**Title:** "Program Assignment One: Multi-threaded Programming in Java"
Points: 100
Due Date: Friday January 24, 2014 by 11:59pm WebCourses time

Objectives: To practice programming an application with multiple threads of execution and synchronizing their access to necessary shared objects.

Description: In this programming assignment you will simulate the pump management system for a water treatment plant like the one depicted here:
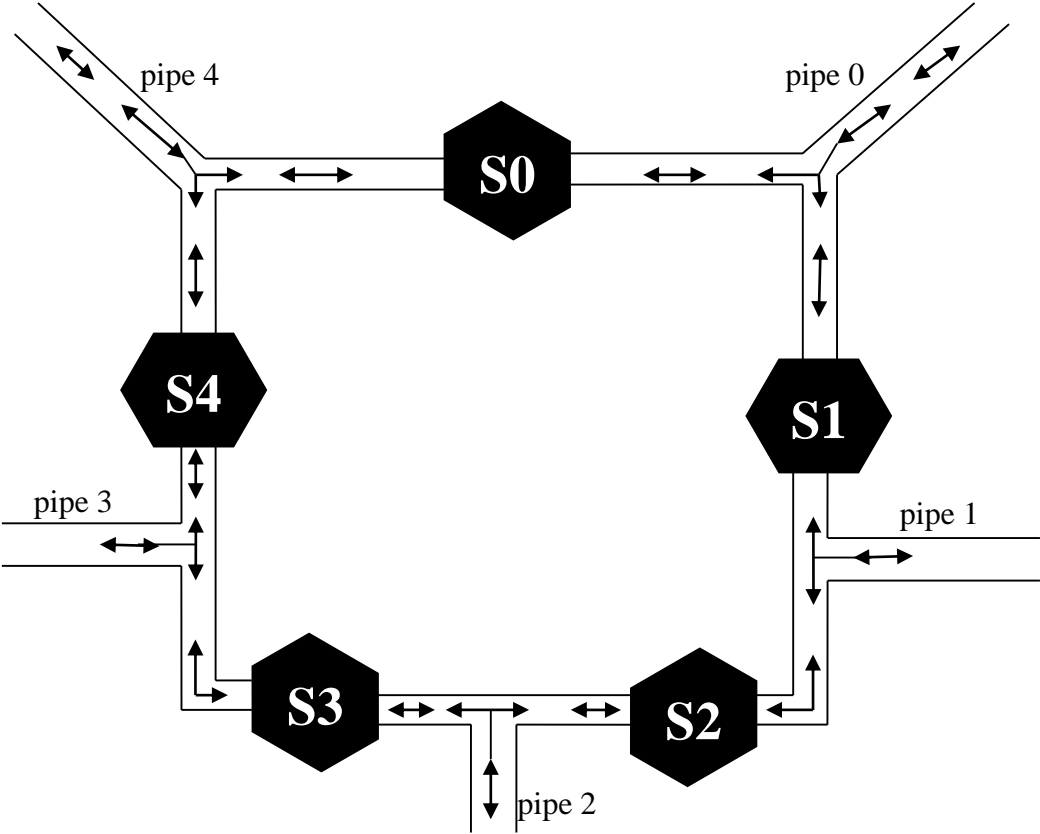
This example water treatment plant has five pumping stations (S0 – S4), each of which has an input and output pipe connecting to lines (P0 – P4) that go elsewhere in the plant. Resources were limited when building the plant so each pipe going to the rest of the facility must be shared between two pumping stations. Since each station simultaneously needs an input and output connection to function, access to the shared lines must be strictly regulated. Flow direction in not important in our simulation.

You have been hired to design a simulator for a new water treatment plant being built with the same design, but possibly fewer/more stations. You are to implement this simulator in Java and have each station function in its own thread. A station's workload is the amount of time that a station needs to have access to the input and output pipes during the simulation. Once a station is granted access to

both pipes it calls its `doWork()` method during which it will attempt to flow water down each pipe (of course it must verify that it has access and isn't in conflict with another station). After the flow-in and flow-out methods are run, the workload of the station is reduced by 1 and the station releases both pipes and signals waiting stations that the pipes are available. After executing a flow and releasing its pipes, a station should sleep for some random period of time. A station's thread stops running when its workload reaches 0.

Restrictions:

1. Your source files should begin with comments containing the following information:

```
/*
        Name: <your name goes here>
        Course: CNT 4714 Spring 2014
        Assignment title: Program 1 – Multi-threaded programming in Java
        Date:  January 24, 2014

        Class:  <name of class goes here>
*/
```

2. Do **not** use a monitor to control the synchronization in your program (i.e., do not use the Java synchronize statement).

