Q5

Zijian Yue

Z5188675

The problem can be solved by solving following subproblems.

Subproblem1: find all possible pairs of (u,v) from u ∈ V, v ∈ V, note this should include the case of u == v.

Subproblem2: for a pair of(u,v) find a path from u to v which obtains the maximum total weight which takes exactly k edges.

Subproblem3: find the maximum value among the (u,v) pairs

* For Q5 assume the Graph is given as a 2D array called Graph such that Graph[u][v] = weight .

* For Q5 Assume that all the arrays are started from index 0

* For Q5 all the pairs with their maximum weight will be stored in an array in the form of A[u][v] = W

Subproblem1(not recursion):

```
For every int u < V:
    For every int v < V:
            A[u][v] = -1
```

Subproblem2:

```
MaxPath(u,v,k) : max{MaxPath(u,v,k) + max(Array[u][v]) if Array[u][v] != -1}
BaseCase: MaxPath(u,v,k) = 0 if u == v and k == 0
                MaxPath(u,v,k) = Graph[u][v]  if k == 1 and Graph[u][v] != -1
                MaxPath(u,v,k) = -1 if k < 0
```

Note * the code for Subproblem2 is rewrite from the open source code

https://www.geeksforgeeks.org/shortest-path-exactly-k-edges-directed-weighted-graph/

Subproblem3(not recursion)

Return max(A[u][v]) for every u < V,v < V

Time Complexity O(V^K): for subproblem1 the time complexity is 0(V^2) , for subproblem2 the time complexity is O(V^k), for subproblem 3 the time complexity is O(V^2), this gives the overall time complexity O(V^K).

Python3 code for subproblem 2

```
def MaxPath(graph, u, v, k):
    V = 4
    INF = -1

    # Base cases
    if k == 0 and u == v:
        return 0
    if k == 1 and graph[u][v] != INF:
        return graph[u][v]
    if k <= 0:
        return INF

    # Initialize result
    res = INF

    # Go to all adjacents of u and recur
    for i in range(V):
        if graph[u][i] != INF:
            rec_res = MaxPath(graph, i, v, k - 1)
            if rec_res != INF:
                res = max(res, graph[u][i] + rec_res)
    return res

# Test MaxPath
# Define number of vertices in
# the graph and inifinite value
V = 4
```

```
INF = -1


# Let us create the graph shown
# in above diagram
graph = [[0, 10, INF, INF],
        [INF, INF, 10, INF],
        [INF, INF, INF, 4],
        [5, INF, INF, INF]]
u = 0
v = 3
k = 5
print("Weight of the shortest path is",
         MaxPath(graph, u, v, k))
```