CS2040S Data Structures and Algorithms **AY2022/2023 Semester 2**

CS2040S: Midterms Extra Practice

By: Audrey Felicio Anwar

Problem 1

Solve the following recurrence relations, give the tightest bound possible.

1. $T(n) = 2T(n-1) + \Theta(1)$
2. $T(n) = \sum_{i=1}^{n-1} T(i) + \Theta(1)$
3. $T(n) = 3T(\frac{n}{2}) + \Theta(n^2)$
4. $T(n) = T(\frac{n}{2}) + T(\sqrt{n}) + \Theta(n)$

Problem 2

Audrey is not impressed with the sorting algorithms taught in lectures and decided to design his own sorting algorithm. He would like to sort an array A of N integers in the following way.

1. Run `QuickSort` algorithm on A, but Audrey modified the `QuickSort` algorithm to stop when the current size of the array during recursion is $\leq K$.
2. After the modified `QuickSort` has finished, Audrey runs `InsertionSort` on the modified array A.

Does this algorithm always sorts A correctly? What is the running time of this algorithm?

Problem 3

Audrey is busy with his midterms, however he still has K homeworks that needs to be finished this week. Due to the lack of time, Audrey decided to outsource his homeworks to his friends. Audrey has N friends. Each friend f_i has an initial annoyance level of a_i and every time Audrey ask to do an homework, the friend's annoyance level increases by d_i. Audrey does not want to annoy his friends too much. Design an efficient algorithm that will assign all of Audrey's homeworks to his friends, such that the maximum level of annoyance among his friends is as small as possible.

As an example, when $K = 4$ and `friends` $= [(1, 2), (2, 3), (3, 4), (4, 5)]$, a possible desired assignment would be give 2 homeworks to f_1, 1 homework to f_2, and 1 homework to f_3, resulting in a maximum annoyance level of 7. In this test case, we cannot have maximum annoyance less than 7, no matter what the assignments are.

Problem 4

You are given an array A of N integers. You want to construct an array B of N integers, such that

$$B[i] = \begin{cases} x & \text{where } x \text{ is the smallest index} > i \text{ such that } A[x] \geq A[i] \\ -1 & \text{if all the elements after } A[i] \text{ are smaller than } A[i] \end{cases}$$

Design an efficient algorithm to construct B. As an example, if $\texttt{A} = [1, 3, 2, 2, 1]$, then $\texttt{B} = [1, -1, 3, -1, -1]$.

Problem 5

You are given an array A of N distinct integers. You are also given an array B consisting $N-1$ characters `<` or `>`. You want to find a permutation of A such that it satisfies the constraints given in array B. For example, if $\texttt{A} = [100, 2, 9, 19]$ and $\texttt{B} = [>, <, >]$, then a desired result can be $\texttt{A} = [19, 9, 100, 2]$, since $19 > 9, 9 < 100$, and $100 > 2$. Design an efficient algorithm to solve this problem.

Problem 6

You are given N distinct points on a plane. The manhattan distance between two points $(x_1, y_1), (x_2, y_2)$ is defined as $|x_1 - x_2| + |y_1 - y_2|$. Design an efficient algorithm to compute the sum of manhattan distances between all pairs of points. As an example, when $\texttt{points} = [(1, 2), (2, 3), (3, 5)]$, the answer is $2 + 3 + 5 = 10$.

Problem 7

Audrey has been kidnapped and is forced to do some tasks in order to survive. For each day, a stream of tasks will be given to Audrey and has to be executed in the order of arrival. The tasks that arrive are of two types.

1. Audrey will be asked to write down a name X.
2. Audrey will be asked to convert all written names X to Y.

At the end of the day, after executing all the tasks, Audrey is asked to scream out loud the final names of each name that he has been asked to write, in order of their arrival.

As an example, if the tasks are `(1, "Alice"), (1, "Bob"), (2, "Alice", "Charlie"), (1, "Alice"), (2, "Bob", "Charlie"), (2, "Charlie", "Malory")`, then at the end of the day, Audrey will have to scream `["Malory", "Malory", "Alice"]`, since the arrival order is `"Alice", "Bob", "Alice"` and the first `"Alice"` and `"Bob"` has been changed to `"Malory"`.

Audrey will be given punishment if he cannot figure out what names to scream quickly. Design an efficient algorithm to help Audrey survive.