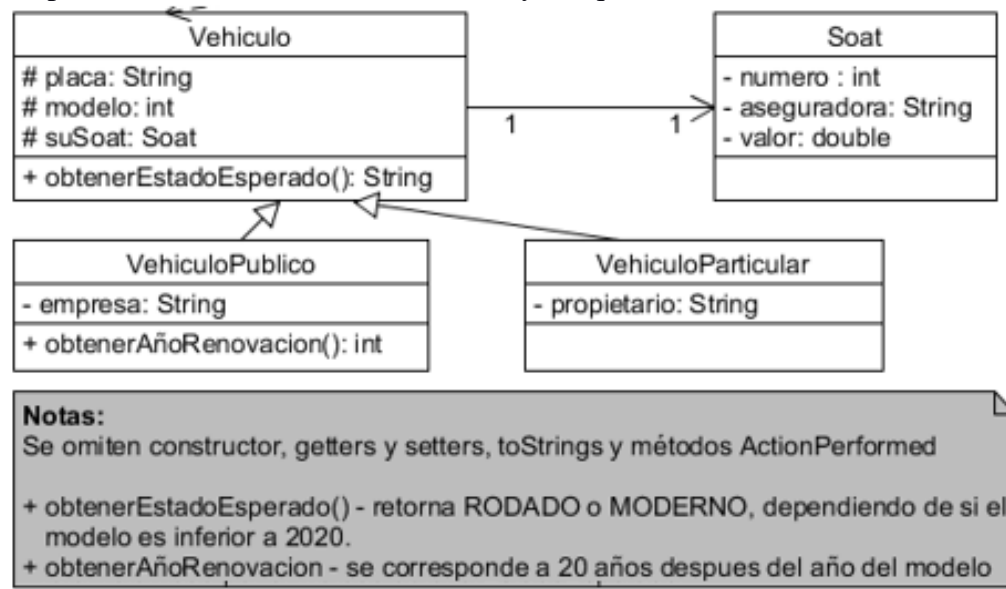


	<b>UNIVERSIDAD AUTÓNOMA DE OCCIDENTE</b>				<i>Valoración</i>
	<b>FACULTAD DE INGENIERÍA NUCLEO MIDIA</b>		<b>ESTRUCTURAS DE DATOS Y ALGORITMOS 1</b>	<b>GRUPO 1 HL</b>	
	<b>CÓDIGO:</b>		<b>NOMBRE:</b>		
<b>Caso 3</b> <b>POO, Ordenamiento y Búsqueda</b>					<b>FECHA:</b>

**Profesor:**  
**Orlando Arboleda Molina**

Tomando como referencia el proyecto **SolucionCaso3** suministrado, el cual debe ser procesado en el framework de desarrollo **Visual Studio Code**, se solicita un aplicativo web frontEnd que permita gestionar vehículos que pueden ser públicos o privados, como se indica en el siguiente diagrama de clase y realizar procesos eficientes de ordenamiento y búsqueda:



La solución debe satisfacer los siguientes requerimientos:

Estructura del proyecto a realizar:

- Se debe utilizar la estructura existente en el proyecto dado, en el que se tienen los siguientes archivos:
  - Logo-50-años.png – imagen que debe existir en el encabezado del aplicativo solicitado
  - ordenamientos.js – modulo con las funciones para realizar ordenamientos Quicksort o usando el Api del lenguaje
  - main.js – la lógica del aplicativo solicitado, que se debe realizar en la función *procesarFuncionalidad*, que dependiendo de la opción requerida desde la página web, permite ingresar nuevos vehículos y ordenarlos por placa, ordenar por modelo, o realizar su búsqueda.
  - Soat.js, Vehículo.js, VehiculoParticular.js, VehiculoPublico.js – la lógica de las clases indicadas en el diagrama de clase.
  - default.css – la hoja de estilos

- index.html – página web del aplicativo

Funcionalidades al procesar cada solicitud (realizar en la función *procesarFuncionalidad* según lo mostrado en la figura 1):

