Final Exam — Introduction to Algorithms (CS 300)

May 21, 2012, 10:00 to 12:00

- Before you start: Write your name and student number on *every page* of your exam sheet.
- This is a closed book exam. You are not allowed to consult any book or notes.
- To ensure a quiet exam environment, we will not answer questions during the exam. If you think there is a mistake in the question, explain so on your answer sheet, and use common sense to answer the question.
- The questions have to be answered in *English*. Write clearly!

Problem 1: (30 pts) Let X be a set of n intervals on the real line. We say that a set P of points *stabs* X if every interval in X contains at least one point in P. Describe and analyze an algorithm to compute a *smallest* set of points that stabs X. Assume that your input consists of two arrays `L[1..n]` and `R[1..n]`, representing the left and right endpoints of the intervals in X.

Your algorithm should be as efficient as possible. As always, you must prove that your algorithm is correct.

You will receive 8 points if you write "I don't know" and nothing else.

Problem 2: (40 pts) This week is the K-Pop (KAIST-Pop) dancing contest you have been training for this entire semester!

You have the list of the n songs that the judges will play during the contest, in chronological order. You know all the songs, all the judges, and your own dancing ability extremely well. For each integer k, you know that if you dance to the kth song on the schedule, you will be awarded exactly `Score[k]` points, but then you will be physically unable to dance for the next `Wait[k]` songs (that is, you cannot dance to songs $k+1$ through $k + \texttt{Wait[k]}$).

The dancer with the highest total score at the end of the night wins the contest, so you want your total score to be as high as possible. Describe and analyze an efficient algorithm to compute the maximum total score you can achieve. The input to your algorithm is the pair of arrays `Score[1..n]` and `Wait[1..n]`.

You will receive 10 points if you write "I don't know" and nothing else.

Problem 3: (30 pts) Consider the two following decision problems:

- SubsetSum: Given a set X of positive integers and a positive integer t, does X have a subset whose elements sum up to t?
- Partition: Given a set Y of positive integers, can Y be partitioned into two disjoint subsets whose sums are equal?

(a) (15 pts) We want to show that SubsetSum is easier than Partition, that is that SubsetSum $\preceq$ Partition.

To prove this, we have to give a polynomial-time algorithm.

What is the input and output for this algorithm, and what function is it allowed to call as a subroutine?

You will receive 4 points if you write "I don't know" and nothing else.

(b) (15 pts) Now prove that SubsetSum $\preceq$ Partition.

You will receive 4 points if you write "I don't know" and nothing else.