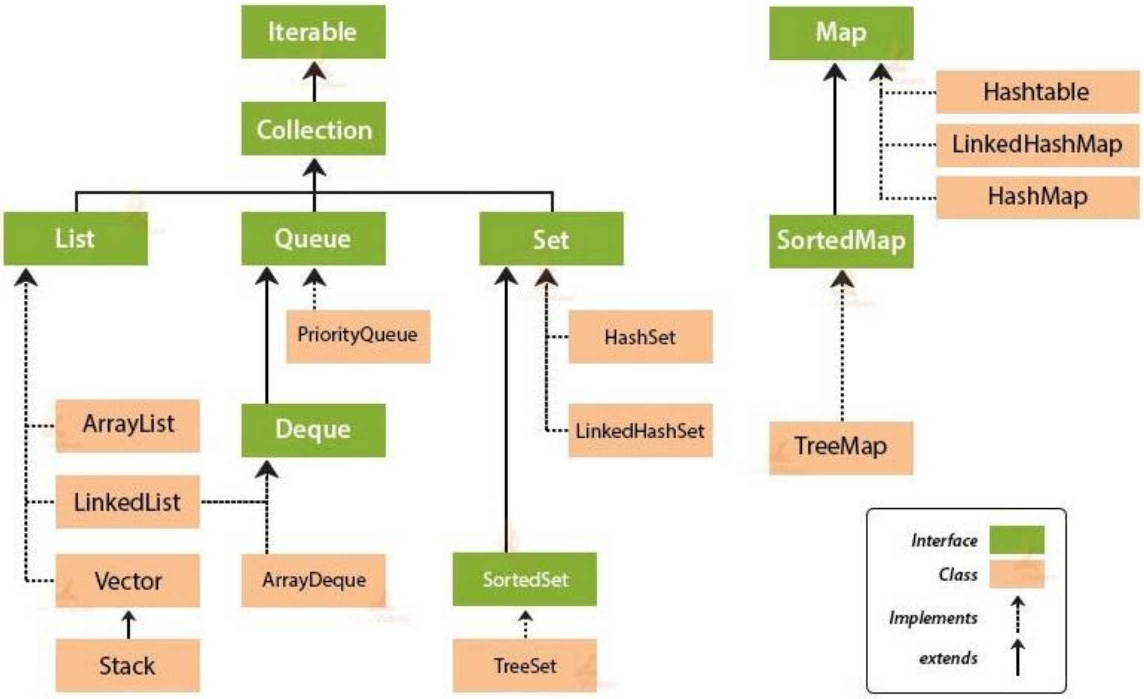
Day 21

Collections - ArrayList

Collection

- In Java, a **collection** represents a group of objects, often referred to as **elements**.
- Collections provide a way to **store, manipulate, and retrieve data** in a more organized manner.
- These collections are part of the **Java Collection Framework**.
- Java collections can only store **object types** (i.e., reference types such as String, Integer, etc.), and cannot store **primitive types** (e.g., int, char).
- To store primitive types, Java provides **Wrapper classes** (like Integer for int, Double for double, etc.).
- Java provides different types of collections, each with its specific use cases. Common collections include:
    - **List**: Stores elements in an ordered sequence (e.g., ArrayList, LinkedList).
    - **Set**: Stores unique elements without any particular order (e.g., HashSet, TreeSet).
    - **Map**: Stores key-value pairs (e.g., HashMap, TreeMap).
- **Iterable** and **Collection** are root interfaces in the Java Collection Framework

Hierarchy of Collection Framework:



ArrayList

1. **ArrayList is implemented List interface.**

2. **Heterogeneous Data**

   An ArrayList can store both **homogeneous** (same type) and **heterogeneous** (different types) data. However, if generics are used, it is typically constrained to homogeneous types.

   a. Example without generics: ArrayList list = new ArrayList(); – can store different types like String, Integer, Double, etc.

   b. Example with generics: ArrayList<String> list = new ArrayList< String >(); – can store only String elements.

3. **Duplicate Elements Allowed**

   An ArrayList allows **duplicate elements**. You can add the same object multiple times, and it will be stored as individual elements.

4. **Maintains Insertion Order**

   An ArrayList **maintains the insertion order** of elements. The order in which elements are added will be the order in which they are retrieved.

5. **Indexing Support**

   ArrayList supports **index-based access** to its elements. You can access, update, or remove elements using their index positions.

   a. Example: list.get(0); – retrieves the element at index 0.

6. **Multiple Nulls Allowed**

   An ArrayList allows storing **multiple null values**. There is no restriction on the number of null elements.