Problem Set 7

data structures and algorithms 2 - fall 2024

your name here

due november 6, 2024 at 11:59p

Collaboration Policy: You are encouraged to collaborate with up to 4 other students, but all work submitted must be your own *independently* written solution. List the computing ids of all of your collaborators in the `collabs` command at the top of the tex file. Do not share written notes, documents (including Google docs, Overleaf docs, discussion notes, PDFs), or code. Do not seek published or online solutions for any assignments. If you use any published or online resources (which may not include solutions) when completing this assignment, be sure to cite them. Do not submit a solution that you are unable to explain orally to a member of the course staff. Any solutions that share similar text/code will be considered in breach of this policy. Please refer to the syllabus for a complete description of the collaboration policy.

Collaborators: list your collaborators

Sources: list your sources

problem 1 *Thinking About Dynamic Programming*

1. In Unit 2 we discussed divide and conquer algorithms. In Unit 4 we are discussing dynamic programming algorithms. For each of the following statements, explain if that statement is true for either (or both) of those algorithm styles.

   (a) A problem needs to be able to be defined recursively.

   Solution:

   (b) It's contains subproblems that overlap and are repeated.

   Solution:

   (c) A solution is created by combining solutions to independent, non-overlapping subproblems.

   Solution:

2. As part of our process for creating a dynamic programming solution, we search for a good order for solving the subproblems. Briefly (and intuitively) describe the difference between a top-down approach and a bottom-up approach.

   Solution:

problem 2 *Nesting Boxes*

The "Nesting Boxes Prank" has become popular in recent years. Someone is given a box to open, and inside that box is a smaller box. Opening that box, reveals a smaller box, and so on. Eventually the smallest box contains something of value. You have access to a large assortment of boxes of different sizes and styles. Given this set of n boxes, you need to find the deepest nesting possible. Describe a **dynamic programming** algorithm which, given a $fits(b_i, b_j)$ function that determines if box b_i fits inside box b_j, returns the maximum number of boxes that can be nested (i.e. gives the count of the maximum number of boxes that must be opened). You do not need

to determine the specific boxes used for the maximum nesting depth, just calculate the maximum nesting depth that can be achieved.

Note: Boxes can be any shape. There could be a set of 4 boxes b_1, b_2, b_3, b_4 such that b_1 can fit in b_2 and b_2 can fit in b_3, but b_4 can not fit in any other box, and no other box can fit in b_4. Given the possible variety of shapes, the boxes can not be sorted in any way.

1. Describe the top-down approach to solving this problem.

   Solution:

2. Describe the bottom-up approach to solving this problem.

   Solution: