**COLLECTIONS**

In Java, the Collections framework provides a set of classes and interfaces to work with groups of objects, or collections. Collections in Java allow you to store, manipulate, and retrieve objects efficiently. The Collections framework includes interfaces such as List, Set, Queue, and Map, along with their various implementations.

Here's an overview of the main interfaces and their implementations in the Java Collections framework:

1. **List Interface**:
   * Ordered collection of elements where duplicate elements are allowed.
   * Main implementations: ArrayList, LinkedList, Vector.
2. **Set Interface**:
   * Unordered collection of unique elements. No duplicate elements are allowed.
   * Main implementations: HashSet, LinkedHashSet, TreeSet.
3. **Queue Interface**:
   * Ordered collection of elements designed for holding elements before processing.
   * Main implementations: LinkedList, PriorityQueue.
4. **Map Interface**:
   * Maps keys to values, where each key is unique.
   * Main implementations: HashMap, LinkedHashMap, TreeMap, ConcurrentHashMap.
5. **Deque Interface**:
   * Double-ended queue that supports insertion and removal of elements at both ends.
   * Main implementations: ArrayDeque, LinkedList.
6. **Stack Class**:
   * Represents a last-in, first-out (LIFO) stack of objects.
   * Main methods: push(), pop(), peek().
7. **Collections Class**:
   * Provides utility methods for working with collections, such as sorting, searching, and synchronizing.
   * Main methods: sort(), binarySearch(), reverse(), shuffle(), synchronizedCollection().

**EXAMPLE CODE:**

**package** com.dxc;

**import** java.util.\*;

**public** **class** CollectionsExample {

**public** **static** **void** main(String[] args) {

// List example

List<String> list = **new** ArrayList<>();

list.add("Apple");

list.add("Banana");

list.add("Orange");

System.***out***.println("List: " + list);

// Set example

Set<Integer> set = **new** HashSet<>();

set.add(10);

set.add(20);

set.add(30);

System.***out***.println("Set: " + set);

// Map example

Map<String, Integer> map = **new** HashMap<>();

map.put("One", 1);

map.put("Two", 2);

map.put("Three", 3);

System.***out***.println("Map: " + map);

// Queue example

Queue<String> queue = **new** LinkedList<>();

queue.add("John");

queue.add("Alice");

queue.add("Bob");

System.***out***.println("Queue: " + queue);

// Stack example

Stack<Integer> stack = **new** Stack<>();

stack.push(100);

stack.push(200);

stack.push(300);

System.***out***.println("Stack: " + stack);

// Deque example

Deque<Character> deque = **new** ArrayDeque<>();

deque.addFirst('A');

deque.addLast('Z');

deque.addFirst('B');

System.***out***.println("Deque: " + deque);

}

}

**OUTPUT:**

List: [Apple, Banana, Orange]

Set: [20, 10, 30]

Map: {One=1, Two=2, Three=3}

Queue: [John, Alice, Bob]

Stack: [100, 200, 300]

Deque: [B, A, Z]