SISTEMA DE RESERVAS DE TIQUETES AÉREOS

Gabriel Lozano, Andrés Ortega, Zamir Velásquez

No. de Equipo Trabajo: {*Número de Equipo de trabajo*}

I. INTRODUCCIÓN

En un aeropuerto convergen un sinfín de datos como lo son los tiempos de salida de los distintos vuelos, las distintos aviones y sus respectivas capacidades, además de la afluencia de pasajeros que toman distintos vuelos.

Para la implementación de este sistema de reserva de tiquetes aéreos se realiza este documento en donde se describen las características, requerimientos, estructuras de datos usadas y demás aspectos concernientes a la realización del mismo.

II. DESCRIPCIÓN DEL PROBLEMA A RESOLVER

El problema por tratar en este proyecto se centra en el desarrollo de un software que permita la gestión de reservas de tiquetes aéreos, lo cual conlleva al manejo apropiado de los distintos datos provistos para la correcta operación de un aeropuerto. Esto con el objetivo de brindar una herramienta amigable con el usuario y a la vez estructurada para su fácil mantenimiento y actualización.

III. USUARIOS DEL PRODUCTO DE SOFTWARE

El software tendrá tres clases de usuarios una vez implementado, los cuales son:

- Usuario final: Este podrá reservar y cancelar vuelos dando una ruta específica.
- Aerolíneas: Estas podrán agregar vuelos y modificar las horas de salida, llegada, origen y destino de estos.
- Desarrollador: Este tiene acceso total al código del software y permiso tanto de lectura como de edición sobre el mismo.

IV. REQUERIMIENTOS FUNCIONALES DEL SOFTWARE

- *Añadir Ruta*
    - *Descripción:* La aerolínea ofrece un nuevo vuelo de un aeropuerto a otro, especificando el modelo del avión, cantidad de pasajeros y hora de salida.
    - *Acciones iniciadoras y comportamiento esperado*: En este caso es la aerolínea quien realiza esta acción, la cual consta en añadir un nuevo elemento a la lista ordenada **Ruta**.
- *Eliminar Ruta*
    - *Descripción:* La aerolínea cancela un vuelo previamente ofertado en el aeropuerto en cuestión
    - *Acciones iniciadoras y comportamiento esperado*: En este caso es la aerolínea quien realiza esta acción, la cual consta en eliminar un elemento existente de la lista ordenada **Ruta**.
- *Añadir Pasajero a Ruta*
    - *Descripción:* El pasajero reserva un vuelo entre dos aeropuertos especificados, teniendo como datos conocidos el tiempo de salida, el aeropuerto de origen y de llegada.
    - *Acciones iniciadoras y comportamiento esperado*: Al reservar un vuelo, el pasajero es añadido a la ruta cubierta entre los dos aeropuertos especificados. Esto implica añadir a un pasajero al objeto
- *Eliminar Pasajero de Ruta*
    - *Descripción:* El pasajero cancela una reservación previamente hecha.
    - *Acciones iniciadoras y comportamiento esperado:* Al cancelar una reservación de vuelo, el pasajero es eliminado de la ruta cubierta entre los dos aeropuertos especificados. Esto implica eliminar a un pasajero del objeto
- *Sugerir Ruta a Pasajero con una ya Existente*
    - *Descripción:* El pasajero ingresa el aeropuerto de partida y de llegada en el software, para luego obtener el mejor vuelo para cubrir esa ruta.
    - *Acciones iniciadoras y comportamiento esperado*: Al pedir una sugerencia de vuelo el pasajero ingresa el aeropuerto de origen y de destino. Esto implica buscar el camino óptimo entre los objetos **Aeropuerto** requeridos por el usuario.

V. DESCRIPCIÓN DE LA INTERFAZ DE USUARIO PRELIMINAR

La interfaz preliminar planteada para este proyecto contará, en primer lugar, con dos menús desplegables en donde se podrán elegir las ciudades de origen y destino. En segundo lugar, se muestra una lista de vuelos sugeridos que cubren dicha ruta y se tendrá la opción de reservar el vuelo en cuestión. Por último, habrá una interfaz que permita ver el estado del vuelo reservado, sus características y una opción para cancelar la reserva.

VI. ENTORNOS DE DESARROLLO Y DE OPERACIÓN

El software se desarrollará principalmente en el entorno de desarrollo NetBeans haciendo uso de Java, además de este se implementará una base de datos en Oracle Database haciendo uso de SQL.

