Name: Abdelrahman Medhat Saad Nawar

ID: 120180025

Assignment 7: Dynamic Programming

Question 1

as the matrix is not sorted in a way that makes doing a modified binary search possible we will be searching for the element by:

1. we start form the element in the first row and last column
2. if we are at an element that is bigger than the one we search for we go to the element to the left of it
3. if it is smaller we go to the element we go to the element under it

Time complexity:
since on each iteration we either increase i or decrease j we can only go
$2n$
iterations and in each one we only do a simple comparison
$\therefore \mathcal{O}(n)$

Space complexity:
$\mathcal{O}(1)$

Question 2

it is not efficient as it has a lot of redundant calls and will calculate some values more than once, a way to improve it is to use memoization technique and store the value of each call once it is done so it will only be done once

Question 3

1. We start traversing from
   $i = 0, j = 0$
   until we get any out of range values.
   this way in each iteration we will not calculate every value it will rather be calculated in on of the iterations that happened before it.

2. In this recursive call the function will always go to the diagonal values and evaluate them. thus we can traverse it in any order as long as we find the diagonal values first. we can calculate the diagonal value of each row first then loop on that row
3. there is no way of traversing that will solve this problem as in each other call we will be back calculating the same values.
   e.g.
   $i = 3, j = 3$
   will lead to
   $i = 1, j = 1$
   and
   $i = 5, j = 5$
   which will give
   $i = 3, j = 3$
   and
   $i = 7, j = 7$

Question 4

$$MC(i,j) = \begin{cases} 0 & if i = j+1 \\ min_{i<k<j}(MC(k,j) + MC(i,k) + (d_j - d_i)) & otherwise \end{cases}$$

Time complexity:

The work needed in each call is
$\mathcal{O}(n)$
in order to do the summation and finding the minimum value.

and we need to evaluate a total of
$\mathcal{O}(n^2)$
points
$\therefore \mathcal{O}(n^3)$

Space complexity:

we need a matrix that is
$n * n$
to hold intermediate values
$\therefore \mathcal{O}(n^2)$

Question 5

We need first to calculate the total number of ways to get from
s
to
t

.
we do this by exploring every path from
s
to
t
in topological order.

$$Paths(s,t) = \begin{cases} 1 & if s = t \\ \sum_{(s,i)\in E} Paths(i,t) & otherwise \end{cases}$$

Time complexity:

since we need to visit every edge and vertices and all intermediate values will be stored so it wont be calculated more that once
$\mathcal{O}(|V| + |E|)$

Space complexity:

we need a matrix to hold the values of the vertices we are at to reach
t

$\therefore$
space complexity is
$\mathcal{O}(|V|)$