Assignment 3 + Quiz 03

Answer to the Question 1

```
~~def meet (G):~~
  ~~temp=dijkstra (G,1)~~
  ~~Temp2=dijkstra (G,50)~~
    ~~distan = inf~~
    ~~point = 0~~
    ~~for~~
```

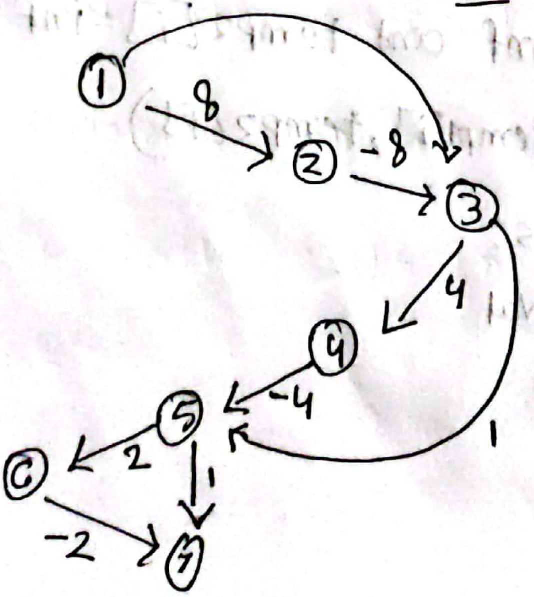
```
def  meet (G):
    temp1 = dijkstra(G,1)
    temp2 = dijkstra(G,50)
    idx = 0
    dis = inf
    point = 0
    for i in range (1, 50):
        if temp1[i]!= inf and temp2[i]=inf:
            val = max(temp1[i],temp2[i])
            if val<dis:
                dis = val
                idx = i
    return dis, idx
```

Answer to the Question 2

```
def MIM (G, kfc):

    min_dis = inf
    area = None
    temp1 = dijkstra (G, [illegible])

    temp2 = dijkstra (G, 50)

    for i in kfc:
        val = temp1 [i] + temp2[i]

        if val < min_dis:
            min_dis = val
            area = i

    return area
```

Answer to the Question 3

a



Hene:-

Initialization:-

distanee = [0, inf, inf, inf, inf, inf, inf]

step 1:-

distanee = [0, 8, 1, inf, inf, inf, inf]

Step 2:-

distance = [0, 1, 1, 5, 3, inf, inf]

step 3:-

distanee = [0, 1, 1, 2, 6, 3]

Setep 4:-

distanee = [0, 1, 1, 2, 2, 4, 3]

step 5:-

distanee = [0, 1, 1, -2, 2, 4, 3]

step 6:-

distance = [0, 1, -2, 2, 2, 4, 3]

∴ Final distanee = [0, 1, -7, 2, -2, 4]

b

Dijkstra algorithm is a greedy appoach which assumes all of the given weights are positive as a result wheren when a negative weight is provided It will not work properly.

c

With the changes the modified version of Dijkstra's algorith becomes an implementation in which the queue manages that every node in visited. Becase of its special architecture, the Bellman-Ford alogorith can accuratly determine the shortest path costs in graphs with negative edges.

d

The updated algorithm has time complexity O(V*E).

Answer to the Question 4

The divide and conquer method for detemining an undirected graphs minimum spanning Tree may or may not provide the correct result in all cases.

The algorithm assumes that dividing the set of vetices into two halves will alway give us three parts, ~~left, right~~ lelf and right half and the edges connecting them. But there is a possibility That splitting does not give us three parts which may give us an incorrect MST..

let's consider as example, {(1,2), (2,3), (3,4), (4,1), (2,4}} has four vertices and five edges.

If the vertices are divided in half, vertex 1 and 2 will be on the left and 3 and 4 on the right. The algorith will not be able to find the connect MST.

To conclude, even though the divide and conquer algorithm might be effective in some cases, It is not a ~~r~~ reliable way to find MST.

Answe to the Question 5

1. The appropriate graph theory algo for this problem will be Dijkstras alogorithm.

2. Here the edges will be the notes (1, 2, 5, 10) and the vertices will be the amount to make (21, 20, 7).

Here the Dijkstras algo finds the shortest from Vertex 0 to the ~~vetex~~ ~~dis~~ desired amount or vertex. The minimum number of notes required to reach amount k is the shortest path. The specific notes required can the found out by ehamining to ~~Aelated~~ edges in the shontest path.

If thene is not path from venten 0 to the disined venten ~~the~~ we can not make te nequined amount.