FirstName LastName
StudentID acct@ucsc.edu

CMPS 102 — Winter 2019 – Homework 4

Three problems, 28 points, due 11:50 pm Wednesday March 6th, read the *Homework Guidelines*

1. (10 pts) Let B(n) be the number of different binary search trees containing the n keys 1, 2, . . . , n. For this problem you will develop a dynamic programming algorithm that, given n, calculates B(n). Note that B(1) = 1 since there is only one one-node binary tree. For two nodes B(2) = 2 (either node can be root, and there is only one binary search tree consistent with each choice).

   (a) (1 pt) Calculate by hand B(3).

   (b) (1 pt) How many n = 6 key binary trees are there with keys $\{1, 2, 3, 4, 5, 6\}$ and key 3 at root (so the left subtree has two nodes, and the right one has 3 nodes) are there?

   (c) (3 pts) Construct a recurrence for B(n), I expect something with a sum. Include boundary conditions in your recurrence; what is a convenient boundary value for B(0)? Briefly justify (formal proof not required) that your recurrence computes the correct value.

   (d) (4 pts) Give an iterative (bottom up) algorithm based on the recurrence that computes B(n) by filling in a table.

   (e) (1 pt) What is the running time of your iterative algorithm as a function of n (use asymptotic notation)?

2. (8 pts) Assume that you have a list of n home maintenance/repair tasks (numbered from 1 to n) that must be done *in list order* on your house. You can either do each task i yourself at a positive cost (that includes your time and effort) of $c[i]$. Alternatively, you could hire a handyman who will do the next 4 tasks on your list for the fixed cost h (regardless of how much time and effort those 4 tasks would cost you). You are to create a dynamic programming algorithm that finds a minimum cost way of completing the tasks. The inputs to the problem are h and the array of costs $c[1], \ldots, c[n]$.

   - (3 pts) First, find a justify a recurrence (with boundary conditions) giving the optimal cost for completing the tasks.
   - (1 pts) Give an $O(n)$-time recursive algorithm with memoization for calculating the value of the recurrence.
   - (1 pt) Give an $O(n)$-time bottom-up algorithm for filling in the array
   - (2 pts) Describe how to determine which tasks to do yourself, and which tasks to hire the handyman for in an optimal solution.

3. (10 pts) Assume you are planning a canoe trip down a river. The river has n trading posts numbered 1 to n going downstream. You will start your trip at trading post number 1 and end at trading post number n. Let $R(i, j)$ be the cost of renting a canoe at trading post i and returning it at trading post j, where $j > i$. Assume that you always want to go down river, so the costs if $j \leq i$ are irrelevant. Find the cheapest sequence of rentals that allow you to complete your trip. Aim for an algorithm running in $O(n^2)$ time.

For example, if $n = 4$ and the costs are:

R(i,j)	j		
i	2	3	4
1	15	25	35
2	$--$	12	16
3	$--$	$--$	5

then the cheapest sequence of canoe rentals to travel the river would be to rent from 1 to 3, and then from 3 to 4 for a cost of $25 + 5 = 30$.

On the other hand, if the costs were:

R(i,j)	j		
i	2	3	4
1	20	15	30
2	$--$	5	10
3	$--$	$--$	20

then taking one canoe all the way from 1 to 4 and renting 1 to 2 and then 2 to 4 are the cheapest solutions (both cost 30). (Note that renting from 1 to 3 is cheaper than going 1 to 2, but the 1 to 3 rental is not in any of the cheapest 1 to 4 solutions.)

Let $C(k)$ be the cost of the cheapest sequence of canoe rentals starting from trading post 1 and returning the last canoe rented at trading post k.

(a) (3 pts) Assume an optimal sequence of rentals changes canoes at some trading post j. What subproblems are also solved optimally by (parts of) this rental sequence? Prove your answer (probably using a proof by contradiction)

(b) (2 pts) Derive a recurrence for $C(k)$ in terms of $C(j)$ values where $j < k$.

(c) (4 pts) Give a bottom-up iterative algorithm for computing the $C(j)$ values.

(d) (1 pt) Finally, show how keeping a little (i.e. $O(n)$) additional information allows the a cheapest sequence of canoe rentals to be printed out in $O(n)$ time.

For part (c), it may be helpful to first construct a recursive algorithm for computing the $C(k)$ values.