Nachdenkzettel Collections

Aufgabe 1:

- **ArrayLists:**
  Wenig Speicherplatzbedarf, da keine Verweise auf vorherige/nachfolgende Elemente.
  Schneller zufälliger Zugriff per Index (O(1)).
  Effizient bei statischen oder selten geänderten Daten.
- **LinkedLists:**
  Häufige Einfüge- oder Löschoperationen ohne Verschiebung nachfolgender Elemente.
  Dynamisches Verhalten, effizient bei sich häufig ändernder Größe.
  Iterative Operationen erfordern Durchlauf, weniger effizienter zufälliger Zugriff.

Aufgabe 2:

- Deutliche Unterschiede in der Effizienz verschiedener Collection-Arten.

- **CopyOnWriteArrayList:**
  Ineffizient bei Datenänderungen.
- **HashMap und ArrayList:**
  Vergleichsweise schnelle Leistung.
- **LinkedList:**
  Bei allen Aktionen, außer "retain all", langsamer oder ähnlich schnell wie ArrayList.

Aufgabe 3:

CopyOnWriteArrayList kopiert das vorhandene Array bei jeder Veränderung und erstellt ein neues. Bei hohem Änderungsbedarf ist das sehr ineffizient.

Aufgabe 4:

Durch list.remove() wird die list während des Iterierens verändert. Das könnte zum Beispiel dazu führen, das beim Iterieren Elemente übersprungen werden oder es Endlosschleifen kommt.
Mit itr.remove() können die Elemente sicher entfernt werden