VII. PROTOTIPO DE SOFTWARE INICIAL

Luego de la creación y prueba de este prototipo, se subió el código correspondiente junto a la documentación requerida al siguiente repositorio en GitHub:

https://github.com/adol2299/DS-2019-II-Final-Proyect-

En este repositorio, las distintas carpetas y documentación de apoyo se encuentran organizadas de la siguiente manera:

```
root
|   README.txt
|--src
    |--BusinessLogic
       |--Aeropuerto.java
       |--Avion.java
       |--Controlador.java
       |--Lector.java
       |--Main.java
       |--Ruta.java
       |--Tiempo.java
    |--Build
       |--classes                          //Clases usadas en el software
    |--estructuras_de_datos_propias
       |--ListaOrdenada.java
       |--Main.java
|--docs
    |--ED_definicion_del_problema_final.pdf
|--data
    |--Data_generada_2.txt                 //Datos usados en las pruebas
    |--data_test.txt                       //Datos usados en el main del software
|--dist
|--lib
    |--librerias.txt
```

De esta manera, se pueden encontrar los ejecutables de la aplicación junto con sus clases necesarias, los documentos con los datos de prueba utilizados para el benchmarking de este prototipo, el documento de soporte para el proyecto y las librerías usadas en el mismo.

VIII. PRUEBAS DEL PROTOTIPO

Para las pruebas del prototipo se escogieron tres funcionalidades que son las de mayor consumo computacional en cuestión de tiempo, siendo estas:

- Añadir un elemento a la lista
- Eliminar un elemento de la lista
- Buscar un elemento en la lista

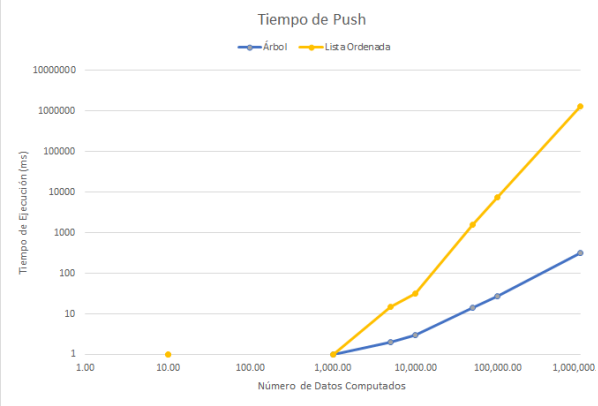
Como estructura de referencia para el benchmarking se hizo uso de árboles, siendo estos los tiempos de cómputo en milisegundos usando ambas estructuras de datos con un número n de datos a computar:

	Árboles de Búsqueda			Lista Ordenada		
n-elementos	push	encontrar	eliminar	push	encontrar	eliminar
10.00	0	0	0	1	0	0
100.00	0	0	0	0	0	0
1,000.00	1	1	0	1	2	1
5,000.00	2	1	2	15	16	4
10,000.00	3	3	2	32	71	11
50,000.00	14	10	10	1581	3836	479
100,000.00	27	24	23	7582	17299	2541
1,000,000.00	324	284	205	1265289	511771	153019
10,000,000.00	6893	6395	5961			
100,000,000.00	132634	110740	114756			

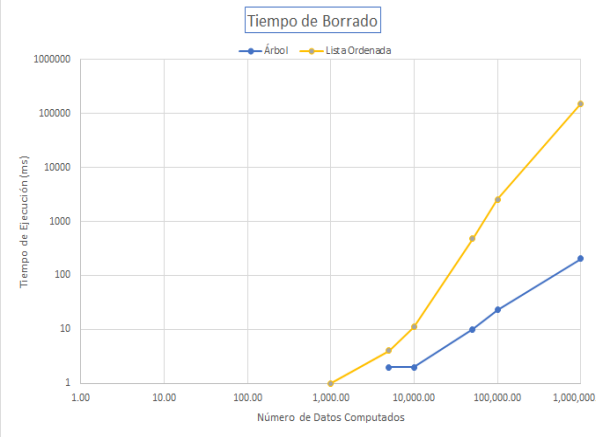
Como se puede apreciar, los tiempos de cómputo para 10 y 100 millones de datos concernientes a la lista ordenada no están, esto debido a lo mucho que se tardaba este proceso en terminar y de la gran demanda de memoria de este.

Con los datos provistos por estas tablas podemos hacer las siguientes tablas comparativas:

- Insertar Datos



- Eliminar Datos



- Buscar Datos