añade código de control para ignorar las líneas del fichero que sean incorrectas y se procese el resto. Indica por consola el error que corresponda: "faltan datos", "un dato entero o real es erróneo", "el dato booleano no es correcto", "ha ocurrido un error inesperado".

Descomenta la llamada a este método en el main de la clase Principal para cargar los datos de las misiones. Comprobarás que existen tres errores en líneas del fichero de entrada.

TAREA 1E) [1.25 puntos] (ficheros: DeustoSpace, Mision, Principal)

Programa un método **asignarDestino()** en la clase **DeustoSpace** que:

1. Cree 2 planetas con 2 compuestos cada uno y que no se repitan los compuestos
2. Añada estos planetas a la lista de planetas
3. Asigne a cada misión un planeta de destino aleatorio.

*Para que funcione tendrás que cambiar el tipo del atributo destino en la clase Mision y actualizar el método toString()

TAREA 1F) [1.25 puntos] (ficheros: DeustoSpace, Principal)

Programa un método **guardarPlanetasCSV()** en la clase **DeustoSpace** que guarde en un csv los datos de cada planeta y cuantas veces se ha viajado a él.
Por ejemplo:

Marte;Rojo,Marrón,Beige;0.642;247;228;-65;No;Roca,Hielo,4

Importante: Si no has podido completar las tareas 1A, 1B, 1C, 1D Y 1F, descomenta la llamada al método datosIniciales() en el main de la clase Principal para tener datos suficientes para poder realizar el resto de tareas.

TAREA 2: Programación Orientada a Objetos

TAREA 2E) [1.5 puntos] (ficheros: Mision, Alcanzable, Principal)

Crea una interfaz **Alcanzable** con los métodos boolean **esAlcanzable()** y double **getCosteAdicional()**. Los planetas podrán ser subvencionables:

- Si el destino de la misión (el planeta) está a una distancia del sol menor a 50 x **10^6 km,** el planeta es alcanzable.
- Si el planeta no es alcanzable, significa que una nave no puede llegar a tanta distancia sin repostar, por lo que, **getCosteAdicional()** calculará el incremento de la siguiente forma: duracionDia*distanciaAlSol. En caso contrario devolverá 0.

*Recuerda que deberás cambiar el método getCosteTotal() de la clase Mision

TAREA 2F) [1.5 puntos] (ficheros: Mision, Principal)

Haz que las misiones se puedan ordenar en función del nombre de su planeta de destino de forma descendente.
Para ello tendrás que comentar la tarea 2D.

TAREA 3: Colecciones de datos

TAREA 3C) [1 puntos] (ficheros: DeustoSpace)

Programa un método **misionesPorDestino()** en la clase DeustoSpace que devuelva un mapa con el Planeta de destino como clave y una lista de misiones como valor.

TAREA 3D) [1 puntos] (ficheros: DeustoSpace)

Programa un método **personalPorPais()** en la clase DeustoSpace que devuelva un mapa con el país clave y un contador de su personal como valor.

Luego imprímelo. Deberá tener una pinta similar a la siguiente:

Austria 1...
- Astronauta Carmen Possnig (Austria): [INVESTIGACION, MEDICINA]

Belgium ..
- Astronauta Raphaël Liégeois (Belgium): [INVESTIGACION, MEDICINA]
- Personal de tierra Ana García (Belgium nivel 1)

- Personal de tierra Andrea Johnson (Belgium nivel 1)
- Personal de tierra Mark Becker (Belgium nivel 1)

…

TAREA 4: Pruebas unitarias

TAREA 4C) [1.5 puntos] (ficheros: PlanetaTest)

Crea un test unitario para el **constructor** de la clase Planeta. Crea un objeto de la clase Planeta y comprueba que sus atributos se han asignado correctamente. Crea otro objeto y llama al método **provocarError()** y asegúrate de que se genera una excepción de **ArithmeticException**.