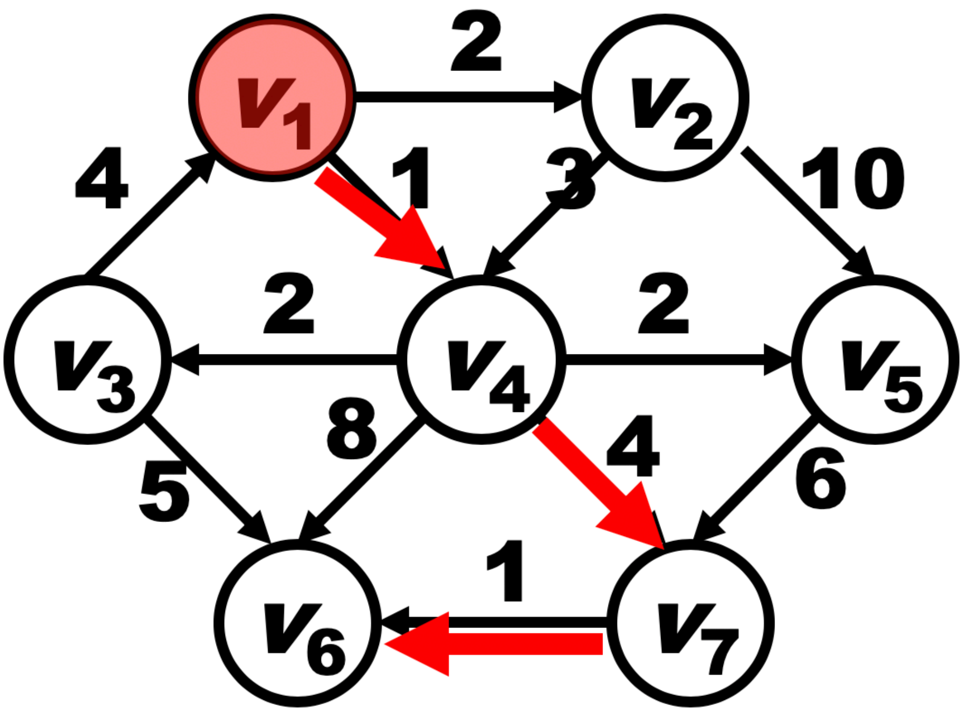
Lab 14 Report

1. Single-Source Shortest-Path Problem：

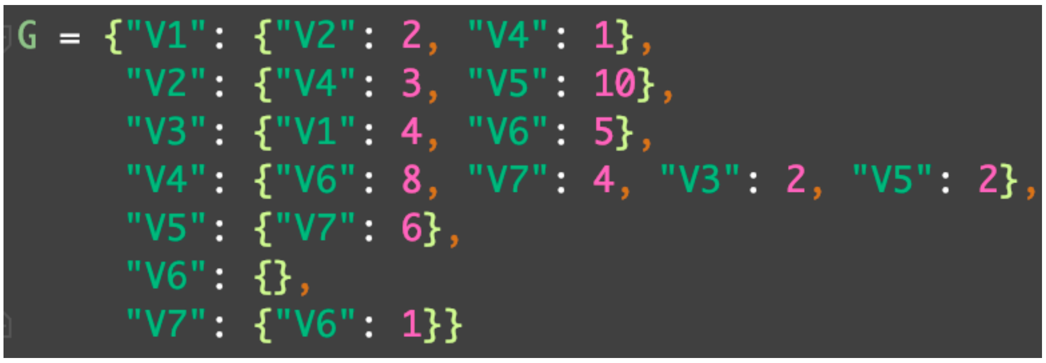
Given as input a weighted graph, G = (V, E), and a distinguished vertex, s, find the shortest weighted path from s to every other vertex in G.