Input Specification:

Your program must initially read from a text file (`config.txt`) to gather configuration information for the simulator. The first line of the text file will be the number of stations to use during the simulation. Afterwards, there will be one line for each station. These lines will hold the amount of work each station needs to process. Only use integers in your configuration file, decimals will not be needed.

Output Specification:

Your simulator must output the following text to let the user know what the simulator is doing in each of these situations:

1. An input connection is set:
   ```
   Station X: In-Connection set to pipe n
   ```

2. An output connection is set:
   ```
   Station X: Out-Connection set to pipe n
   ```

3. A stations workload is set:
   ```
   Station X: workload set to n
   ```

4. A station is granted access to a pipe:
   ```
   Station X: granted access to pipe n
   ```

5. A station is releasing access to a pipe:
   ```
   Station X: released access to pipe n
   ```

6. A station has completed its workload:
   ```
   * * Station X: Workload successfully completed. * *
   ```

7. A station successfully flows water down a pipe:
    ```
    Station X: successfully flows on pipe n
    ```
**Deliverables:**

Submit the following items via WebCourses no later than 11:59pm January 24, 2014.
   (1) All of your `.java` files.
   (2) A copy of a sample execution of your program, i.e., the output produced by your simulator (this should just be a text file). In Eclipse you can redirect console output to a file, do this and include a copy of the output file produced by your program. This should be the entire output file produced by your simulation.

Additional Information:

An example execution of the simulator (with `config.txt` containing 3 2 3 4)

config.txt

```
3
2
3
4
```
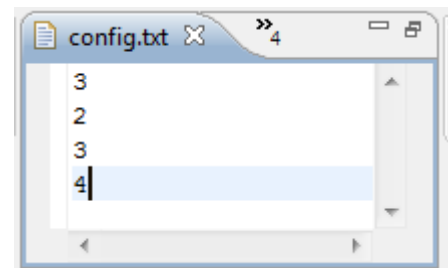
* * * MAIN SIMULATION BEGINS * * *

Station 0: In connection set to pipe 0.
Station 0: Out connection set to pipe 2.
Station 1: In connection set to pipe 1.
Station 0: Workload set to 2.
Station 2: In connection set to pipe 2.
Station 2: Out connection set to pipe 1.
Station 2: Workload set to 4.
Station 1: Out connection set to pipe 0.
Station 2: granted access to pipe 2.

 * * * MAIN SIMULATION ENDS * * *

Station 0: granted access to pipe 0.
Station 0: released access to pipe 0.
Station 2: granted access to pipe 1.
Station 1: Workload set to 3.
Station 2: successfully flows 2.
Station 2: successfully flows 1.
Station 0: granted access to pipe 0.
Station 0: released access to pipe 0.
Station 0: granted access to pipe 0.
Station 0: released access to pipe 0.
Station 2: released access to pipe 2.
Station 2: released access to pipe 1.
Station 1: granted access to pipe 1.
Station 2: granted access to pipe 2.
Station 2: released access to pipe 2.
Station 1: granted access to pipe 0.

