Here are some extra practice problems taken from past semesters. You are welcomed to check your solutions with me or discuss them in the telegram group chat.

1. **Freivalds' algorithm** (New Question). Instead of choosing $v = \{0, 1\}^n$ as in lecture, we modify the algorithm to choose $v = \{0, 1, \ldots, 999\}^n$ instead. What is the probability that Freivalds' algorithm is successful? Prove the new bound.

2. **Expectation Arguments** (CS5275 Tutorial 3).Let $\mathcal{M} = \{0, 1, \ldots, N^2 - 1\}$ for some positive integer N, and let $\mathcal{A}$ be a subset of $\mathcal{M}$ of size N. Show that there exists a subset $\mathcal{B}$ of $\mathcal{M}$ of size at most N such that the set
$$\mathcal{C} = \{a + b \bmod N^2 \mid a \in \mathcal{A} \text{ and } b \in \mathcal{B}\}$$
has cardinality at least $\frac{N^2}{2}$.

3. **Markov's Inequality** (CS5275 Tutorial 3, Simplified). Suppose that there are n nodes, and between each pair of nodes, an edge is created with probability p, independently of all the others. A random graph obtained through this procedure is called the *Erdős–Rényi random graph*. Let T_n be the random variable denoting the (random) number of triangles for a random graph obtained through the procedure, where a triangle is a set of 3 nodes that are all connected to each other.

   (a) Find $\mathbb{E}[T_n]$.

   (b) Using the results from (a), show that $\mathbb{P}[T_n = 0] \geq 1 - \frac{1}{6}n^3p^3$.

4. **Union Bound** (AY 22/23 Sem 2, Modified). Given a permutation π over $\{1, 2, \cdots, n\}$, let $L(\pi)$ be the length of the longest increasing sequence in π. For example, in the permutation $\pi = (1, 6, 4, 5, 2, 7, 3)$, the longest increasing subsequence is $(1, 4, 5, 7)$ and thus $L(\pi) = 4$.
   Now, we consider a random permutation π over $\{1, 2, \cdots, n\}$.

   (a) Show that $\mathbb{P}[L(\pi) \geq 10\sqrt{n}] \leq \left(\dfrac{ne^2}{(10\sqrt{n})^2}\right)^{10\sqrt{n}}$.
   (You may assume the inequalities $\binom{n}{k} \leq (\frac{en}{k})^k$ and $k! \geq (\frac{k}{e})^k$ without proof.)

   (b) Using the results from (a), show that $\mathbb{E}[L(\pi)] = O(\sqrt{n})$.

5. **Dynamic Programming** (Classical Questions). Here are some simpler "quickfire" exercises for DP.

   (a) There are N stones with heights $h_1, h_2, \ldots, h_N$ respectively. For each step, the frog can jump from stone i to any of stones $i + 1, i + 2, \ldots, i + K$ (say, stone j), incurring cost $|h_i - h_j|$. Find the minimum cost for the frog to jump from stone 1 to stone N. Your algorithm should run in $O(NK)$ time.

   (b) The vacation consists of N days. For day i, you can either choose one of activities A, B, C, incurring happiness a_i, b_i, c_i respectively. However, you cannot pick the same activity for two consecutive days. Find the maximum happiness you can get over the N days. Your algorithm should run in $O(N)$ time.

   (c) There are N items with weights $w_1, w_2, \ldots, w_N$ and values $v_1, v_2, \ldots, v_N$. Pick a subset of items with maximum total value, subject to the constraint where the total weight of the items must be $\leq W$. Design two algorithms, one solving this in $O(NW)$ and another solving this in $O(NV)$, where $V = \sum_{i=1}^{N} v_i$.

   (d) You are multiplying N matrices of sizes $m_0 \times m_1$, $m_1 \times m_2$, ..., $m_{N-1} \times m_N$ respectively. Multiplying a $x \times y$ matrix with a $y \times z$ matrix takes cost xyz. What is the minimum cost required for you to multiply all these N matrices together to get a single $m_0 \times m_N$ matrix? Your algorithm should run in $O(N^3)$ time.

