

CSE 7345

Assignment 3

Due: In class: Nov 12; Distance: Nov 14

Note: to make it easier to test your program, do not use packages.

Part A.

Write a class called A3A.java with a main method that:

- reads a file (whose name is passed in as a parameter) containing text. The file should contain at least 100 lines. Store each line of text as an entry in an **ArrayList**.
- starts a millisecond clock (use `System.currentTimeMillis()`)
- launches an instance of a thread class called `SearchThread1` with a constructor that takes a string parameter and that implements a `run` method that searches through the List and find lines that contain the search string.
- when the thread finds an entry with one or more matches, it should append the entry in the ArrayList with:
`//match: <text> found`
-
- when the thread has searched through the ArrayList, it should print on `System.out`:
`SearchThread1 search for <text> found: <numberLinesWithMatch>`
-
- when the thread has completed, the caller should display the amount of time in milliseconds it took for the thread to complete its task from the call to start until thread termination.
- Use `System.currentTimeMillis()` for timing
- Design your solution so that the thread instance obtains access to the List defined in A3A.java.
- Include in your `readme.txt` file, your strategy for making the List available to the thread and include an alternative strategy you rejected and why.
- Include in `readme.txt`, the output of your program.

Part B.

Observer/Observable/Threads

Write a class **A3B.java** that is **Observable**.

- The class should create an ArrayList using the same file as part A.
- The class should contain an instance variable called **stringVal** and a method called **setStringVal** which is used to set the value of **stringVal**.
- The setting of **setStringVal** should trigger any Observers that have been registered with the Observable.
- Write a class called **MyObserver** that can act as an Observer and that reacts to notification by MyObservable by examining the string for commas. If there are commas, split the string and launch a SearchThread2 class instance thread for each token.
- When the thread finds a line with one or more matches, it should append the entry in the ArrayList with:
`//match: <text> found`
- Use Thread.yield() to give other threads a chance to do work
- When the thread has completed searching through the ArrayList, it should print on System.out:
`Thread search for <text> found: <numberMatches>`
- When all threads have completed, display the total time for all threads and the average time per thread.
- Include in readme.txt, the output of your program.
-

Part C.

- ArrayList is not thread-safe, since multiple threads may interfere with each other during writing and reading. However, concurrency problems will be silent.
- Write A3C.java which is the same as A3B.java except for the fact that it uses a synchronized wrapper from the Collections library to wrap the ArrayList. This will create a thread-safe List.
- Use:
`public static ArrayList list = new ArrayList();`
- `public static List synchList = Collections.synchronizedList(list);`
- When all threads have completed, display the total time for all threads and the average time per thread.
- Is there a performance difference? Explain? (in readme.txt)

- It is possible to get your program to throw an Exception due to concurrency issues. Run your program several times and see if you can get it to crash. Explain why the program should crash and if you do get it to crash, include the error msg in your readme.txt file.
- Include in readme.txt, the output of your program.
-
-
-

Part D. A3D.java.

- Modify your code from part C, creating A3D.java. After the threads have been launched, add a loop that iterates over the list (it does not matter what it does, just that it should use an iterator).
- Get your program to throw a ConcurrentModificationException.
- Include the message that appears in your readme.txt.

Submit:

A zip file called A4.zip with:

- all your source code
- all class files (no packages please)

Very Important: a readme.txt organized by sections: A-D.

•

