Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y Sistemas
Estructuras de Datos
Sección A
Catedrático: Rene Ornelis
Auxiliar: Daniel Monterroso

PRÁCTICA

SISTEMA DE GESTIÓN DE AEROPUERTO

Objetivos

General

Aplicar los conocimientos del curso de Estructuras de Datos en la creación de soluciones de software.

Específicos

- Hacer un uso correcto de la memoria dinámica y apuntadores en el lenguaje de programación C++.
- Aplicar los conocimientos adquiridos sobre estructuras de datos lineales.
- Utilizar la herramienta graphviz para la generación de reportes.

Descripción general

Las aplicaciones de escritorio, tanto de interfaz gráfica como de consola, son muy utilizadas cuando no se tiene un sitio web o una conexión a internet para otras alternativas, lo cual tener una aplicación instalada en el sistema operativo facilita la consulta y visualización de información en cualquier momento sin la conexión a internet.

En este proyecto se le solicita al estudiante del curso de Estructuras de Datos desarrollar un sistema de gestión para un aeropuerto que permita gestionar vuelos, pasajeros, equipajes y algunas otras características más. Este sistema utilizará

diversas estructuras de datos como listas enlazadas, listas circulares, pilas y colas para manejar las diferentes operaciones y procesos del aeropuerto.

Implementación

Un aeropuerto busca realizar la simulación de la llegada de aviones y de los pasajeros, cada uno de estos con sus respectivas características y atributos.

Aviones

Al inicio de la simulación se establecerá el listado de aviones a partir de un archivo de entrada el cual tendrá toda la información necesaria de cada avión. Estos se estarán almacenando en 2 **listas circulares dobles**; el primer listado corresponderá a los aviones con estado “Disponible”, y el otro listado será para los aviones con estado “Mantenimiento”. Posteriormente, a través del menú (ver Imagen 4), los aviones pueden ser movidos de ambas listas por medio de su estado.


```
[
    {
        "vuelo": "A100",
        "numero_de_registro": "N12345",
        "modelo": "Boeing 737",
        "fabricante": "Boeing",
        "ano_fabricacion": 2015,
        "capacidad": 180,
        "peso_max_despegue": 79000,
        "aerolinea": "AirlineX",
        "estado": "Disponible"
    },
    {
        "vuelo": "A102",
        "numero_de_registro": "A54321",
        "modelo": "Airbus A320",
        "fabricante": "Airbus",
        "ano_fabricacion": 2018,
        "capacidad": 150,
        "peso_max_despegue": 77000,
        "aerolinea": "AirlineY",
        "estado": "Disponible"
    }
]
```


Imagen 1: Entrada en formato json para los aviones

Pasajeros

Los pasajeros se registrarán a partir de un archivo de entrada el cual tendrá toda la información necesaria de cada pasajero, este simulará la ventanilla de registro que luego se almacenará en una **cola** de llegada.

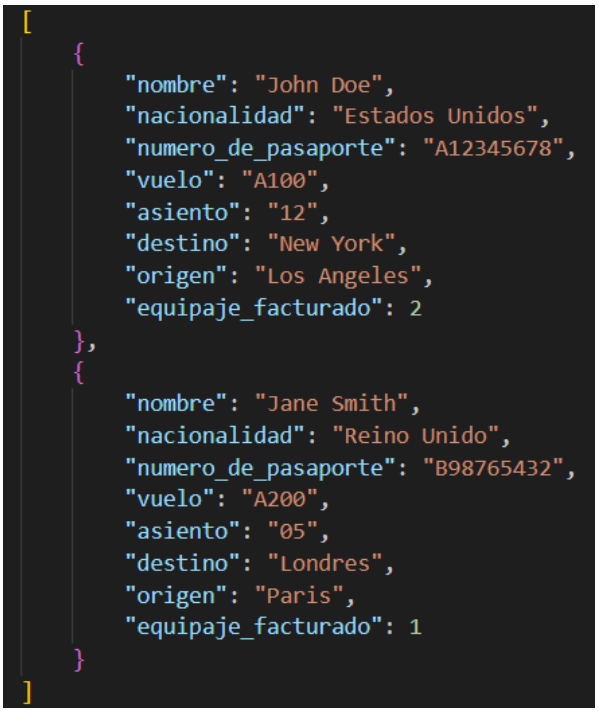

Imagen 2: Entrada en formato json para los pasajeros

Equipaje

Esta información se extraerá de los pasajeros que hayan pasado la ventanilla de registro. Se creará una **pila** que almacene la cantidad de equipaje de cada pasajero según vayan saliendo de la cola de registro. Si el pasajero no tiene equipaje, será únicamente sacado de la cola sin ingresar a la pila.
Nota: estos movimientos se harán a través de un archivo de entrada (ver Imagen 4).

Simultáneamente con la creación de la pila, se generará una **lista enlazada doble** con los pasajeros que hayan salido de la cola de registro, estos serán ordenados por el número de vuelo, en caso que este se repita, se tendrá como segundo criterio de ordenamiento el número de asiento. Los pasajeros almacenados en esta lista podrán ser consultados por el usuario en cualquier momento durante la ejecución del programa y deberá mostrar toda su información correspondiente en la consola.

Flujo de la aplicación

La aplicación se ejecutará por medio de una consola, la cual pedirá la información necesaria para su correcto funcionamiento. La aplicación tendrá un menú donde el usuario podrá navegar y realizar el ingreso de los datos, lectura de archivos, generación de reportes y las diferentes operaciones en las estructuras de datos.

Ejemplos para la interfaz de la consola con las opciones mínimas que debe incluir:

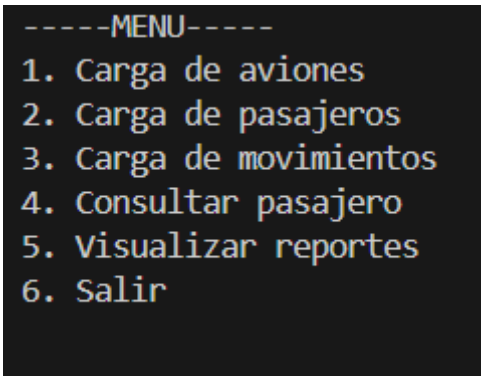

Imagen 3: Menú principal

Nota: Al realizar la consulta del pasajero se solicitará su número de pasaporte.

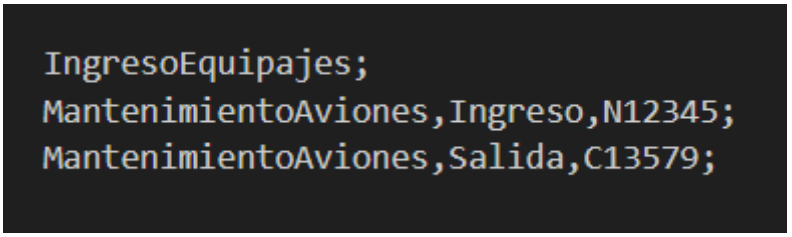

Imagen 4: Carga de movimientos

Nota: El comando *IngresoEquipaje* será para la cola de pasajero, lista doble de pasajeros y pila de equipaje; el comando *MantenimientoAviones*, seguido de su estado y número de registro, será para la lista circular doble de mantenimiento.

Reportes

Por medio del menú en la consola el usuario podrá generar y visualizar los siguientes reportes:

- Lista de aviones disponibles
- Lista de aviones en mantenimiento
- Cola de registro
- Pila de equipaje
- Lista de pasajeros

Manual técnico

Este manual tendrá los requerimientos del sistema operativo para llevar con éxito la ejecución del programa, así como también las dependencias instaladas y utilizadas para su desarrollo.

Restricciones

- El proyecto se realizará individualmente.
- El lenguaje de programación será C++.
- El IDE de desarrollo para la aplicación queda a discreción del estudiante.
- Los reportes deben ser implementados con la herramienta Graphviz.
- Los reportes deben abrirse desde la aplicación. No se permitirá buscar en directorios.
- Las estructuras de datos deben ser implementadas por el estudiante.
- **Las copias serán merecedoras de una nota de 0 puntos y los responsables serán reportados con el catedrático de la sección para su respectiva sanción**.

Entrega de la proyecto

- **La fecha de entrega será el jueves 13 de junio del 2024 antes de las 23:59 horas.**
- La entrega en UEDI será el link de la carpeta de drive que tendrá todos los archivos del proyecto.
- El archivo del manual técnico deberá estar en formato pdf y ser subido al repositorio.
- **El repositorio deberá tener como nombre el número de carnet de estudiante, ejemplo: 200000000_EDD_Practica**