Faculty of Arts & Sciences
Department of Computer Science
CMPS 212—Intermediate Programming with Data Structures
Spring 2013-2014 - Quiz 3

Question1. Given two queues with some of their standard operations (`add`, `remove`, `isEmpty`, `size`), implement a stack, `StackUsingQueues`, with the following standard operations: `push`, `pop`, `isEmpty`, `size`. A skeleton of the class `StackUsingQueues` is given below. **You are not allowed to use any collection other than the two queues defined in the class**.

```
public class StackUsingQueues {
	private Queue<Integer> q1;
	private Queue<Integer> q2;

	public StackUsingQueues() {
		// TODO
	}

	public int size() {
		// TODO
	}

	public int pop() {
		// TODO
	}

	public void push(int value) {
		// TODO
	}

	public boolean isEmpty() {
		// TODO
	}

	public static void main(String[] args) {
		StackUsingQueues s = new StackUsingQueues();
		s.push(1);
		s.push(5);
		s.push(7);
		s.push(-2);

		System.out.println(s.pop());
		System.out.println(s.pop());
		System.out.println(s.size());
		System.out.println(s.isEmpty());
	}
}
```

Sample output:

```
-2
7
2
false
```

Question2. Given a text as input, you are to develop a class, `ConcordanceWords`, that computes the concordance for the words in the text. A ***concordance*** consists of each word in the text followed by a sequence of pairs (`line`, `count`) where: (1) `line` indicates the line number where the word occurred and, (2) `count` indicates the number of times the word appeared in that line. All concordances of the input file should be sorted alphabetically according to the words, and for each word the sequence of pairs should be sorted according to the line number (check sample run below). Create three methods (see sample `main` method below):

1. A constructor that takes the name of a text file as input and build the appropriate data structure(s).
2. A method that returns the number of occurrences of a word in a given line.
3. A `toString` method to display all the concordances of the file.

For a sample `input.txt` (*assume that all letter are lowercase and there are no punctuation marks*) that contains:

```
i cannot imagine how nice a science as computer science is it is so so nice
however after i was taught by not so so nice instructors
i do not like computer science anymore as a science
i want a major where instructors are so so nice and so so flexible
i like though the computer challenges in computer science so so much
```

and a sample main method:

```
public static void main(String[] args) {
       Concordance concordance = new Concordance("input");
       System.out.println(concordance);
       System.out.println(concordance.getOccurrences("so", 5)); // 2: "so" appears two times in the 5th line
       System.out.println(concordance.getOccurrences("hello", 1)); // 0: "hello" does not appear in the 1st line
}
```

The output is:

```
a: (1,1) (3,1) (4,1)
after: (2,1)
and: (4,1)
anymore: (3,1)
are: (4,1)
as: (1,1) (3,1)
by: (2,1)
cannot: (1,1)
challenges: (5,1)
computer: (1,1) (3,1) (5,2)
do: (3,1)
flexible: (4,1)
how: (1,1)
however: (2,1)
i: (1,1) (2,1) (3,1) (4,1) (5,1)
imagine: (1,1)
in: (5,1)
instructors: (2,1) (4,1)
is: (1,2)
it: (1,1)
like: (3,1) (5,1)
major: (4,1)
much: (5,1)
nice: (1,2) (2,1) (4,1)
not: (2,1) (3,1)
science: (1,2) (3,2) (5,1)
so: (1,2) (2,2) (4,4) (5,2)
taught: (2,1)
the: (5,1)
though: (5,1)
want: (4,1)
was: (2,1)
where: (4,1)

2
0
```

Submission Instructions

- Your exam submission must consist of a single zip archive named s#_exam3_*netid* that contains your properly commented Java files. We will compile and run your programs.