

The grading policy for this homework is as follows: If you leave a question blank, you receive 1 point for that question. If you answer a question, the question will be graded on a scale from 0 to 5. This homework has five questions.  
You do not need to rewrite the question or copy down pseudo-code that was presented in class.

1. The dynamic programming algorithm for finding the length of the maximum total length set of non-overlapping jobs is:

MaxLengthSched( $S = [(s_1, f_1), (s_2, f_2), \dots, (s_n, f_n)]$ )

1. Sort jobs so that  $f_1 \leq f_2 \leq \dots \leq f_n$ .
2. Calculate last[ $j$ ] for  $j = 1, 2, \dots, n$
2.  $L[0] = 0$
3. For  $j = 1$  to  $n$   
     $L[j] = \max\{L[\text{last}[j]] + (f_j - s_j), L[j - 1]\}$
4. Return  $L[n]$

Recall that last[ $j$ ] is the largest index less than  $j$  of a job that doesn't overlap job  $j$  (or 0 if no such job exists).

We can obtain three other algorithms by modifying step 1. For each of the following proposed replacements for step 1, either: write "works" if the resulting algorithm always produces the length of an optimal (maximum total length) schedule for input  $S$  (no proof is necessary), or give an input for which the resulting algorithm fails to produce the optimal length. (Note: The algorithm may fail by reporting a length that is smaller or larger than the optimal length.)

- (a) 1. Sort jobs so that  $f_1 \geq f_2 \geq \dots \geq f_n$ .
  - (b) 1. Sort jobs so that  $s_1 \leq s_2 \leq \dots \leq s_n$ .
  - (c) 1. Sort jobs so that  $s_1 \geq s_2 \geq \dots \geq s_n$ .
2. Suppose you want to travel down the Mississippi River by canoe. You don't own a canoe but you can rent them at  $n$  different cities along the river. We'll number these cities in downstream order from 1 (your starting point) to  $n$  (your ending point). For each pair of cities  $i, j$  where  $i < j$  there is a price  $p_{ij}$  to rent a canoe from city  $i$  to city  $j$ . Given this set of prices, find the cheapest rental cost to travel from city 1 to city  $n$ . Note that you cannot paddle upstream. (There is an  $O(n^2)$ -time solution.)
  3. (Exercise 6.1 in Algorithms by Dasgupta, Papadimitriou, and Vazirani) A *contiguous subsequence* of a list  $S$  is a subsequence made up of consecutive elements of  $S$ . For instance, if  $S$  is

5, 15, -30, 10, -5, 40, 10

then 15, -30, 10 is a contiguous subsequence but 5, 15, 40 is not. Give a linear time algorithm for the following task:

*Input:* A list of numbers  $a_1, a_2, \dots, a_n$ .

*Output:* A contiguous subsequence of maximum sum (a subsequence of length zero has sum zero).

For the preceding example, the answer would be 10, -5, 40, 10, with a sum of 55.

(*Hint:* For each  $j \in \{1, 2, \dots, n\}$ , consider contiguous subsequences ending exactly at position  $j$ .)