Due: Friday, Feb 3, 2017, 11am on MarkUs

You will receive 20% of the points for any (sub)problem for which you write "I do not know how to answer this question." You will receive 10% if you leave a question blank. If instead you submit irrelevant or erroneous answers you will receive 0 points. You may receive partial credit for the work that is clearly "on the right track."

1. (20 pts) You are given an infinite array A that contains n integers in the first n positions (indexing starts with 1) and ∞ in all other positions. You do *not* know the value of n. Your task is to design an iterative algorithm that finds the value of n using $O(\log n)$ accesses to array A. Asymptotically slower solutions receive the grade of 0.
   (a) Describe your algorithm in plain English (maximum 5 short sentences).
   (b) Describe your algorithm in pseudocode.
   (c) For each loop in your algorithm, state a useful loop invariant (without proof). State the corresponding termination condition(s) and how correctness follows from the termination condition(s).
   (d) Argue that the running time is $O(\log n)$ (as measured by the number of accesses to array A). Mention the running time of each logical block of your algorithm.

2. (20 pts) You are given an array A of n images. Some of these images might be identical. For $i \neq j \in [n]$ you can invoke a comparison procedure that returns whether the images $A[i]$ and $A[j]$ are identical or not. This procedure is denoted by $A[i] == A[j]$. Design a divide and conquer algorithm to decide whether there is an image that appears more than $n/2$ times in A using $O(n)$ invocations of the comparison procedure. Solutions using asymptotically more invocations of the comparison procedure receive the grade of 0.
   (a) Describe your algorithm in plain English (maximum 5 short sentences).
   (b) Describe your algorithm in pseudocode.
   (c) Provide a concise argument of correctness of your algorithm.
   (d) State the recurrence of the number of invocations of the comparison procedure (do not forget the base case).

3. (20 pts) You are given two arrays D (of positive integers) and P (of positive reals) of size n each. They describe n jobs. Job i is described by a deadline $D[i]$ and profit $P[i]$. Each job takes one unit of time to complete. Job i can be scheduled during any time interval $[t, t+1)$ with t being a positive integer as long as $t + 1 \leq D[i]$ and no other job is scheduled during the same time interval. Your goal is to schedule a subset of jobs on a single machine to maximize the total profit - the sum of profits of all scheduled jobs. Design an efficient greedy algorithm for this problem.
   (a) Describe your algorithm in plain English (maximum 5 short sentences).
   (b) Describe your algorithm in pseudocode.
   (c) Formally prove correctness of your greedy procedure. You may use the framework introduced in class - argue by induction that the partial solution constructed by the algorithm can be extended to an optimal solution.

(d) Analyze the running time of your algorithm (in terms of the total number of operations).

4. (20 pts) You are given an array A of $n \geq 3$ points in the Euclidean 2D space. The points $A[1], A[2], \ldots, A[n]$ form the vertices (in clockwise order) of the convex polygon P. A triangulation of P is a collection of $n-3$ interior diagonals of P such that the diagonals do not intersect except possibly at the vertices. The total length of a triangulation of P is the sum of the lengths of the $n-3$ interior diagonals used to form that triangulation. Design an efficient dynamic programming algorithm to find the total length of a triangulation with the minimum total length.

   (a) Describe the semantic array.

   (b) Describe the computational array. Don't forget the base case.

   (c) Justify why the above two arrays are equivalent.

   (d) What is the running time of your algorithm (as measured by the total number of operations)?

5. (20 pts) You are a manufacturer of chocolate goods. You start with a large rectangular chocolate bar that consists of $W \times H$ tiles arranged into W rows and H columns. You have a machine that can make a horizontal or a vertical break along tile edges. There are n possible goods that you can manufacture. Good i is described by w_i, h_i and p_i. This means that manufacturing good i requires a rectangular chocolate bar consisting of exactly $w_i \times h_i$ tiles (w_i rows and h_i columns). The value of p_i is how much profit you can make from good i. You can manufacture the same good more than once. What is the maximum overall profit that you can get starting with the large rectangular chocolate bar, performing a series of break operations, and manufacturing goods? Design an efficient dynamic programming algorithm. Note: W, H, w_i, h_i are positive integers and the p_i are reals.

   (a) Describe the semantic array.

   (b) Describe the computational array. Don't forget the base case.

   (c) Justify why the above two arrays are equivalent.

   (d) What is the running time of your algorithm (as measured by the total number of operations)?