CS2110 Homework 6
Concurrency and Recursion

Due via WebCAT on April 19, 2019 at 11:30pm

Collaboration/Plagiarism Policy: This assignment is **individual work**. Sending, receiving, posting, reading, viewing, comparing, or otherwise copying any part of course assignments or solutions is not allowed except when explicitly permitted in assignment instructions. This includes solutions from other students in the course, past or present. See the course syllabus for penalties for collaboration policy violations.

Learning Objectives

- Practice your Java programming.
- Experiment with locking and concurrency.
- Understand recursion.

Grading Rubric

- 50% Recursion Functionality
  - 10% `palindrome` method
  - 10% `reverseString` method
  - 10% `handShakes` method
  - 20% `ackermann` method
- 20% Concurrency Functionality
  - 15% Locking in `PrintQueue`
  - 5% Queueing and waiting in `CardCreator`
- 30% Coding Style for both sections
  - 10% Correct indentation
  - 10% Naming Conventions
  - 10% Well-commented
    - Comment at the head of each file (see "Style" section below)
    - Comment at the head of every major (and new) method
    - General in-line commenting as needed to highlight interesting logic
- Up to -20% late penalty as defined in the syllabus. The latest submission or resubmission will count as the official submission time.

Recursion

In the single class called `Recursion`, write the body of each of the following static methods. Also include in the main method of the class at least two tests per method. Be sure to include a comment for each test that states the expected output and the actual output. Remember to comment the rest of your code as well! A few examples are given below each problem; please test others as well!

1. Write a static recursive method, called `palindrome(String s)` that takes in a string and returns `true` if the string is a palindrome (i.e., "abba" or "racecar") and `false` otherwise.

Input	Return
`abba`	`true`
`racecar`	`true`
`smith`	`false`

2. Write a static recursive method, called `reverseString(String s)` that takes in a string and returns the reverse of the string.

Input	Return
`racecar`	`racecar`
`CS2110`	`0112SC`
`I love CS!`	`!SC evol I`

3. There are n people in a room, where n is an integer greater than 1. Each person shakes hands once with every other person. What is the total number, $h(n)$, of handshakes? Write a recursive function, called `handShakes(int n)`, to solve this problem. It should return an int for the total number of handshakes for n people.

Method Call	Return
`handShakes(0)`	`0`
`handShakes(3)`	`3`
`handShakes(6)`	`15`

4. Ackermann's function is a recursive mathematical algorithm that can be used to test how well a computer performs recursion (It is also important mathematically as well, as it is the first known function that is totally computable but not primitive recursive). Write a method `ackermann(long m, long n)`, which solves Ackermann's function and returns a `long`. The function is given by the following:

$$A(m,n) = \begin{cases} n+1 & \text{if } m = 0 \\ A(m-1,1) & \text{if } m > 0 \text{ and } n = 0 \\ A(m-1, A(m,n-1)) & \text{if } m > 0 \text{ and } n > 0. \end{cases}$$

Note when testing this method: keep m and n small, as $A(4,2)$ is an integer of $19,729$ digits!

Method Call	Return
`ackermann(0,0)`	`1`
`ackermann(2,0)`	`3`
`ackermann(3,4)`	`125`

Concurrency

The MarkAntonyms card company has asked you to implement their printer queue for printing greeting cards. You must implement `CardCreator Runnable` tasks for the company that will "think up" greeting card slogans to be printed, which will be read from their respective "inspiration" files. Each line of the inspiration file (for example `roses.txt`) contains one new greeting card slogan. The `CardCreator` will then submit that slogan into the `PrintQueue`. The printer will remove the slogans from the queue and print them out in a card-formatted design.
For this assignment, you must implement or improve the following methods in `PrintQueue`, ensuring the proper locking mechanisms are in place:

- `public String dequeue()` : Remove the head element off the queue and return it. (The printer will call this to get the next job to print).
- `public void enqueue(String greeting)` : Add the greeting given by the String parameter onto the end of the print queue. The creators will use this method to add greetings to the print queue.
- `public void turnOff()` : Turn off the print queue (i.e. set a `boolean` field denoting that the PrintQueue is no longer accepting jobs). The last running creator should turn off the queue so that the printer thread(s) know to stop at some point.
- `public boolean isOn()` : Returns true if the `PrintQueue` is still accepting jobs, false if it has been "turned off."

Note that for your `PrintQueue`, you will need a `Lock` and `Condition` to ensure that two `CardCreators` do not attempt to add to the queue at the same time and for the Printer to be able to wait for a job to be queued.
You must also finish the implementation of `CardCreator`, which reads input line-by-line from the `filename` given as a parameter to the constructor and add that to the `PrintQueue`. We have provided the code required to read from the file; you must write the code necessary to enqueue the new greeting card slogan onto the `PrintQueue` and then sleep for 1 second (i.e., 1000 ms) before submitting the next slogan. (`CardCreators` must rest between slogan creation so as not to face creative burnout or writers block.) When the last `CardCreator` thread has finished executing, they must turn off the `PrintQueue`, which will stop the printer thread and end your program. (For this part, you will need a counter of running threads inside the `CardCreator` class, like a former in-class activity, and you will want to signal the printer which may be waiting on the Condition of an empty queue).

Submission Information

Style: Follow the CS2110 Coding Style Guide, posted under Resources on Collab. This includes:

- Correct naming conventions, including appropriate camelCasing and TitleCasing.
- Comment each file, with a block at the top of the file denoting assignment information and comments for each field and method of your classes. You should also comment portions of your code that may be difficult to follow. We would like you to get into the habit of commenting your code. This adds to the readability of your code which contributes to "good quality" code.
- Use correct indentation. Eclipse makes this easy: select all code and choose "Correct Indentation" or Control-I (Windows/Linux) or Command-I (Mac).
- Do not put your classes into a package. (If you don't know what this means, don't worry about it.)

- If two methods share identical logic, you should factor that out into a separate method (a "helper" method).

Submission: You must submit on Web-CAT. Submit all files (`Recursion.java`, `PrintQueue.java`, `CardCreator.java`, `Printer.java`, and `GreetingCardCreator.java`) by "Exporting" that project from Eclipse. They must have those names exactly (including capitalization). Make sure you do not submit the `.class` files. **When you submit, Web-CAT will NOT give feedback. You should therefore perform main method testing to ensure your results match our examples.** If you fail our tests your grade will be based on how many tests you pass. If you fail a test, Web-CAT will give you a small hint about what was being tested. If your code does not compile, you will receive a 0 and Web-CAT will try to point out why it was unable to compile your code. (Usually, this is because something was spelled differently than specified above. Be sure to follow all naming conventions for getters and setters.)

To submit on Collab you need to zip up your files. If you are not sure how to do this you can follow the instructions below:

1. Right-click your `src` folder in Eclipse
2. Select `Export...`, `General`, `Archive File`
3. Check the `.java` files you want to submit
4. Browse to a save location you can find again
5. Finish
6. In a browser, go to `web-cat.cs.vt.edu`
7. Log in
8. Click the Submit button next to an assignment
9. Browse and select the `.zip` file you just created
10. Upload
11. Confirm