Tienda Virtual: Hermes

Jesús Andrés Rojas Montenegro, Santiago Díaz González.
No. grupo del curso: {*5*} No. de Equipo de Trabajo: {*9*}

I. INTRODUCCIÓN

En el presente documento, se realizará un informe de los avances en el proyecto propuesto para la clase de estructuras de datos, así como la descripción del segundo prototipo del software que se ha estado desarrollado a lo largo del semestre.

El documento incluirá la descripción del problema a resolver, la caracterización de los usuarios del producto, requerimientos funcionales del software, los avances realizados en la implementación de la interfaz de usuario, descripción de los entornos de desarrollo y operación, descripción del prototipo de software, pruebas del prototipo, dificultades y lecciones aprendidas y un link al repositorio del proyecto en Github.

I. DESCRIPCIÓN DEL PROBLEMA

El objetivo es aplicar los conocimientos adquiridos en clase a lo largo del semestre para crear una tienda virtual que sea accesible e intuitiva para los usuarios, y que sea eficiente tanto en tiempo de ejecución como en uso de memoria al hacer las funcionalidades básicas de una tienda virtual, tales como la búsqueda, creación y eliminación de productos. Además se requiere el poder añadir dichos productos a un carrito, realizar la compra de éstos y tener una cuenta de usuario ya sea como comprador o administrador.

II. USUARIOS DEL PRODUCTO DE SOFTWARE

Durante el proceso de desarrollo del software, se ha decidido tener cuatro tipos de usuario de acuerdo a ciertos privilegios de acceso a la hora de manejar productos. Dichos usuarios y sus privilegios serán descritos a continuación:

1. Visitante: Es un usuario que puede acceder a la página principal de la tienda, ver la página de cada producto, realizar búsquedas, utilizar filtros, ordenar productos, añadirlos a un carrito y puede entrar a la página de Sign Up para registrarse o Login para acceder a una cuenta creada con anterioridad. Sin embargo, no podrá realizar compras ni crear o eliminar productos, además no tendrá sugerencias relacionadas a sus preferencias en la página de inicio y tampoco podrá ver estadísticas de la tienda.
2. Cliente: Puede realizar las mismas funciones que un visitante, exceptuando las pantallas de Login y Sign Up (pues ya es un usuario registrado), Además podrá realizar compras, aparecerán sugerencias en su página de inicio basándose en sus preferencias, hacer sugerencias para nuevos productos por medio de un correo electrónico interno, podrá acceder a una ventana de editar perfil y podrán dar a un botón de logout que cerrará su sesión.
3. Empleado: Éste tipo de usuario es un funcionario de la tienda, por lo que podrá modificar la página de inicio de la tienda de forma arbitraria, podrá buscar y ordenar productos, además de acceder a funciones como la creación o eliminación de éstos, además podrá usar un correo electrónico interno para ver los mensajes de sugerencia de los clientes. Sin embargo, no podrá realizar compras ni no tendrá un carrito. Tampoco tendrá una página para editar perfil.
4. Administrador: Tiene los mismos privilegios del empleado, además, podrá tener acceso a estadísticas de ventas de la tienda e información financiera.

III. REQUERIMIENTOS FUNCIONALES DEL SOFTWARE

A. Manejo de productos:

Descripción: Los productos están pensados como objetos con varios parámetros básicos, como lo son: ID, Nombre, precio, tags, fotos, reseña del producto, etc. Por lo cual se necesita una **Base de datos de Productos (Almacén)** donde se puedan eliminar, añadir y editar. Además, que en este Almacén no todo el mundo tenga acceso.

Acciones iniciadoras, comportamiento esperado y requerimientos funcionales : Un **empleado** está habilitado para manejar el Almacén, es decir, puede crear, añadir, remover y editar productos. El sistema debería rectificar con proveedores en la vida real si los cambios hechos son válidos o no. Por consiguiente, requiere una constante cantidad de actualizaciones y consultas de los datos.
Así, esto requiere que el sistema tenga una sólida comunicación con los datos de la vida real , previniendo errores humanos, y un eficiente manejo de productos.

Para esta entrega se ha implementado un **Árbol AVL** para el manejo de los productos, el cual permite todo lo anterior dicho (menos validar con datos de la vida real).

B. Uso del carrito de compra:

Descripción: Cada **Cliente**, al tener una cuenta, tiene un **Carrito**, el que sirve para apartar los productos que sea posible que compre. Además, este hace que sea fácil la decisión final de qué comprar y qué no comprar.

