PARTE 2: Ejercicio de resolución de un problema mediante un programa en JAVA

Duración 90 minutos

Este ejercicio comprende **2** actividades principales:

1. Comprensión del problema con su conjunto de datos asociado, análisis de métodos de ordenación que puedan aplicarse para resolver el problema, elección de uno de ellos y justificación.
2. Resolución del problema:
   a) Desarrollo del programa JAVA necesario para resolver el problema (de ordenación) planteado.
   b) Desarrollo de los casos de prueba ("test case", al menos uno) para verificar la corrección de la funcionalidad implementada.

ESCENARIO

Una empresa global de asesoramiento comercial para la venta de automotores tiene información de una gran cantidad de automóviles, modelos, características, etc., a nivel global, con el objetivo de brindar a sus clientes – empresas de venta de vehículos, mayoristas, usuarios finales, distribuidos en todo el mundo – información que les ayude a tomar decisiones de negocios.

La empresa cuenta con una gran base de datos, que se integra continuamente con la información de automotores que se van poniendo en el mercado.

Una consulta típica que los clientes desean que la empresa les resuelva, es obtener un listado de vehículos, ordenados por marca, luego por potencia y en tercer lugar por precio (en forma anidada).

Para poder trabajar en este ejercicio, se provee una muestra **reducida** de un conjunto de datos descargado y ensamblado en un cierto momento, "**autos*.csv***" (ten en cuenta que la base de datos completa contiene millones de registros de este tipo).

Cada automóvil se encuentra representado en una línea. El primer campo es "***id***", el identificador único de cada automotor en la base de datos

Descarga de la webasignatura el archivo "**Parcial3-2019.zip**" que contiene el conjunto de datos indicado y la clase java "**Automovil**".

Observa los atributos / datos asociados a cada automotor (al reverso):

id	Alfanumérico, identificador único de la base de datos
marca	alfa-romero, audi, bmw, chevrolet, dodge, honda,isuzu, jaguar, mazda, mercedes-benz, mercury,mitsubishi, nissan, peugot, plymouth, porsche,renault, saab, subaru, toyota, volkswagen, volvo
combustible	diesel, nafta
puertas	dos, cuatro
tipo	hardtop, wagon, sedan, hatchback, convertible
traccion	4wd, fwd, rwd
posmotor	delantero, trasero
largo	continuo de 141.1 a 208.1.
ancho	continuo de 60.3 a 72.3
alto	continuo de 47.8 a 59.8.
tipo_motor	dohc, dohcv, l, ohc, ohcf, ohcv, rotor
cilindros	dos, tres, cuatro, cinco, seis, ocho, doce
tam_motor	continuo de 61 a 326.
sist_comb	1bbl, 2bbl, 4bbl, idi, mfi, mpfi, spdi, spfi
diam	continuo de 2.54 a 3.94
carrera	continuo de 2.07 a 4.17.
compresion	continuo de 7 a 23
hp	continuo de 48 a 288.
rpm_max	continuo de 4150 a 6600.
consumo_ciudad	continuo de 13 a 49
consumo_autopista	continuo de 16 a 54.
precio	continuo de 5118 a 45400.

SE REQUIERE:

Análisis de los datos, alternativas de algoritmos de ordenación, elección y justificación.

1. Realiza una inspección de los datos para observar los campos incluidos en cada línea.
    a. Analiza posibles algoritmos de ordenación apropiados para, a partir del conjunto de datos de entrada indicado, emitir un listado de los automóviles, ordenados en forma anidada por los siguientes criterios: Marca (ascendente alfanumérico), Potencia (HP) (ascendente numérico), Precio (ascendente numérico)
2. Selecciona / Diseña el algoritmo que te parezca más apropiado, tomando en cuenta criterios de tiempo de ejecución, consumo de memoria y complejidad del algoritmo. Justifica esta elección.
3. Reporta todas estas consideraciones en un **breve** (no más de 1 carilla) documento "***analisis.doc***" (a entregar junto con todo el código que desarrolles y el archivo de salida)

Funcionalidad a desarrollar:

1. Genera una estructura apropiada para representar el problema y cargarla a partir del archivo de entrada indicado. Para facilitar, se provee una clase "**Automovil**" cuyo constructor toma como parámetro una línea completa del archivo de entrada y crea una instancia de esta clase, utilizando el campo "***id"*** como "clave" primaria (la clave por la cual deberá hacerse la ordenación).
2. Realiza la ordenación del conjunto de datos **con la implementación que decidiste en la primera parte del trabajo (análisis), en orden compuesto por:**
    a. **Marca**
    b. **Potencia HP**
    c. **Precio**
3. DEBES IMPLEMENTAR **SOLAMENTE** EL ALGORITMO QUE HAS DISEÑADO o SELECCIONADO Y DOCUMENTADO PREVIAMENTE (NO SE ACEPTARÁN OTROS, pero sí puedes documentar e implementar MAS de uno si lo deseas – será tenido en cuenta-).
4. Emite un archivo de salida "**informe_autos.txt**", que contenga los datos de los automóviles agrupados de la siguiente manera (ver ejemplo de formato del archivo de salida al final de este documento):
    - **Marca**
        - **Potencia**
            - **Precio**

TEST CASES.

Implementa los **Casos de Prueba** necesarios para verificar el correcto funcionamiento de los métodos desarrollados.

ENTREGA

Subir a la webasignatura, en la tarea "**PARCIAL3-2019**":

- El documento "analisis.doc"
- El proyecto completo realizado,
- El archivo de salida "**informe_autos.txt**".

RUBRICA DE CALIFICACIÓN:

Se utilizarán los siguientes criterios en la evaluación del trabajo remitido:

1. **ANÁLISIS: 30%**
    - Identificación del problema, revisión de los datos.
    - Evaluación de algoritmos de ordenación
    - Selección de mejor opción y justificación de la misma

2. DESARROLLO y EJECUCIÓN 40%

- Estructura de datos utilizada (pertinencia Y eficiencia)
- Implementación del método de ordenación con el algoritmo **SELECCIONADO EN LA PARTE DE ANÁLISIS**
- Correcta lectura del archivo de datos y creación de las estructuras definidas.
- Programa principal: ejecución en tiempo razonable, aplicación correcta del método de ordenación y generación **correcta** del archivo de salida.

3. CALIDAD DEL CÓDIGO (10 %)

- Selección inteligente de estructuras auxiliares en aras de reducir complejidad, tiempo de ejecución, uso de recursos en general.
- Nombres de variables y métodos
- Aplicación correcta y rigurosa del paradigma de programación orientada a objetos
- Invocación racional y eficiente de métodos y funciones
- Encapsulación, modularidad
- Utilización apropiada de clases de colecciones y genéricos necesarios

4. PRUEBAS DE UNIDAD 20%.

- Calidad de los tests desarrollados, todas las condiciones normales y de borde, se testean todos los métodos, uso del enfoque inductivo en los tests.

Ejemplo de formato de archivo de salida:

Marca: AUDI
50 HP
Id Auto1, Precio, combustible, puertas, tipo,cilindros, rpm_max
Id Auto2, Precio, combustible, puertas, tipo,cilindros, rpm_max
50 HP
Id Auto1, Precio, combustible, puertas, tipo,cilindros, rpm_max
Id Auto2, Precio, combustible, puertas, tipo,cilindros, rpm_max
Marca: FORD
30 HP
Id Auto7, Precio, combustible, puertas, tipo,cilindros, rpm_max
Id Auto15, Precio, combustible, puertas, tipo,cilindros, rpm_max
90 HP
Id Auto11, Precio, combustible, puertas, tipo,cilindros, rpm_max
Id Auto21, Precio, combustible, puertas, tipo,cilindros, rpm_max
Marca: VOLVO
110HP
Id Auto121, Precio, combustible, puertas, tipo,cilindros, rpm_max
Id Auto13, Precio, combustible, puertas, tipo,cilindros, rpm_max
Id Auto51, Precio, combustible, puertas, tipo,cilindros, rpm_max