## Quiz 8 Name: ___________________________

1. Write a method getStringWithNthOccurrences that accepts a List of strings as a parameter. This method returns the values that occur **at least** n times in the list, and false otherwise. **You should only traverse through the list once**. When the program is executed, it shows the following output:

```
[apple, durian]
[apple, pear, durian]
[]
[]
[]
```

```
import java.util.*;

public class Q1 {

   public static void main(String[] args) {
       {
           List<String> values = new ArrayList<>();
           values.add("durian");
           values.add("durian");
           values.add("durian");
           values.add("apple");  // appeared 3 times
           values.add("orange");
           values.add("apple");
           values.add("pear");
           values.add("apple");
           values.add("pear");
           List<String> result = getStringWithNthOccurrences(values, 3);
           System.out.println(result);
       }

       {
           List<String> values = new ArrayList<>();
           values.add("durian");
           values.add("durian");
           values.add("durian");
           values.add("apple");
           values.add("orange");
           values.add("apple");
           values.add("pear");
           values.add("apple");
           values.add("pear");
           // apple appeared 3 times, pear 2 times, durian 3 times.
           // orange appeared 1 time
           List<String> result = getStringWithNthOccurrences(values, 2);
           System.out.println(result);
       }
```

```
        {
            List<String> values = new ArrayList<>();
            values.add("durian");
            values.add("durian");
            values.add("durian");
            values.add("apple");
            values.add("orange");
            values.add("apple");
            values.add("pear");
            values.add("apple");
            values.add("pear");
            // none of the values appear 5 times
            List<String> result = getStringWithNthOccurrences(values, 5);
            System.out.println(result);
        }

        {
            List<String> values = new ArrayList<>();
            // none of the values appear 5 times
            List<String> result = getStringWithNthOccurrences(values, 1);
            System.out.println(result);
        }

        {
            List<String> values =null;
            // none of the values appear 5 times
            List<String> result = getStringWithNthOccurrences(values, 1);
            System.out.println(result);
        }
    }
}
```

2. **[ 4 marks ]** Given the following class:

```
public class Student {
    private int age;

    public Student(int age) {
        this.age = age;
    }

    public int getAge() {
        return age;
    }

    public void setAge(int age) {
        this.age = age;
    }

    @Override
    public String toString() {
        return "" + age;
    }
}
```

Draw the memory state diagram for the following program at the point of time when the program reaches line 12:

```
1    import java.util.ArrayList;
2    import java.util.List;
3
4    public class Quiz {
5       public static void doMagic(List<Student> everyone) {
6           Student a = everyone.get(0);
7           everyone.add(a);
8
9           a = new Student(3);
10          everyone.add(a);
11
12          //How does the memory state diagram look here?
13      }
14
15      public static void main(String[] args) {
16          List<Student> everyone = new ArrayList<>();
17          Student s1 = new Student(1);
18          everyone.add(s1);
19
20          doMagic(everyone);
21      }
22   }
```

## ANSWER

```
// Q1
public class Q1 {









}
```

**Answer (Q2):**