Algorithms Laboratory (CS29203)
Assignment 4: Dynamic Programming
Department of CSE, IIT Kharagpur

15th September 2023

Question-1 (50 points)

In this question, you will be extending the Longest Common Subsequence (LCS) problem in some ways. Given two similar strings, our goal is to find the difference between them by some specific metrics. The idea is to find out the smallest number of deletion and insertion operations needed to create one string from the other.

For example, let us consider two strings X = "XMJYAUZ" and Y = "XMJAATZ". Let us say that '-' operation indicates that the character is deleted from Y but it was present in X, and '+' operation indicates that the character is inserted in Y but it was not present in X. Then the output will be "X M J -Y A -U +A +T Z".

Your task is to use the idea of dynamic programming to develop an algorithm to solve the problem. The implementation should be in C. For your convenience, a possible direction is given as hint: think of using LCS to solve this problem. If you find the LCS between the two strings, then if a particular character is not present in the subsequence, but that is present in X, then the character has been deleted (i.e. '-' operation). Similarly if a particular character is not present in the subsequence, but that is present in Y, then the character has been inserted (i.e. '+' operation).

Example:

```
Input: X = "ABCDFGHJQZ", Y = "ABCDEFGIJKRXYZ"
Output: "A B C D +E F G -H +I J -Q +K +R +X +Y Z"

Input: X = "ABCDFGHJQZ", Y = "ABCDEFGIJKRXYZ"
Output: "A B C D +E F G -H +I J -Q +K +R +X +Y Z"

Input: X = "ABCDEFGHIJKL", Y = "LKJIHGFEDCBA"
Output: "-A -B -C -D -E -F -G -H -I -J -K L +K +J +I +H +G +F +E +D +C +B +A"

Input: X = "ABCDEFGHIJKL", Y = " "
Output: "-A -B -C -D -E -F -G -H -I -J -K -L"
```

Question-2 (50 points)

You have entered in a casino for trying your luck on winning gambling games. The casino has a series of gambling machines connected in sequence. There are some specific rules for playing in the casino. Each machine in the casino has two specific attributes. For each machine, there are some fixed points that you can earn. In addition, each machine also has a number associated, called the "skip-number". It means that you won't be able to access next skip number of machines. You can think it as a (zero-indexed) 2D integer array 'machines' where machines[i] = [points_i, skipNumber_i].

While playing the game, you can access the machines in order (i.e. starting from machine #0), and make a decision whether to play in that machine or skip that machine. The final goal is to gain maximum points after having access to all the machines in the casino. For example let us consider the following sequence: machines = [[3, 2], [4, 3], [4, 4], [2, 5]]. In this case, if you play in machine #0, then you can get 3 points, but you will not have access to machine #1, #2. Instead, if you skip playing machine #0 and play machine #1, you will get 4 points, but you won't have access to machine #2, #3. That means, the second solution is better than the first one.

Your task is to develop and implement an algorithm to solve this problem using the idea of dynamic programming.

Example 1:

```
(Input) machines = [[1,1],[2,2],[3,3],[4,4],[5,5]]
(Output) 7

Explanation: The maximum points can be earned by accessing machine #1 and #4.
- Skip machine #0
- Access machine #1: Get 2 points, won't have access to next 2 machines
- No access to machine #2 and #3
- Access machine #4: Get 5 points
Total points gained: 2 + 5 = 7. There is no other way to earn 7 or more points.
```

Example 2:

```
(Input) machines = [[3,2],[4,3],[4,4],[2,5]]
(Output) 5

Explanation: The maximum points can be earned by accessing machine #0 and #3.
- Access machine #0: Get 3 points, won't have access to next 2 machines
- No access to machine #1 and #2
- Access machine #3: Get 2 points
Total points earned: 3 + 2 = 5. There is no other way to earn 5 or more points.
```