Name: Tayyab Kamran (24)

Roll Number: 22i-2505

Date: 19th September 2024

Weight: **5 Absolutes**

Software Construction and Development Quiz 1

Part 1: Dry Run (3 marks * 5)

```
List<Integer> numbers = new ArrayList<>();
numbers.add(10);
numbers.add(20);
numbers.add(30);

for (int i = 0; i < numbers.size(); i++) {
  numbers.set(i, numbers.get(i) * 2);
}
System.out.println(numbers);
```

20, 40, 60

```
class Animal {
  String name;
  Animal(String name) {
    this.name = name;
  }
}
class Dog extends Animal {
  Dog(String name) {
    super(name);
  }
  void bark() {
    System.out.println(name + " is barking!");
  }
}

Animal animal = new Dog("Buddy");
((Dog) animal).bark();
```

Buddy is barking

```
class GenericBox<T> {
  private T value;
  public void setValue(T value) {
    this.value = value;
  }
  public T getValue() {
    return value;
  }
}

GenericBox<Integer> intBox = new
GenericBox<>();
intBox.setValue(42);
System.out.println(intBox.getValue());
```

```
GenericBox<String> strBox = new
GenericBox<>();

strBox.setValue("Hello");

System.out.println(strBox.getValue());
```

42 ✓

Hello

```
try {
  int[] arr = {1, 2, 3};
  System.out.println(arr[3]);
} catch (ArrayIndexOutOfBoundsException e) {
  System.out.println("Array index out of
bounds.");
} finally {
  System.out.println("Finally block executed.");
}
```

Array index out of bounds.
Finally block executed.

```
class Parent {
  void show() {
    System.out.println("Parent show");
  }
}
class Child extends Parent {
  void show() {
    System.out.println("Child show");
  }
}

Parent obj = new Parent();
Child c = (Child) obj;
c.show();
```

✓

Parent Object cannot be typecasted into a Child Object. After Error remove output will be: Child show.

Part 2: Coding Problem (15 Marks)

Given an array of integers (which may contain duplicates), write a program to generate all the unique subsets of the array. Your program should not output any duplicate subsets, and the subsets should be printed in lexicographically sorted order.

Input: An array of integers, which may contain duplicates.

Output: A list of unique subsets, sorted in lexicographical order.

Example:

[1, 2, 3]

[], [1], [1, 2], [1, 2, 3]

Input:

[1, 2, 2]

Output:

[], [1], [1, 2], [1, 2, 2], [2], [2, 2]

Explanation:

- The input array has duplicates, but the output subsets should be unique.
- The output subsets should be sorted lexicographically (by set length, then elements).
- You must ensure that there are no duplicate subsets in the output.

```
import java.utils.ArrayList;
public class RemoveDuplicate {
    void ~~main~~ {
        ~~ArrayList myList = new ArrayList();~~

    ArrayList sortFunc( ArrayList myList)
    {
        ArrayList * newList = new ArrayList[];
        newList.push([]); ~~System.out.println("[],");~~
        for(int i=0; i < myList.size(); i++){
            for( int j= 0; j < myList.size(); i++) {
            if (i == j) && ~~!~~(newList.contains(myList[i]))
            { ~~myList~~newList.push( myList[i]);
              ~~System.out.println("[" + myList[i] + "]");~~
            } else { continue;}
            newList[i].push (myList[j]);
        }
    }
}
```

newList ←

(7+2)

```
public static void main ( String [] argc){
    ArrayList myList = new ArrayList();
    myList.push(1); myList.push(2); myList.push(2);
Arraylist new = sortFunc( myList);
    for( i=0; i< new.size() ; i++){
} sout ( new[i] );
```

77