Here are some **Collections manipulation coding questions** to help you practice for HackerRank assessments:

---

1. Sort a List of Strings by Length

Write a program to sort a list of strings based on their lengths, and if two strings have the same length, sort them lexicographically.

Input:

- A list of strings.

Output:

- A sorted list of strings first by length, then lexicographically.

java

Sample Code

```
import java.util.*;


public class SortStringsByLength {

   public static void main(String[] args) {

      List<String> strings = Arrays.asList("apple", "banana", "kiwi", "pear", "grape");

      Collections.sort(strings, (a, b) -> {

         if (a.length() == b.length()) {

            return a.compareTo(b);

         }

         return a.length() - b.length();

      });

      System.out.println(strings);

   }

}
```

---

2. Find Duplicates in an ArrayList

Given an ArrayList of integers, find and print the duplicates.

Input:

- A list of integers.

Output:

- The duplicate elements from the list.

java

Sample Code

```
import java.util.*;


public class FindDuplicates {
    public static void main(String[] args) {
        List<Integer> list = Arrays.asList(1, 2, 3, 4, 5, 1, 2, 6);
        Set<Integer> seen = new HashSet<>();
        Set<Integer> duplicates = new HashSet<>();


        for (int num : list) {
            if (!seen.add(num)) {
                duplicates.add(num);
            }
        }


        System.out.println(duplicates);
    }
}
```

---

3. Frequency Count of Each Element

Given a list of integers, count the frequency of each element in the list and print it.

Input:

- A list of integers.

Output:

- A frequency count of each element.

java

Sample Code

```
import java.util.*;


public class FrequencyCount {
  public static void main(String[] args) {
    List<Integer> list = Arrays.asList(1, 2, 2, 3, 4, 3, 4, 4);
    Map<Integer, Integer> frequencyMap = new HashMap<>();


    for (int num : list) {
      frequencyMap.put(num, frequencyMap.getOrDefault(num, 0) + 1);
    }


    for (Map.Entry<Integer, Integer> entry : frequencyMap.entrySet()) {
      System.out.println(entry.getKey() + ": " + entry.getValue());
    }
  }
}
```

---

4. Remove Duplicates from a List

Write a program to remove duplicates from a list while preserving the original order of the elements.

Input:

- A list of integers.

Output:

- A list with duplicates removed.

java

Sample Code

```
import java.util.*;


public class RemoveDuplicates {
  public static void main(String[] args) {
    List<Integer> list = Arrays.asList(1, 2, 3, 2, 4, 5, 1, 6);
    Set<Integer> seen = new LinkedHashSet<>(list);
    System.out.println(new ArrayList<>(seen));
  }
}
```

---

5. Merge Two Sorted Lists

You are given two sorted lists. Write a program to merge them into a single sorted list.

Input:

- Two sorted lists of integers.

Output:

- A single merged sorted list.

java

Sample Code

```
import java.util.*;


public class MergeSortedLists {
  public static void main(String[] args) {
    List<Integer> list1 = Arrays.asList(1, 3, 5, 7);
    List<Integer> list2 = Arrays.asList(2, 4, 6, 8);


    List<Integer> merged = new ArrayList<>(list1);
```

```
        merged.addAll(list2);

        Collections.sort(merged);

        System.out.println(merged);
    }
}
```

---

6. Find Intersection of Two Sets

Write a program to find the intersection of two sets of integers.

Input:

- Two sets of integers.

Output:

- A set that contains only the elements common to both input sets.

java

Sample Code

```
import java.util.*;

public class IntersectionOfSets {
    public static void main(String[] args) {
        Set<Integer> set1 = new HashSet<>(Arrays.asList(1, 2, 3, 4, 5));
        Set<Integer> set2 = new HashSet<>(Arrays.asList(3, 4, 5, 6, 7));

        set1.retainAll(set2);

        System.out.println(set1);
    }
}
```

---

7. Sort Map by Values

You are given a map of string keys and integer values. Write a program to sort the map by values.

Input:

- A map of strings and integers.

Output:

- The sorted map by values.

java

Sample Code

```
import java.util.*;


public class SortMapByValues {
    public static void main(String[] args) {
        Map<String, Integer> map = new HashMap<>();
        map.put("apple", 3);
        map.put("banana", 1);
        map.put("cherry", 2);


        List<Map.Entry<String, Integer>> list = new ArrayList<>(map.entrySet());
        list.sort(Map.Entry.comparingByValue());


        for (Map.Entry<String, Integer> entry : list) {
            System.out.println(entry.getKey() + ": " + entry.getValue());
        }
    }
}
```

---

8. Find Kth Largest Element in a List

Given a list of integers, find the Kth largest element in the list.

Input:

- A list of integers and an integer K.

Output:

- The Kth largest element in the list.

java

Sample Code

```
import java.util.*;


public class KthLargestElement {
   public static void main(String[] args) {
      List<Integer> list = Arrays.asList(3, 2, 1, 5, 6, 4);
      int k = 2;


      Collections.sort(list, Collections.reverseOrder());


      System.out.println(k + "th largest element: " + list.get(k - 1));
   }
}
```

---

9. Rotate a Deque

Given a Deque of integers, rotate it K times to the right.

Input:

- A deque of integers and an integer K.

Output:

- The deque after K rotations.

java

Sample Code

```
import java.util.*;


public class RotateDeque {
  public static void main(String[] args) {
    Deque<Integer> deque = new ArrayDeque<>(Arrays.asList(1, 2, 3, 4, 5));
    int k = 2;


    for (int i = 0; i < k; i++) {
      int last = deque.removeLast();
      deque.addFirst(last);
    }


    System.out.println(deque);
  }
}
```

---

10. Group Anagrams

Given an array of strings, group anagrams together.

Input:

- A list of strings.

Output:

- A list of lists where each list contains strings that are anagrams.

java

Sample Code

```
import java.util.*;


public class GroupAnagrams {
  public static void main(String[] args) {
    List<String> list = Arrays.asList("eat", "tea", "tan", "ate", "nat", "bat");
    Map<String, List<String>> anagramMap = new HashMap<>();

    for (String str : list) {
      char[] chars = str.toCharArray();
      Arrays.sort(chars);
      String sorted = new String(chars);
      anagramMap.computeIfAbsent(sorted, k -> new ArrayList<>()).add(str);
    }

    System.out.println(new ArrayList<>(anagramMap.values()));
  }
}
```

These questions will help you become familiar with collections in Java such as List, Set, Map, and Deque, and allow you to practice important operations like sorting, merging, and finding duplicates.