Acciones iniciadoras, comportamiento esperado y requerimientos funcionales : El Cliente necesita entrar al navegador de productos, donde hay únicamente productos válidos, es decir que están revisados por el empleado. A lo que, el cliente escoge un producto para añadirlo a su carrito. Va a la página de carrito y escoge si comprar únicamente ese producto, eliminarlo porque ya no lo va a comprar o seguir buscando más productos.
Así, que se requiere una página especial de Carrito, una validación de productos y una muy eficiente consulta y ordenamiento de productos para que el Cliente navegue con facilidad.
Para esta entrega se ha implementado un **Arreglo Dinámico** para cada objeto cliente, permitiendo toda la lógica y manejo de datos requeridos.

C. Ordenar los productos:

Descripción: Teniendo en cuenta los parámetros básicos de un producto, podemos ver que estos pueden ser ordenados con respecto a cualquiera de dichos parámetros. Este orden puede variar en función de la necesidad del cliente o el empleado de la tienda. Por lo que ambos tiene acceso a ésta función.

Acciones iniciadoras, comportamiento esperado y requerimientos funcionales : En el caso de un cliente, puede elegir cómo quiere ver los productos ordenados en su cuenta, para un empleado, el orden influirá en la página de inicio de la tienda, afectando así a otros usuarios. En ambos casos, se debe elegir algún parámetro (id, precio, nombre, etc.), luego debe escoger si quiere que se muestran los productos en un orden de mayor a menor o menor a mayor respecto a dicho parámetro. El sistema debe comparar los parámetros de los productos entre sí para poder ordenarlos correctamente. Por lo tanto, se necesita una estructura de datos que permita comparar y organizar de forma eficiente.
Aún no se ha implementado, ya que se requiere tener una interfaz gráfica bien definida. Hemos pensado en implementar próximamente con las estructura de Disjoint Sets para la organización por categorías .

D. Busqueda de producto:

Descripción: Para la búsqueda de un producto, se siguen teniendo en cuenta los parámetros básicos de éste objeto. Un cliente puede buscar un producto para añadirlo a su carrito y comprarlo posteriormente, mientras que un empleado puede buscarlo para revisar el stock, actualizar algún parámetro o eliminar el producto, o simplemente cualquier visitante puede realizar una búsqueda para ver u producto.

Acciones iniciadoras, comportamiento esperado y requerimientos funcionales : Tanto un administrador como un cliente, ingresarán información en una barra de búsqueda. Ésta información será comparada con los parámetros de los productos para encontrar coincidencias entre éstos y la información ingresada en la barra de búsquedas. Si hay productos con parámetros que coinciden con la información dada, se mostrarán éstos productos en orden de mayor coincidencia a menor coincidencia, si no hay coincidencias, aparecerá un mensaje diciendo “No se ha encontrado el producto”.
Esto significa que se requiere un sistema de comparación eficiente entre los datos introducidos en la barra y los parámetros de los productos.
Como se dijo antes, se implementó un Árbol AVL de productos, en el cual se puede buscar por el nombre del producto o el id. Sin embargo, no es la búsqueda por coincidencias que se quiere.

E. Reservar/Comprar producto:

Descripción: Establecimiento final y crucial del trato con el cliente para hacer la transacción y en donde se actualizará el Almacén para efectuar la entrega.

Acciones iniciadoras, comportamiento esperado y requerimientos funcionales : El cliente va al carrito de compra y selecciona un botón de efectuar compra y se apartaran los artículos escogidos y se procederá a la compra. . Si la transacción no es posible se notificará a un empleado, pero si el error es del cliente (malos datos, no tiene dinero, no le alcanza, etc.) el sistema le avisará.
Así pues, se requiere acceder a un sistema de dinero externo a la página (que va usar el cliente) y uno interno que controle las cuentas de la tienda. Además de un proceso de generación de facturas que valide los datos (tanto de dinero como de productos).
Se implementó un sistema básico de reserva y compra de productos, en el que se disminuye la cantidad de un producto y se efectúa una compra.

F. Registro y fácil manejo de la cuenta para los usuarios:

Descripción: Los usuarios, tanto clientes como empleados, necesitan que su información personal esté segura, además de poder actualizar el estado de su cuenta de manera eficiente. Por lo tanto se requiere un sistema sencillo pero seguro para que puedan manejar dicha información.

