Semaphores and Monitors in Java

For this problem, you'll get a chance to solve a simple synchronization problem using semaphores and monitors in Java. This is just toy problem, but it will give you a chance to try out these synchronization mechanisms. Also, I've tried set up the programs you're completing so that their behavior will give you a pretty good idea of whether your solutions are correct.

On the course homepage, you'll find two simple multi-threaded Java programs, SmileySemaphore.java and SmileyMonitor.java. These programs are supposed to print out a series of randomly-generated ASCII-art smiley faces (you knowIn, like this :-)). In their current state, the programs just print out random-looking sequences of smiley face characters. You're going to fix this by just adding synchronization code. When you're done, the output should be 3000 smiley faces, one per line (well, really some are neutral faces and some are sad).

For SmileySemaphore.java, add code using java.util.concurrent.Semaphore. You shouldn't need to change any of the existing code or what it does, just create semaphores and add calls to acquire() (probably acquireUninterruptibly()) and release() to constrain the timing of thread execution. When it's time to print a pair of eyes, any of the eye-printing threads should be able to do it. Likewise for printing the nose and the mouth. Your synchronization code will just guarantee that you always get a pair of eyes, followed by a nose and then a mouth.

For SmileyMonitor.java I've already started the monitor you need, MyMonitor. You will need to add state to the monitor (i.e., fields), additional logic inside the monitor's member functions (i.e., checking/updating the state) and calls to wait()/notify() (or, probably, notifyAll()) to guarantee only ASCII art faces are printed (eyes, then nose then mouth). For example, if it's not time to print a pair of eyes, threads may have to wait in the printEyes() until it's OK for one of them to go ahead.

Once I got my programs working, they only printed faces (lots of them) and I got a lot of different types. In fact, I got at least one example of every possible smiley face every time I ran the program. You can try out these programs (either SmileySemaphore or SmileyMonitor) in a few different ways to see if you're getting what you'd like:

```
# Just print all the smile faces generated
$ java SmileySemaphore

# Get wc to count faces, you should get 3000
$ java SmileySemaphore | wc

# Just see the unique (different) faces you print
$ java SmileySemaphore | sort | uniq

# Count unique smiley faces printed
$ java SmileySemaphore | sort | uniq | wc
```

When your two programs are working, submit them to the exercise_11 assignment on Moodle.