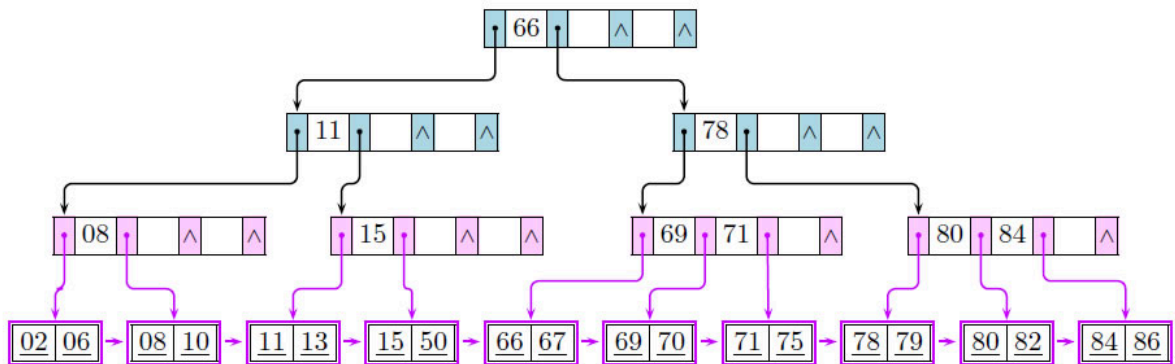
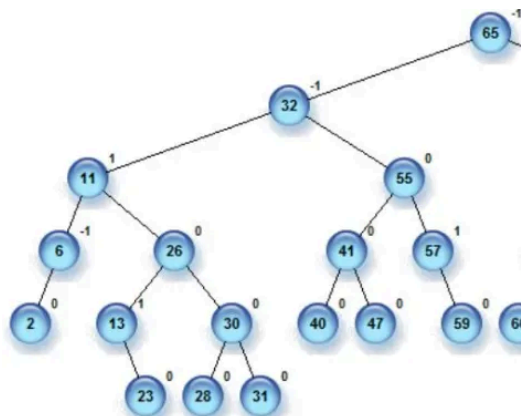



MANUAL TÉCNICO




TAD's

Libro:

 Libro
<ul style="list-style-type: none">▢ cantidad: int▢ titulo: string▢ isbn: string▢ genero: string▢ fecha: string▢ autor: string
<ul style="list-style-type: none">● Libro()● Libro(titulo, isbn, genero, fecha, autor)● getTitulo(): string● getIsbn(): string● getGenero(): string● getFecha(): string● getAutor(): string● getCantidad(): int● setTitulo(titulo: string): void● setIsbn(isbn: string): void● setGenero(genero: string): void● setFecha(fecha: string): void● setAutor(autor: string): void● incrementarCantidad(): void● toString(): string● compararPorTitulo(otro: Libro): int● compararPorIsbn(otro: Libro): int● compararPorFecha(otro: Libro): int● getFechaInt(): int

Catalogo: lista para la comparación lineal

 Catalogo
<ul style="list-style-type: none">▢ cabeza: Nodo*▢ cola: Nodo*▢ tamaño: int
<ul style="list-style-type: none">● Catalogo()● ~Catalogo()● agregarLibro(libro: Libro*): void● eliminarLibroPorISBN(isbn: string): bool● estaVacio(): bool● getTamaño(): int● buscarTituloSecuencial(titulo: string): Libro*● buscarISBNSecuencial(isbn: string): Libro*● obtenerIterador(): Iterador
<div>----- Iterador -----</div> <ul style="list-style-type: none">● Iterador(nodo: Nodo*)● tieneSiguiente(): bool● siguiente(): Libro*

Árbol AVL de títulos:

C ArbolAVL
<p>■ raiz: NodoAVL*</p>
<ul style="list-style-type: none"> ● ArbolAVL() ● ~ArbolAVL() ● insertar(libro: Libro*): void ● eliminarPorISBN(isbn: string, titulo: string): void ● buscar(titulo: string, isbn: string = ""): NodoAVL* ● buscarPorTitulo(titulo: string): Libro* ● buscarTodosPorTitulo(titulo: string): ListaEncontrados* ● obtenerLibrosEnOrdenAlfabetico(): ListaLibros* ● estaVacio(): bool ● getRaiz(): NodoAVL* ● guardarComoDOT(ruta: string): void
<ul style="list-style-type: none"> ■ altura(nodo: NodoAVL*): int ■ factorBalance(nodo: NodoAVL*): int ■ nodoMinimo(nodo: NodoAVL*): NodoAVL* ■ insertarNodo(nodo: NodoAVL*, libro: Libro*): NodoAVL* ■ eliminarNodoEficiente(nodo: NodoAVL*, titulo: string, isbn: string): NodoAVL* ■ buscarNodo(nodo: NodoAVL*, titulo: string, isbn: string): NodoAVL* ■ destruir(nodo: NodoAVL*): void ■ rotarIzquierda(nodo: NodoAVL*): NodoAVL* ■ rotarDerecha(nodo: NodoAVL*): NodoAVL* ■ rotacionDobleIzquierda(nodo: NodoAVL*): NodoAVL* ■ rotacionDobleDerecha(nodo: NodoAVL*): NodoAVL* ■ balancear(nodo: NodoAVL*): NodoAVL* ■ compararLibros(a: Libro*, b: Libro*): int ■ compararISBN(a: Libro*, b: Libro*): int ■ buscarPorTituloRecurso(nodo: NodoAVL*, titulo: string): Libro* ■ buscarTodosPorTituloRecurso(nodo: NodoAVL*, titulo: string, lista: ListaEncontrados*): void ■ recorrerEnOrdenRecurso(nodo: NodoAVL*, lista: ListaLibros*): void

Árbol B de fechas:

C ArbolB	
<p>□ raiz: NodoB*</p> <ul style="list-style-type: none"> ● ArbolB() ● ~ArbolB() ● insertar(libro: Libro*): void ● buscarPorRango(fechaInicio: int, fechaFin: int): ListaLibros* ● buscarPorFecha(fecha: string): Libro* ● eliminarPorISBN(isbn: string, fecha: string): void ● eliminar(fecha: string): void ● getRaiz(): NodoB* ● imprimirParaPrueba(): void ● verificarDuplicados(): void 	<ul style="list-style-type: none"> ■ dividirHijo(padre: NodoB*, i: int): void ■ insertarNoLleno(nodo: NodoB*, fecha: int, libro: Libro*): void ■ buscarPorRangoRecursivo(nodo: NodoB*, inicio: int, fin: int, resultados: ListaLibros*): void ■ destruirRecursivo(nodo: NodoB*): void ■ obtenerMaximo(nodo: NodoB*): EntradaFecha* ■ obtenerMinimo(nodo: NodoB*): EntradaFecha* ■ fusionar(nodo: NodoB*, idx: int): void ■ prestarDelzquierda(nodo: NodoB*, idx: int): void ■ prestarDeDerecha(nodo: NodoB*, idx: int): void ■ eliminarRecursivo(nodo: NodoB*, fecha: int): void ■ buscarFechaGlobal(fecha: int): EntradaFecha* ■ buscarFechaEnNodo(nodo: NodoB*, fecha: int): EntradaFecha* ■ recorrerAVLyAgregarLibros(nodoAVL: NodoIndiceISBN*, resultados: ListaLibros*): void

Árbol B+ de géneros:

