Explanation (Assignment 4)

ID: 2210459

Task 1 (a): To print the graph which is directed and weight as adjacency matrix, we need a list and its elements will also be lists. Then initial elements of the 2nd list will be 0 (The number of total elements will be highest number+1). After taking the inputs, the elements will change according to a[u_i][v_i] = w_i. To add them in the output file, we need to use a nested loop.

1(b). To print the directed and weighted graph, we need a dictionary. After taking m, n from the input file, we will take the other values from the input file and store them in the dictionary as dictionary[u_i] = (v_i, w_i). Then we will store them in the output file with a nested loop.

Task 2: To perform bfs (breadth first search) we need a queue structure. After taking the inputs as directed, the graph will be stored as a dictionary. Then we it will be sent to the bfs method with necessary arguments along with the result array, where the result will be stored. The method will implement the bfs algorithm and store the nodes in the result array. Then it will be added to the output folder.

Task 3: DFS we is an recursive algorithm where the depest vertex of a node will be discovered first. To implement the method, we use a recursive method. As before we store the graph from the input as a dictionary. In the we create another dictionary, we store the initial color of each node as '0'. In the dfs method, We take a node as u and check weather it is visited or not. Before that, we add it to the result. we find its vertex, change its color and then recall the method with the vertex as parameter. finally, we change return the result.

Task 4: We use the dfs method to find a cycle in a graph. We use to dictionary to store the graph itself and to store the color of nodes. After storing them the graph and putting the initial color, we call the recursive dfs method. We used a global flag to store output. The method will change the nodes color as 'G' at the bogining. If any of it's vertex's ~~vertex's~~ color is found 'G' that means there's a cycle. It will be stored in flag and returned. At the end of recursion the nodes color will be changed to 'B'.

Task 5: BFS method is used to check the shortest path. the graph will be stored from the input as dictionary. Then the bfs method will be called with destination node. While traversing, the method will check weather the node is the destination or not. If it is, then it will return the path stored in queue. If the node is not found, it will return None.

Task 6: To find the number of diamonds, we will use dfs method. The graph will be stored as grid, in adjacency matrix style. To find the first diamond as start, we will use two nested loops. After finding it, we will call the dfs method with the index position of it as parameter. At first the method will check the conditions given in question. Then it will check if the selected value is 'D' or not and increase the value of counter if necessary. Then it will recursively call itself for the 4 neighbours of the current node. Finally it will return the value of counter.

Task 7: To find the two nodes who have the longest distance, we use the bfs method. At first we find out the farthest val node from the start. Then we again call the bfs with that node as start and find its farthest node. These two are the output value.