CMSC 651, Analysis of Algorithms, University of Maryland, Fall 2013
Mid-term exam, due at the beginning of class on October 22, 2013

Instructions: (i) Submit a written exam, or email to Khoa by the deadline. (ii) You can use your own notes and Emo Welzl's lecture-notes. No other material is permitted, including the Web, books, etc. (iii) This mid-term is to be worked on by each student separately. Discussions with anyone is disallowed. Write your solutions neatly. (iv) Please write and sign the following pledge: "I pledge on my honor that I have not given or received any unauthorized assistance on this examination." (iv) *Good luck!*

1. Answer only the first question ("What is the expected number of comparisons ...") of Exercise 3.6 from Emo Welzl's notes. **(10 points)**

2. We are given a connected, undirected graph $G = (V, E)$ with a cost for each edge. We are also given an integer k, where $1 \leq k \leq n-1$; we aim to find a minimum-cost forest in G with exactly k edges. Give a polynomial-time algorithm for this problem, and prove its correctness. **(15 points)**

3. We have a single server on which to schedule n tasks, one-by-one. Task i requires some positive processing time of p_i on the server; each task i also has a positive "weight" w_i. Time starts at 0, and we schedule the tasks one-by-one, each of them to completion. Given a schedule, let F_i denote the finish-time of task i in it. (For instance, if we schedule the tasks as $5, 2, 7, ...$ with $p_2 = 4$, $p_5 = 2$, and $p_7 = 3$, then $F_5 = 2$, $F_2 = 2 + 4 = 6$, $F_7 = 2 + 4 + 3 = 9$, etc.) The *cost* we pay for a schedule is $\sum_{i=1}^{n} w_i F_i$. Give a polynomial-time algorithm to find a schedule of minimum cost, and provide a proof of correctness. (**Hint:** Considering examples with $n = 2$ will help.) **(15 points)**

4. We have two servers X and Y each with some idle cycles available; we have a program that can be run on these idle cycles, and switched between the two servers, as follows. Our program has a large number of computational "steps". We have n time units, numbered 1 to n. For each time unit i, server X has announced that it will have free cycles to run $x_i > 0$ steps, and server Y has announced that it will have free cycles to run $y_i > 0$ steps. Our scheduling problem is as follows: for each i, we can schedule our program on X (in which case we get a "payoff" of x_i), or on Y (in which case we get a payoff of y_i). We want to maximize total payoff.

As stated, this problem is easy: for each i, schedule on X if $x_i \geq y_i$, and on Y if $x_i < y_i$. But we have a catch: if we decide to switch servers, we need to spend one full time-unit doing so, in which we get no payoff. An example schedule with $n = 11$ is

X X Switch Y Y Y Switch X Switch Y Y,

in which the total payoff is $x_1 + x_2 + y_4 + y_5 + y_6 + x_8 + y_{10} + y_{11}$.

Give a polynomial-time algorithm to find just the total payoff of an optimal schedule (i.e., one that maximizes total payoff; you do not have to output the actual optimal schedule, just its payoff). The numbers x_i, y_i are all given to you. Prove the correctness of your algorithm, and specify its runtime. **(20 points)**