Acciones iniciadoras, comportamiento esperado y requerimientos funcionales : Para los usuarios administrativos, solo se permitirá crear una cuenta en un computador con permisos especiales, puede ser por medio de una contraseña. Además, deberá haber más personal de la empresa que verifique su identidad. En cuanto los empleados, solo los puede crear un Administrador con una contraseña especial.
En cuanto a los clientes, se les pedirán sus datos personales, y se les enviará un correo de verificación cada vez que alguien intente loguearse a su cuenta. Además, la primera vez que se haga una compra, se hará una verificación con la entidad bancaria correspondiente.
Después cada usuario podrá editar su perfil, con normalidad, ingresando al panel de su cuenta.
Aún no se ha implementado por la falta de una interfaz gráfica establecida, pero ya se creó la lógica para hacer Objetos de tipo usuario con sus parámetros, darles una id única y sus Getters y Setters.
Estos usuarios, están guardados en una **Base de Datos de Usuarios** implementada con un **Árbol AVL,** que permite añadir, eliminar y actualizar usuarios y buscar un usuario por nombre y contraseña (para el LogIn) o por id.
.

G. Mostrar a los clientes productos seleccionados

Descripción: Los clientes al loguearse, en su página de inicio, se le mostrarán productos recomendados basados en sus compras o en publicidad.

Acciones iniciadoras, comportamiento esperado y requerimientos funcionales : El cliente se loguea y ve productos que probablemente le gustarán y si es publicidad estará marcado el producto para dejar claro que es publicidad.
Entonces, el software necesitará un algoritmo de sugerencias basado en los gustos del cliente o en compras anteriores y que pueda ser modificado para poner productos que se quieran (publicidad), el cual debe ser muy eficiente ya que se ejecutará cuando el usuario entre.
Aún no implementado, quizá se pueda en un futuro con Disjoint Sets.

H. Los clientes pueden dar a la tienda sugerencias para nuevos productos:

Descripción: La afinidad con los clientes es una prioridad en cada tienda, por lo tanto, los clientes pueden enviar sugerencias a la tienda para que oferten nuevos productos.

Acciones iniciadoras, comportamiento esperado y requerimientos funcionales : Desde una cuenta creada en la página de la tienda, los clientes tienen la opción de enviar una sugerencia, la cual es un e-mail que va dirigido al correo electrónico de la tienda, al cual tienen acceso las personas que trabajen en soporte. Esto significa que la tienda requiere de un sistema de correo electrónico, en el que cada cuenta de cliente puede escribir a un único correo que está siendo administrado por el personal.
Aún no implementado a falta de una interfaz gráfica sólida.

I. Mostrar datos estratégicos a los dueños de la tienda

Descripción: Los empleados y dueños podrán tener consultas para ver estadísticas de los distintos productos.
Acciones iniciadoras, comportamiento esperado y requerimientos funcionales : Los Dueños tendrán una página especial donde acceder a las estadísticas de los productos. Por lo que se necesita un algoritmo de Big Data para esto.
Aún no implementado a falta de una interfaz gráfica sólida, pero se avanzó, al ya estar implementada la base de datos de productos.

IV. AVANCE EN LA IMPLEMENTACIÓN DE LA INTERFAZ DE USUARIO

1. DESCRIPCIÓN DE LA INTERFAZ DE USUARIO:

Página de inicio:

La página de inicio consta de una barra de búsqueda, los botones de home, login y sign out en la parte superior, tiene un banner amarillo en el que se anunciarán promociones y eventos de la tienda. En el costado izquierdo tiene una lista con las categoría de los productos. En la parte central tiene una cuadrícula con los productos más populares en la tienda. Por último, tiene botones de “página anterior” y “página siguiente” en la parte inferior.

Página de inicio de sesión:

Al presionar el botón “Log In”, aparecerá el botón “Sign In” y se reducirá el tamaño de la barra de búsqueda, para que aparezcan los campos para nombre de usuario y contraseña.

