1. **Clase PrendaRopa**

```python
import uuid

class PrendaRopa:  19 usages  alejandrobarreche *
    def __init__(self, nombre, precio, stock, color, tipo_de_prenda, material):  alejandrobarreche
        self._id_producto = str(uuid.uuid4())  # Genera un UUID único
        self.nombre = nombre
        self._precio = precio
        self._stock = stock
        self.color = color
        self.tipo_de_prenda = tipo_de_prenda
        self.material = material

    @property  3 usages  alejandrobarreche
    def id_producto(self):
        """Getter para id_producto."""
        return self._id_producto

    @property  4 usages (1 dynamic)  alejandrobarreche
    def precio(self):
        """Getter para precio."""
        return self._precio

    @precio.setter  1 usage (1 dynamic)  alejandrobarreche
    def precio(self, precio):
        """Setter para precio."""
        if precio < 0:
            raise ValueError("El precio no puede ser negativo")
        self._precio = precio

    @property  4 usages (1 dynamic)  alejandrobarreche
    def stock(self):
        """Getter para stock."""
        return self._stock

    @stock.setter  1 usage (1 dynamic)  alejandrobarreche
    def stock(self, stock):
        """Setter para stock."""
        if stock < 0:
            raise ValueError("El stock no puede ser negativo")
        self._stock = stock
```

La clase PrendaRopa representa una prenda individual, y sus métodos principales son los getters y setters para los atributos.

- **Getters y Setters**: Cada getter y setter es O(1) debido a que solo acceden o modifican atributos individuales. Por lo tanto también son Ω(1)

2. **Clase Arbol (AVL)**

El árbol AVL es una estructura de datos balanceada, lo que garantiza que las operaciones de búsqueda, inserción y eliminación sean eficientes. Dado que se mantiene balanceado, la altura del árbol es O(log N), donde N es el número de nodos.

- **Inserción (insertar)**: O(log N) / Ω(1)
    - Insertar un nodo en un árbol AVL toma O(log N), ya que el árbol se balancea con cada inserción. En el caso más sencillo tarda una ejecución
- **Eliminación (eliminar)**: O(log N) / Ω(1)
    - Similar a la inserción, eliminar un nodo implica operaciones de rebalanceo, por lo que es O(log N). En el caso más sencillo vuelve a tardar una ejecución
- **Búsqueda (buscar)**: O(log N) / Ω(1)
    - Localizar un nodo específico también toma O(log N). Sin embargo, si el primer elemento es el que busca tarda una sola iteración
- **Altura (obtener_altura)**: O(1) / Ω(1)
    - La altura de cada nodo se almacena, por lo que acceder a ella es una operación constante.
- **Balanceo (balancear)**: O(log n) / Ω(1)

  O(1) para cada nodo; sin embargo, en el peor caso de una operación de inserción o eliminación, se aplicará a O(log N) nodos.
    - Esta operación depende de la rotación requerida para el balanceo (rotación simple o doble), por lo que cada llamada de balanceo es O(1).
- **Listar en Orden (listar_en_orden)**: O(N) / Ω(n)
    - Generar una lista ordenada de los nodos requiere un recorrido in-order, que visita cada nodo una vez.
- **Buscar en Rango (buscar_en_rango)**: O(log N) / Ω(log n)
    - Al ser un árbol balanceado tarda log n en encontrar todos los elementos que estén dentro de un rango

---

3. Clase Catalogo

La clase Catalogo gestiona el inventario de ropa mediante un diccionario y dos árboles AVL (para atributos de precio y stock). Este diseño permite acceder a prendas rápidamente por su ID y realizar búsquedas eficientes por precio y stock.

- **Añadir Prenda (añadir_prenda)**: O(log N) / Ω(1)
    - Insertar una prenda en el diccionario es O(1).
    - Insertarla en los árboles AVL de precio y stock toma O(log N), ya que se realiza una operación de inserción en un árbol balanceado; que como he mencionado previamente, en el mejor caso tarda Ω(1).
- **Eliminar Prenda (eliminar_prenda)**: O(log N) / Ω(1)
    - Similar a añadir_prenda, eliminar una prenda del diccionario es O(1), pero requiere O(log N) para eliminarla de los árboles AVL.
- **Buscar Prenda (buscar_prenda)**: O(1) / Ω(1)
    - La búsqueda por ID en el diccionario es O(1).
- **Listar Prendas (listar_prendas)**: O(N) / Ω(n)
    - Listar todas las prendas implica recorrer cada entrada en el diccionario, lo cual es O(N) y Ω(n) para cualquier caso.
- **Listar por Precio (listar_por_precio)**: O(N) / Ω(n)
    - Esta operación depende del recorrido in-order en el árbol AVL de precios, que es O(N) en el peor y en el mejor caso, puesto que recorre todos los elementos del árbol
- **Listar por Stock (listar_por_stock)**: O(N) / Ω(n)
    - Similar a listar_por_precio, este método también requiere un recorrido completo en el árbol AVL de stock.
- **Buscar en Rango de Precio (buscar_en_rango)**: O(log N) / Ω(log n)
    - Utiliza el método buscar_en_rango del árbol AVL para precios, que es O(log N) en el peor y en el mejor caso puesto que el árbol está balanceado.
- **Buscar en Rango de Stock (buscar_en_rango)**: O(log N) / Ω(log n)
    - Igualmente, el árbol AVL de stock se recorre en O(log N).

---

4. Flujo Principal (main.py)

El flujo principal se encarga de manejar el menú interactivo, ejecutando funciones específicas en Catalogo según la entrada del usuario.

- **Opciones del Menú**: O(1) / Ω(1) para mostrar opciones y procesar la entrada.
- **Operaciones del Catálogo**: La complejidad de cada opción del menú depende de la operación que se llame en la clase Catalogo.

Resumen de Complejidades

Clase	Operación	Complejidad O	Complejidad Ω	Complejidad Θ
PrendaRopa	Creación de Prenda	O(1)	Ω(1)	Θ(1)
	Getters/Setters	O(1)	Ω(1)	Θ(1)
Arbol (AVL)	Insertar	O(log N)	Ω(1)	-
	Eliminar	O(log N)	Ω(1)	-
	Buscar	O(log N)	Ω(1)	-
	Balanceo	O(log n)	Ω(1)	
	Listar en Orden	O(N)	Ω(n)	Θ(n)
	Buscar en Rango	O(log N)	Ω(log n)	Θ(log n)
Catalogo	Añadir Prenda	O(log N)	Ω(1)	-
	Eliminar Prenda	O(log N)	Ω(1)	-
	Buscar Prenda	O(1)	Ω(1)	Θ(1)
	Listar Prendas	O(N)	Ω(n)	Θ(n)
	Listar por Precio	O(N)	Ω(n)	Θ(n)
	Listar por Stock	O(N)	Ω(n)	Θ(n)
	Buscar en Rango (Precio)	O(log N)	Ω(log n)	Θ(log n)

Clase	Operación	Complejidad O	Complejidad Ω	Complejidad Θ
	Buscar en Rango (Stock)	O(log N)	Ω(log n)	Θ(log n)
Main	Menú Principal	O(1)	Ω(1)	Θ(1)