CS 1324 Spring 2021 Homework 13 Classes with Generics

Jordan McFadden

TOTAL POINTS

20 / 20

QUESTION 1

1 Question 1 **8 / 8**

✓ **- 0 pts Correct**

- 1 pts The main stack identifier is missing/incorrect

- 1 pts The main stack Contents is missing/incorrect

- 2 pts The heap identifiers are missing/incorrect

- 2 pts The heap Contents are missing/ incorrect

- 1 pts Size and/or Capacity are missing/incorrect in heap

- 0 pts ArrayList contents in heap are not in order

- 8 pts Incorrect/ Empty submission

QUESTION 2

6 pts

2.1 Question 2a **2 / 2**

✓ **- 0 pts Correct**

- 2 pts Incorrect/Empty submission

- 1 pts Incorrect parameters

- 1 pts Incorrect return type/ method name

2.2 Question 2b **2 / 2**

✓ **- 0 pts Correct**

- 2 pts Incorrect/Empty Submission

- 1 pts Incorrect parameters

- 1 pts Incorrect return type/ method name

- 0 pts Missing parameters

2.3 Question 2c **2 / 2**

✓ **- 0 pts Correct**

- 2 pts Incorrect/Empty Submission

- 1 pts Incorrect parameters

- 1 pts Incorrect return type/ method name

- 0 pts Missing parameters

QUESTION 3

4 pts

3.1 Question 3a **2 / 2**

✓ **- 0 pts Correct**

- 2 pts Incorrect/Empty Submission

- 1 pts new ArrayList with existing data is missing

- 1 pts List is not shuffled

3.2 Question 3b **1 / 1**

✓ **- 0 pts Correct**

- 2 pts Incorrect/Empty Submission

- 1 pts Collection methods are not used correcty

3.3 Question 3c **1 / 1**

✓ **- 0 pts Correct**

- 2 pts Incorrect/Empty Submission

- 1 pts Missing arguments

QUESTION 4

4 Everyone gets 2 free points because Dr. Trytten can't add well **2 / 2**

✓ **- 0 pts Correct**

Homework 13: Classes with Generics

CS 1323/4 Spring 2021

Name: Jordan McFadden

Student ID (usually 112-XXX-XXXX or 113-XXX-XXXX): 113502650

1. (8 points) Trace the code below in the given memory diagram. Since PDF files do not allow strikethroughs, separate words that are replaced with commas and put the new word on the right. For example, if "a" were replaced by "B" in the heap, the table would look as follows:

heap		
Identifier	**Address**	**Contents**
	500	"a", "B"

```
public class ArrayListTest {
     public static void main(String[] args) {
          ArrayList<String> ambroggio
               = new ArrayList<String>(8);//[1]
          ambroggio.add("The");
          ambroggio.add("sublimity");
          ambroggio.add(2,"of");
          ambroggio.add("eagle's");
          ambroggio.remove("of");
          ambroggio.add("flight");
          ambroggio.remove(1);
          ambroggio.set(1, "winged");
          ambroggio.add("purpose");
     }
}
```

[1] From a lovely poem by Luis Alberto Ambroggio. https://www.poetryfoundation.org/poems/150383/we-are-all-whitman-30-animal-song

main stack frame		
Identifier	**Address**	**Contents**
ambroggio	100	1000
	101	
	102	
	103	

heap		
Identifier	**Address**	**Contents**
0	1000	null, The
1	1001	null, sublimity, eagle's, winged
2	1002	null, of, eagle's, flight
3	1003	null, eagle's, null, flight, null, purpose
4	1004	null
5	1005	null
6	1006	null
7	1007	null
capacity	1008	8
size	1009	0, 1, 2, 3, 4, 3, 4, 3, 4
	1010	
	1011	
	1012	

2. (6 points) Write the **signature of the methods** described below. Do not write the methods.

a) The method determines whether or not an ArrayList<Integer> contains at most a given number of copies of a given int value. For example: this method would return true if the ArrayList that contains {1, 3, 5, 3, 1}, 1, and 2 were given as arguments. If that same ArrayList, 5 and 2 were given as arguments, the method would return false.

```
public static boolean isRepeated(ArrayList<Integer> list, int value, int timesRepeated)
```

b) The method returns a newly constructed ArrayList<String> that contains three given String values repeated as many times as necessary for a given size. For example: If the method was given the values "A", "B", and "C" and the given size was 5, the returned ArrayList<String> would contain {"A", "B", "C", "A", "C"}.

```
public static ArrayList<String> repeatedString(String first, String second, String third,
int size)
```

c) The method returns a newly constructed ArrayList<Integer> that contains the values in a given array of int values repeated as many times as necessary for a given size. For example: If the method was given an int array that contained {3, 5, 7} and the given size was 5, the returned ArrayList<Integer> would contain {3, 5, 7, 3, 5}.

```
public static ArrayList<Integer> repeatedInt(int[] array, int size)
```

3. (6 points) Use method(s) in the Collections class to write a **code fragment** to solve the problems below.

a) Take a given ArrayList<String> with reference data and create a new ArrayList<String> with the same values in random order. For example: if data contained {“b”, “a”, “c”}, the new ArrayList<String> might contain {“c”, “b”, “a”} after being randomized. The ArrayList<String> data should not be modified.

```
public static ArrayList<String> randomizeElements(ArrayList<String> data)
{
   ArrayList<String> result = new ArrayList<String>(data.size());
   for (int i = 0; i < data.size(); ++i)
{
     result.add(data.get(i));
}
     Collections.shuffle(result);
     return result;
}
```

b) Print out the range of values in an ArrayList<Integer> with reference list to the console. For example: If list contained {5, 3, 2, 1, 4, 9, 7, 3}, the printout should say “1 to 9”.

```
System.out.println(Collections.min(list) + " to " + Collections.max(list));
```

c) Swap the first and last values in an ArrayList<String> with reference list. For example: if list contained {“A”, “F”, “C”} initially, it should contain {“C”, “F”, “A”} after the operation.

```
Collections.swap(list, 0, list.size() - 1);
```