Programa de repaso espaciado para el aprendizaje de vocabulario.

Camilo García Martinez, Nikolái Guzmán Gómez

No. de Equipo Trabajo: 5

[1] INTRODUCCIÓN

En este informe se expone una descripción detallada del avance del proyecto de "Repaso Espaciado" que se está desarrollando a lo largo del curso Estructuras de Datos 2020-1.

I. DESCRIPCIÓN DEL PROBLEMA

Uno de los problemas de los programas de repaso espaciado en el mercado, es la falta de estructuras de relación que hay entre las tarjetas con excepción de ser parte de una lista común. Ya que se sigue usualmente sólo una estructura lineal de ítems, sin tomar en cuenta la relación que pueda haber entre estos.

Como el repaso espaciado se suele usar en un contexto educativo-académico, donde la interconexión entre diferentes conceptos es natural, proponemos un software que haga uso de diferentes estructuras de datos para proveer una clasificación de tarjetas más eficaz, que reduzca los tiempos de estudio y de repaso diario a hacer gracias a una clasificación y ordenamiento más efectiva.

Cada palabra va a estar relacionada con otras palabras, (carro con llantas, autopista, manejar, etc..) para establecer esta relación (que es determinada por el usuario), usaremos la estructura de árbol. Ya que con esta estructura podemos identificar la palabra rápidamente y localizar a su vez a su padre, sus hermanos y sus hijos inmediatos, que serían las palabras relacionadas más estrechamente con la palabra inicial.

II. USUARIOS DEL PRODUCTO DE SOFTWARE

El programa está dirigido principalmente a personas que quieran hacer uso del repaso espaciado para el aprendizaje de léxico de un idioma.

El programa cuenta con un solo rol, de usuario. El usuario instala el programa en su computador y puede acceder a todas las funcionalidades propuestas como crear listas, crear tarjetas, eliminar, modificar y crear relaciones entre tarjetas. El usuario también puede exportar o importar listas de tarjetas.

III. REQUERIMIENTOS FUNCIONALES DEL SOFTWARE

Nombre de la funcionalidad: Creación de Deck/Lista de Tarjetas

Descripción: El programa permite la creación de diferentes listas de tarjetas, por ejemplo, una lista de léxico francés y otra de léxico italiano. No se permite relación de ningún tipo entre tarjetas de una lista y las de otra. Las listas de tarjetas siguen una estructura de lista ordenada al momento de mostrarle al usuario..

Acción Iniciada y comportamiento esperado: Se crea una nueva lista. En principio vacía. El tipo de tarjeta, el ordenamiento y la programación de las tarjetas (horario) no tiene relación (por defecto) con ninguna otra lista.

Nombre de la funcionalidad: Crear Tarjeta y añadirla a un Deck.

Descripción: Las tarjetas son el producto principal del programa, son creadas por el usuario. No se pueden crear tarjetas vacías. Las tarjetas pueden tener una prioridad asociada y una etiqueta (verbo, adjetivo, sustantivo, etc…) además de indicar el Deck al que pertenece.

Acción Iniciada y comportamiento esperado: Se crea un objeto tarjeta con una cara y un revés, el usuario tiene que indicar a cuál Deck pertenece la tarjeta, si no lo hace, no se dejará crear la tarjeta. Cuando el usuario indica el Deck, el programa busca que la tarjeta no se encuentre ya en ese Deck, en caso de que eso ocurra, se le notificará al usuario por medio de un mensaje que esa tarjeta ya se encuentra en ese mazo. Ya que las tarjetas se almacenan en el Deck en estructura de árbol binario, en caso de que recorra todas las tarjetas y no encuentre la tarjeta, el usuario debe elegir a qué tarjeta de ese mismo Deck está relacionada directamente, osea, cúal va a ser su padre. Si el padre que eligió ya tiene 2 hijos, se le informará por medio de un mensaje al usuario que ese espacio ya está ocupado, y no dejará hacer la inserción. En caso tal de que sí esté ese lugar disponible, el programa procederá a agregarla en ese Deck. Aunque para realizar las pruebas con los diferentes tipos de datos, lo que se hizo fue añadir las tarjetas por un archivo de texto e ir organizando el árbol por comparación alfabética.

