UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

CENTRO UNIVERSITARIO DE OCCIDENTE

DIVISIÓN CIENCIAS DE LA INGENIERIA

ESTRUCTURA DE DATOS


# BIBLIOTECA MAGICA

**BRANDON GUSTAVO GÜINAC ROMÁN**

**201931217**

**FECHA DE ENTREGA: 02/11/2025**

**DOCUMENTACIÓN DE TAD'S IMPLEMENTADOS**

## 1. Introducción

El presente documento describe los **Tipos Abstractos de Datos (TAD's)** implementados en el sistema **"Biblioteca Mágica Alrededor del Mundo"**, desarrollado en **Python con interfaz gráfica**.
El proyecto tiene como objetivo gestionar una red de bibliotecas interconectadas, cada una con su propio catálogo de libros, aplicando estructuras de datos avanzadas para optimizar la búsqueda, almacenamiento, transferencia y visualización de información.

Todas las estructuras fueron implementadas desde cero, siguiendo los principios de encapsulamiento, modularidad y eficiencia.
Las representaciones gráficas de las estructuras se generan mediante **Graphviz**, el cual renderiza automáticamente las estructuras lógicas en diagramas visuales.

## 2. Lista de TAD's Implementados

1. **Lista Enlazada / Lista Doblemente Enlazada**
2. **Pila**
3. **Cola**
4. **Árbol AVL**
5. **Árbol B**
6. **Árbol B+**
7. **Tabla Hash**
8. **Grafo (Bibliotecas interconectadas)**

## 3. Descripción de Estructuras

### 3.1 Lista Enlazada y Lista Doblemente Enlazada

**Propósito:**
Administrar las colecciones de libros en cada biblioteca, permitiendo inserciones, eliminaciones y recorridos eficientes.

**Principales operaciones:**

- insertar(libro)
- eliminar(isbn)
- buscar(título)
- recorrer()

**Uso en el sistema:**
Cada biblioteca mantiene una lista enlazada con sus libros disponibles.
Cuando se ingresan nuevos registros desde archivos CSV, estos se almacenan en listas organizadas por colección o género.

**Complejidad temporal:**

- Inserción: O(1)
- Eliminación: O(n)
- Búsqueda: O(n)

### 3.2 Pila (Stack)

**Propósito:**
Controlar los libros devueltos y las operaciones de *"deshacer" (rollback)* en el registro de libros o transferencias.

**Principales operaciones:**

- push(libro) → Inserta un elemento.
- pop() → Elimina el elemento superior.
- peek() → Consulta el elemento superior.

**Uso en el sistema:**
Permite restaurar el estado anterior del sistema ante errores o registros duplicados, así como mantener un historial de acciones del usuario.

**Complejidad temporal:**

- Inserción: O(1)

- Eliminación: O(1)

### 3.3 Cola

**Propósito:**
Simular las colas de ingreso, preparación y despacho de libros entre bibliotecas.

**Principales operaciones:**

- enqueue(libro) → Agrega un libro al final de la cola.
- dequeue() → Retira el primer libro.
- peek() → Consulta el próximo libro a enviar.

**Uso en el sistema:**
Cada biblioteca posee tres colas:

1. **Cola de ingreso** (libros recién llegados).
2. **Cola de traspaso** (libros en tránsito intermedio).
3. **Cola de salida** (libros listos para despacho).

**Complejidad temporal:**

- Inserción: O(1)
- Eliminación: O(1)

### 3.4 Árbol AVL

**Propósito:**
Almacenar y ordenar libros por **título** o **ISBN**, manteniendo un equilibrio óptimo para búsquedas rápidas.

**Principales operaciones:**

- insertar(libro)
- eliminar(libro)
- buscar(título)
- recorrer_inorden()

**Uso en el sistema:**
El Árbol AVL es la estructura base para las búsquedas principales de libros por título.
Su equilibrio automático garantiza tiempos de búsqueda consistentes incluso con grandes volúmenes de datos.

