University of New Mexico
Department of Computer Science

Midterm Examination

CS 561 Data Structures and Algorithms
Fall, 2025

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 $\log n$ level of a skip list that contains 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: $T(n) = 2T(n/8) + n$

   (d) Solution to the recurrence: $T(n) = 8T(n/2) + n$

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

2. **Induction (20 points)** You are given a graph over n nodes. You must color each node red or black. An edge in the coloring is *good* if the endpoints of that edge are two *different* colors.

   Prove by induction on n that there exists some coloring where at least half the edges are good.

3. **The Creaky Stairs**

On a dark and quiet fall night, you find yourself climbing a creaky staircase. There are n stairs, and every stair, $i \in [1, n]$ has a creakiness value, or cost, of c_i. In each step, you can go up either 2 or 3 stairs. Your goal is to get to the top of the staircase in a way that minimizes the sum of costs for all the stairs you visit. The bottom and top of the staircase are not creaky and so have no cost. At the end, any step size that exceeds the number of stairs remaining will take you to the top.

For example, for $n = 4$ and costs $[2, 1, 5, 8]$, you can first go up 2 stairs, at a cost of 1, and then go up 3 stairs at a cost of 0, for a total cost of 1.

(a) (3 points) Consider a greedy algorithm which always takes the next step with least cost. Show this algorithm may not be optimal.

(b) (12 points) Now, write a recurrence relation for a dynamic program to solve this problem. First, define a function, in words, whose solutions help solve the big problem.

(c) (5 points) Describe a dynamic program to solve the problem using your recurrence. What are the dimensions of your table? How do you fill it in? What is the final value returned? What is the runtime of your algorithm?

4. **Probability** There is a grid with n by n nodes, n^2 **nodes total**. Each node is independently colored green with probability p and red otherwise.

   (a) (4 points) What is the expected number of green nodes?

   (b) (4 points) Use Markov's inequality to get an upper bound on the probability that at least one node is green.

   (c) (4 points) Now use a union bound to bound the probability that no node is green.

   (d) (8 points) A node is an *interior* node if it has 4 neighbors. So, the grid has $(n-2)^2$ interior nodes. Now, let $p = 1/2$. What is the expected number of interior nodes where the node and all 4 of its neighbors have the same color?

5. **Nim**

The ancient game of *NIM* is played by two players who alternate taking any positive number of stones from one of 3 piles. The person taking the last stone loses. An example game starting with piles of size 13, 9 and 1 is below. Here, the first move of Player 1 is to take the single stone in pile 1, and player 1 eventually wins.

Player Turn	1	2	1	2	1	2
Stones left	(13, 9, 1)	(13, 9, 0)	(13, 2, 0)	(2, 2, 0)	(1, 2, 0)	(1, 0, 0)

In this problem, you will write a dynamic program to determine if Player 1 can force a win for an given input (x_1, x_2, x_3) giving the number of stones in each of the 3 piles.

(a) Describe in words a function whose solutions for smaller problems will help you solve the big problem.

(b) Write a recurrence relation for the dynamic program using the function you described above.

(c) Describe a dynamic program to solve the problem for any initial input consisting of x_1, x_2, x_3 stones in the piles. What are the dimensions of your table? How do you fill it in? What is the final value returned? What is the runtime of your algorithm if $x_1 = x_2 = x_3 = n$?

5. **Nim, continued.**