ID: 20101266

Task 3

With reference to the pseudocode given in Task 1,

DIK (G, s)

$$\left.\begin{array}{l} \text{for each } v \in V[G] \\ \quad \text{do } d[v] \leftarrow \infty \\ \quad \pi[v] \leftarrow \text{nil} \end{array}\right] O(V)$$

$$\left.\begin{array}{l} d[s] \leftarrow 0 \\ S \leftarrow \phi \\ Q \leftarrow V[G] \end{array}\right] \rightarrow O(1)$$

while $Q \neq \phi$ —— $O(V)$

do $u \leftarrow$ extract-min $[Q] \rightarrow \log V$

$S \leftarrow S \cup \{u\}$ → vertices i.e $O(V)$

for each $v \in Adj[u]$ → edges i.e $O(E)$

do if $d[v] > d[u] + w(u,v)$

$d[v] \leftarrow d[u] + w(u,v)$

$\pi[v] \leftarrow u$

So Time for first task is $= O(V) + O(1) + O(V\log V) + O(E+V)$

$= O(V \log V)$

for task 2 we have same complexity, but an additional code block for calculating path, i.e,

node = z for each node in graph

```
def path(node, source):
    while (node != source)
        append node to path
        node = parent [node]
    return path (node, source).
```

runs for V vertices

also $T(n)$

$\rightarrow O(1)$

$\rightarrow T(n-1)$.

$T(n) = T(n-1) + O(1)$

$T \rightarrow O(n)$. i.e $O(V) \times V = V^2$

so for Task 2,

Time is $O(V) + O(1) + O(V \log V) + O(E+V) + O(V^2)$

If all the edges are of same weight, then we can take the graph to be an unweighted graph and for that we can use BFS whose time complexity is $O(V+E)$

$$\equiv O(N+M)$$

So in input we will just give the N and M and the weight is not needed. This will also give us our solution for a graph with edges of all 1.