**CESAR AUGUSTO TAMAYO URRIAGO**

**Materia: Estructura De Datos**

**EVIDENCIAS**

**Texto

El contenido generado por IA puede ser incorrecto.Pilas:**

Esta clase utiliza dos estructuras Stack (una para deshacer y otra para rehacer) para simular el funcionamiento de un editor de texto. Cada vez que el usuario escribe algo, se guarda el estado anterior en la pila de “undo”. Al deshacer, se transfiere ese estado a la pila de “redo”, y viceversa.

**Complejidad:**

* push() y pop() son operaciones de tiempo constante: **O(1)**.
* Espacio: proporcional al número de acciones almacenadas → **O(n)**.

**Colas:** Captura de pantalla de computadora

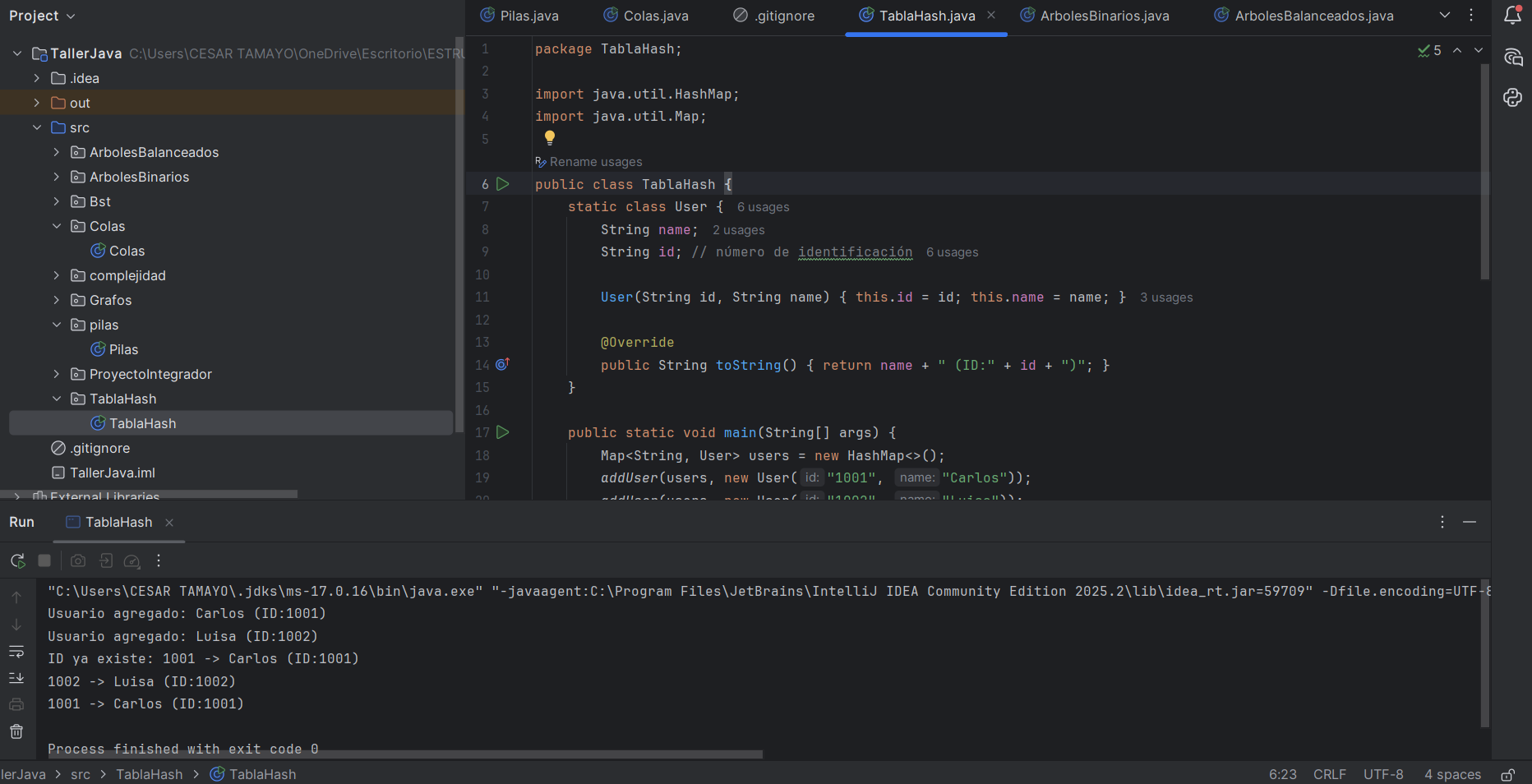
El contenido generado por IA puede ser incorrecto.

