CS 201, Spring 2023 — May 4th — Exam 3 Name: _____

Question 1. [6 points] State a big-O upper bound on the worst-case running time of the following method, where the problem size N is the number of elements in the ArrayList passed as the parameter. Briefly justify your bound.

```
public static<E extends Comparable<E>>
Collection<E> getSortedCollection(ArrayList<E> list) {
    TreeSet<E> treeSet = new TreeSet<E>();
    for (E elt : list) {
        treeSet.add(elt);
    }
    return treeSet;
}
```

Question 2. [6 points] State a big-O upper bound on the worst-case running time of the following method, where the problem size N is the number of elements in the LinkedList passed as a parameter. Briefly justify your bound.

```
public static void removeEven(LinkedList<Integer> list) {
    Iterator<Integer> i = list.iterator();
    while (i.hasNext()) {
    Integer val = i.next();
        if (val % 2 == 0) {
            i.remove();
        }
    }
}
```

Question 3. [6 points] What output is printed by the following code?

```
Queue<Integer> q = new LinkedList<Integer>();
Stack<Integer> s = new Stack<Integer>();

for (int i = 1; i <= 4; i++) {
    q.add(i);
    s.push(i);
}
for (int i = 1; i <= 4; i++) {
    System.out.println(s.pop());
    System.out.println(q.remove());
}
```

Question 4. [6 points] Briefly explain under what circumstances adding additional threads will likely not speed up (and could even slow down) the performance of an algorithm that can be divided into N independent tasks.

Question 5. [6 points] Complete the following code to update the inventory count for a given make of car. Note: There are 3 separate lines of code to fill in.

```
public Integer updateCarInventory(
    TreeMap<String, Integer> carInventory, String car, Integer count) {

    // if car is not in the inventory...
    if (!carInventory.containsKey(car)) {
        // ...add it to the map (add CODE here)


    }
    else {
        // otherwise, update the car's count in the map (add CODE here)


    }
    // return the updated count for the car (add CODE here)


}
```

Question 6. [10 points] What output is printed by the following code?

```
import java.util.Arrays;
import java.util.Iterator;
import java.util.List;

public class Foo {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		List<Integer> list = Arrays.asList(5,6,4,3,8,1);
		int result = 1;
		Iterator<Integer> i = list.iterator();

		while(i.hasNext()) {
			if ((i.next()) %2 ==0) {
				result *= i.next();
				}
			else if(i.next()%2 != 0) {
				result += i.next();
			}
			System.out.println("Result: " +result);
		}
		System.out.println("Done");


	}

}
```

Programming Questions

To get started, use a web browser to download the zipfile as specified by your instructor. Import it as an Eclipse project using File → Import... → General → Existing Projects into Workspace → Archive file.

Important: You may use the following resources:

- The textbook
- The lecture notes posted on the course web page
- Your previous labs and assignments

Do not open any other files, web pages, etc.

Question 7. [30 points] A "tall" character is one of the lower-case letters b, d, f, h, k, l, or t, or an upper-case letter. Implement the `countTall` method so that it returns the number of "tall" characters in the string passed as the parameter.

Hint: You may use a loop.

Hints:

- `s.charAt(i)` returns the character at index `i` in the string `s`
- `s.substring(start, end)` returns the substring of `s` from index `start` (inclusive) to index `end` (exclusive)
- `s.substring(start)` returns the substring of `s` from index `start` (inclusive) to the end of the string
- `Character.isUpperCase(c)` returns true if a character `c` is an upper case letter, false otherwise

Unit tests are provided in `Q7Test`. Make sure the unit tests pass.

Question 8. [30 points] Complete the implementation of the `Stats` class. The idea is that each time the `updateStats` method is called, information is being provided about the performance of one player in one particular softball game, specifically the player's name (`player`), the game number (`game`), and the number of runs scored by that player (`numRuns`). The `updateStats` method should keep track of the total number of runs by each player (over all games), and the total number of runs in each game (over all players).

Once a `Stats` object has been populated with data, the `getRunsForPlayer` and `getRunsForGame` methods can be used to get the total number of runs for a particular player or game, respectively.

The stats.csv file contains the raw data. The Q8.readStats method creates a new Stats object, reads the data from stats.csv, and calls updateStats for each data record in the file. You should not modify either stats.csv or Q8, but you may find it helpful to refer to them. (To open stats.csv, right click and choose Open with → Text editor.)

The Q8Test class has unit tests. Make sure all of the unit tests pass.

Hints:

- Use a map of String (player) to Integer (runs) for keeping track of the number of runs for each player
- Use a map of Integer (game) to Integer (runs) for keeping track of the number of runs for each game
- updateStats should create or update one entry in each map
- The getRunsForPlayer and getRunsForGame methods should look up the desired number of runs in the appropriate map