Java Collections Framework – Cheat Sheet

1️⃣ Collection Interfaces

Interface	Description	Implementations
List	Ordered, allows duplicates	ArrayList, LinkedList, Vector, Stack
Set	Unique elements, no order	HashSet, LinkedHashSet, TreeSet
Queue	FIFO, used for task scheduling	LinkedList, PriorityQueue, ArrayDeque
Deque	Double-ended queue	ArrayDeque, LinkedList
Map	Key-value pairs, keys unique	HashMap, LinkedHashMap, TreeMap, Hashtable

---

2️⃣ List

- ArrayList → Resizable array, fast random access, slow insert/delete middle.
- LinkedList → Doubly linked list, fast insert/delete middle, slow random access.
- Vector → Thread-safe ArrayList (legacy).
- Stack → LIFO, extends Vector (legacy).

Common Methods: `add()`, `get()`, `remove()`, `set()`, `size()`, `contains()`

3️⃣ Set

- HashSet → Unordered, allows null, fastest.
- LinkedHashSet → Ordered by insertion.
- TreeSet → Sorted, navigable, no null key.

Common Methods: `add()`, `remove()`, `contains()`, `size()`, `isEmpty()`

4️⃣ Queue & Deque

- Queue (FIFO) → `offer()`, `poll()`, `peek()`
- Deque → `offerFirst()`, `offerLast()`, `pollFirst()`, `pollLast()`
- PriorityQueue → Min-heap by default, use comparator for max-heap

---

5️⃣ Map

- HashMap → Unordered, 1 null key, multiple null values, O(1) average.
- LinkedHashMap → Ordered, supports LRU cache.
- TreeMap → Sorted, O(log n), no null key.

- Hashtable → Synchronized, legacy.

Common Methods: `put()`, `get()`, `remove()`, `containsKey()`, `containsValue()`, `keySet()`, `values()`, `entrySet()`

HashMap Internals: - Collision handled with LinkedList → Java 8 converts to Red-Black Tree if bucket size > 8 - Load factor = 0.75 → triggers rehash

---

6️⃣ Utility Class – Collections

- `Collections.sort(list)` → Sort list
- `Collections.reverse(list)` → Reverse
- `Collections.shuffle(list)` → Shuffle
- `Collections.frequency(list, obj)` → Count frequency
- `Collections.synchronizedList(list)` → Thread-safe wrapper

7️⃣ Concurrent Collections

- ConcurrentHashMap → Thread-safe replacement for HashMap
- CopyOnWriteArrayList → Thread-safe list for reading > writing
- BlockingQueue → Producer-Consumer pattern

8️⃣ Key Interview Points

- Difference: HashMap vs Hashtable vs TreeMap vs LinkedHashMap
- Difference: HashMap vs TreeMap
- HashMap allows 1 null key, many null values → bucket 0 for null key
- LinkedHashMap → LRU Cache (`accessOrder = true`)
- Queue vs Stack vs Deque → FIFO/LIFO behavior