CSCE 3613 Operating Systems Homework #4, ver. 4.7
Producer-Consumer Programming Assignment

40 points

Instructions

- Name your Java program "ProducerConsumer.java".
- Make sure that your code can be run from the **command line** as "java ProducerConsumer 20 10 5", which means that it will run of 20 seconds, create 10 producer threads, and 5 consumer threads.
- You may debug the program on other platforms but you must compile and run the final program on the machine turing.uark.edu.
- Put your name and UA ID in a comment at the top of your code.
- Create one ZIP file of your assignment consisting of the following and upload it to Blackboard:
    - All source code
    - A readme.txt file that describes your program
    - An output.txt file that is the output of your program from "java ProducerConsumer 20 10 5" and should look similar to the example below

Implement the Producer-Consumer Problem programming assignment at the end of Chapter 5 in the textbook **using the programming language Java** instead of the described C code. **You must use Java with Threads instead of Pthreads.** A brief overview is below.

This program should work as follows: The user will enter on the command line the *sleep time*, *number of producer threads*, and the *number of consumer threads*. One example of the Java application from the command line could be "java ProducerConsumer 20 10 1". **The code must use threads, mutexes, and semaphores. Note that the number of producer and consumer threads does not have to be equal.** The `ProducerConsumer` application will create the specified number of producer and consumer threads and then sleep letting both producer(s) and consumer(s) place 100 random integers into a bounded buffer that has a finite number of slots to hold data. Each producer and consumer thread will also choose a random amount of time to sleep using `Random()` (make sure to look up what kind of random number `Random()` provides) and `Thread.sleep()` that is *less than the given sleep time* as it places 100 random integers into this one bounded buffer. **Set the sleep time of each producer and consumer thread to be a random amount of time between 0 to 0.5 seconds because we will test it over a 20-second period.** Each producer and consumer thread will print the random number that it placed in the bounded buffer or removed from the bounded buffer to the screen. **Set the bounded buffer equal to five**.

The producer will 100 times go through the cycle of sleeping a random amount of time, creating a random integer, inserting it into the bounded buffer of size 5, and then printing the random integer.

The consumer will 100 times go through the cycle of sleeping a random amount of time, consuming an item from the bounded buffer, and then printing the random integer.

Use a mutex variable named `mutex` to protect the buffer and counting semaphores named `empty` and `full` so that a consumer does not try to remove an item from an empty buffer (`empty`) and a producer does not try to place an item into a full buffer (`full`). At the beginning initialize `empty = 5` and `full = 0` before creating the producer and consumer threads. Below is the usage and an example of executing this application.

Usage.

java ProducerConsumer `<sleep time> <producer threads> <consumer threads>`

The following command creates 5 producer threads and 1 consumer thread and lets them run for 20 seconds. **Notice the pattern!**

```
$ java ProducerConsumer 20 5 1

Using arguments from command line
Sleep time = 20
Producer threads = 5
Consumer threads = 1

Producer produced 6595
        Consumer consumed 6595
Producer produced 97415
        Consumer consumed 97415
Producer produced 50798
Producer produced 25626
Producer produced 95610
Producer produced 88100
Producer produced 14130
        Consumer consumed 50798
Producer produced 95845
        Consumer consumed 25626
Producer produced 71052
        Consumer consumed 95610
Producer produced 64862
        Consumer consumed 88100
Producer produced 26986
        Consumer consumed 14130
Producer produced 17117
        Consumer consumed 95845
Producer produced 33333
        Consumer consumed 71052
Producer produced 28460
```

```
        Consumer consumed 64862
Producer produced 11266
        Consumer consumed 26986
Producer produced 46761
        Consumer consumed 17117
```

Rubric

The program will be graded using the following rubric.

- Program does not compile (-40).
- Program compiles but does not run (-20).
- The program will be run with several different parameters. Example parameters are below. If any of this set create deadlock or errors, (-10). In addition, if we find other parameters that cause deadlock or errors, (-5).
  - `java ProducerConsumer 20 1 1`
  - `java ProducerConsumer 20 5 1`
  - `java ProducerConsumer 20 1 5`
  - `java ProducerConsumer 20 5 5`
  - `java ProducerConsumer 20 20 20`
- Program correctness (-10).
- Minor mistakes (-5).
- If name and ID not in a comment in the program (-1).