CS 473 ✧ Fall 2024

Homework 3

Due Tuesday, September 24, 2024 at 9pm Central Time

This is the last homework before Midterm 1.

Full credit for any dynamic programming problem requires the following ingredients:

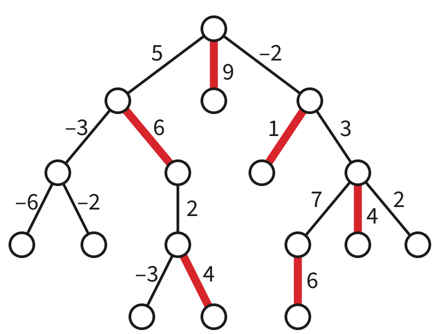
- A clear English specification of the underlying recursive function. In particular, this description should name the input parameters and describe how the function value depends on those parameters. This is the most important part of the problem!
- A recurrence or recursive backtracking algorithm that evaluates this recursive function.
- Iterative details: A description of an appropriate data structure (which is *never* a hash table), an evaluation order, and the resulting running time.

Complete iterative pseudocode is *not* required for full credit. If you provide iterative pseudocode, you do *not* need to separately provide a recurrence, a memoization structure, and an evaluation order, but you do still need an English specification and (as always) the running time.

---

1. Suppose we are given a tree T with weighted edges. Each edge weight could be positive, negative, or zero, and the degrees of vertices in T could be arbitrarily large. A subset M of edges of T is called a *matching* if every vertex of T is incident to at most one edge in M. Describe and analyze an algorithm to find a matching in T with maximum total weight.

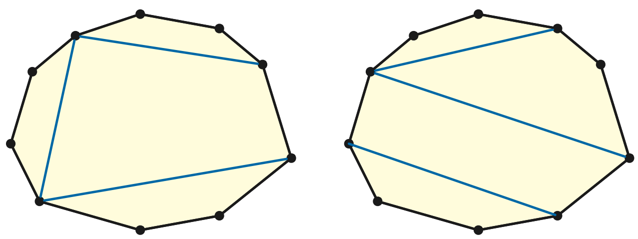
   For example, given the tree shown below, your algorithm should return the integer 30, which is the total weight of the indicated matching.



2. A *polygon* is the area enclosed by a cycle of straight line segments in the plane, connected end-to-end. The line segments are called the *edges* of the polygon, and the endpoints of the segments are called its *vertices*. A *diagonal* of a polygon is a line segment between two vertices that is not an edge. A polygon is *convex* if every diagonal lies entirely inside the polygon. Cutting a convex polygon along one of its diagonals partitions it into two smaller convex polygons.

   Suppose we are given a convex polygon P with an even number of vertices $p_0, p_1, \ldots, p_{2n-1}$, indexed in order around the boundary. Describe an algorithm to compute diagonals that partition P into convex quadrilaterals (4-sided polygons), where the area of the smallest quadrilateral is as large as possible. *[Hint: Why did we assume the number of vertices is even? Why is a partition into quadrilaterals always possible? What can happen at p_0?]*

   Assume you have access to a subroutine QuadArea(i, j, k, l) that computes the area of the quadrilateral with vertices p_i, p_j, p_k, p_l in $O(1)$ time. This function is actually easy to implement with the right math,[1] but you don't need to know anything about its implementation to design your algorithm.



Two different ways to partition a convex decagon into four convex quadrilaterals.

3. The (Eleventh) Doctor and River Song decide to play a game on a directed acyclic graph G, which has one source vertex s and one sink vertex t.[2]

   Each player has a token on one of the vertices of G. At the start of the game, The Doctor's token is on the source s, and River's token is on the sink t. The players alternate turns, with The Doctor moving first. On each of his turns, the Doctor moves his token forward along a directed edge; on each of her turns, River moves her token *backward* along a directed edge.

   If the two tokens ever meet on the same vertex, River wins the game. ("Hello, Sweetie!") If the Doctor's token reaches t or River's token reaches s before the two tokens meet, then the Doctor wins the game.

   Describe and analyze an algorithm to determine who wins this game, assuming both players play perfectly. That is, if the Doctor can win *no matter how River moves*, then your algorithm should output "Doctor", and if River can win *no matter how the Doctor moves*, your algorithm should output "River". (Why are these the only two possibilities?) The input to your algorithm is the graph G.

[1] QuadArea$(i, j, k, l) = x_i y_j - x_j y_i + x_j y_k - x_k y_j + x_k y_l - x_l y_k + x_l y_i - x_i y_l$.

[2] The labels s and t are abbreviations for the Untempered **S**chism and the **T**ime Vortex, or the Shining World of the Seven Systems (also known as Gallifrey) and Trenzalore, or Skaro and Telos, or Something else Timey-wimey.