1.1 Populate the graph representation：

I use the double dictionary to simulate the function of the multiple lists, and the weighted graph is given in the question.


$$\begin{tikzcd}
 & v_1 \arrow[r, "2"] \arrow[dr, "1"] & v_2 \arrow[d, "3"] \arrow[dr, "10"] & \\
v_3 \arrow[ur, "4"] \arrow[dr, "5"] & & v_4 \arrow[ll, "2"'] \arrow[r, "2"] \arrow[dll, "8"] \arrow[d, "4"] & v_5 \arrow[dl, "6"] \\
 & v_6 & v_7 \arrow[l, "1"] &
\end{tikzcd}$$

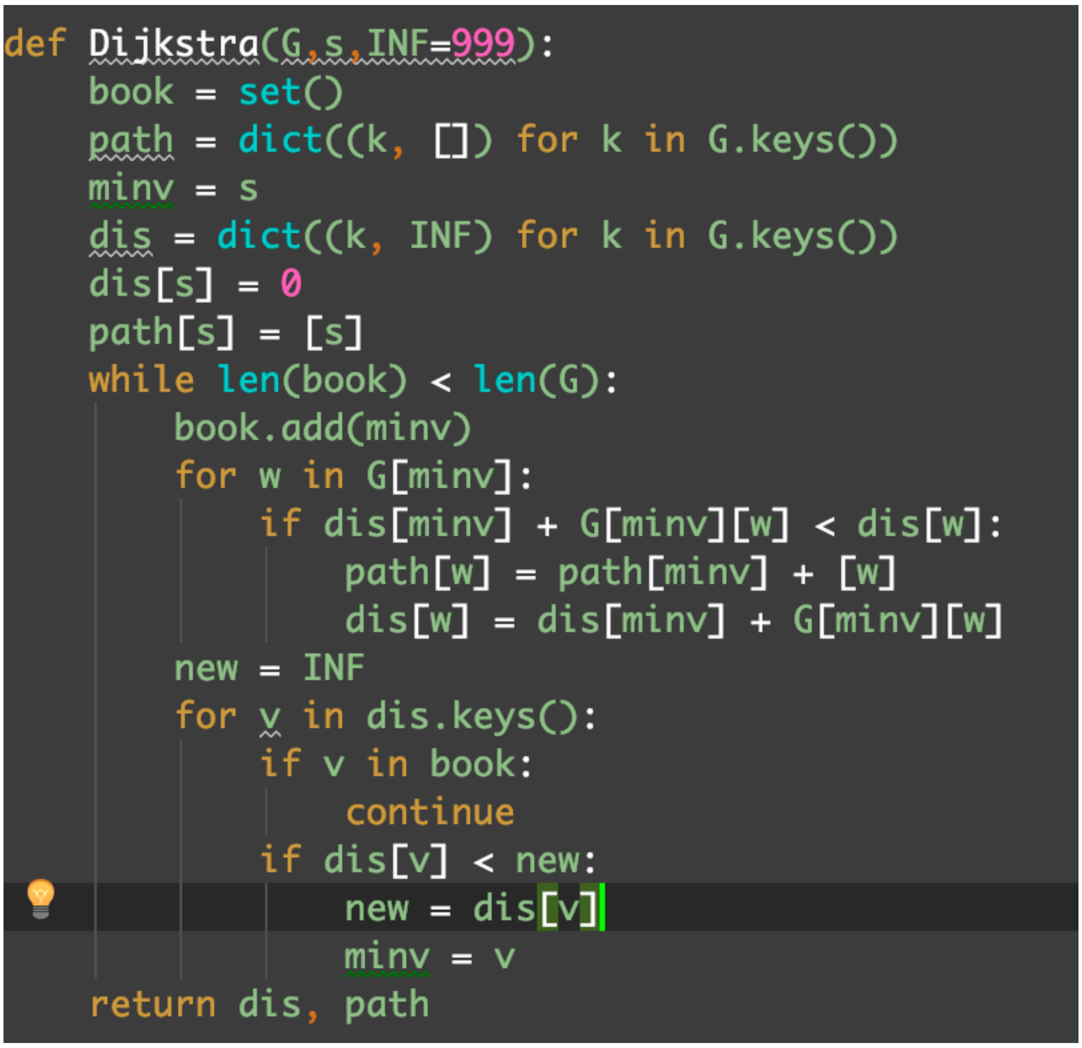
Code:


```
G = {"V1": {"V2": 2, "V4": 1},
     "V2": {"V4": 3, "V5": 10},
     "V3": {"V1": 4, "V6": 5},
     "V4": {"V6": 8, "V7": 4, "V3": 2, "V5": 2},
     "V5": {"V7": 6},
     "V6": {},
     "V7": {"V6": 1}}
```