Nombre de la funcionalidad: Eliminar Tarjeta De un Deck.

Descripción: El usuario decide eliminar una tarjeta de un deck.

[1]

Acción Iniciada y comportamiento esperado: El usuario selecciona una tarjeta por un identificador determinado (como la cara de esta). El programa elimina la tarjeta que corresponda con el identificador, si no se encuentra una tarjeta que corresponda, no se hace nada y se le notifica al usuario que esa tarjeta no existe. La tarjeta sigue existiendo en la lista total de tarjetas.

Nombre de la funcionalidad: Impresión de deck +

Descripción: Se imprime la lista de tarjetas que son parte de un deck.

Acción Iniciada y comportamiento esperado:
Cuando el usuario escoge un deck para mostrar, el programa recorre todas las tarjetas que están dentro y las va imprimiendo una por una de manera individual, cada tarjeta indica cuáles son sus hijos.

Nombre de la funcionalidad: Presentación de tarjetas o Presentación de Revisión Diaria

Descripción: Dada una lista, se presentan una por una la cara y luego anverso de las tarjetas. La presentación de tarjetas es la funcionalidad principal del programa, es mediante esta función que el usuario determina si ha memorizado o no las tarjetas.

Acción Iniciada y comportamiento esperado: Se presenta la cara de una tarjeta. Luego se espera que el usuario presione una tecla y tras esto se muestra el anverso. El usuario decide si su suposición sobre el anverso fue correcta o no e ingresa una opción por teclado: 1 Correcta, 0 Incorrecta. Si la suposición fue correcta la tarjeta no aparece más durante la presentación de tarjetas, si fue incorrecta se volverá a mostrar mas tarde en la presentación de tarjetas. La presentación de tarjetas acaba cuando el usuario ha contestado a todas las tarjetas programadas para esa ocasión

Nombre Funcionalidad: Buscar una tarjeta en un deck.

Descripción: Se realiza una búsqueda por el usuario de una tarjeta, bien sea par modificarla o eliminarla o agregarla a un Deck.

Acción Iniciada y comportamiento esperado: El usuario realiza la búsqueda de una tarjeta poniendo el Front de la tarjeta, por ejemplo “car” o directamente el ID propio de cada tarjeta, el programa recorre la el árbol de tarjetas y busca coincidencias entre ellas y lo suministrado por el usuario. En caso de que haya encontrado, se le mostrarán al usuario, en caso contrario se le informará que no se hallaron coincidencias.

Nombre funcionalidad: Cambiar relación de tarjetas de un Deck.

Descripción: El usuario quiere cambiar la relación que había establecido primeramente entre las tarjetas.

Acción iniciada y comportamiento esperado: Sólo se pueden cambiar al mismo tiempo la relación entre 2 tarjetas. Las 2 tarjetas ya tiene que estar dentro del mismo Deck. Un ejemplo sería que el usuario quiere cambiar la tarjeta A al lugar donde está la tarjeta B. El programa guardará la información de la tarjeta tarjeta B en una memoria temporal y pondrá a la tarjeta A en ese sitio con los mismos apuntadores de hijos y padre que tenía la tarjeta B originalmente. Cuando ya se haya insertado la tarjeta A en su nuevo sitio. El programa inserta la tarjeta B en el lugar que anteriormente estaba la A, y queda con las mismas referencias que la tarjeta A tenía anteriormente.

IV. AVANCE EN LA IMPLEMENTACIÓN DE LA INTERFAZ DE USUARIO

Se implementaron algunas ventanas y escenas, entre ellas un Inicio con los decks cargados desde la base de datos. Ventanas para la creación e inserción de los decks y flashcards a la base de datos. Se añadió una presentación ordenada de las tarjetas de un deck previamente cargado desde la base de datos.