1. opción **Ingresar-OrdenarPlaca** – se debe leer los datos suministrados de forma masiva (cadena de tamaño  $m$ , en la cual se suministra la información de  $n$  vehículos), crear los vehículos que correspondan, almacenarlos en un arreglo, luego ordenar de forma ascendente por la placa usando el método *quickSort\_por\_Placa* u *ordena\_Placa*, que son definidos en *ordenamientos.js* y desplegar los datos suministrados de cada vehículo y valor retornado por la función *obtenerEstadoEsperado* (ver figura 2).
2. opción **Ordenar por Modelo** – se debe realizar el ordenamiento ascendente de los vehículos según su modelo, usando el método *quickSort\_por\_Modelo* u *ordena\_Modelo*, que son definidos en *ordenamientos.js* (ver figura 3).
3. opción **Buscar por Placa** – se debe realizar la búsqueda binaria de un vehículo, según su placa, usando el método suministrado en *busqueda.js*. La salida debe tener el siguiente formato (ver figuras 4 y 5)

No existe el vehiculo <placa>  
o  
<posicion>. <toString del vehiculo> Estado:<obtenerEstadoEsperado del vehiculo>

1 Recreativo CBD156 2010 1234 MAFRE 1200000;2 Juan TAB291 2015 4254 SURA 900000;1 Papagayo FDA647 2019 9872 SURA 900000;2 Miriam DAC064 2021 7209 MAFRE 750000;1 Mio BPX870 2021 4903 MAFRE 2500000

Los datos suministrados fueron

Ingresar-OrdenarPlaca ▼ Ejecutar funcionalidad

Ingresar-OrdenarPlaca  
Ordenar por Modelo  
Buscar por Placa

profesor Orlando A. ... el curso de EDyA1 en el Universidad  
Occidente

Figura 1- opciones proporcionadas por el aplicativo

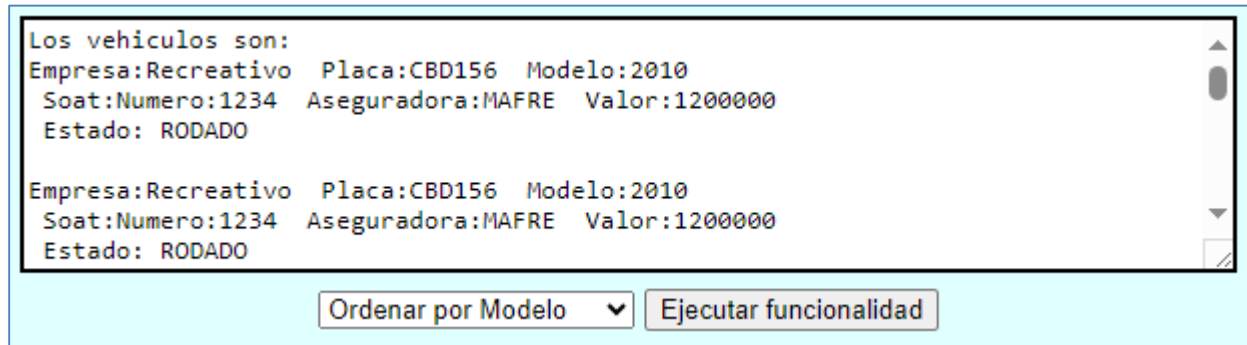
Los vehiculos son:

Empresa:Mio Placa:BPX870 Modelo:2021  
Soat:Numero:4903 Aseguradora:MAFRE Valor:2500000  
Estado: MODERNO

Empresa:Mio Placa:BPX870 Modelo:2021  
Soat:Numero:4903 Aseguradora:MAFRE Valor:2500000  
Estado: MODERNO

Ingresar-OrdenarPlaca ▼ Ejecutar funcionalidad

Figura 2- salida Ingresar-OrdenarPlaca



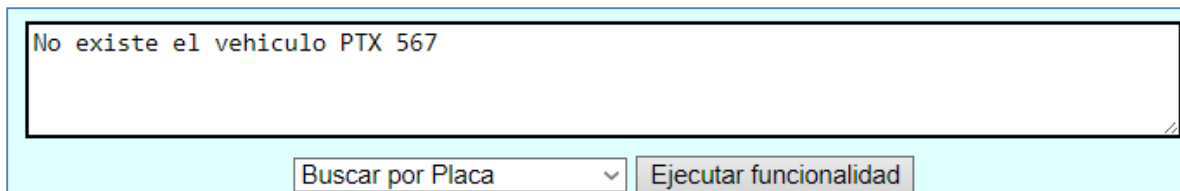
Los vehiculos son:  
Empresa:Recreativo Placa:CBD156 Modelo:2010  
Soat:Numero:1234 Aseguradora:MAFRE Valor:1200000  
Estado: RODADO

Empresa:Recreativo Placa:CBD156 Modelo:2010  
Soat:Numero:1234 Aseguradora:MAFRE Valor:1200000  
Estado: RODADO

Ordenar por Modelo ▾ Ejecutar funcionalidad

This screenshot shows a text area with vehicle information. The text is repeated twice. Below the text area, there is a light blue bar containing a dropdown menu set to 'Ordenar por Modelo' and a button labeled 'Ejecutar funcionalidad'.

