CS401 Lab 3

This lab is to be completed individually.
This lab is for you to understand runtime complexity. This lab will provide you with some practice on understanding the runtime complexity of programs.

What to do?

Part-1

Write down complexity in terms of Big-O for methodA, methodB and methodC
Note:

There's no need to run the code. Run by your eyes and analyze by your brain.
You should write down the complexity in terms of Big-O in a word or PDF document.
Explain why you conclude your answer.

```
package lab3;

import java.util.Date;
import java.util.Timer;
public class ComplexityExperiment {
public static void main (String args []) throws InterruptedException {
run_method_A(250);
run_method_A(500);
run_method_A(1000);
run_method_A(2000);
System.out.println();
run_method_B(250);
run_method_B(500);
run_method_B(1000);
run_method_B(2000);
System.out.println();
run_method_C(250);
run_method_C(500);
run_method_C(1000);
run_method_C(2000);
System.out.println();
}
public static void run_method_A(int n) { // n must be bigger than 0 always
       int i = 0;
       double start, end;
       start = System.currentTimeMillis();
       methodA(n);
       end = System.currentTimeMillis() - start;
       System.out.println("methodA(n = " + n + ") time = " + end + "ms");
}
public static void run_method_B(int n) { // n must be bigger than 0 always
       int i = 0, loop = 1000;
       double start, end;
       start = System.currentTimeMillis();
       for (i = 0; i < loop; i++) {
              methodB(n);
       }
       end = System.currentTimeMillis() - start;
       System.out.println("methodB(n = " + n + ") time = " + end/loop + "ms");
}
public static void run_method_C(int n ) { // n must be bigger than 0 always
       int i = 0;
```

```
	double start, end;
	start = System.currentTimeMillis();

	methodC(n);
	end = System.currentTimeMillis() - start;
	System.out.println("methodC(n = " + n + ") time = " + end + "ms");
}
public static void methodA(int n) {
	int i = 0;
	int j = 0;
	int total = 0;
	while (i < n) {
		while (j < n) {
			total++;
			j++;
		}
		i++;
	}
}
public static void methodB(int n) {
	int i = 0;
	int j = 0;
	int k = 0;
	int total = 0;
	while (i < n) {
		while (j < n) {
			while (k < n) {
				total++;
				k++;
			}
			k = 0;
			j++;
		}
		j = 0;
		i++;
	}
}
public static void methodC(int n) {
	int i = 0;
	int j = 0;
	int k = 0;
	int total = 0;
	j = n;
	while (k < n) {
		while ((j = j / 2) > 0) {
			for (i = 0; i < 100 * n; i++) {
				total++;
			}
		}
		k++;
	}
}
}
```

Part-2

You need to implement Selection Sort (chapter 1 describes) such that it takes an array of integer and perform sort routines using Selection Sort. You can only use the textbook's algorithm but no other online or reference article resources. You can use chapter 1 codes OR you can design your own but must use the selection sorting steps described in chapter 1.
You should also write down complexity in terms of Big-O of your Selection Sort program. In addition, you should capture the execution time of your Selection Sort program.

What type of data? Any type of data: integer values, float values, or strings (up to your own convenience).

Test Case: Generate 10,000+ random values and sort them out. (you can use the "random" function from the math library)

In an event that your execution time is "0 ms", try nanoseconds

Make sure that your code is well documented i.e., in-line comments with a simple README would be ideal. For instance, every function and complex portion of code should have comments that describe what it does.

What to turn in?

1. Complexity.pdf: This file contains your answers to part1 and complexity of your code in part 2
2. SelectionSort.java: This file contains the source code of your selection sort implementation. Put your test code into the main function of your SelectionSort class.
3. Your program's outputs/screenshots in a PDF file
4. Runnable JAR file.
5. README file to demonstrate how your program works. Include a command to determine how to run the JAR file.

Please submit on Blackboard before the assigned due date