Implementa una PriorityQueue que ordena automáticamente los elementos según la prioridad de atención (número mayor = más urgente). Los pacientes se atienden de mayor a menor prioridad.

**Complejidad:**

* Inserción (add) y eliminación (poll): **O(log n)** por el reordenamiento interno del heap.
* Espacio: **O(n)** (un nodo por paciente).

**Tablas Hash:**



Usa un HashMap para almacenar usuarios por su identificación. Antes de agregar un usuario nuevo, verifica si el ID ya existe para evitar duplicados. Muestra mensajes según si el usuario se agregó o ya estaba registrado.

**Complejidad:**

* Inserción, búsqueda y eliminación: promedio **O(1)** (gracias a la función hash).
* Peor caso (colisiones): **O(n)**.
* Espacio: **O(n)** (una posición por usuario).

**Captura de pantalla de computadora

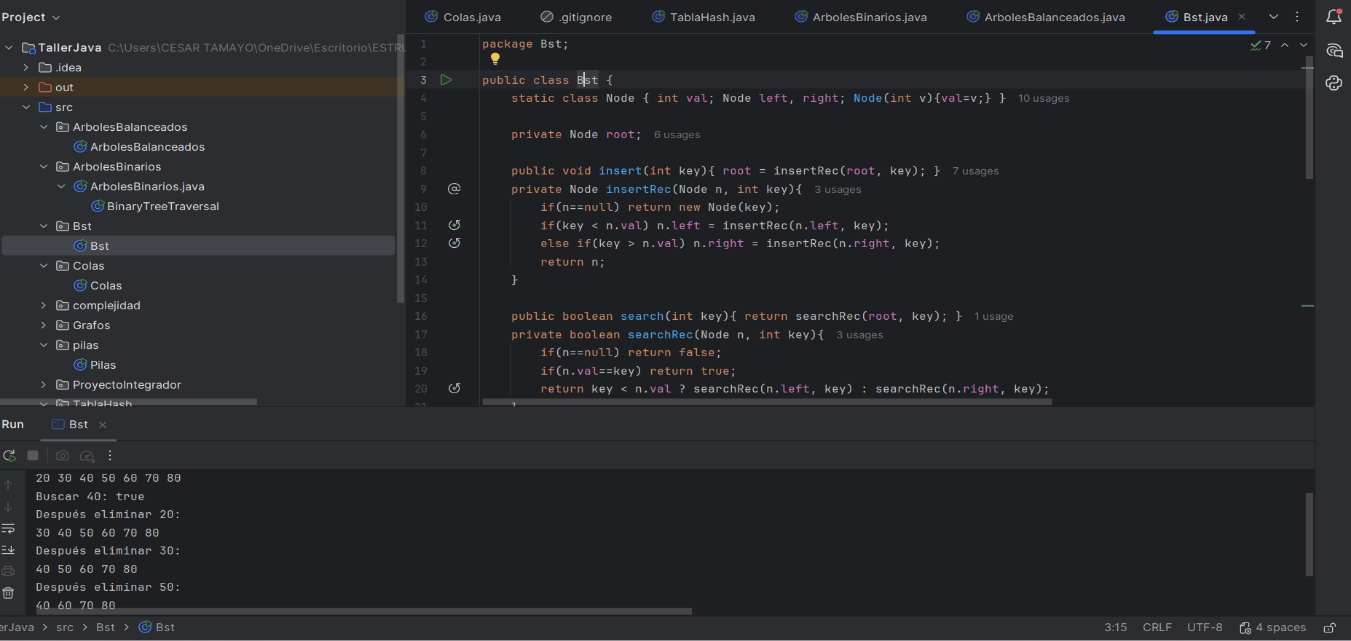
El contenido generado por IA puede ser incorrecto.Arboles Binarios:**

Crea un árbol binario con nodos conectados por referencias left y right. Implementa tres métodos de recorrido:

* Preorden: visita raíz → izquierda → derecha
* Inorden: izquierda → raíz → derecha
* Postorden: izquierda → derecha → raíz

**Complejidad:**

* Todos los recorridos visitan cada nodo una vez → O(n).
* Espacio: O(h) (altura del árbol, por la pila de recursión).

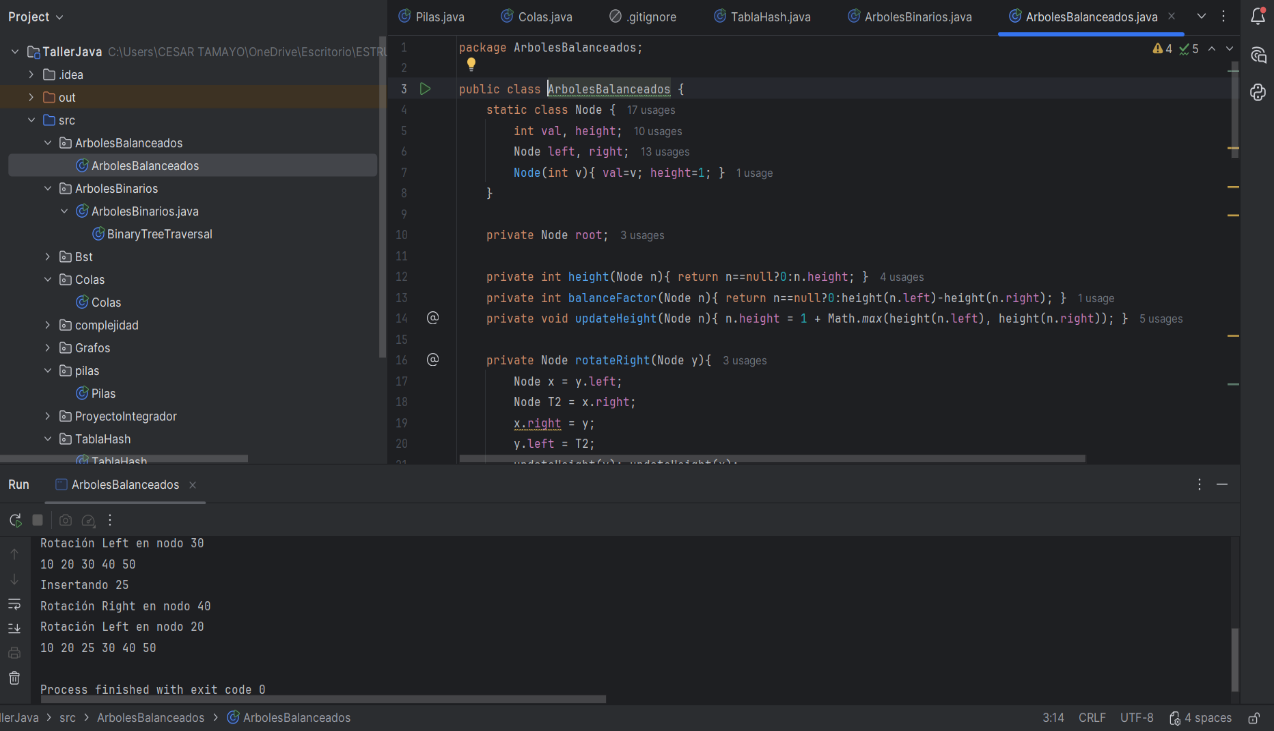
**Árboles Balanceados:**

Extiende el árbol binario garantizando que la diferencia de alturas entre subárboles sea como máximo 1. Después de cada inserción, calcula el factor de equilibrio y realiza rotaciones (Left, Right, Left-Right o Right-Left) para mantener el balance.

**Complejidad:**

* Inserción y búsqueda: O(log n) (árbol siempre balanceado).
* Rotaciones: O(1) por operación.
* Espacio: O(n).

