CS 325- Homework Assignment 4

Problem 1: *(10 points)* **Scheduling jobs with penalties**:

For each $1 \le i \le n$ job j_i is given by two numbers d_i and p_i, where d_i is the deadline and p_i is the penalty. The length of each job is equal to 1 minute and once the job starts it cannot be stopped until completed. We want to schedule all jobs, but only one job can run at any given time. If job i does not complete on or before its deadline, we will pay its penalty p_i.

a) Design a greedy algorithm to find a schedule such that all jobs are completed and the sum of all penalties is minimized. Provide a verbal description and pseudocode.
b) What is the running time of your algorithm?

Problem 2: *(5 points)* **CLRS 16-1-2 Activity Selection Last-to-Start**

To solve the activity-selection problem (16.1 CLRS) , suppose that instead of always selecting the first activity to finish, we instead select the last activity to start that is compatible will all previously selected activities. Describe how this approach is a greedy algorithm, and prove that it yields an optimal solution.

Problem 3: *(15 points)* **Activity Selection Last-to-Start Implementation**

You may use any language you choose to implement the activity selection last-to-start algorithm described in problem 3. If you use a sorting function in your program you will need to write that yourself. You can use any of the sorting algorithms from previous homework assignments or implement another sorting algorithm. Include the time to sort in your overall running time.

In Canvas, submit a verbal description of your algorithm, pseudocode and analysis of the theoretical running time.

You do not need to collected experimental running times.

The program should read input from a file named "act.txt". The file contains lists of activity sets with number of activities in the set in the first line followed by lines containing the activity number, start time & finish time.

Example act.txt:

```
11
1 1 4
2 3 5
3 0 6
4 5 7
5 3 9
6 5 9
7 6 10
8 8 11
9 8 12
```

```
10 2 14
11 12 16
3
3 6 8
1 7 9
 2 1 2
```

In the above example the first activity set contains 11 activities with activity 1 starting at time 1 and finishing at time 4, activity 2 starting at time 3 and finishing at time 5, etc. The second activity set contains 3 activities with activity 3 starting at time 6 and finishing at time 8 etc. The activities in the file are not in any sorted order.

Your results including the number of activities selected and their order should be outputted to the terminal. For the above example the results are:

Set 1
Number of activities selected = 4
Activities: 2 4 9 11

Set 2
Number of activities selected = 2
Activities: 2 1

***Note**: There is an alternative optimal solution for Set 1. Since activities 8 and 9 have the same start time of 8, a2 a4 a8 a11 would be an alternative solution. Your program only needs to find one of the optimal solutions. For either solution the activities differ from the solution presented in the text which uses the earliest-finish time criteria.*

Submit a copy of all your files including the txt files and a README file that explains how to compile and run your code in a ZIP file to TEACH. We will only test execution with an input file named act.txt.