Programa de repaso espaciado para el aprendizaje de vocabulario.

Camilo García Martinez, Nikolái Guzmán Gómez

No. de Equipo Trabajo: 5

I. INTRODUCCIÓN

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

II. DESCRIPCIÓN DEL PROBLEMA A RESOLVER

Uno de los problemas de los programas de repaso espaciado en el mercado es la falta de estructuras de relación entre las tarjetas con excepción de ser parte de una lista común. Esto es, 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, 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.

III. 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.

IV. REQUERIMIENTOS FUNCIONALES DEL SOFTWARE

La tarjeta (Flashcard) es el objeto principal del programa. Está conformada principalmente por dos campos, un caray un anverso. Usualmente el objetivo es memorizar la relación entre estos campos. El usuario puede crear, editar, eliminar y ordenar estas tarjetas dentro de listas de tarjetas contenidas por objetos "Decks", que encapsulan otra información como la configuración de horarios.. La cara y el anverso no necesariamente esta compuestos de un solo elemento; Si la cara es una palabra en español, el anverso puede contener la traducción en el idioma extranjero deseado y además una frase usando la palabra, un audio o imagen asociada, etc.

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.

Acción Iniciada y comportamiento esperado: Se crear una nueva lista. 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.

Descripción: Las tarjetas se crean y se añaden primeramente a una lista donde están todas las tarjetas. No se pueden crear tarjetas vacías. Las tarjetas pueden tener una prioridad asociada y una etiqueta (verbo, adjetivo, sustantivo, etc…).

Acción Iniciada y comportamiento esperado: Se crea un objeto tarjeta y se añade a la lista total. Si la tarjeta ya se encuentra en la lista, se le notifica al usuario que ya está y que no puede crear la misma tarjeta 2 veces.

Nombre de la funcionalidad: Eliminar Tarjeta De un Deck.

Descripción: Se elimina una tarjeta de un deck..

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.

Nombre de la funcionalidad: Impresión de deck +

Descripción: Se imprime la lista de tarjetas que son parte de un deck, en el orden de la lista ordenada.

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.

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 una palabra clave por ejemplo “car” o directamente el ID propio de cada tarjeta, el programa recorre la lista 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: Añadir una tarjeta a un Deck.

Descripción: El usuario desea agregar una tarjeta a un Deck determinado.

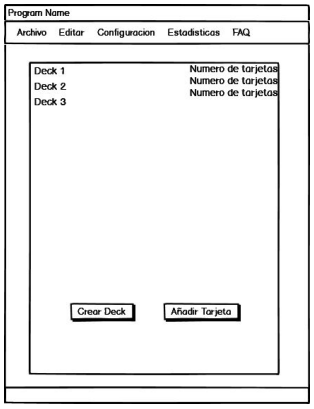
Acciones Iniciadas y Comportamiento Esperado: Primero, el programa busca que la tarjeta no se encuentre ya en ese Deck, en caso de que eso ocurra, se le mostrará al usuario por medio de un mensaje que esa tarjeta ya se encuentra en ese mazo. En caso de que recorra toda la lista y no encuentre la tarjeta, el programa procederá a agregarla en la lista de tarjetas de ese Deck.

V. DESCRIPCIÓN DE LA INTERFAZ DE USUARIO PRELIMINAR

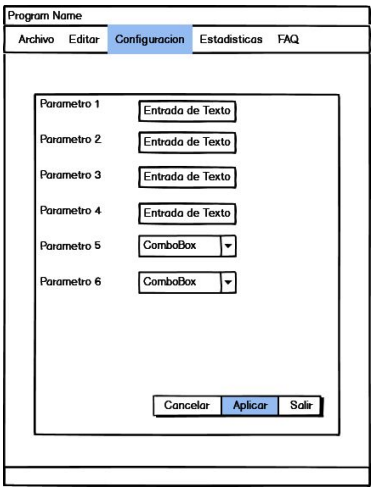
La interfaz del programa propuesta está conformada por las siguientes ventanas.

Todas las ventanas comparten la misma barra de menú.

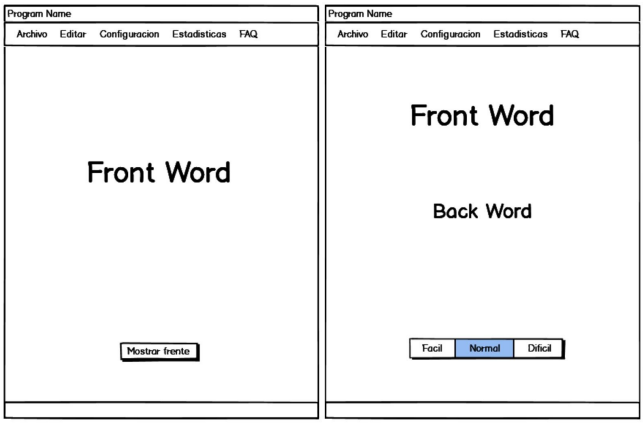
1. Una ventana de inicio, que presenta las listas del usuario.



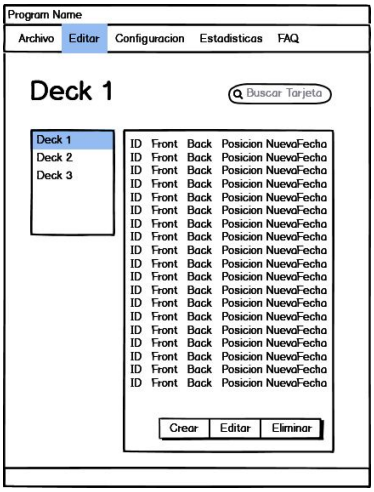
2. Una ventana de configuración para definir parámetros generales del programa.



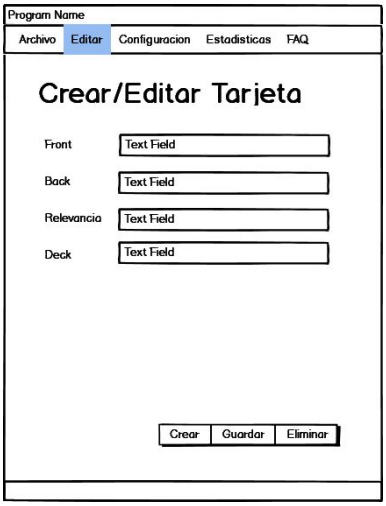
3. Una ventana de revisión o lección periódica donde se muestran las tarjetas una por una. Se muestra la cara de una tarjeta, luego dada una interacción con el usuario se muestra el adverso.



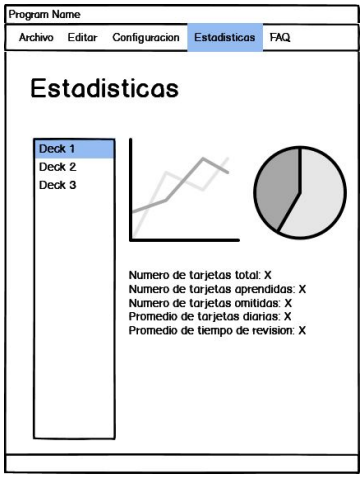
4. Una ventana que muestra todas las palabras pertenecientes a una lista y los atributos de estas como la cara, el anverso, la dificultad, la posición.



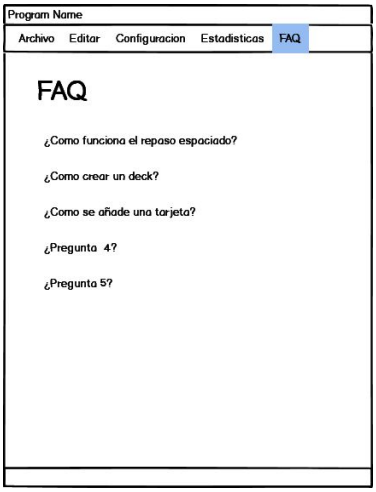
5. Una ventana de edición de tarjetas. Esta ventana es la misma que se usa al añadir o eliminar una tarjeta. Eso es, si la tarjeta ya existe hay un botón de eliminar



6. Una ventana de estadísticas



7. Una ventana de FAQ (Frequently Asked Questions) para introducir al usuario al uso del programa.



presente, si no existe permite escoger la lista a la que se va a añadir la tarjeta.

VI. Entornos de desarrollo y de operación

El entorno de desarrollo IDE es IntelliJ IDEA. El programa corre sobre la máquina virtual de Java. Inicialmente usa un archivo de texto común para el almacenaje de datos, pero se implementará una base de datos para la entrega final.

El entorno de operación planeado es el sistema operativo Windows, especialmente se planea que sea compatible con versiones desde Windows XP hasta Windows 10.

VII. PROTOTIPO DE SOFTWARE INICIAL

El primer prototipo se compone del Deck y las Flashcards mediante listas implementadas por medio de arreglos. El programa contiene los suficientes medios para poder crear Decks, insertar flashcards en estos decks, eliminar flashcards, buscar flashcards, e imprimir flashcards, dado el título del Frente de la flashcard o dado un índice de la lista asociada al deck.

Se usó un archivo txt para cargar un Deck de prueba, con el objetivo de mostrar la inserción de datos.

