Ejercicio 4: Ventajas y Desventajas de Cada Estructura

---

1. ArrayList

- **Ventajas**:
  - Acceso aleatorio rápido a elementos por índice (tiempo de acceso constante).
- **Desventajas**:
  - Inserciones y eliminaciones en posiciones intermedias pueden ser lentas, ya que requieren el desplazamiento de elementos.
- **Uso recomendado**:
  - Ideal cuando se necesita un acceso rápido a elementos por índice y no se planea realizar muchas inserciones o eliminaciones.

---

2. LinkedList

- **Ventajas**:
  - Eficiente en inserciones y eliminaciones al inicio o final, ya que solo requiere cambiar referencias de nodos.
- **Desventajas**:
  - Acceso a elementos específicos puede ser lento debido a la necesidad de recorrer la lista hasta el elemento deseado.
- **Uso recomendado**:
  - Útil cuando se realizan muchas inserciones y eliminaciones en ambos extremos de la lista y no se necesita un acceso rápido por índice.

---

3. HashMap

- **Ventajas**:
  - Búsqueda y actualizaciones rápidas (tiempo constante promedio) gracias a su estructura hash.
- **Desventajas**:
  - No mantiene un orden de los elementos.
- **Uso recomendado**:
  - Excelente para almacenar pares clave-valor cuando el orden de los elementos no es importante y se necesita acceso rápido.

---

4. TreeMap

- **Ventajas**:

    - Mantiene el orden de los elementos según la clave y permite operaciones de rango eficiente.
- **Desventajas**:
    - Ligeramente más lento que HashMap debido al mantenimiento del orden.
- **Uso recomendado**:
    - Útil cuando es importante tener los elementos ordenados y realizar operaciones de rango en las claves.

---

5. LinkedHashMap

- **Ventajas**:
    - Proporciona acceso rápido similar a HashMap y mantiene el orden de inserción.
- **Desventajas**:
    - Consume ligeramente más memoria para mantener el orden de inserción.
- **Uso recomendado**:
    - Ideal para aplicaciones donde es importante mantener el orden de los elementos, como en cachés que dependen del orden de uso o de inserción.