Página de registro/crear usuario:

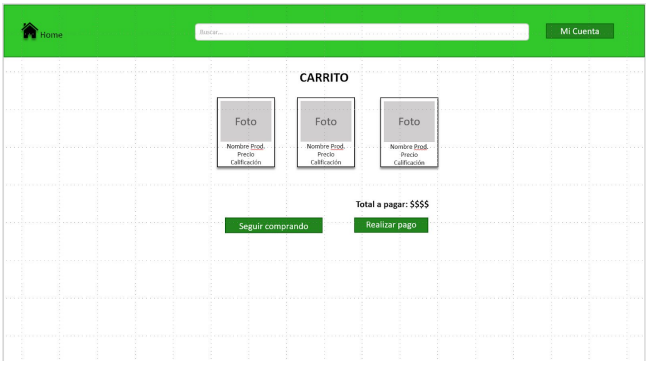
Al presionar el botón de "sign up", el usuario será redirigido a una página en la que se le pedirán los datos personales tales como: Nombre, apellidos, correo electrónico, fecha de nacimiento, un nombre de usuario y una contraseña. En la esquina inferior derecha aparecerá un botón para confirmar y completar su registro como un n

```
root
    README.md
    src                 // Carpeta donde se almacena el código fuente
    └── my.packages     // Paquete principal del proyecto (kernel)
        ├── Main.(*)  // Clase principal que ejecuta la aplicación. Ejemplo. Java: clase Main.java
        ├── ...         // Otros archivos java con el código fuente de la app
    docs
    └── Entrega_(Plantilla).pdf     //Documento plantilla con la información de la entrega requerida
    └── ......
        OtrosDocumentos_X.pdf      //Otros documento según se requiera
    data                //Carpeta con los datos de prueba del proyecto
    dist                //Carpeta en donde se almacenan los archivos que ejecutan la aplicación
    └── proyectoEntrega1.jar
    └── proyectoEntrega2.jar
    └── proyectoEntrega3.jar
    └── ......
    lib                 // Carpeta con las librerías requeridas en la aplicación
    Otros archivos...   // Otros archivos de ejecución y configuración necesarios.
                        // Ejemplo: project.xml, build.xml (ant), run.sh, run.bat (ejecución), etc.
```

uevo usuario de la tienda.

Página Carrito:

En la página del carrito se muestra una lista con los productos que han sido añadidos, también se muestra el valor total de los productos, un botón para realizar la compra y un botón de "seguir comprando" que redirigirá al usuario a la página de inicio de la tienda.

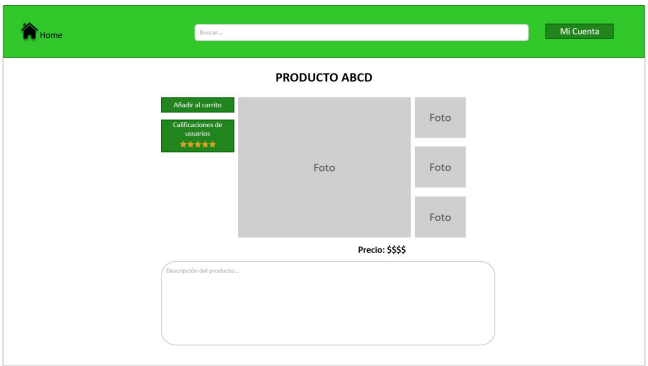


Página de crear producto:

Ésta pantalla solo será accesible para el empleado y el administrador, en ella se pedirán datos como: nombre del producto, precio, descripción y por lo menos una foto. Además tendrá un botón de confirmación en la esquina inferior derecha.

Página de producto:

En ésta pantalla se muestran el botón home, una barra de búsqueda y un botón de "mi cuenta" (o login y sign up en caso de no estar logueado) en la parte superior. Debajo está el nombre del producto, un botón de añadir al carrito, un espacio con la calificación de los usuarios y una galería de fotos del producto. En la parte inferior están el precio y un cuadro con la descripción del producto.

