Trabajo 2 Estructuras de Datos, 2020-2S

Universidad Nacional de Colombia, Sede Medellín

Integrantes: Máximo 4

Fecha de entrega y sustentación: Responder encuesta: https://forms.gle/uqUTAUCh4b8BYrH27

Este trabajo está basado en el trabajo anterior. Debe continuar con el mismo modelo del dominio. Puede realizar cambios al modelo del dominio si lo considera necesario, pero debe enviar un correo al docente indicando que cambio se realiza y por qué motivo, con el fin de ser autorizado.

Si bien este trabajo está basado en el anterior, solo las funcionalidades aquí descritas serán evaluadas.

1. **Migración del modelo a grafos**
   Realice los cambios necesarios en la declaración de clases de Java de modo tal que:
   - Las clases solo contengan sus atributos primitivos
   - Las instancias de las clases sean un vértice o nodo del grafo
   - Las relaciones entre las instancias sean aristas del grafo

   El CRUD para cada clase y sus relaciones debe continuar existiendo, pero será migrado a interfaz gráfica (Ver numeral más adelante). Recuerde que debe existir una opción para crear, editar, ver y eliminar cada una de las clases con sus atributos, así como las relaciones entre ellas (ver nota al final).

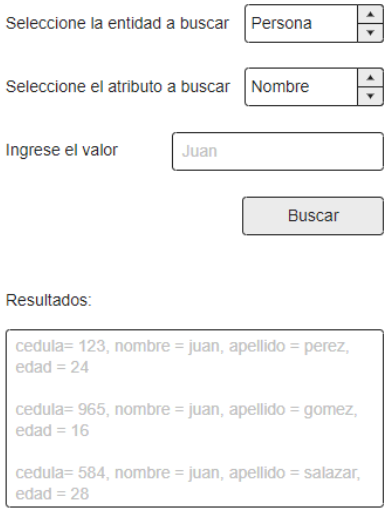
   La autenticación de usuario debe continuar existiendo, pero será migrado a interfaz gráfica (Ver numeral más adelante)

2. **Búsqueda**
   Debe existir un menú en el cual se permita realizar búsquedas a través de ciertos atributos. Elija tres atributos (clave primaria, uno numérico y uno de texto) para cada una de las clases a través de los cuales el usuario puede buscar.
   El menú debe permitir elegir una clase y un atributo, utilizando checkbox o radio button. Luego debe mostrar los dos atributos a buscar según la clase seleccionada y un textfield para ingresar el valor a buscar, luego debe mostrar en la consola de la UI los resultados (ver imagen), imprimiendo los objetos que cumplen con dicho criterio de búsqueda, junto con los valores de todos sus atributos primitivos.

   Debe determinar para cada clase una clave primaria. Tenga presente que en algunos casos no existe una clave primaria evidente, por lo que se debe crear una ID autogenerado

incremental, pero debe agotar primero las opciones lógicas que puedan servir como claves primarias.

Por facilidad, convierta todos los valores de texto ingresados por el usuario a minúscula, tanto en el CRUD como en los campos de búsqueda.



Las búsquedas se deben realizar utilizando árboles y/o tablas hash. Tenga presente que el modelo de grafo es la representación “real” de el sistema y la que tiene las relaciones entre los objetos, sin embargo, por tema de eficiencia, se mantendrán estructuras de datos alternas ordenando por los atributos de búsqueda de interés, de modo tal que al realizar una búsqueda se utilice dicha estructura de dato y sea más eficiente el proceso.
Para ello, cree árboles de búsqueda con eficiencia promedio log(n) para los atributos de búsqueda que no son clave primaria, y tablas hash para los atributos de búsqueda que son clave primaria. Estas estructuras de datos NO deben crear de nuevo los objetos, si no que deben apuntar al mismo objeto que el modelo del grafo contiene, para no crear objetos duplicados ni desperdiciar memoria. Por ejemplo, si el modelo del grafo tiene 10 personas, existirá un árbol con las 10 personas ordenadas por su nombre, existirá otro árbol de las 10 personas ordenadas por su edad, y una tabla hash de las 10 personas utilizando la cédula como clave; pero siempre son las mismas 10 personas (El mismo objeto, el mismo espacio de memoria), es decir que si se cambia un atributo de una de esas personas en el grafo, se debe cambiar “automáticamente” en los dos árboles y la tabla hash.
Al momento de dar clic en el botón “buscar”, el sistema debe utilizar la estructura de datos (árbol o hash) para realizar la búsqueda y mostrar los resultados.

Importante: Por definición, los árboles no permiten elementos duplicados, pero ya que se puede buscar por atributos que no son la clave primaria, se encontrará con el problema de que varios objetos pueden tener el mismo valor. Por ejemplo, varias personas pueden tener edad 23 en el árbol ordenado por edad. Por lo tanto, debes solucionar el problema de forma

tal que no se empeore la eficiencia de la búsqueda del árbol y se continúe utilizando el árbol para la búsqueda.

3. **Interfaz gráfica**
   Debe existir una interfaz gráfica funcional para:
   a. Iniciar sesión (Registrarse no estará disponible en este trabajo y por lo tanto no tendría UI). Si los datos son correctos debe enviar al menú principal de la UI, pero si son incorrectos debe mostrar un mensaje indicando el error.
   b. Administración: Interfaces para el CRUD de cada una de las clases.
   c. Búsqueda
   d. Salir

Notas:

- El guardado y lectura de archivos Json no estará disponible en este trabajo (Si lo quiere dejar está bien pero no será revisado). Debido a esto, "queme" el usuario y password con el que se puede iniciar sesión en el código.
- El proyecto debe continuar en Git (**debe ser una rama nueva en el mismo repositorio**) y utilizar Maven
- **Por facilidad, seleccione tres clases del modelo para este trabajo y realice las interfaces de CRUD y búsqueda únicamente para esas clases. Sin embargo, esa "porción" del modelo de clases seleccionada debe tener relaciones entre todas sus clases.**
- Utilice la librería para el grafo enseñada en clase
- Puede utilizar la librería para UI en Java que desee

Entregable: Link del repositorio de GitHub. El archivo README del repositorio debe contener los nombres y correos de los integrantes del equipo. **Además del usuario y clave con la que acceder al sistema**

Nota: El trabajo deberá ser sustentado en la fecha indicada. La asistencia de todos los integrantes del equipo es obligatoria y la única excusa para no asistir es aquella validada por la universidad. Se seleccionará aleatoriamente un integrante de cada equipo para sustentar, así que asegúrate de incluir en el trabajo solo aquellos que realmente trabajaron. La sustentación se realizará por Google Meet. Si la sustentación se realiza de manera adecuada, la nota final es la nota del trabajo según la calificación de los parámetros. Si la sustentación no se realiza de manera adecuada, la nota final del trabajo se puede ver disminuida hasta en 1.5. Si algún integrante no está trabajando de manera adecuada por favor reporte al profesor con debida anticipación para tomar acciones.