Dijkstra's Algorithm

P. Sai Deekshith
1BM18CS148

```
# include <bits/stdc++.h>
int main()
{
    int i, j, v;

    cout << "Enter the no. of vertices:";
    cin >> v;
    int graph[v][v];
    cout << "Enter the adjacency Matrix:";
    for(int i=0 ;i< v; i++)
    {   for(int j=0 ; j< v; j++)
        {   cin >> graph[i][j];

        }

    }
                        # source
                           ↓
    ShortestPath( graph , 0,  v);

    return 0;
```

```
Void ShortestPaths(int graph[V][V], int src,
                   int V)
{
    int dist[V];
    bool visited[V];
    for (int i=0; i<V; i++)
    { dist[i] = INT_MAX;
      visited[i] = false;
    }
    dist[src] = 0;
    for(int i=0; i<V-1; i++)
    {
        int u = minKey(dist, visited);
        visited[u] = true;
        for (int v=0; v<V; v++)
            if(visited[v] == false &&
               graph[u][v] &&
               dist[u] != INT_MAX &&
               dist[u] + graph[u][v] < dist[v])
               dist[v] = dist[u] + graph[u][v]
}
```

```
printSol(dist);

int minKey(int dist[], bool visited[])
{
    int min = INT_MAX;
    int min_index;
    for (int i=0; i<V; i++)
        if (visited==false && dist[i]<=min)
        {
            min = dist[v];
            min_index = v;
        }
    return min_index;
}

void printSol(int dist[])
{
    cout << "Vertex" << "    " <<
    "Distance from Source \n" << endl;
    for (int i=0; i<V; i++)
        cout<< i << "    " <<dist[i]
        <<endl;
}
```