C ArbolBPlus
□ raiz: NodoBPlus* □ primeraHoja: NodoHoja*
<ul style="list-style-type: none"> ● ArbolBPlus() ● ~ArbolBPlus() ● insertarSoloGenero(genero: string): void ● insertarLibroEnGenero(libro: Libro*): void ● eliminarPorISBN(isbn: string, genero: string): void ● buscarPorGenero(genero: string): ListaLibros* ● getRaiz(): NodoBPlus* ● getPrimeraHoja(): NodoHoja* ● recorrerEstructura(nodo: NodoBPlus*, nivel: int = 0): void ■ buscarHoja(genero: string): NodoHoja* ■ insertarEnHoja(hoja: NodoHoja*, libro: Libro*): void ■ dividirHoja(hoja: NodoHoja*): void ■ dividirInterno(interno: NodoInterno*): void ■ liberarRecursivo(nodo: NodoBPlus*): void ■ buscarPadre(actual: NodoBPlus*, hijo: NodoBPlus*): NodoInterno* ■ recorrerAVLyAgregarLibros(nodoAVL: NodoIndiceISBN*, resultados: ListaLibros*): void ■ buscarGeneroAux(genero: string, hojaOut: NodoHoja**, pos: int): bool ■ buscarPosicionGenero(genero: string, hojaResultado: NodoHoja**, posicionResultado: int): bool ■ eliminarGeneroDeHoja(hoja: NodoHoja*, posicion: int): void ■ balancearHoja(hoja: NodoHoja*): void ■ redistribuirHojas(hojaIzq: NodoHoja*, hojaDer: NodoHoja*, padre: NodoInterno*, posClavePadre: int): void ■ fusionarHojas(hojaIzq: NodoHoja*, hojaDer: NodoHoja*, padre: NodoInterno*, posClavePadre: int): void ■ balancearInterno(interno: NodoInterno*): void ■ redistribuirInternos(internoIzq: NodoInterno*, internoDer: NodoInterno*, padre: NodoInterno*, posClavePadre: int): void ■ fusionarInternos(internoIzq: NodoInterno*, internoDer: NodoInterno*, padre: NodoInterno*, posClavePadre: int): void

Árbol AVL de ISBN de apoyo:

C IndiceISBN
□ raiz: NodoIndiceISBN*
<ul style="list-style-type: none"> ● IndiceISBN() ● ~IndiceISBN() ● insertar(isbn: string, libro: Libro*): void ● eliminar(isbn: string): void ● buscar(isbn: string): Libro* ● estaVacio(): bool ● getRaiz(): NodoIndiceISBN* ■ altura(nodo: NodoIndiceISBN*): int ■ factorBalance(nodo: NodoIndiceISBN*): int ■ balancear(nodo: NodoIndiceISBN*): NodoIndiceISBN* ■ rotarIzquierda(nodo: NodoIndiceISBN*): NodoIndiceISBN* ■ rotarDerecha(nodo: NodoIndiceISBN*): NodoIndiceISBN* ■ insertarNodo(nodo: NodoIndiceISBN*, isbn: string, libro: Libro*): NodoIndiceISBN* ■ eliminarNodo(nodo: NodoIndiceISBN*, isbn: string): NodoIndiceISBN* ■ nodoMinimo(nodo: NodoIndiceISBN*): NodoIndiceISBN* ■ destruirRecursivo(nodo: NodoIndiceISBN*): void

Complejidad:

AVL de títulos:

INSERCIÓN AVL - $O(\log n)$

Búsqueda de posición: $O(\log n)$ - Recorre desde raíz hasta hoja

Inserción del nodo: $O(1)$ - Creación e inicialización

Balanceo recursivo: $O(\log n)$ - Actualiza alturas y aplica rotaciones

Rotaciones: $O(1)$ por nivel - Reorganización local

Caso crítico: Inserción ordenada requiere máximo $\log_2(n)$ rotaciones

BÚSQUEDA AVL - $O(\log n)$ exacta / $O(n)$ múltiple

Búsqueda exacta: $O(\log n)$ - Compara título + ISBN, divide espacio de búsqueda

Búsqueda múltiple: $O(n)$ - Recorrido inorden forzado para encontrar coincidencias

Comparaciones: Máximo altura del árbol $\approx 1.44 \times \log_2(n+2)$

ELIMINACIÓN AVL - $O(\log n)$

Localización: $O(\log n)$ - Búsqueda binaria por título e ISBN

Reemplazo:

0 hijos: $O(1)$ - Eliminación directa

1 hijo: $O(1)$ - Promoción del hijo

2 hijos: $O(\log n)$ - Búsqueda del sucesor + reemplazo

Balanceo post-eliminación: $O(\log n)$ - Rotaciones hasta raíz si es necesario

Árbol B de fechas:

Inserción: $O(T \times \log_{\square} n)$

Búsqueda de fecha existente: $O(\log_{\square} n)$ - Recorrido altura árbol

Inserción en AVL interno: $O(\log m)$ - Donde m = libros por fecha

División de nodos: $O(T)$ - Reorganización claves/hijos

Altura del árbol: $O(\log_{\square} n)$ - Con $t = T$ (grado mínimo)

Características:

Cada nodo tiene entre $T-1$ y $2T-1$ claves

Búsqueda binaria interna en nodos: $O(\log T)$

Inserción preserva propiedades B-tree

Búsqueda por rango: $O(\log_{\square} n + k)$

$\log_{\square} n$: Recorrido altura del árbol para encontrar el rango

k : Número total de libros en el rango de fechas

Desglose:

Navegación al rango: $O(\log_{\square} n)$ - Búsqueda primeros nodos del rango

Recorrido inorden dentro del rango: $O(k)$ - k = libros encontrados

Procesamiento AVL interno: $O(m)$ por fecha - m = libros en cada fecha

Características:

Eficiente para consultas de rango temporal

Recorre solo nodos relevantes al rango

Escala lineal con resultados, no con tamaño total

Eliminación: $O(T \times \log n)$

Búsqueda de la fecha: $O(\log n)$ - Recorrido altura árbol

Eliminación en AVL interno: $O(\log m)$ - Eliminar ISBN del AVL

Operaciones de balanceo B-tree:

Fusión: $O(T)$ - Combinar nodos hermanos

Préstamo: $O(T)$ - Rotación de claves entre hermanos

Reorganización: $O(T)$ - Ajuste de punteros

Casos de eliminación:

Caso 1 (hoja): $O(T)$ - Eliminación directa + desplazamiento

Caso 2 (interno): $O(\log n)$ - Búsqueda sucesor/predecesor

Caso 3 (fusión): $O(T)$ - Combinación de nodos

Árbol B+ de géneros:

Inserción: $O(T \times \log n)$

Búsqueda de hoja: $O(\log n)$ - Recorrido desde raíz hasta hoja apropiada

Búsqueda binaria en hoja: $O(\log T)$ - Localizar posición dentro del nodo hoja

Inserción en AVL interno: $O(\log m)$ - Insertar ISBN en el índice del género

División de hojas/nodos: $O(T)$ - Reorganización de claves y punteros

Casos específicos:

Insertar solo género: $O(\log n)$ - Sin AVL interno

Insertar libro en género: $O(T \times \log n)$ - Con gestión completa

Operaciones de división:

División de hoja: $O(T)$ - Crear nueva hoja, redistribuir entradas, ajustar enlaces

División de nodo interno: $O(T)$ - Promover clave, redistribuir hijos

Mantenimiento: Preserva propiedades B+ con splits recursivos

Búsqueda por género: $O(\log n + m)$

Búsqueda de hoja: $O(\log n)$ - Recorrido desde raíz hasta la hoja que contiene el género

Búsqueda lineal en hoja: $O(T)$ - Escaneo secuencial dentro del nodo hoja (en el peor caso)

Recorrido AVL interno: $O(m)$ - m = número de libros en ese género específico

Desglose detallado:

Navegación árbol: $O(\log n)$ - Cada nivel realiza búsqueda binaria $O(\log T)$ en nodos internos

Procesamiento hoja: $O(T)$ - Búsqueda secuencial dentro del nodo hoja (máximo $2T-1$ elementos)

Extracción resultados: $O(m)$ - Recorrido inorden del AVL interno para obtener todos los libros del género

Caso óptimo: $O(\log n)$ cuando el género no existe

Caso típico: $O(\log n + m)$ cuando se encuentra el género y se recuperan m libros

Eliminación: $O(T \times \log n)$

Búsqueda de género: $O(\log n)$ - Localizar hoja y posición específica

Eliminación en AVL interno: $O(\log m)$ - Eliminar ISBN del árbol AVL del género

Balanceo post-eliminación: $O(T \times \log n)$ - Operaciones de redistribución/fusión recursivas

Operaciones de balanceo:

Redistribución hojas: $O(T)$ - Transferencia de entradas entre nodos hermanos

Fusión hojas: $O(T)$ - Combinación de nodos y actualización de enlaces

Balanceo nodos internos: $O(T \times \log n)$ - Propagación recursiva hacia raíz

Casos de eliminación:

Caso simple: $O(\log n)$ - Eliminación sin underflow

Caso complejo: $O(T \times \log n)$ - Con redistribución/fusión múltiple

Catalogo Lista Lineal:

Inserción (agregarLibro): $O(1)$ - La inserción se realiza siempre al final de la lista mediante el puntero cola, sin necesidad de recorrer la estructura, manteniendo referencias directas tanto al inicio como al final de la lista.