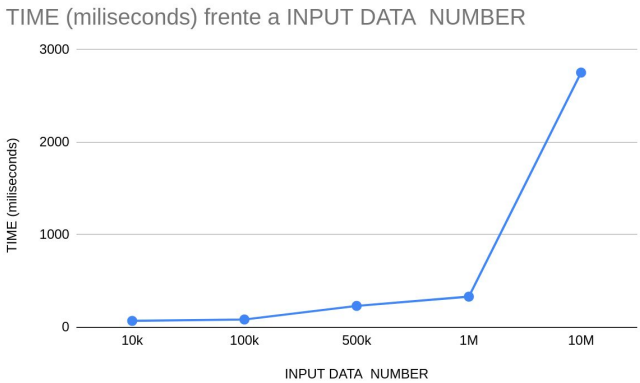
Se implementó una versión básica del algoritmo de repaso espaciado por medio de un método revisión() de la clase Deck, que presenta uno por una las tarjetas pertenecientes a un Deck, y que dada una respuesta por parte del usuario, modifica la fecha en la que el usuario revisará la tarjeta de nuevo.

La rama del primer prototipo se encuentra en https://github.com/KiokuDS/Kioku/tree/FirstPrototype

VIII. PRUEBAS DEL PROTOTIPO

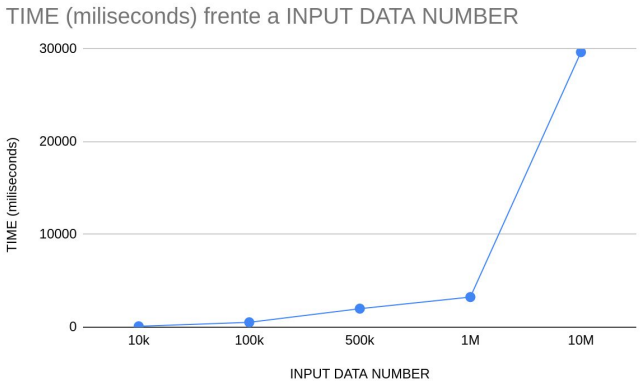
Se realizaron pruebas para las funcionalidades de añadir tarjeta a deck, e imprimir la lista de tarjetas pertenecientes a un deck. Con un número N de tarjetas de 10.000, 100.000, 500,000, 1,000,000 y 10,000,000.

Funcionalidad Añadir Tarjeta



INPUT DATA NUMBER	TIME (milliseconds)
10k	70
100k	85
500k	232
1M	332
10M	2754

Funcionalidad Imprimir Deck



INPUT DATA NUMBER	TIME (milliseconds)
10k	95
100k	526
500k	1999
1M	3251
10M	29656

XIX. Análisis de Comparativo

Como se usó el la función de insertar que siempre lo hace en la cola, solamente se toma el y se recorre de forma ascendente asignando un valor a cada
espacio del array, por lo tanto se tiene una complejidad de O(n)
La función que imprime la lista del Deck sencillamente recorre el array de la lista de forma ascendente desde 0 hasta n-1, imprimiendo los n elementos de esta, y su complejidad es O(n).
El método de eliminación delete(Item, Position) sin embargo requiere que para cada valor borrado sea incluso necesario mover los n-1 valores restantes, si se quieren
eliminar los n datos del array entonces se requiere acceder a los n datos y para cada dato mover n-1 elementos, por lo que se tiene una complejidad

de O(n^2). Este método no fue mostrado en las tablas pues después de de 50 minutos no se terminaron de eliminar los datos de la lista con 1 millón de
Flashcards y se predijo que el método iba a tomar tiempos muy altos para hacer lo mismo con la tarjeta de 10 millones, tiempos que hacen que el método
de eliminar ítems de la lista actualmente implementada no sea práctica.

X. ROLES Y ACTIVIDADES

Defina para cada integrante, el rol asignado y las actividades realizadas en la actual entrega.

INTEGRANTE	ROLES	ACTIVIDADES REALIZADAS
Camilo García	- Líder - Coordinador - Experto - Técnico	- Desarrollo del prototipo.. - Parte del trabajo escrito. - Cuadrar horas de reunión. - Definir las ideas principales del proyecto y cómo se iba a desarrollar.
Nikolái Guzmán	- Investigador - Observador - Secretario - Animador	- Algunas pruebas sobre el prototipo desarrollado por Camilo - Parte del trabajo escrito. - Preparación de la presentación. - Algunas ideas y opiniones sobre las ideas principales del proyecto.

XI. DIFICULTADES Y LECCIONES APRENDIDAS

Este trabajo fue bastante difícil poner en marcha, ya que al ser completamente virtual, es difícil ponerse de acuerdo con la otra persona, y más si nunca se han visto en persona. Pero poco a poco se va cogiendo experiencia y esperamos que para la próxima entrega podamos organizarnos mejor y dar mejores resultados como grupo de 2.