1.2 Work and print out the paths.

To get the shortest weighted path from ***s*** to every other vertex in G, where the ***s*** is given as V1, I used the Dijkstra's method. It tries to find the min-cost path of every small sub-path in the whole path from ***s*** to the vertex to find the min-cost path form ***s*** to the vertex.

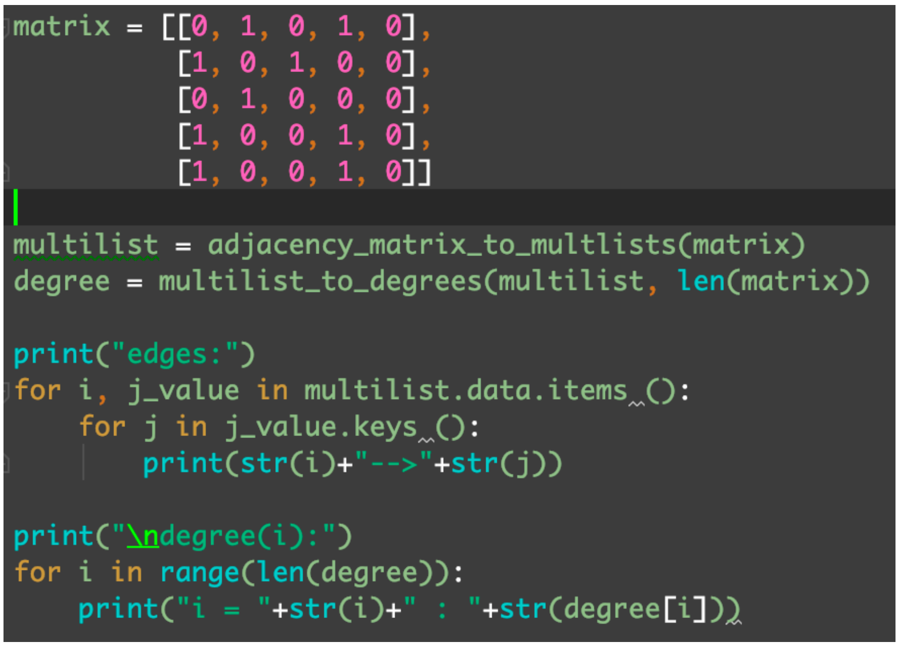
Code:



```
def Dijkstra(G,s,INF=999):
    book = set()
    path = dict((k, []) for k in G.keys())
    minv = s
    dis = dict((k, INF) for k in G.keys())
    dis[s] = 0
    path[s] = [s]
    while len(book) < len(G):
        book.add(minv)
        for w in G[minv]:
            if dis[minv] + G[minv][w] < dis[w]:
                path[w] = path[minv] + [w]
                dis[w] = dis[minv] + G[minv][w]
        new = INF
        for v in dis.keys():
            if v in book:
                continue
            if dis[v] < new:
                new = dis[v]
                minv = v
    return dis, path
```

In this function, the ***G*** is the graph while the ***s*** is the distinguished vertex. The output ***dis*** is a dictionary, whose ***keys*** are the vertexes and the relative ***values*** is the shortest weighted path from ***s*** to every other vertex in ***G***. The output ***path*** has the ***same keys*** but the ***values*** are the relevant paths.

