Programming Assignment 4 - Rishab Dudhia (SID: 862141444)

a. This problem was a simple dynamic programming problem. We started at the last element and calculated the longest decreasing substring up until that element. The recursive structure will first reach the base case to then calculate the longest decreasing substring of the first element which would just return 1. Then looking back onto the before calculated longest decreasing subsequences, we compare the current value to the longest decreasing subsequences and we try to see if we can add the value to the longest decreasing substring before the current element. If the current element cannot be added onto any substring it will represent the beginning of a new decreasing subsequence with a value of 1 in the array to remember the longest decreasing subsequence of the i^{th} element.

b. To calculate the minimum weighted edit distance, we had to manipulate the recurrence given in class. Instead of each operation (insert, delete, edit) having the same cost of 1, we must calculate 6 values. First, we must call the three operations while changing the first string, and then, we must call the same operations while changing the second string. Then, we choose the lowest value to represent the minimum weighted edit distance in the two dimensional array for subproblem s[i,j]. The subproblem represents the minimum weighted edit distance for the first i elements of string 1 and the first j elements of string 2. If the current values of both strings are equal, we do not need to call any of the operations and just return the value for s[i-1,j-1].