**Complejidad temporal:**

- Inserción: O(log n)
- Eliminación: O(log n)
- Búsqueda: O(log n)

**Renderizado:**
Cada operación de inserción o eliminación puede generar un nuevo archivo .dot, renderizado con **Graphviz** para visualizar el árbol actualizado.

### 3.5 Árbol B

**Propósito:**
Organizar libros por **año de publicación**, permitiendo búsquedas por rango de fechas.

**Principales operaciones:**

- insertar(libro)
- buscar_rango(año_inicial, año_final)
- recorrer()

**Uso en el sistema:**
Optimiza la búsqueda y agrupación de libros por períodos o décadas.
Se utiliza cuando el usuario selecciona “Ordenar por año”.

**Complejidad temporal:**

- Inserción: O(log n)
- Búsqueda: O(log n)

**Renderizado:**
Genera visualizaciones en Graphviz mostrando nodos con múltiples claves ordenadas.

### 3.6 Árbol B+

**Propósito:**
Organizar libros por **género**, facilitando recorridos secuenciales y agrupados.

**Principales operaciones:**

- insertar(libro)
- buscar(género)
- recorrer_hojas()

**Uso en el sistema:**
Permite recorrer todos los libros de un mismo género rápidamente sin perder el orden.
Ideal para reportes de tipo “Listar libros por género”.

**Complejidad temporal:**

- Inserción: O(log n)
- Búsqueda: O(log n)

### 3.7 Tabla Hash

**Propósito:**
Indexar libros mediante su **ISBN**, garantizando búsquedas directas y únicas.

**Principales operaciones:**

- insertar(isbn, libro)
- buscar(isbn)
- eliminar(isbn)

**Uso en el sistema:**
Evita duplicidades y permite validar ISBN mágicos según las reglas del proyecto.
Maneja colisiones mediante **encadenamiento con listas** o **sondeo lineal**.

**Complejidad temporal:**

- Búsqueda promedio: O(1)
- Inserción promedio: O(1)

**Renderizado:**
Graphviz muestra la tabla hash con índices, celdas ocupadas y punteros a listas de colisiones.

### 3.8 Grafo Ponderado (Red de Bibliotecas)

**Propósito:**
Modelar la red nacional de bibliotecas interconectadas por rutas con pesos de **tiempo** o **costo mágico**.

**Principales operaciones:**

- agregar_biblioteca(nombre)
- conectar(origen, destino, peso)
- ruta_optima(origen, destino, criterio)

**Uso en el sistema:**
Permite calcular la ruta más corta o económica entre bibliotecas, utilizando algoritmos de **Dijkstra** o **Floyd-Warshall**.
Además, controla las colas de despacho y el flujo de libros en tránsito.

**Complejidad temporal:**

- Inserción de nodo: O(1)

- Cálculo de ruta (Dijkstra): $O(V^2)$
- Cálculo de ruta (Floyd): $O(V^3)$

**Renderizado:**
Graphviz genera nodos circulares (bibliotecas) y aristas ponderadas (rutas) con etiquetas de tiempo o costo.

**4. Integración de TAD's en la Interfaz Gráfica**

La interfaz gráfica del sistema permite interactuar directamente con las estructuras, ofreciendo opciones como:

- **Agregar / eliminar bibliotecas o libros.**
- **Visualizar estructuras renderizadas.**
- **Comparar métodos de búsqueda u ordenamiento.**
- **Simular transferencias de libros en tiempo real.**

Cada estructura posee un método graficar() que:

1. Genera un archivo .dot.
2. Llama a graphviz.render() para producir una imagen .png.
3. Muestra el resultado dentro de la GUI o en una ventana externa.

## 5. Conclusión

Los TAD's implementados en "Biblioteca Mágica Alrededor del Mundo" representan la base lógica y funcional del sistema.
Su correcta integración permite una gestión eficiente, rápida y visualmente comprensible de los datos.
El uso de **Graphviz** como motor de renderizado complementa la experiencia gráfica, proporcionando una representación clara de las estructuras y su comportamiento dinámico durante la ejecución del programa.