University of New Mexico
Department of Computer Science

Midterm Examination

CS 561 Data Structures and Algorithms
Fall, 2023

Name:	
Email:	

Directions:

- This exam lasts 75 minutes. It is closed book and notes, and no electronic devices are permitted. However, you are allowed to use 2 pages of handwritten "cheat sheets"
- *Show your work!* You will not get full credit, if we cannot figure out how you arrived at your answer.
- Write your solution in the space provided for the corresponding problem.

Question	Points	Score	Grader
1	20		
2	20		
3	20		
4	20		
5	20		
Total	100		

1. **Short Answer (4 points each)**

   Answer the following using *simplest possible* Θ *notation.*

   (a) Expected number of items at the $(1/3)\log n$ level of a skip list containing n items?

   (b) Time to insert n items into a count-min sketch with k hash functions and m counters?

   (c) Solution to the recurrence: $T(n) = 6T(n/2) + n^3$

   (d) Solution to the recurrence: $f(n) = 2f(n-1) - f(n-2) + 10$ (answer in big-O)

   (e) Consider a certain operation OP. Over n calls to OP, the i-th call, for $i \in [1, n])$, takes time i if i is a power of 3 or 1 otherwise. What is the amortized cost of OP?

2. **Induction (20 points)** An *elegant* string is defined recursively as follows:

   - The empty string is elegant
   - If S is an elegant string, then the string $aSbb$ is elegant
   - If S_1 and S_2 are non-empty, elegant strings, then the string $S_1 \cdot S_2$ is elegant

   Thus, for example, the string $aabbbbabb$ is elegant since it's the concatenation of string $aabbbb$ and abb, which are both elegant.

   Prove that any elegant string of length n contains $n/3$ a's. Prove this by induction on n. Don't forget the BC, IH and IS.

3. **Donuts**

A bakery sells donuts in boxes of three different increasing quantities: x_1, x_2, and x_3. Each of the types of boxes has probability, p_1, p_2, p_3, of holding a special coupon; these probabilities are all independent. You want to buy *at most* n donuts and want to *maximize* the expected number of coupons that you receive. For example, if $x_1 = 2$, $x_2 = 3$, $x_3 = 4$; $p_1 = .1$, $p_2 = .7$ and $p_3 = .2$; and $n = 7$, then you should buy two of the $x_2 = 3$-donut boxes to get an expected number of coupons of $2 * .7$.

(a) (7 points) For any positive x, let $m(x)$ be the maximum expected number of coupons you can get by buying at most x donuts. Write a recurrence relation for $m(x)$.

(b) (6 points) Describe a dynamic programming algorithm based on your recurrence. How big is your array as a function of n? How do you fill it in? What value do you return? What is the runtime?

(c) (7 points) **Hard!** For this month only, the special coupon is for a lifetime supply of a new "Kreamy Kale"$^{\text{TM}}$ donut. The value of x_1 now equals 1, but the other x_i and p_i values are arbitrary. Now your goal is to buy exactly n donuts, while *minimizing the probability of receiving any coupon.*

Warning: Minimizing the probability of getting one coupon is *not* the same as minimizing the expected number of coupons.

Write a recurrence relation to obtain the minimum probability of receiving a coupon while buying n donuts. Hint: recall $\log(x \cdot y) = \log x + \log y$.

4. **Probability and Expectation**

There is a grid of n nodes, where n is a perfect square.

(a) (2 points) Assume each node fails independently with probability p. What is the expected number of nodes that fail?

(b) (4 points) Use Markov's inequality to bound the probability that at least 1 node fails. Hint: Let X be the number of failed nodes, use $E(X)$ from above.

(c) (4 points) Use a union bound to bound the probability that no node fails. Hint: Let E_i be the event that node i fails. Bound $Pr(\cup_{i=1}^{n} E_i)$.

(d) (5 points) A row is said to be *bad* if any of the nodes in the row fail. What is the expected number of bad rows?

(e) (5 points) Now assume that there are exactly $n/2$ failed nodes, uniformly distributed among all nodes. What is the expected number of bad rows?

5. **Palindromes (Hard!)**

(a) (10 points) **Longest Palindrome Subsequence.** Design an algorithm to find the longest subsequence of a string that is also a palindrome. In particular, you are given a string A of length n, and you want to find the longest subsequence of A that reads the same forwards as backwards. For example if A is the string $\underline{E}Z\underline{D}IS\underline{E}N\underline{GAGED}X\underline{E}$ then the longest palindrome subsequence is $EDEGAGEDE$, and your algorithm should return 9.

Hint: For appropriate i and j, let $m(i, j)$ be the longest palindrome subsequence of the substring $A[i, \ldots, j]$.

(b) (10 points) **Shortest Palindrome Supersequence.** Now design an algorithm to find the *shortest* supersequence of a string that is also a palindrome. For example, if your string A is $UNOONON$, then the shortest palindrome supersequence is $\underline{UNOONO}O\underline{N}U$, and your algorithm should return 9.

Hint: Now, for appropriate i and j, let $m(i,j)$ be the *shortest* palindrome *supersequence* of the substring $A[i,\ldots,j]$.

5. **Palindromes (Hard!), continued.**