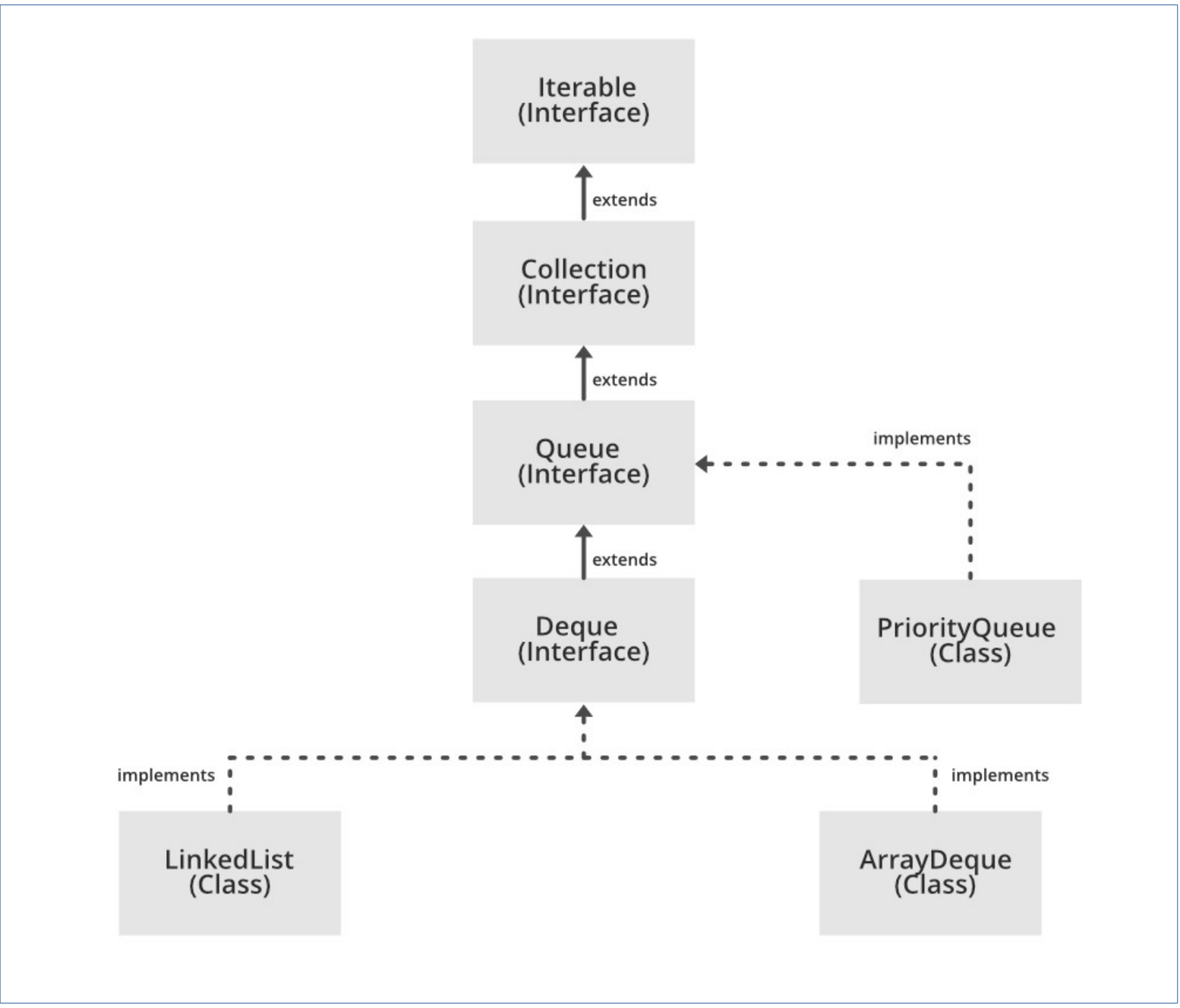
Iterable
(Interface)
extends
Collection
(Interface)
extends
Queue
(Interface)
implements
extends
Deque
(Interface)
PriorityQueue
(Class)
implements
implements
LinkedList
(Class)
ArrayDeque
(Class)

Queue:

- **FIFO-Prinzip (First In, First Out):** fügt am Ende ein und entfernt am Anfang.
- **Einfügen am Ende, Entfernen am Anfang**
  - **enqueue** => Elemente am Ende der Queue einfügen **(Zeitkomplexität O(n))**
  - **dequeue** => Elemente am Anfang der Queue entfernen **(Zeitkomplexität O(n))**
- **keinen direkten Zugriff** auf die Elemente, kann nur das Element am **Anfang** der Queue betrachten oder entfernen
- **Zugriffsoperationen:**
  - **Enqueue (Hinzufügen eines Elements):** Fügt ein Element am Ende der Queue hinzu.
  - **Dequeue (Entfernen eines Elements):** Entfernt das Element am Anfang der Queue.
  - **Peek (Betrachten des vordersten Elements):** Gibt das Element am Anfang der Queue zurück, ohne es zu entfernen.
- kann eine **dynamische Größe** haben
- **Varianten von Queues:**
  - **Simple Queue:** Standard-Queue, bei der **Einfügen am Ende und Entfernen am Anfang** erfolgt.
  - **Circular Queue (Kreis-Queue)**: Bei dieser Variante **zeigt das Ende der Queue auf den Anfang**, um Speicher effizienter zu nutzen.
  - **Deque (Doppelt-Ended Queue)**: In einer Deque können Elemente **sowohl am Anfang als auch am Ende eingefügt und entfernt werden**.
  - **Priority Queue**: Bei einer Prioritäts-Queue werden **Elemente basierend auf ihrer Priorität sortiert. Elemente mit höherer Priorität werden vor anderen bearbeitet, unabhängig von der Reihenfolge,** in der sie hinzugefügt wurden.
- Speicherbedarf einer Queue ist **O(n)**