**BST:**

****

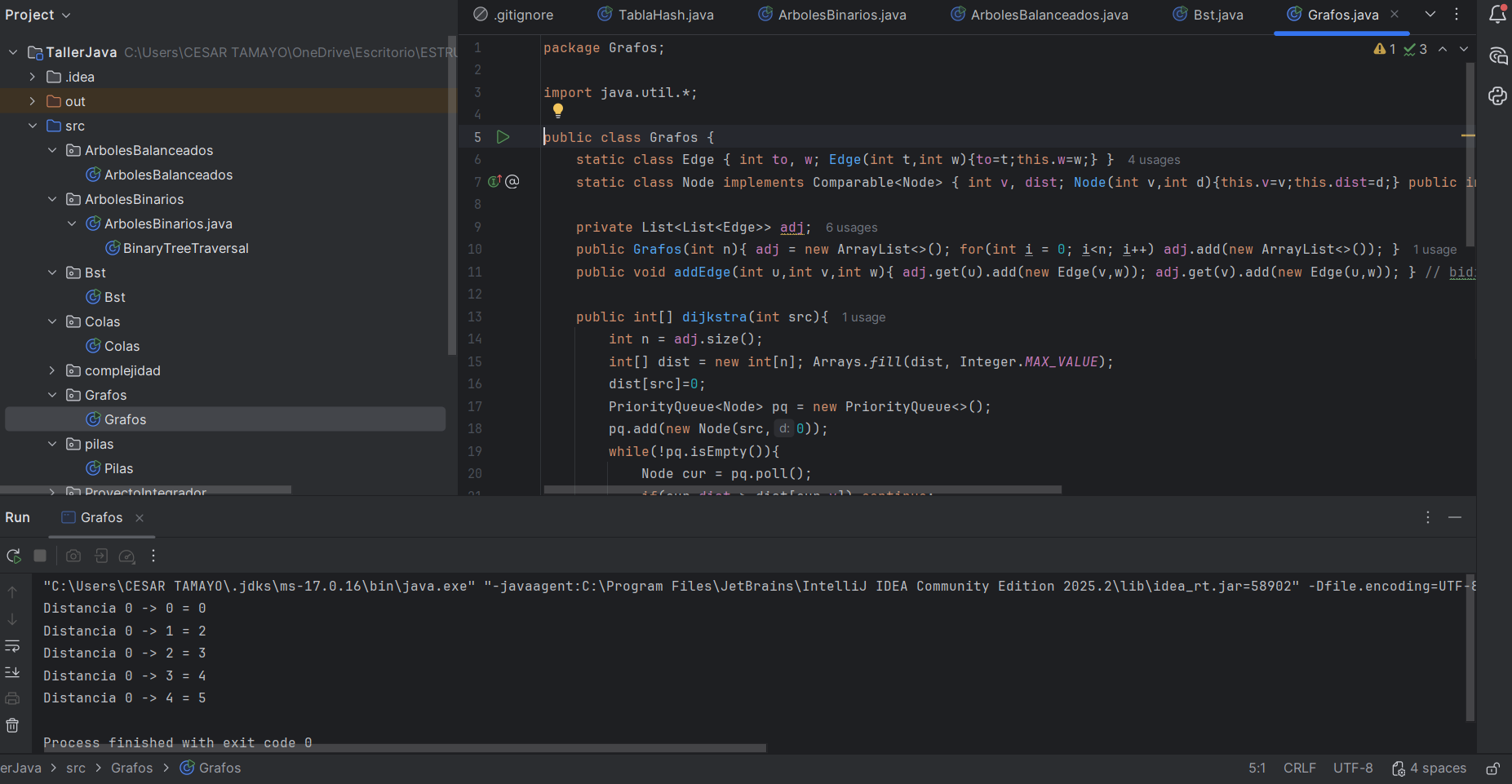
Árbol binario que organiza los datos de forma ordenada:

* Nodos menores a la izquierda
* Nodos mayores a la derecha  
  Permite insertar, buscar y eliminar valores preservando el orden.

**Complejidad:**

* Promedio: O(log n)
* Peor caso (si está desbalanceado): O(n)
* Espacio: O(n)

**Grafos:**

****

Representa un conjunto de ciudades como nodos conectados por aristas con pesos (distancias). Utiliza el algoritmo de Dijkstra con una PriorityQueue para hallar la ruta más corta desde una ciudad origen hacia las demás.

**Complejidad:**

* Dijkstra con cola de prioridad: O((V + E) log V), donde
  + *V* = número de vértices
  + *E* = número de aristas
* Espacio: O(V + E) (lista de adyacencia + distancias).

**Captura de pantalla de computadora

El contenido generado por IA puede ser incorrecto.Complejidad:**

Muestra por consola el resumen de las complejidades de cada estructura implementada, sirviendo como repaso teórico del taller.

**Complejidad:**Solo imprime texto, por tanto su costo computacional es O(1).

**Proyecto Integrador:**

**Captura de pantalla de computadora

El contenido generado por IA puede ser incorrecto.**

integra todas las estructuras en un solo programa.

* Usa la pila para deshacer/rehacer texto.
* La cola para simular atención prioritaria.
* La tabla hash para registrar usuarios.
* Los árboles para organizar e imprimir datos.
* El grafo para calcular rutas más cortas.
* Finalmente, imprime el análisis de complejidad.

**Complejidad:**  
Varía según la estructura usada dentro del flujo, pero globalmente el integrador ejecuta cada bloque de forma independiente → **O(n)** total aproximado considerando todas las operaciones.