

COMP 182 Project 5: Process Scheduling

(15 points)

Due: 20:00 Friday May 4, 2007

Overview: You will be writing a simulation of jobs running on a computer. Every job will have an arrival time and a processing time. The arrival time is the earliest possible time the job can start. The processing time is the total amount of time that it takes to complete the job. Your computer will be simple: once a job starts it is run to completion (there is no swapping/preemption of processes). This project is a bit different from earlier projects. Further description will be given during the Wednesday 4/18 lab.

Requirements: You will write 6 classes and a file called Analysis.txt.

The first class will be called MyJob. It will contain two floats arrivalTime and processingTime. Instances of this will represent each job.

The second class will be called MyEvent. It will contain a float time and a boolean jobArrival.

The third class will be MyPQ which is a priority queue of MyEvent (ordered by time).

The fourth and fifth classes will be MyQ which is a queue of MyJob and MyStack which is a stack of MyJob.

Your final class will be a MyDriver5. This is the portion of the project which will be discussed 4/18. Rough description: you will create 2 instances of MyPQ, you will create a random number (10-10000) job arrival events (the time for each will be between 0 and 10000) and enqueue each into both priority queues. For each priority queue you will perform the following operations: as long as the priority queue is not empty dequeue an event. If the event is a job arrival and the processor is not busy then create a job completion event (jobArrival=false) with time (5-10 greater than the current time). If the event is a job arrival and the processor is busy then create a job and add it to the stack (queue). If the event is a job completion and the stack (queue) is not empty then remove a job from the stack (queue) and create a new job completion (5-10 greater than the current time). If the event is a job completion and the stack (queue) is empty then do nothing.

Once you have finished programming and debugging your code, add code to determine the longest time between the arrival of a job and its completion. Your analysis file should discuss which of the two structures (stack/queue) is better.

Suggestions: If you plan out your classes in detail before coding you will drastically reduce both the length of the code and the amount of debugging. Get the program working with just one of the data structures first. Start as soon as possible and finish programming at least a couple days early (to allow intelligent testing for the Analysis.txt file).

Submission: When you finish (prior to the deadline), log in to WebCT, go into the Assign Submits area, and upload/submit only the files 6 java files and the 1 ascii file. It would be a very good idea to submit the project at least several hours before the deadline (you can resubmit later if necessary). I suggest that when you submit, you request a confirmation email from the server. If you later want to claim the server lost your submission, I will want to see this email.

Grading: This assignment will be graded based upon coding style (comments, indenting, structure, useful var names, etc), whether it compiles and runs properly, how closely your output matches the

sample output, whether you followed directions (don't submit class files, don't change the names of the classes, etc), the quality of your tests and explanation of results, and the degree to which your program handles exceptions.

Sample Interaction:

```
jnoga@nogahome:$ java MyDriver5
```

```
Creating 5137 job arrivals.
```

```
Queue: The maximum time a job took from arrival to completion was 847.
```

```
Stack: The maximum time a job took from arrival to completion was 1237.
```