Java Collections Framework Summary

1. List ? Ordered Collection, Allows Duplicates

- Maintains insertion order
- Allows duplicate elements
- Access elements by index
- Useful when order matters

Example:

```
ArrayList<String> list = new ArrayList<>();
list.add("A"); list.add("B"); list.add("A");
Output: [A, B, A]
```

2. Set ? Unordered Collection, No Duplicates

- Stores unique elements only
- Does not maintain insertion order
- Duplicate values are ignored

Example:

```
HashSet<Integer> set = new HashSet<>();
set.add(1); set.add(2); set.add(1);
Output: [1, 2] (order not guaranteed)
```

3. LinkedHashSet ? Unique + Maintains Insertion Order

- No duplicates
- Maintains insertion order

Example:

```
LinkedHashSet<String> names = new LinkedHashSet<>();
names.add("A"); names.add("B"); names.add("A");
Output: [A, B]
```

4. TreeSet ? Unique + Sorted Order

- No duplicates
- Automatically sorts elements (ascending)
- Cannot store null if mixed

Example:

```
TreeSet<Integer> set = new TreeSet<>();
set.add(5); set.add(2); set.add(3);
```

Output: [2, 3, 5]

5. Map ? Key-Value Pairs

- Stores key ? value pairs
- Keys must be unique
- Values can repeat

Example:

```
HashMap<Integer, String> map = new HashMap<>();
map.put(1, "A"); map.put(2, "B"); map.put(1, "C");
```

Output: {1=C, 2=B}

6. LinkedHashMap ? Key-Value + Insertion Order

- Maintains order of keys
- Good for predictable iteration

Example:

```
LinkedHashMap<Integer, String> map = new LinkedHashMap<>();
map.put(1, "Java"); map.put(2, "Python");
```

Output: {1=Java, 2=Python}

7. TreeMap ? Sorted Map

- Keys stored in sorted order
- Null keys not allowed

Example:

```
TreeMap<Integer, String> map = new TreeMap<>();
map.put(3, "C"); map.put(1, "A"); map.put(2, "B");
```

Output: {1=A, 2=B, 3=C}

Common Methods in Collections

- add(), put(), remove(), contains(), containsKey(), get()
- addAll(), putAll() for merging collections

When to Use What?

- Use ArrayList for ordered duplicates
- Use LinkedList for fast insert/remove

Java Collections Framework Summary

- Use HashSet for unique items
- Use LinkedHashSet for unique + order
- Use TreeSet for sorted unique
- Use HashMap for fast key-value
- Use LinkedHashMap for ordered key-value
- Use TreeMap for sorted key-value