   (e) You are trying to distribute K candies to N children. Child i must receive between 0 and a_i candies (both inclusive). Also, no candies should be left over. Find the number of ways to distribute the candies. Your algorithm should run in $O(NK^2)$ time. (Bonus: Optimize it to $O(NK)$.)

6. **Dynamic Programming** (AY 23/24 Sem 2). Given two strings x and y the edit distance, denoted by $ed(x, y)$, between them is the minimum number of character insertion, deletion and substitution operations (all of these operations are also known as edit operations) required to transform x into y. Given two strings x and y of length n and m respectively, design and analyze an algorithm that computes the value of $ed(x, y)$ in $O(mn)$ time.

7. **Dynamic Programming** (AY 23/24 Sem 2). Suppose a seller wants to cut a rod into several pieces and then sells them to his/her customers. The main objective of the seller is to maximize his/her earning. Given a rod of length n and a list of prices of rod of length i for all $1 \leq i \leq n$, find an optimal way to cut the rod into smaller pieces in order to maximize the earning. (Try to design a dynamic programming algorithm.)

8. **Dynamic Programming** (CLRS Chapter 14 Problem 14-3). In the euclidean traveling-salesperson problem, you are given a set of n points in the plane, and your goal is to find the shortest closed tour that connects all n points. Figure 1(a) shows the solution to a 7-point problem. The general problem is NP-hard, and its solution is therefore believed to require more than polynomial time.

   J. L. Bentley has suggested simplifying the problem by considering only bitonic tours, that is, tours that start at the leftmost point, go strictly rightward to the rightmost point, and then go strictly leftward back to the starting point. Figure 1(b) shows the shortest bitonic tour of the same 7 points. In this case, a polynomial-time algorithm is possible.

   Describe an $O(n^2)$-time dynamic programming algorithm for determining an optimal bitonic tour. You may assume that no two points have the same x-coordinate and that all operations on real numbers take unit time. Provide a clear description of your algorithm, including the optimal substructure, and a detailed running time analysis.

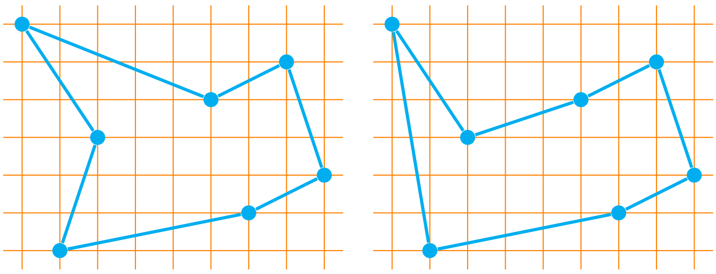

Figure 1: Seven points in the plane, shown on a unit grid. **(a)** The shortest closed tour, with length approximately 24.89. This tour is not bitonic. **(b)** The shortest bitonic tour for the same set of points. Its length is approximately 25.58.

Hints

2. Create a random subset $\mathcal{B}$ uniformly by selecting N items from $\mathcal{M}$ uniformly at random with replacement, and use linearity of expectation to evaluate the expectation of the number of integers in $\mathcal{M}$ but not $\mathcal{C}$. The inequality $(1 - 1/N)^N < 1/e$ is useful for the bound.
3. (a) Use linearity of expectation on the $\binom{n}{3}$ triples of vertices.
   (b) Apply Markov's Inequality. It might be a good idea to consider $\mathbb{P}[T_n \geq 1]$ first.
4. (a) Consider all subsequences of length $10\sqrt{n}$. Apply union bound to get an upper bound for $\mathbb{P}[L(\pi) \geq 10\sqrt{n}]$. Then, use the inequalities $\binom{n}{k} \leq (\frac{en}{k})^k$ and $k! \geq (\frac{k}{e})^k$.
   (b) Write the formula of $\mathbb{E}[L(\pi)]$. Upper bound terms $< 10\sqrt{n}$ and terms $\geq 10\sqrt{n}$ separately.
6. Modify the LCS algorithm taught in lecture.
7. Relate this to the Knapsack problem. Note that each item can now be selected multiple times, so the algorithm has to be modified.
8. Scan left to right, maintaining optimal possibilities for the two parts of the tour.