Eliminación (eliminarLibroPorISBN): $O(n)$ - Requiere búsqueda lineal secuencial a través de todos los nodos en el peor caso, con operaciones de reenlace de punteros $O(1)$ una vez localizado el nodo objetivo, manejando casos especiales de eliminación en cabeza y cola.

Búsqueda secuencial (buscarTituloSecuencial/buscarISBNSecuencial): $O(n)$ - Recorrido lineal exhaustivo desde la cabeza hasta encontrar la coincidencia o llegar al final, sin optimizaciones de ordenamiento ni estructuras auxiliares de índice.

Operaciones auxiliares:

Verificación vacío: $O(1)$ - Consulta directa del puntero cabeza

Obtención tamaño: $O(1)$ - Variable interna mantenida

Iteración: $O(1)$ por elemento - Avance secuencial constante

Características:

Estructura: Lista enlazada simple con punteros cabeza-cola

Orden: Inserción por orden de llegada (FIFO implícito)

Uso principal: Referencia base para medición de rendimiento contra estructuras complejas

Tabla de comparaciones

Estructura	Inserción	Búsqueda Exacta	Búsqueda múltiple	Eliminación	Recorrido
AVL por ISBN	$O(\log n)$	$O(\log n)$	- -	$O(\log n)$	$O(n)$
AVL por título	$O(\log n)$	$O(\log n)$	$O(n)$	$O(\log n)$	$O(n)$
B por fechas	$O(T \log_t n)$	$O(\log n)$	$O(\log n + k)$	$O(T \log n)$	$O(n)$
B+ por géneros	$O(T \log_t n)$	$O(\log n + m)$	- -	$O(T \log n)$	$O(n)$
Catalogo Lista	$O(1)$	$O(n)$	$O(n)$	$O(n)$	$O(n)$

Notación

GLOSARIO DE NOTACIÓN BIG O

Símbolo	Significado	Contexto
O	Orden de complejidad	Límite superior asintótico (peor caso)
n	Número total de elementos	Tamaño total de la estructura
T	Grado del árbol B/B+	Número máximo de claves por nodo (parámetro)
t	Grado mínimo del árbol B/B+	$T = 2t - 1$ (mínimo $t-1$ claves)
$\log_t n$	Altura del árbol B/B+	Logaritmo base t de n ($t \approx T/2$)
k	Número de elementos en rango	Resultados de búsqueda por rango
m	Número de elementos en categoría	Libros por género/fecha específica
$\log n$	Altura del árbol AVL	Logaritmo base 2 de n

RANKING DE EFICIENCIA POR OPERACIÓN

INSERCIÓN:

- 1ro Catálogo: $O(1)$ - Inserción directa al final
- 2do AVLs: $O(\log n)$ - Balanceo automático
- 3ro Árboles B/B+: $O(T \log n)$ - Divisiones ocasionales

BÚSQUEDA EXACTA:

- 1ro AVLs: $O(\log n)$ - Búsqueda binaria óptima
- 2do Árboles B/B+: $O(\log n)$ - Búsqueda multi-nivel
- 3ro Catálogo: $O(n)$ - Búsqueda secuencial

BÚSQUEDA MÚLTIPLE:

- 1ro Árbol B por Fecha: $O(\log n + k)$ - Rangos eficientes
- 2do AVL por Título: $O(n)$ - Recorrido completo
- 3ro Catálogo: $O(n)$ - Secuencial

ELIMINACIÓN:

- 1ro AVLs: $O(\log n)$ - Balanceo eficiente
- 2do Árboles B/B+: $O(T \log n)$ - Operaciones de balanceo
- 3ro Catálogo: $O(n)$ - Búsqueda + eliminación