University of Toronto at Scarborough
CSCC73 - Algorithm Design and Analysis

Assignment #1: Greedy Scheduling

Due: September 22, 2018 at 11.59pm This exercise is worth 5% of your final grade.

Warning: Your electronic submission on MarkUs affirms that this exercise is your own work and no one else's, and is in accordance with the University of Toronto Code of Behaviour on Academic Matters, the Code of Student Conduct, and the guidelines for avoiding plagiarism in CSCC73. Late assignments will not be accepted. If you are working with a partner your partners' name must be listed on your assignment and you must sign up as a "group" on MarkUs. Recall you must not consult **any outside sources except your partner, textbook, TAs and instructor**.

In this assignment we will look at variations of the *Interval Scheduling Problem.*

1. In this question we consider a variation of the scheduling problem in which we are given n distinct jobs j_i each of which can be completed independently. Each job has two stages to be completed. The first stage of each job is the bottleneck in the sense that there is only one resource that can handle it and for job j_i takes f_i time. The second stage is less complicated and can be completed in parallel with other jobs as there are many resources available to complete this portion of the job. We will denote the time required to complete stage 2 of job j_i as s_i. [10]

   For example, consider a factory where there is a unique machine that must be used in the production of each product before it can go through the final stages with an employee. Assume that there are enough employees (i.e., n employees) to handle the second stage of all the jobs at the same time. Give a polynomial-time algorithm to return a *schedule* that orders the jobs for stage 1 so that the *completion time* of stage 2 for all the jobs is minimized (ie, the total time for all jobs to be completed is minimized). Prove your algorithm is correct and explain it's complexity.

2. We now consider another variation of the scheduling problem in which we have a processor that operates 24 hours a day, each day. Jobs requests are submitted to run daily on the processor (ie, every day). Each job j_i has a start time s_i and finish time f_i. The jobs selected to run on the processor run every day during their required interval. The processor can handle one job at a time. The goal is to accept as many jobs as possible. [10]

   Given a list J of n jobs, return the largest sublist of J such that no two jobs conflict. Note that job intervals may begin before midnight and end after. You may assume that no two jobs start (or end) at the same time. An example of such a list J, with $n = 4$ is:

   $$(1am, 12pm), (7pm, 2am), (4pm, 4am), (11am, 5pm)$$

   An optimal solution could be the two jobs (7pm, 2am) and (11am, 5pm). Your algorithm should run in $O(n^2)$ time. Prove your algorithm is correct and explain the complexity.

[Total: 20 marks]