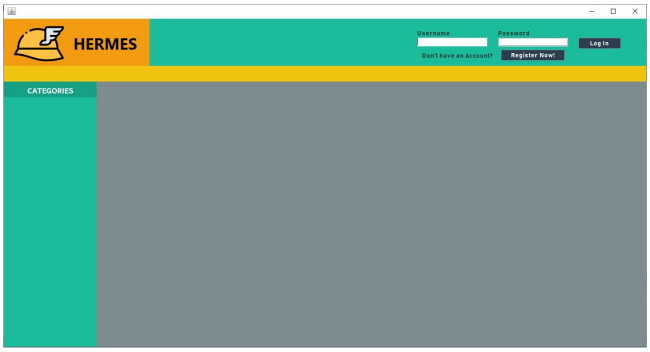


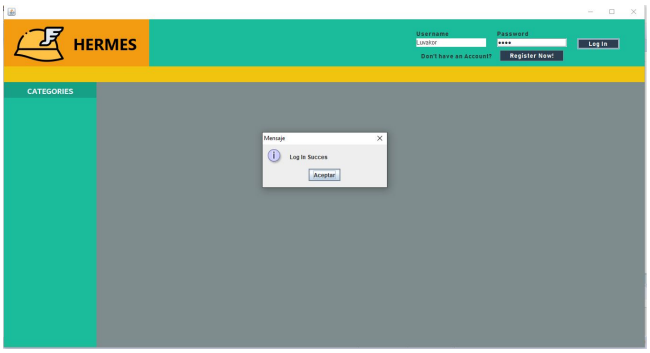
2. AVANCES EN LA IMPLEMENTACIÓN DE LA INTERFAZ DE USUARIO

La interfaz de usuario está siendo desarrollada en Swing, biblioteca para crear GUIs, siendo así se ha avanzado en la creación de:

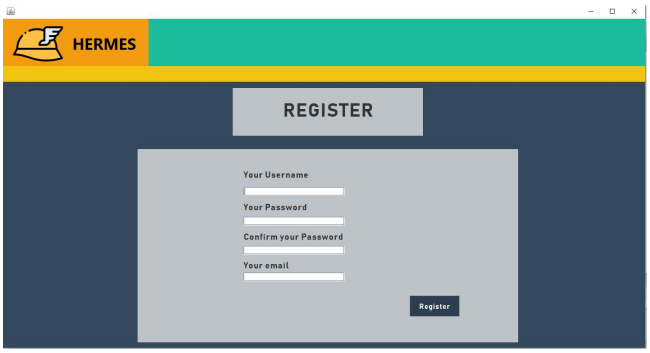
Diseño de la cabecera de la aplicación:

Ventana de Home con validación de Login funcional:





Diseño de la ventana de Registro:



V. ENTORNOS DE DESARROLLO Y DE OPERACIÓN

El Software será desarrollado en Java con Eclipse IDE. El código será subido a un repositorio en BitBucket, donde será actualizado periódicamente.

Para la interfaz gráfica se hará uso de Swing una biblioteca de interfaces gráficas de usuario (GUI) para Java que usa WindowBuilder, viene incluida con el entorno de desarrollo de Java (JDK).

Se pretende que, para la entrega final, se ejecute en un computador con el ejecutable de la aplicación generado por Java.

VI. DESCRIPCIÓN DEL PROTOTIPO DE SOFTWARE

Link del repositorio:
https://Luvakor@bitbucket.org/Luvakor/hermes.git
Hasta la fecha, no se pudo implementar la Pila para hacer y deshacer cambios o navegar por la interfaz.

VII. PRUEBAS DEL PROTOTIPO

Para estas pruebas, se presentará primero cuánto tiempo tomó "subir" los datos, para lo cual se hizo un documento de texto con 10.000 filas, que representan un Producto o Cliente. Así, para el caso de más datos, se hizo un ciclo, por lo que hay que tener en cuenta que hay datos duplicados.
Además, es de aclarar que, quizá la forma de pasar los datos al prototipo no sea la más óptima y por ello no se pueda probar con más datos.

Funcionalidad: "Subir" los productos.

Cantidad de datos probados	Tiempos de ejecución (s)
10000	0.288704
100000	0.941032
1000000	8.695898
10000000	193.8321

Funcionalidad: "Subir" los productos.

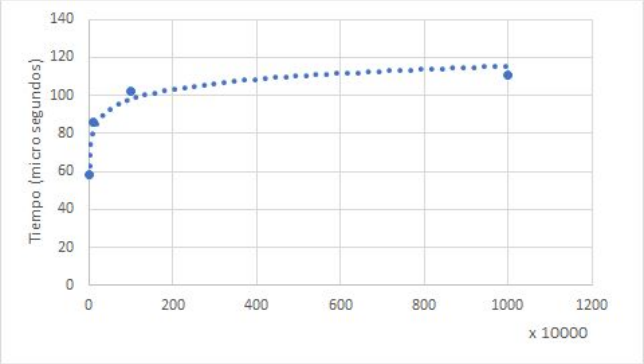
Cantidad de datos probados	Tiempos de ejecución (s)
10000	0.145852
100000	0.731281
1000000	6.82006
10000000	*

*Error de memoria
Visto lo anterior , se van a nombrar las funcionalidades probadas:
Funcionalidad: Buscar un producto.
Tipo de estructura de datos: Árbol AVL

Análisis realizado (Notación Big O): $O(\log_2 n)$

Cantidad de datos probados	Tiempos de ejecución (µs)
10000	58
100000	86
1000000	102
10000000	111

