B503 Homework 4

Due date: November 19 Lecturer: Qin Zhang

Your Name:________________ Your University ID:________________

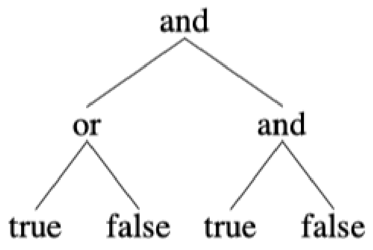
Instruction: Please submit your homework via Canvas before the deadline. Please add references to ALL the resources you have used for completing the assignment. You are allowed to discuss the assignment with other students, and if you do so, please list their names in the submission. Remember, you have to write all answers by yourself even if you have discussed with other students.

Unless mentioned otherwise, for each algorithm that you design, please **give a proof of its correctness**. If the question asks you to come up with an algorithm of a specified runtime complexity, please **give the running time analysis**. If the task doesn't specify the runtime complexity, you are allowed to present an algorithm of any time complexity. However, you still need to prove the correctness of your algorithm and analyze its running time.

Total points: 37

Late Policy: No extensions or late homeworks will be granted, unless a request is made to the course instructor before due date and written documents are provided to support the reason for late submission.

Problem 1 (5 points). At Aperture Bakeries, every cake comes with a binary boolean-valued tree indicating whether or not it is available. Each leaf in the tree has either a *true* or a *false* value. Each of the remaining nodes has exactly two children and is labeled either *and* or *or*; the value is the result of recursively applying the operator to the values of the children. One example is the following tree:



If the root of a tree evaluates to *false*, like the one above, the cake is a lie and you cannot have it. Any *true* cake is free for the taking. You may modify a tree to make it *true*; the only thing you can do to change a tree is to turn a *false* leaf into a *true* leaf, or vice versa. This costs \$1 for each leaf you change. You can't alter the operators or the structure of the tree.

Describe an efficient $O(n)$ algorithm to determine the minimum cost of a cake whose tree has n nodes. Prove the correctness and analyze running time of your algorithm.

Problem 2 (5 points). Consider two vertices, s and t, in a *directed acyclic graph* $G = (V, E)$. Give an efficient $O(|V| + |E|)$ algorithm to determine whether the number of paths in G from s to t is odd or even.

Problem 3 (5 points). You are given a sequence of integers. You can choose to remove a *single* number from the list, or not remove any at all. Your goal is to maximize the length of the strictly increasing contiguous subsequence in the resulting array.

Create an algorithm to find the maximum length of such a sequence with a linear time complexity.

N.B. A contiguous subsequence $b_1, \ldots, b_m$ of sequence $a_1, \ldots, a_n$ is such a subsequence that its elements are next to each other in the original sequence, i.e. $b_i = a_{l-1+i}$ for some l and all $i \in [1..m]$.

Problem 4 (5 points).

During an all too common electricity outage, Jane and Joe decide to pass the time by playing a game. They use a set of n cards, each labeled with a distinct number from 1 to n. They shuffle the cards and place them in a line. The resulting order of labels is denoted as $a_1, a_2, \ldots, a_n$.

To start the game, Jane and Joe take an old Monopoly figure (the iron) and put it on one of the cards. They take turns moving the iron, with Jane taking the first turn. The game has two rules for moving the iron:

1. When the iron is on the ith card, it can only be moved forward, to a card with a greater label, i.e. to a card j where $a_j > a_i$.
2. The iron can only be moved by the number of cards divisible by the label on the current card, i.e. $(j - i) \bmod a_i = 0$.

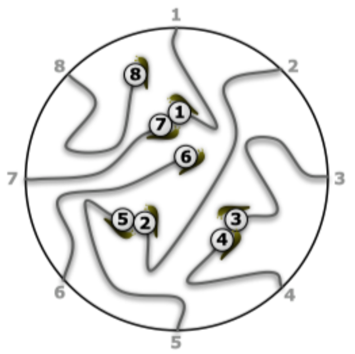
If a player can't make a move, they lose the game.

Create an $O(n \log n)$ algorithm that, for each position $i \in [1..n]$, determines who will win the game starting with this position: Jane or Joe.

Problem 5 (5 points).

Every year, as part of its annual meeting, the Antarctican Snail Lovers of Upper Glacierville hold a Round Table Mating Race. Several high-quality breeding snails are placed at the edge of a round table. The snails are numbered in order around the table from 1 to n. During the race, each snail wanders around the table, leaving a trail of slime behind it. The snails have been specially trained never to fall off the edge of the table or to cross a slime trail, even their own. If two snails meet, they are declared a breeding pair, removed from the table, and whisked away to a romantic hole in the ground to make little baby snails. Note that some snails may never find a mate, even if the race goes on forever.

For every pair of snails, the Antarctican SLUG race organizers have posted a monetary reward, to be paid to the owners if that pair of snails meets during the Mating Race. Specifically, there is a two-dimensional array $M[1..n, 1..n]$ posted on the wall behind the Round Table, where $M[i, j] = M[j, i]$ is the reward to be paid if snails i and j meet.



The end of a typical Antarctican SLUG race. Snails 6 and 8 never find mates.
The organizers must pay $M[3,4] + M[2,5] + M[1,7]$.

Describe and analyze an $O(n^3)$ algorithm to compute the maximum total reward that the organizers could be forced to pay, given the array M as input.

Problem 6 (6 points). You are given a weighted oriented graph $G = (V, E)$, $|E| \geq |V|$. You want to find a path in G, longest *by the number of edges*, such that each consequent edge is greater than the previous one.

Create an $O((|V| + |E|) \log |V|)$ algorithm to find the *length* of such path.

Problem 7 (6 points).

You have a string s with only lowercase English letters. You can take any contiguous substring of s such that it has identical symbols and delete them all together from s. For example, if $s = xyyzqqqzzq$, you can delete qqq and get $xyyzzzq$.

Devise an algorithm to find the minimum number of such subsequent deletions to make s empty. The running time should be $O(n^3 \log n)$ (or, naturally, better).