The goal of this assignment is to gain experience working with concurrent programming via multiple threads and thread synchronization. You’ll write an implementation that simulates producer and consumer processes. The simulated consummation process will take far more time to complete than the production procedure, causing the machine to run out of memory if the threads aren’t synchronized such that no more than a specified number of units are produced prior to being consumed.

1. [20] Create a new class with a public static void main method that first creates a java.util.concurrent.ArrayBlockingQueue with enough capacity for 100,000 String entries.

2. [30] Create a new class which implements Runnable to simulate the producer. When executed, this process should add 2,000,000 random Strings (hint: see the UUID class) to the above queue, waiting until there is space in the queue if necessary (i.e., using the ArrayBlockingQueue put() method). Print your progress once every 1000 Strings produced.

3. [30] Create another new class which implements Runnable to simulate the consumer. When executed this process should consume Strings from the queue, keeping track of the overall max String found (i.e., using the String compareTo() method). The process should continue as long as there are Strings in the queue or the producer hasn’t finished (i.e. so the consumer doesn’t quit if the queue happens to be empty before the producer finishes). Call Thread.sleep(10) (i.e., sleep for 10 ms) between each String comparison to ensure that the consumer takes longer to execute than the producer. Print your progress once every 1000 Strings consumed. Print the total number of Strings consumed and the max String found when the process completes.

4. [20] Use a java.util.concurrent.ExecutorService to execute your producer and at least two consumers concurrently. Shut down the service after starting the two processes.

The following is output for producing 10000 Strings with a queue of size 1000 and 2 consumers:

produced: 1000
produced: 2000
produced: 3000
consumer 1 consumed: 1000
consumer 2 consumed: 1000
produced: 4000
produced: 5000
consumer 2 consumed: 2000
consumer 1 consumed: 2000
produced: 6000
produced: 7000
consumer 2 consumed: 3000
consumer 1 consumed: 3000
produced: 8000
produced: 9000
consumer 2 consumed: 4000
consumer 1 consumed: 4000
produced: 10000
done producing! 10000 produced
consumer 1 consumed: 5000
csumer 1 done consuming! 5000 consumed
consumer 1 max String: ffe53515-0ccc-49ed-a61f-e9ed10d21188
csumer 2 consumed: 5000
csumer 2 done consuming! 5000 consumed
csumer 2 max String: ffeca27c-d5f2-4c15-9635-14dd909cbaa4

Zip the Assignment7 folder in your Eclipse workspace directory and upload the .zip file to Canvas (see Assignments section for submission link).