CMSC131 Practice Final Exam

December 11, 2017

Name: ______________________________

Preface: This is a final exam for Dr. Bill Pugh's & Dr. Ilchul Yoon's Fall 2017 CMSC131 course at the University of Maryland– College Park written by Shriraj Gandhi and Jonathan Saewitz, teaching assistants for the course. It is meant to be used for studying, and should not be considered a guide for what the exam will or will not cover — it should serve merely as a resource to use when studying. If you have any questions, feel free to email saewitz@terpmail.umd.edu.

Topics covered: Arrays, ArrayLists, Sets, Maps, Classes, Interfaces, Exceptions, Recursion

1. **Arrays**

   Write a function named `averageLength` that, given an array containing `String`s, returns the average length of the `String`s in the array.

   For example, with input `["I", "love", "pizza", "!"]`, return `2.75`

2. **ArrayLists**

   Write a function named `getSquares` that, given an `ArrayList` containing `Integer`s, returns an `ArrayList` with the square root of all numbers that are perfect squares. A perfect square is an integer that is the square of an integer (Wikipedia) (e.g. 36 is a perfect square because $6^2 = 36$).

   For example, with input `[30, 26, 25, 1, 3, 4]`, return `[5, 1, 2]`

3. **Sets**

After the following code is run:

```
List<Integer> a = new ArrayList<Integer>();
Set<Integer> s = new HashSet<Integer>();

for (int i = 0; i < 8; i++) {
    for (int j = 0; j < 3; j++) {
        a.add(i);
        s.add(i);
    }
}
```

How many elements are in `a`? ________

How many elements are in `s`? ________

When might you use a `Set` instead of a `List`?

4. **Maps**

(a) Write a function `biggestClass` that, given a `Map<String, Integer>` mapping courses to number of enrolled students, returns the course with the most students.

For example, with input `{"CMSC131"=332, "ENGL101"=3000}`, return `"ENGL101"`

(b) Consider a class `Salesperson` defined with `private` instance variables `String name and int revenueGenerated` which has methods `getName()` and `getRevenueGenerated()` that returns a Salesperson's name and revenue generated, respectively. Write a method named `consolidate` that, given an `ArrayList<Salesperson> sales`, creates and returns a `HashMap<String, Integer>` of `Salespersons` where each Salesperson appears only once with revenue aggregated.

For example, with input `[("Alice", 10), ("Bob", 30), ("Alice", 20), ("Charlie", 40), ("Bob", 10)]`, return `{"Alice"=30, "Bob"=40, "Charlie"=40}`

(c) Why might `map_name.keySet()` return a `Set`, rather than a `List`?

5. **Interfaces & Classes**

Write a class `Dinner` that implements the `Meal` interface:

```java
public interface Meal {
    // Print “eat x” where x is the name of the Meal object
    public void eat();

    // Returns the name of the Meal object
    public String getName();

    // Changes the name of the Meal
    // Must throw IllegalArgumentException if newName contains the word "moldy" anywhere
    public void setName(String newName);
}
```

A Dinner must have two properties: a `name` and a count of the number of times eaten (you can call this `numEaten`). `numEaten` is increased every time the `eat()` method is called.

The `Dinner` class must have

- The two fields mentioned above, privately encapsulated
- A constructor that takes a `String name` and initializes `numEaten` to 0 and a copy constructor
- A getter for `numEaten`
- A static method `getShorter` that takes two objects of type `Dinner` as parameters and returns the object with the shorter name
- An `equals` method that returns `true` if a passed `Object` is a `Dinner` and has the same `name` and `numEaten` and `false` otherwise

6. **Exceptions**

   You are starting to work for a software development company named Epsilon Productions. The programmer before you wrote a method named `epsilon()` that takes a `HashMap<String, Integer>` and prints meaningful statistical data based on it. However, he made some mistakes and his method throws many exceptions, leading to complaints from customers of your software. Your job is to write a method named `safeEpsilon()` with the following header:

   ```
   public static void safeEpsilon (HashMap<String, Integer> map)
   ```

   Your method will simply call the epsilon method by passing the map, but your code must stop gracefully in the case of exceptions.

   - If there is a `ConcurrentModificationException`, print "Please don't change the map while we are using it"
   - If there is a `NullPointerException`, print "Please provide a valid map"
   - If there is any other `Exception`, print "Please call us for assistance"
   - You must always print "Thank you for choosing Epsilon Production Inc."

7. **Recursion**

You cannot use any loops for the following problems.

(a) Write a method named `getSum` that, given an array of `int`s named `arr`, recursively finds the sum of its elements and returns it.

(b) Write a method named `power` that, given an n and p, recursively computes the value of n to the p power. For example, `power(3, 2)` returns 9 (compute n^p — you may assume that $n \geq 0$ and $p \geq 0$).