Test:



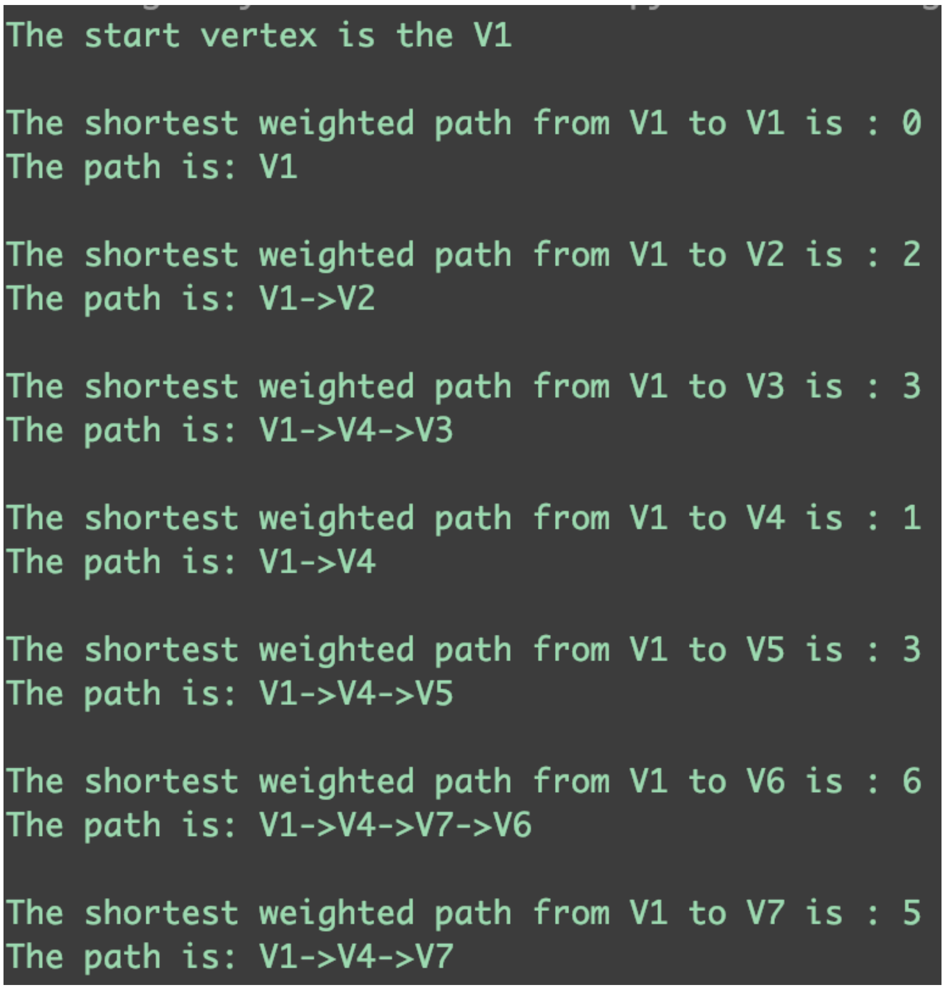
```
matrix = [[0, 1, 0, 1, 0],
          [1, 0, 1, 0, 0],
          [0, 1, 0, 0, 0],
          [1, 0, 0, 1, 0],
          [1, 0, 0, 1, 0]]

multilist = adjacency_matrix_to_multlists(matrix)
degree = multilist_to_degrees(multilist, len(matrix))

print("edges:")
for i, j_value in multilist.data.items():
    for j in j_value.keys():
        print(str(i)+"-->"+str(j))

print("\ndegree(i):")
for i in range(len(degree)):
    print("i = "+str(i)+" : "+str(degree[i]))
```

Output:



```
The start vertex is the V1

The shortest weighted path from V1 to V1 is : 0
The path is: V1

The shortest weighted path from V1 to V2 is : 2
The path is: V1->V2

The shortest weighted path from V1 to V3 is : 3
The path is: V1->V4->V3

The shortest weighted path from V1 to V4 is : 1
The path is: V1->V4

The shortest weighted path from V1 to V5 is : 3
The path is: V1->V4->V5

The shortest weighted path from V1 to V6 is : 6
The path is: V1->V4->V7->V6

The shortest weighted path from V1 to V7 is : 5
The path is: V1->V4->V7
```