Homework 2

ESE 344 [Spring 2025]

Prof. Alex Doboli

Due date: 02/24/2025 midnight

Submit all exercises using Github

Exercise 1: Given an n x n matrix where each of the rows and columns is sorted in ascending order, return *the* k^{th} *smallest element in the matrix*.

Note that it is the k^{th} smallest element **in the sorted order**, not the k^{th} **distinct** element.

You must find a solution with a memory complexity better than $O(n^2)$.

Example 1:

Input: matrix = [[1, 5, 9], [10, 11, 13], [12, 13, 15]]; k = 8

Output: 13

13 is the 8^{th} smallest number in the array.

Example 2:

Input: matrix = [[-5]], k = 1

Output = -5

(**Acknowledgment**: Source leetcode.com)

Exercise 2: A password is considered strong if the below conditions are all met:

- It has at least 6 characters and at most 20 characters.
- It contains at least one lowercase letter, at least one uppercase letter, and at least one digit.
- It does not contain three repeating characters in a row (i.e., "Baaabb0" is weak, but “Baaba0” is strong).

Given a string “**password**”, return the minimum number of steps required to make a password strong. If “**password**” is already strong, return 0.

In one step, you can:

- Insert one character to password,
- Delete one character from password, or

- Replace one character of password with another character.

Example 1:

Input: password = "a"

Output: 5

Example 2:

Input: "aA1"

Output: 3

Example 3:

Input: "1337C0d3"

Output: 0

(**Acknowledgment**: Source leetcode.com)

Exercise 3: You are given an array of CPU tasks, each labeled with a letter from A to Z, and a number **n**. Each CPU interval can be idle or allow the completion of one task. Tasks can be completed in any order, but there's a constraint: there has to be a gap of **at least n** intervals between two tasks with the same label.

Return the **minimum** number of CPU intervals required to complete all tasks.

Constraints:

- 1 <= tasks.length <= 10^4
- tasks[i] is an uppercase English letter.
- 0 <= n <= 100

Example 1:

Input: tasks = [“A”, “A”, “A”, “B”, “B”, “B”], n = 2

Output: 8

Explanation: A possible sequence is: A -> B -> idle -> A -> B -> idle -> A -> B.

After completing task A, you must wait two intervals before doing A again. The same applies to task B. In the 3rd interval, neither A nor B can be done, so you idle. By the 4th interval, you can do A again as 2 intervals have passed.

Example 2:

Input: tasks =["A","C","A","B","D","B"], ["A","C","A","B","D","B"], n = 1

Output: 6

Explanation: A possible sequence is: A -> B -> C -> D -> A -> B.

With a cooling interval of 1, you can repeat a task after just one other task.

Example 3:

Input: tasks = ["A","A","A", "B","B","B"], n = 3

Output: 10

Explanation: A possible sequence is: A -> B -> idle -> idle -> A -> B -> idle -> idle -> A -> B.

There are only two types of tasks, A and B, which need to be separated by 3 intervals. This leads to idling twice between repetitions of these tasks.

(**Acknowledgment**: Source leetcode.com)