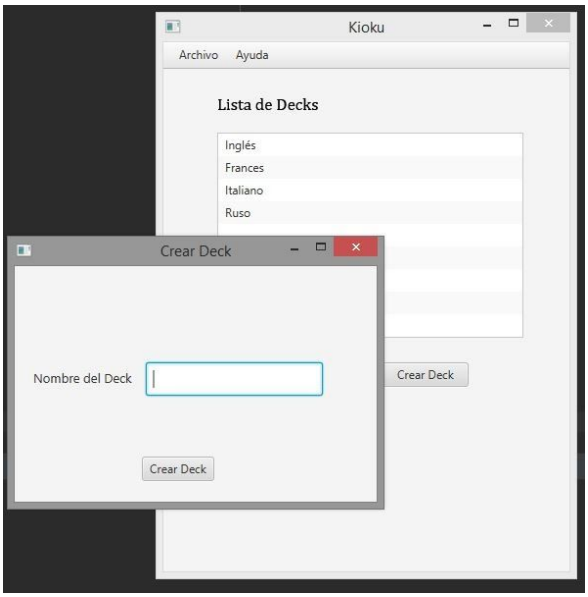


Figura 1. Ventana de Agregar Deck y ventana de Inicio, con una lista de Decks que usa la estructura de datos LinkedList.

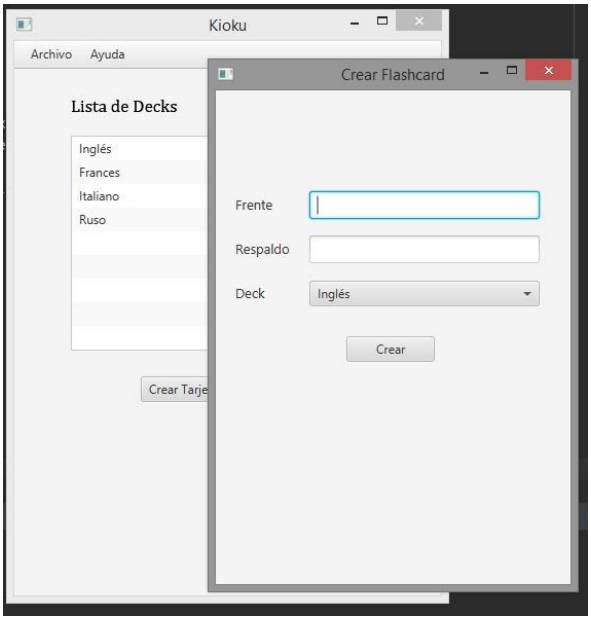


Figura 2. Ventana de Crear e Insertar una tarjeta en un deck.

Figura 3. Escena de la lección (antes de presionar Respuesta). Se usó orden de Pila como se requirió para esta entrega. El usuario puede volver al inicio por medio del botón en la parte superior izquierda.

Figura 4. Escena de lección tras ser presionado respuesta: Se muestra el anverso de la tarjeta y se le pregunta la dificultad al usuario. Tras esto se hace pop(), se modifica la dificultad asociada a la tarjeta y se cambia de tarjeta.

V. Entornos de desarrollo y de operación

El entorno de desarrollo IDE es IntelliJ IDEA. El programa corre sobre la máquina virtual de Java.

Para esta segunda entrega se implementó una interfaz de usuario mediante el uso de la plataforma JavaFX. Tambien se implemento una base de datos con SQLite, que se conecta a la aplicación por medio de la API JDBC (Java Database Connectivity).