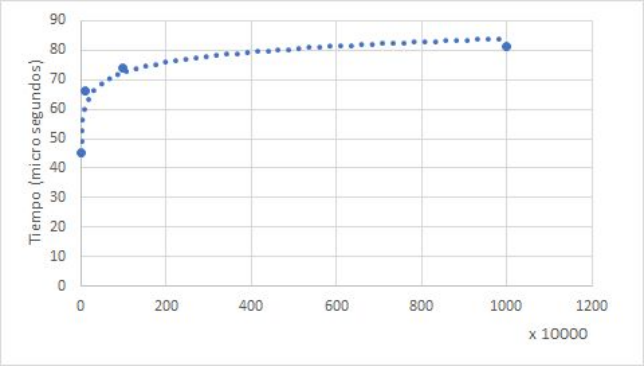
Gráfico:



Funcionalidad: Editar un producto (eliminación-inserción).
Tipo de estructura de datos: Árbol AVL
Análisis realizado (Notación Big O): $O(\log_2 n)$

Cantidad de datos probados	Tiempos de ejecución (µs)
10000	45
100000	66
1000000	74
10000000	81

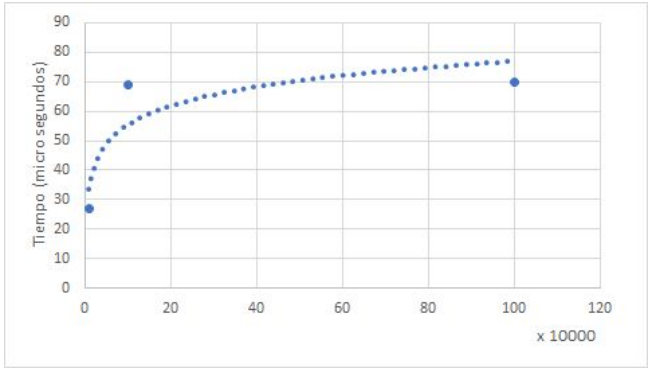
Gráfico:



Funcionalidad: Buscar un usuario.
Tipo de estructura de datos: Árbol AVL
Análisis realizado (Notación Big O): $O(\log_2 n)$

Cantidad de datos probados	Tiempos de ejecución (µs)
10000	27
100000	69
1000000	70
10000000	*

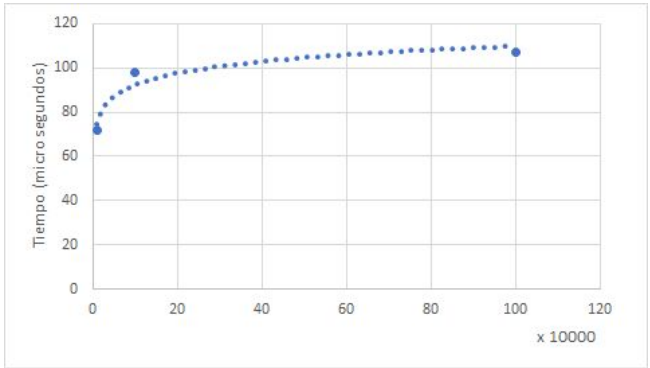
*Error de memoria
Gráfico:



Funcionalidad: Editar un usuario (eliminación-inserción)
Tipo de estructura de datos: Árbol AVL
Análisis realizado (Notación Big O): $O(\log_2 n)$

Cantidad de datos probados	Tiempos de ejecución (µs)
10000	72
100000	98
1000000	107
10000000	*

*Error de memoria
Gráfico:



VIII. DIFICULTADES Y LECCIONES APRENDIDAS

Uno de los problemas con los que nos encontramos fue con la implementación de las interfaz gráficas, ya que en un principio se quería desarrollar en un entorno web, pero al ser un tema tan amplio y en especial, al estar toda la parte del manejo de bases de datos ya implementada con SQL y parecidos, era difícil usar nuestras propias estructuras de datos.

En cambio, en Java ya teníamos implementado todas nuestras estructuras, sin embargo la implementación de interfaces gráficas no es que fuera más sencilla.
Además en las pruebas fue evidente que se demostró lo teorizado por el análisis de complejidad, aunque nos vimos limitados por algunos problemas de memoria.
Aprendimos que hay mucha diferencia en la eficiencia de algoritmos y estructuras con complejidades lineales y logarítmicas, en especial en la búsqueda.