COMP 482 Project 2: Minimize Number Late Algorithm Testing

Due: Wednesday November 9 at 2355

Points: 30 points possible

Overview: The MinimizeNumberLate problem has input of a set of jobs (each with a processing time and a deadline $\{(p_i, d_i\}_{i=1}^{N}\}$ that need to be scheduled on a single processor. A solution will place the jobs in some order and will be judged based on the **number** of jobs that do not complete by their deadline. Note this problem is similar to MinimizeMaximumLateness, but has a different goal. Your program will test the performance of several greedy algorithms.

Details: The input will come from a file called input.txt which will be placed in the same directory as your java file. The first line of the file will have a single integer value N which will be the number of jobs. The next N lines will represent a job, each with 2 values p_i and d_i. See the sample input below for examples.

There are many ways that you could schedule the jobs. You will test the following algorithms:

- EarliestDeadlineFirst which sorts the jobs by deadline d_i and schedules them in that order.
- ShortestJobFirst which sorts the jobs by processing time p_i and schedules them in that order.
- LeastSlackFirst which sorts the jobs by their slack $d_i - p_i$ and schedules them in that order.

Your program will output the number of jobs that each algorithm does not finish on time.

You can discuss ideas for the algorithm to be used with anyone and consult any source (books, internet, etc). However, for this project, you are expected to write the code on your own with limited or no assistance from the professor, no assistance from others, and limited or no assistance from other sources (books, internet, etc). To clarify, you can seek assistance in understanding the task and how it can be solved, but "your code" should be written by you: not written by others, not copied from others, not copied from books/internet.

Picky, but required specifications: Your project must:

- be submitted via canvas.
- consist of 1 or more dot-java files (no class files, zip files, input files or other files should be submitted). Each file must have your name and which project you are submitting as comments on the first 2 lines.
- not be placed into any package (for the java pedants, it must be in the default package).
- have one file called Project2.java.
- compile using the command 'javac Project2.java'.
- run using the command 'java Project2',
- accept input from a file called input.txt in the same directory as the java file(s) formatted precisely as described above.
- be submitted on time (early and multiple times is fine).

If your project fails any of the above, you will receive a zero (recall that each of you may replace 1 and only 1 project that receives a zero this semester). If your project meets the requirements above then it will be graded on whether it:

- is designed and formatted reasonably (correct indentation, no excessively long lines, no excessively long methods, has useful method/variable names, etc) and
- accomplishes the goal of the project. In other words, the output should be the correct answer, computed in a valid way, formatted correctly.

Sample execution: If input.txt contains

```
4
2 2
1 3
4 7
3 4
```

then the output should be

```
EDF 2
SJF 3
LSF 3
```

because the schedules:

- EDF = [(2,2),(1,3),(3,4),(4,7)] has jobs (3,4) and (4,7) late.
- SJF = [(1,3),(2,2),(3,4),(4,7)] has jobs (2,2), (3,4) and (4,7) late.
- LSF = [(2,2),(3,4),(1,3),(4,7)] has jobs (3,4), (1,3), and (4,7) late.

If input.txt contains

```
5
2 3
4 7
6 15
1 6
7 14
```

then the output should be

```
EDF 1
SJF 1
LSF 2
```

because the schedules:

- EDF = [(2,3),(1,6),(4,7),(7,14),(6,15)] has job (6,15) late.
- SJF = [(1,6),(2,3),(4,7),(6,15),(7,14)] has job (7,14) late.
- LSF = [(2,3),(4,7),(1,6),(7,14),(6,15)] has job (1,6), (6,15) late.

If input.txt contains

```
4
3 5
7 10
1 11
5 16
```

then the output should be

```
EDF 0
SJF 1
LSF 0
```

because the schedules:

- EDF = [(3,5),(7,10),(1,11),(5,16)] has all jobs on time.

- SJF = [(1,11),(3,5),(5,16),(7,10)] has job (7,10) late.
- LSF = [(3,5),(7,10),(1,11),(5,16)] has all jobs on time.

Stray Thoughts:
I suggest you finish and submit your project at least several days in advance. This way you have time and opportunity to ask any last questions and verify that what you upload satisfies the requirements. There is nothing wrong with working on the project for a day and uploading your code, working for another day and uploading your improved code, ..., working for another day and uploading your final version. In fact there are advantages: you have a fairly reliable place that keeps your versions and even if you get busy at the last moment you have still uploaded your best version.
Your project should be written and understood by you. Helping or receiving help from others to figure out what is allowed/required is fine, but copying code is not. Significant shared source code indicates that you either did not write/understand what you submitted or you provided code to another (allowing them to submit code they did not write/understand). Both are academic dishonesty.