VI. DESCRIPCIÓN DEL PROTOTIPO DE SOFTWARE

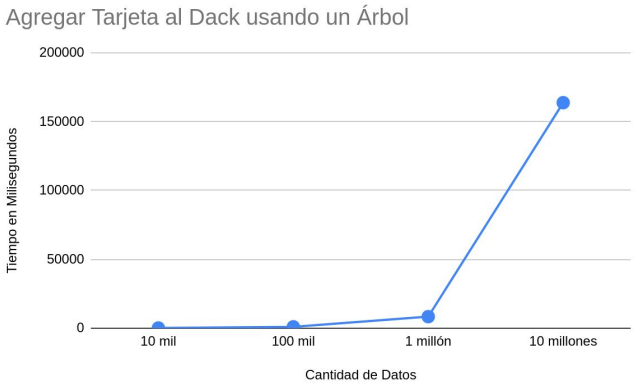
En esta entrega, se implementaron varias nuevas estructuras de datos importantes para el correcto funcionamiento de nuestro proyecto. Una de las más relevantes fue la del Árbol Binario de Búsqueda (BST) cuyos nodos son objetos FlashCards con referencia a su hijo izquierdo e hijo derecho. El árbol se encarga de ordenar las tarjetas dentro del Deck, volviendo las palabras del Front a enteros y comparándolas y ordenándose alfabéticamente, de esa manera se facilita muchísimo la búsqueda de una tarjeta en particular.
También se implementó una PriorityQueue para la misma funcionalidad de almacenar las tarjetas en un Deck, y aunque en datos grandes almacena más rápido los datos, al querer buscar un dato en concreto es muy ineficiente, ya que tiene que ir sacando los diferentes heads, hasta que el head se convierta en el elemento deseado. Ya que esta estructura de datos sólo permite acceder al head. Y no es muy útil para nuestro proyecto..

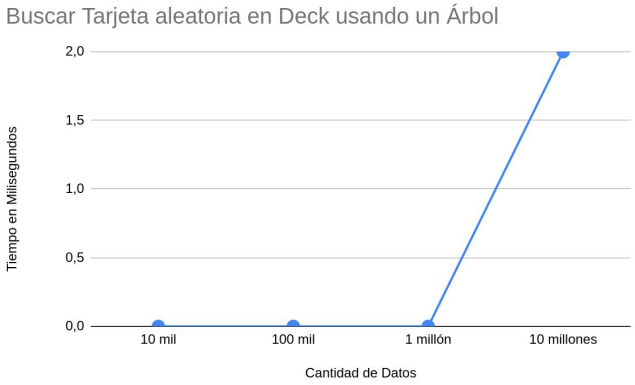
El segundo prototipo se encuentra en la segunda rama del repositorio:
https://github.com/KiokuDS/Kioku

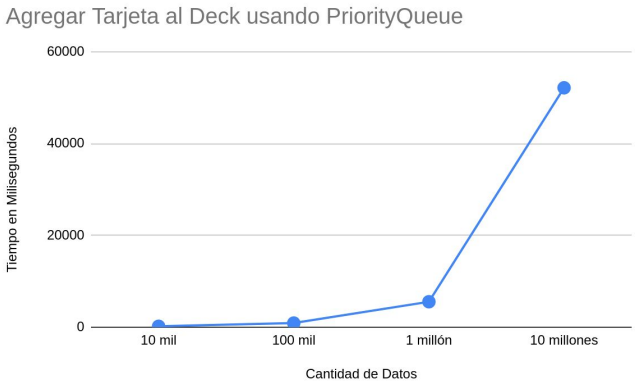
VII. PRUEBAS DEL PROTOTIPO

Tabla 1. Tabla comparativa de las pruebas realizadas.

Nombre de la funcionalidad	Tipo(s) de estructura de datos	Cantidad de datos probados	Análisis realizado (Notación Big O)	Tiempos de ejecución (en milisegundos)
Agregar FlashCards a un Deck	Árbol	- 10 mil -100 mil -1 millón -10 millones		184 1041 8429 163791
Agregar una FlashCard a un Deck	Árbol	- 10 mil -100 mil -1 millón -10 millones		0 0 0 0
Buscar una tarjeta en el Deck	Árbol	-10 mil -100 mil -1 millón -10 millones	O(log N)	0 0 0 2
Agregar FlashCards a un Deck	Priority Queue	- 10 mil -100 mil -1 millón -10 millones	O(log N)	210 913 5557 52158
Retirar elemento de mayor prioridad	Priority Queue	- 10 mil -100 mil -1 millón -10 millones	O(log N)	0 0 0 7







VIII. DIFICULTADES Y LECCIONES APRENDIDAS

Hubo dificultades en cuanto a la base de datos, es muy importante que creación del controlador escalable para la base de datos. Por lo tanto es prioritario mejorar el estado actual de este para facilitar la inserción, modificación y eliminación de tablas dadas las decisiones tomadas en la interfaz gráfica.