Figura 3- salida Ordenar por Modelo

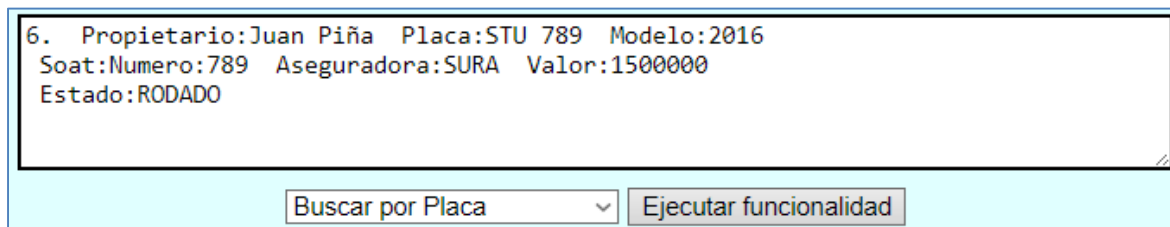


No existe el vehiculo PTX 567

Buscar por Placa ▾ Ejecutar funcionalidad

This screenshot shows a text area with the message 'No existe el vehiculo PTX 567'. Below the text area, there is a light blue bar containing a dropdown menu set to 'Buscar por Placa' and a button labeled 'Ejecutar funcionalidad'.

Figura 4- información desplegada cuando no se encuentra un vehículo



6. Propietario:Juan Piña Placa:STU 789 Modelo:2016  
Soat:Numero:789 Aseguradora:SURA Valor:1500000  
Estado:RODADO

Buscar por Placa ▾ Ejecutar funcionalidad

This screenshot shows a text area with vehicle information for a specific vehicle, identified by the number '6.'. The text includes the owner's name, license plate, model, SOAT number, insurance company, value, and status. Below the text area, there is a light blue bar containing a dropdown menu set to 'Buscar por Placa' and a button labeled 'Ejecutar funcionalidad'.

Figura 5- información desplegada cuando si se encuentra un vehículo

## Actividad Evaluable 3

### 1. Definición de la Tarea

Generar la solución solicitada y su correspondiente análisis de complejidad.

### 2. Entrega

La entrega consiste en un archivo comprimido en formato zip o rar, que contenga:

#### 1. Un documento **Word versión 2013 (o inferior) o pdf**, de máximo 5 páginas, que incluya:

- Una hoja de portada en la que se indique claramente: los códigos y nombres de los alumnos que hacen la entrega, el grupo de EDyA1 en el que están matriculados.
- Una hoja con un párrafo de máximo 10 líneas en que se resuma en que consiste el aplicativo generado, el código/pseudocódigo de la opción **IngresarOrdenar** al interior de la función **procesarFuncionalidad** y su cálculo detallado de complejidad (en la cual deben aparecer los términos  $m$  y  $n$  indicados previamente).

#### 2. El aplicativo web realizado en JavaScript, sin errores de compilación, con datos iniciales de prueba.

**Nota:** La implementación **no será evaluada** si **no se suministra el documento** o en este **no se realiza el cómputo de complejidad solicitado**.

**Fecha de Entrega:** en la plataforma UAOVirtual, a más tardar a las 8pm del **domingo 28 de abril de 2024**.

**Sustentación:** Se realiza en la semana 7 en el horario escogido con anterioridad. Debido a las limitaciones de tiempo, el horario puede corresponder a un día festivo.

### 3. Grupos de trabajo

Grupos de 3 personas. Solo se permite una cantidad diferente, si es consultada al docente y este da su aval.

### 4. Bibliografía Mínima

Los textos guías del curso y enlaces indicados al describir el presente proyecto.