PROGRAMACIÓN II – EXAMEN FINAL
01/07/2021 – MODALIDAD VIRTUAL

OBSERVACIONES

- Descomprimir el archivo Final_2021_Julio_Alumno.zip en el eclipse-workspace de su máquina.
- Cambiar el nombre de la carpeta Final_2021_Julio_Alumno la parte de alumno por su apellido.
- Cambiar el atributo <name> del proyecto en el archivo ".project" de Final_2021_Julio_Alumno la parte de Alumno por su apellido.
- Abra el eclipse y proceda a importar en proyecto.
- Deberá resolver las actividades descriptas a continuación en el proyecto antes mencionado y antes de finalizar el tiempo enviar el proyecto comprimido al mail jorgedat@gmail.com para dar el cumplimiento a la actividad. Esto debe ser realizado antes de la hora de finalización 01/07/2021 12:00 hs. Se recomienda empezar a subir la actividad antes de las 11:55 hs.

ACTIVIDAD 1

Para el siguiente enunciado codificar las clases que den solución. Guiarse con el enunciado y los test.

Un vehículo se caracteriza por el apellido y nombre de su dueño (dos String), la marca (que puede ser FORD, PEUGEOT o FIAT), el modelo (int) y la cilindrada (double).

Se debe contar con un constructor que permita crear un vehículo con todos los argumentos y valide con una excepción personalizada los casos mencionados en el siguiente cuadro.

Parámetro Controlado	*Tipo Control*	*Mensaje mostrado*
duenioApellido	***Cadena de longitud igual a 0.***	***"Apellido Vacío"***
duenioNombre	***Cadena de longitud igual a 0.***	***"Nombre Vacío"***
modelo	***Modelo anterior a 2001***	***"Modelo muy viejo"***

Se deben implementar solo los getter's, no los setter's.

El método toSting() debe devolver:

- ***el o los apellidos del dueño en mayúsculas a simple espacio,***
- ***concatenado con una coma seguida de un espacio,***
- ***concatenado con las iniciales de el o los nombres en mayúsculas con punto y a simple espacio,***
- ***concatenado con la marca entre paréntesis.***

El método equals() debe validar que dos vehículos son iguales cuando coinciden el nombre y apellido del dueño y la cilindrada.

Se debe contar con un gestor de vehículos que debe ser singleton y debe poder realizar las siguientes:

- ***Almacenar un ArrayList de Vehiculos.***
- ***Un método de firma <u>getCantidadDeVehiculos(): int</u> que devuelva un int que es la cantidad de vehículos cargados.***
- ***Un método de firma <u>addVehiculo(Vehiculo v): boolean</u> que permite cargar un nuevo vehículo al ArrayList cuando el mismo no sea duplicado.***
- ***Un método de firma <u>limpiarVehiculos(): void</u> que permite borrar todos los vehículos cargados en el ArrayList.***
- ***Un método de firma <u>getVehiculo(Predicate<Vehiculo> p , Comparator <Vehiculo> c) : ArrayList<Vehiculo></u> que permite recibir un predicado (Stream/Lamdas) y un comparator para realizar búsquedas filtradas por el predicado y ordenadas por el comparator y devuelve un ArrayList con los resultados obtenidos y ordenados.***

REQUISITOS

1) Tomando en cuenta el enunciado anterior hacer pasar los test del uno al ocho del proyecto.
2) No se debe modificar el contenido de los test salvo en los test siete y ocho en donde estos los indican con comentarios.
3) En el test siete completar:
    a. Con un predicado que valida que el apellido del dueño contenga el string parteDuenioapellido1.
    b. Con un comparator que ordene vehículos por cilindrada (de menor a mayor).
    c. Con un predicado que valida que el apellido del dueño contenga el string parteDuenioapellido2.
4) En el test ocho completar:
    a. Con un predicado que valida que el modelo del vehículo se encuentre entre los valores limiteInferior1 y limiteSuperior1.
    b. Con un comparator que ordene vehículos por apellido del dueño y ante empates por su nombre.
    c. Con un predicado que valida que el modelo del vehículo se encuentre entre los valores limiteInferior1 y limiteSuperior1.
5) Codificar un test nueve que valide que se agrega un campo nuevo a vehículo de llamado fechaAlta (GregorianCalendar). El test debe validar que:
    a. Exista un constructor con los ahora seis argumentos,
    b. Que exista un método que devuelva la fecha como un String con el formato “dd/mm/yyyy”.
    c. Que exista un método que calcule y devuelva la antigüedad de alta del vehículo en años (int).

   Sobre este test nueve no se debe escribir el código que lo haga pasar, solo el código del test propiamente dicho. Se evaluará la completitud de lo solicitado y su robustez.