Station 1: successfully flows 1.
Station 1: successfully flows 0.
Station 2: granted access to pipe 2.
Station 2: released access to pipe 2.
Station 2: granted access to pipe 2.
Station 2: released access to pipe 2.
Station 1: released access to pipe 1.
Station 0: granted access to pipe 0.
Station 1: released access to pipe 0.
Station 0: granted access to pipe 2.
Station 1: granted access to pipe 1.
Station 0: successfully flows 0.
Station 1: released access to pipe 1.
Station 0: successfully flows 2.
Station 1: granted access to pipe 1.
Station 1: released access to pipe 1.
Station 1: granted access to pipe 1.
Station 1: released access to pipe 1.
Station 0: released access to pipe 0.
Station 0: released access to pipe 2.
Station 2: granted access to pipe 2.
Station 0: granted access to pipe 0.
Station 2: granted access to pipe 1.
Station 0: released access to pipe 0.
Station 2: successfully flows 2.
Station 2: successfully flows 1.
Station 2: released access to pipe 2.
Station 2: released access to pipe 1.
Station 2: granted access to pipe 2.
Station 2: granted access to pipe 1.
Station 2: successfully flows 2.
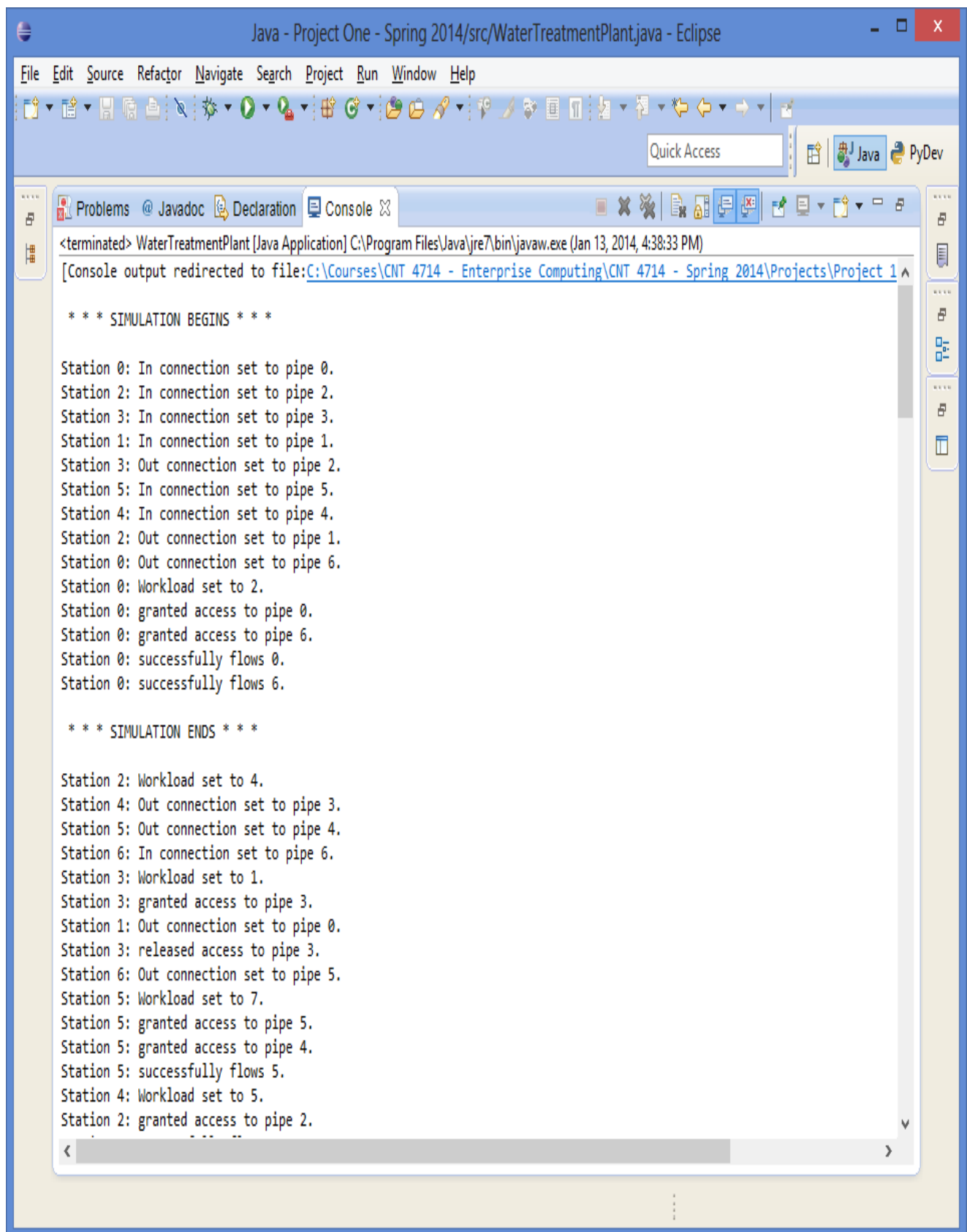**Station 2: successfully flows 1.**
Station 2: released access to pipe 2.
Station 2: released access to pipe 1.
Station 1: granted access to pipe 1.
Station 2: granted access to pipe 2.
Station 1: granted access to pipe 0.
Station 1: successfully flows 1.
Station 2: released access to pipe 2.
Station 1: successfully flows 0.
Station 1: released access to pipe 1.
Station 1: released access to pipe 0.
Station 0: granted access to pipe 0.
Station 1: granted access to pipe 1.
Station 0: granted access to pipe 2.
Station 1: released access to pipe 1.
Station 0: successfully flows 0.
Station 0: successfully flows 2.
Station 1: granted access to pipe 1.

Station 1: released access to pipe 1.
Station 1: granted access to pipe 1.
Station 1: released access to pipe 1.

* * Station 0: workload successfully completed. * *

Station 0: released access to pipe 0.
Station 0: released access to pipe 2.
Station 2: granted access to pipe 2.
Station 2: granted access to pipe 1.
Station 2: successfully flows 2.
Station 2: successfully flows 1.

* * Station 2: workload successfully completed. * *

Station 2: released access to pipe 2.
Station 2: released access to pipe 1.
Station 1: granted access to pipe 1.
Station 1: granted access to pipe 0.
Station 1: successfully flows 1.
Station 1: successfully flows 0.

* * Station 1: workload successfully completed. * *

Station 1: released access to pipe 1.
Station 1: released access to pipe 0.

Actual simulation run in Eclipse (console output not redirected in this example).