This is a diagnostic exam intended to help you evaluate your readiness for the real exam.

The following rules apply to you, whether you think they do or not. Read and understand them; failure to abide by these rules, or directions given by course staff during the exam, may result in disciplinary action, including but not limited to a failing grade in the class.

- This exam is solely for students enrolled in this lecture. Anyone not enrolled in this lecture may not take an exam.
- You may not open the exam or begin writing until the instructor has explicitly given you permission to do so.
- Keep your UCI ID readily accessible during the test. Proctors may request to see it.
- This exam is closed book, closed notes, and is individual effort. Once course staff begin passing out exams, you may not communicate with anyone other than proctors for any reason, nor may you have electronics, including calculators watches and phones, available to you during the test for any reason. **YOU DO NOT NEED A CALCULATOR!**
- If you leave your seat during the test for any reason, your instructor may collect it and deem you to have turned it in. Do not ask proctors for an exemption to this, they are not authorized to grant such.
- You must take the exam in your assigned seat unless the professor (not a TA) tells you otherwise. The instructor will call to cease writing at 11:48 AM, at which point you must immediately cease writing and close the exam. You may not write any further at that point, including finishing one’s current sentence.
- If you believe a question is ambiguous, write at least two reasonable interpretations and indicate clearly which one you will be using. Then answer your question with that assumption. Unless your interpretation makes the problem much more trivial than intended, we will grade your response as if one of us had made that clarification.
- The purpose of the real exam is to evaluate how well you understand the material presented in the course. It is an academic integrity violation to do anything that subverts the goals of this assessment including, but not limited to, not doing your own work or submitting that of anyone else.
- We will only grade responses marked in the space provided for each question.
Nothing you write on this page will be graded. The next page in this booklet contains a spot to answer these questions. You may use this page as scratch paper if you would like, and room to do so exists.

1. (2 points). Suppose I have solved a problem by dynamic programming and my recursive solution is as follows. The top-level call is to $\text{Foo}(0, n)$.

$$
\text{Foo}(i, j) \\
\quad \text{if } i > n \text{ or } j < 0 \text{ then} \\
\quad \quad \text{return } 0 \\
\quad \text{else} \\
\quad \quad \text{return } \max( \text{Foo}(i, j - 1), \text{Foo}(i + 1, j), A[i] \cdot A[j] + \text{Foo}(i + 1, j - 1) ) \\
\quad \text{end if}
$$

What will the running time of this solution be if I use a memoization vector? In which order will I fill in the vector?

2. (1 point) Consider the following table, which is the output from running the optimal binary search tree algorithm for some seven-key input. In each entry, the value printed in the upper half of the cell is $\text{Tree}[i, j]$, the cost of the optimal binary search tree consisting of keys $i \ldots j$. The value printed in the lower half of the cell is the value of $\text{roots}[i, j]$.

Suppose we have computed the following using the algorithm for the optimal binary search tree. However, we accidentally forgot to record $\text{Tree}[1, n]$. What is the missing value for that spot? It is the only one omitted in the following table.

$$
\begin{array}{ccccccc}
\text{k}_1 & \text{k}_2 & \text{k}_3 & \text{k}_4 & \text{k}_5 & \text{k}_6 & \text{k}_7 \\
\hline
1 & 0.17 & 0.41 & 0.8 & 1.29 & 1.87 & 2.17 & 4 \\
2 & 0.12 & 0.41 & 0.83 & 1.31 & 1.61 & 1.73 & \\
3 & 0.17 & 0.55 & 0.95 & 1.25 & 1.37 & \\
4 & 0.21 & 0.61 & 0.82 & 0.91 & \\
5 & 0.2 & 0.4 & 0.49 & \\
6 & 0.1 & 0.16 & \\
7 & 0.03 & \\
\end{array}
$$

3. (1 point) Draw the optimal binary search tree for the input probabilities in the previous question.

© CompSci 161 Winter 2023– Michael Shindler– University of California, Irvine
This exam may not be reposted without the express written permission of the professor teaching this course.
Write your answer to question 1 between this sentence and the dotted line:

Write your answer to question 2 in the box below:

Draw your answer to question 3 between here and the end of the page.
4. (3 points) I was investigating the possibility of giving out candy at office hours and ran into the following problem. Suppose I wanted to give each student one piece of candy for each office hours of mine that they attend. I have a candy jar in my office that can hold \( C \) pieces of candy. At any point, I can ask the department to add candy to the jar; they’ll charge me a fixed price \( P \) for doing so, regardless of how many pieces of candy I ask them for. They also don’t like letting candy stay in the jar, getting stale; to discourage me from letting that happen, they charge me \( d \) for each leftover piece after every office hour.

For the next quarter, I have \( n \) office hours planned; at the \( i \)th office hour, I project that \( s_i \) students will show up, each of whom will get one piece of candy. For this problem, assume that the projected attendance at office hours is accurate. At the start of the quarter, my candy jar is empty, and I aim to have it empty at the end of the semester. My goal is to provide one piece of candy to each student that shows up at each office hour, while spending the least amount of money and having an empty candy jar at the end of the quarter. Give a dynamic programming algorithm to determine the minimum cost for me to do so.

While it is possible to solve this problem in time polynomial only in \( n \), full credit will be given for a correct solution with running time \( \mathcal{O}(nC) \).

To complete these instructions, do the following.

- Give a clear and precise English definition that describes the function you are implementing. Not how it works (yet), but rather what it solves.
- Give that function a meaningful variable name. This is not [just] me being pedantic; I have found it helps students with this topic if they do this. “OPT” is not a meaningful variable name, nor is “table.” Single letters are not meaningful variable names.
- Give a clear recursive formula or algorithm in terms of smaller instances of exactly the same problem.
- Describe the iterative running time correctly. This can either be by writing the iterative algorithm (in which case, you can point out where the previous part is within the solution), or by taking your recursive solution, counting the cases, describing the order in which the table would be filled in, and analyzing the time accordingly.
Write your answer to question 4 on this page. It is possible that you will not use the entire page.