Universidad de San Carlos de Guatemala
Escuela de Ingeniería en Ciencias y Sistemas, Facultad de Ingeniería
Laboratorio de Estructuras de Datos, Vacaciones del 1er. Semestre 2024.

# Manual Tecnico - Practica

**202200075 – Dominic Juan Pablo Ruano Pérez**

**Resumen**

Este manual técnico describe la instalación, configuración y uso del Sistema de Gestión de Aeropuertos desarrollado en C++. El sistema gestiona vuelos, pasajeros y equipajes utilizando diversas estructuras de datos como listas enlazadas, listas circulares, pilas y colas. El manual incluye pasos para instalar C++ en diferentes sistemas operativos, detalla las tecnologías utilizadas (C++, Graphviz, JSON) y enumera los requisitos mínimos del sistema. Proporciona una descripción completa de las estructuras de datos utilizadas, incluyendo listas circulares dobles para aviones, una cola para el registro de pasajeros y una pila para la gestión de equipajes. El menú principal de la aplicación permite a los usuarios cargar archivos de entrada, gestionar aviones, registrar pasajeros, manejar equipajes y generar reportes. Los archivos de entrada para aviones y pasajeros deben estar en formato JSON. Además, el manual incluye espacios reservados para imágenes que ilustran la interfaz de la aplicación y la estructura de los datos. Esta guía asegura una comprensión clara de cómo operar y navegar el sistema de manera efectiva.

**Palabras clave**

1. Estructuras de Datos
2. Listas Enlazadas
3. Listas Circulares
4. Pilas
5. Colas

***Abstract***

*This technical manual outlines the installation, configuration, and usage of the Airport Management System developed in C++. The system manages flights, passengers, and luggage using various data structures such as linked lists, circular lists, stacks, and queues. The manual includes steps for installing C++ on different operating systems, details the technologies used (C++, Graphviz), and lists the minimum system requirements. It provides a comprehensive description of the data structures used, including double circular lists for aircraft, a queue for passenger registration, and a stack for luggage management. The main menu of the application allows users to load input files, manage aircraft, register passengers, handle luggage, and generate reports. The input files for aircraft and passengers must be in JSON format. Additionally, the manual includes placeholders for images that illustrate the application's interface and the structure of the data. This guide ensures a clear understanding of how to operate and navigate the system effectively.*

***Keywords***

1. *Data Structures*
2. Linked Lists
3. Circular Lists
4. Stacks
5. Queues

## Introducción

El Sistema de Gestión de Aeropuertos desarrollado en C++ se presenta como un proyecto universitario fundamental para la formación de un ingeniero en ciencias y sistemas. En un contexto donde la optimización de operaciones aeroportuarias es crucial, este sistema utiliza estructuras de datos avanzadas como listas enlazadas, listas circulares, pilas y colas para manejar grandes volúmenes de información y procesos en tiempo real. La importancia de este proyecto radica en su capacidad para mejorar la organización y rapidez en la gestión de datos críticos, garantizando así un servicio más eficiente y confiable. Basado en teorías de estructuras de datos y algoritmos, este proyecto no solo pone en práctica los conocimientos adquiridos en el curso, sino que también contribuye con herramientas visuales como Graphviz para la generación de reportes. ¿Cómo puede este sistema transformar la administración de aeropuertos? Este manual técnico busca responder a esta interrogante, demostrando los aportes significativos de la solución propuesta y guiando al usuario en su implementación y uso.

### 1.1. Instalación Windows:

1. Descargar e instalar MinGW.
2. Asegurarse de seleccionar la opción "mingw32-gcc-g++" durante la instalación.
3. Agregar la ruta `C:\MinGW\bin` al PATH del sistema.

### 1.2. Linux (Ubuntu):

1. Abrir la terminal.
2. Ejecutar el comando:
    1. “sudo apt-get update”
    2. “sudo apt-get install g++”

### 1.3. macOS:

1. Instalar Xcode desde la App Store.
2. Instalar las herramientas de línea de comandos ejecutando en la terminal:
    1. xcode-select –install

## 2. Tecnologías Empleadas

- *Lenguaje de Programación***:** C++
- *Herramientas de Reportes***:** Graphviz
- *Compilador***:** g++
- *Editor/IDE:* Visual Studio Code

## 3. Requerimientos Mínimos del Sistema

- *Sistema Operativo***:** Windows 7 o superior, Ubuntu 18.04 o superior, macOS Mojave o superior.
- *Memoria RAM***:** 4 GB (mínimo), 8 GB (recomendado).
- *Espacio en Disco***:** 500 MB libres para la instalación de herramientas y compilación del proyecto.
- *Dependencias***:**
    - Graphviz: Para generar reportes visuales de las estructuras de datos.
    - Compilador para C++.

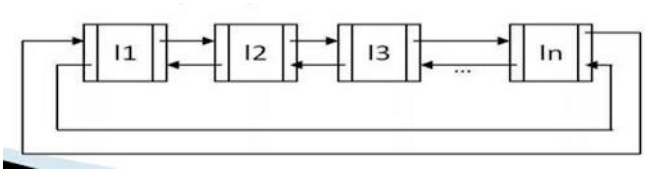
## 4. Descripción de las Estructuras de Datos y Funcionamiento

### 4.1. Aviones

**Listas Circulares Dobles:**

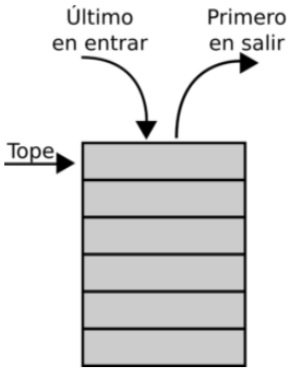
- Se utilizan dos listas circulares dobles para almacenar los aviones: una para los aviones disponibles y otra para los aviones en mantenimiento.
- **Operaciones:**
    - *Inserción:* Agrega un nuevo avión a la lista correspondiente.
    - *Búsqueda*: Encuentra un avión por su identificador.

- *Eliminación*: Remueve un avión de la lista y lo mueve a la otra lista según su estado.



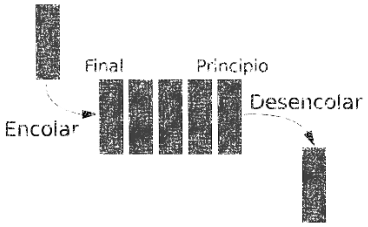

### 4.2. Equipaje

- **Pila de Equipaje:**
  - Los equipajes de los pasajeros se almacenan en una pila según el orden de salida de la cola de registro.
  - **Operaciones:**
    - Push: Agrega un equipaje a la cima de la pila.
    - Pop: Remueve el equipaje de la cima de la pila para su procesamiento.



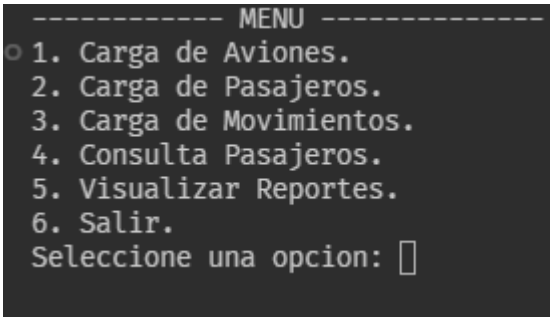

**Cola de Registro:**

- Se utiliza una cola para simular la ventanilla de registro de pasajeros.
- **Operaciones:**
  - Enqueue: Agrega un pasajero al final de la cola.
  - Dequeue: Remueve al pasajero del frente de la cola para su procesamiento.




### 5. Menú Principal

El menú principal de la aplicación permite al usuario realizar varias operaciones y está estructurado de la siguiente manera:




1. **Cargar Archivos de Entrada:**
   - Permite cargar los archivos JSON con la información de los aviones y pasajeros.
2. **Gestionar Aviones:**
   - Opción para mover aviones entre las listas de disponibles y en mantenimiento.
3. **Registrar Pasajeros:**
   - Opción para registrar pasajeros desde un archivo de entrada.
4. **Gestionar Equipaje:**
   - Permite gestionar el equipaje de los pasajeros a través de la pila.
5. **Generar Reportes:**
   - Genera y visualiza reportes de las estructuras de datos usando Graphviz.
6. **Salir:**
   - Termina la ejecución de la aplicación.

### Archivos de Entrada

- **Aviones:** Archivo en formato JSON que contiene la información de cada avión.

```
[
  {
    "id": "A123",
    "modelo": "Boeing 747",
    "estado": "Disponible"
  },
  {
    "id": "A124",
    "modelo": "Airbus A320",
    "estado": "Mantenimiento"
  }
]
```

- **Pasajeros:** Archivo en formato JSON que contiene la información de cada pasajero.

```
[
  {
    "pasaporte": "P001",
    "nombre": "John Doe",
    "vuelo": "A123",
    "asiento": "12A",
    "equipaje": 2
  },
  {
    "pasaporte": "P002",
    "nombre": "Jane Smith",
    "vuelo": "A124",
    "asiento": "15B",
    "equipaje": 1
  }
]
```

## Conclusiones

Este manual técnico evidencia claramente las principales ideas y propuestas generadas durante el desarrollo del Sistema de Gestión de Aeropuertos. A lo largo de este documento, se ha detallado la importancia de utilizar estructuras de datos avanzadas como listas enlazadas, listas circulares, pilas y colas para optimizar la gestión de vuelos, pasajeros y equipajes. La implementación de estas estructuras permite manejar grandes volúmenes de información de manera eficiente y efectiva, lo cual es crucial en el contexto aeroportuario.

Las propuestas destacadas incluyen la integración de Graphviz para la generación de reportes visuales, lo cual añade un valor significativo al proyecto al facilitar la interpretación de los datos y su análisis. Este enfoque no solo pone en práctica los conocimientos teóricos adquiridos en el curso, sino que también mejora la capacidad de los usuarios para desarrollar soluciones tecnológicas complejas.

## Referencias bibliográficas

magjac. (n.d.). Graphviz Visual Editor. Retrieved from http://magjac.com/graphviz-visual-editor/

Graphviz. (n.d.). Gallery: Directed. Retrieved from https://graphviz.org/Gallery/directed/

cplusplus.com. (n.d.). Retrieved from https://cplusplus.com/

nlohmann. (n.d.). nlohmann/json. GitHub. Retrieved from https://github.com/nlohmann/json?tab=readme-ov-file

Universidad de San Carlos de Guatemala
Escuela de Ingeniería en Ciencias y Sistemas, Facultad de Ingeniería
Laboratorio de Estructuras de Datos, Vacaciones del 1er. Semestre 2024.