Experiment 9.

Ans| Dijkshtra Algorithm :- is a solution to the single source shortest path problem in graph theory.

Works on both directed and undirected graphs. However all edges must have non-negative weights.

Approach :- Greedy

Input :- Weighted graph $G = \{E, V\}$ and source vertex $v \in V$ such that all edge weights are non-negative

Output :- Lengths of shortest paths from a given source vertex $v \in V$ to all other vertices.

Pseudocode :-

```
Dijkstra (G,w,s)
    InitializeSingleSource (G,S)
    S = ϕ
    Q = G.V
    while Q ≠ ϕ
        u = ExtractMin(Q)
        S = S U {u}
    for each vertex v ∈ G.Adj[u]
    Relax (u,v,w)
```

InitializeSingleSource (G, S)

```
InitializeSingleSource (G, S)
    for each vertex v ∈ G.V
        v.d = ∞
        v.π = NIL
    s.d = 0
```

Relax(u,v,w)

```
Relax(u,v,w)
    if v.d > u.d + w(u,v)
        v.d = u.d + w(u,v)
        v.π = u
```

Bellmanford Algorithm helps us find the shortest path from a vertex to all other vertices of a weighted graph.
It is similar to Dijkstra's algorithm but it can work with graphs in which edges can have negative weights.

Pseudocode

```
function bellmanford (G, S)
    for each vertex V in G
        distance[V] ← ∞
        previous[V] ← NULL
    distance [S] ← 0
    for each vertex V in G
        for each edge (U,V) in G
            tempDistance ← distance[U] + edge-weight(U,V)
            if tempDistance < distance[V]
```

distance[V] ← tempDistance
previous[V] ← U
for each edge(U,V) in G
if distance[U] + edgeWeight(U,V) < distance[V]
Error: Negative Cycle Exists
return distance[], previous[]

Time Complexity:- Best Case Complexity O(E)
Av. case Complexity O(VE)
Worst Case complexity O(VE)

Space Complexity O(V)