J.S. Meghana
1BM18CS039

Dijkstra's Algorithm

```
# include < bits/stdc++.h>
using namespace std;
# define V 5
int mindist (int dist[], bool sptset[])
{
  int min = 9999, min_index;
  for (int v=0; v< V; v++)
  if (sptset [v] == false && dist [v] <= min)
      min = dist [v], min_index =v;
    return min_index;
  }
  Void printPath (int parent[], int j)
  { if (parent [j] == -1)
    return;
```

J.S. Meghana
1BM18CS039

```
PrintPath (parent, parent [j]);
cout << j << endl;
int printSol(int dist[], int n, int parent[])
{
  int Src = 0;
cout << "\n" << Src << "->" << i << "\t\t" << dist[i] << "\t\t"
                                                << Sc << endl;
    PrintPath (parent, i);
  }
}
  Void dijkstra (int graph[V][V], int Sc)
  {
  int dist [V];
  bool SptSet [V];
  int parent [V];
for (int i=0; i<V; i++)
{ parent [0] = -1;
dist [i] = 9999;
SptSet [i] = false
}
```

Meg.

J.J. Meghana
1BM18CS039

```
dist [src] = 0;
for (int count = 0; count < V-1; count++)
{
  int u = mindist (dist, sptset);
  sptset [u] = true;
for (int v = 0; v<V; v++)
  if (! sptset [v] && graph [u][v] && dist [u] + graph [u][v]
                                    < dist [v])
{
  parent [v] = u;
  dist [v] = dist [u] + graph [u][v];
}
}
printsol (dist, v, paunt);
}
int main ()
{ int graph [v][v];
Cout << "Enter the graph: " << endl;
                    (99 for infinity)
```

JS. Meghana
1BM18CS039

```
for (int i=0; i<v; i++)
{
  for (int j=0; j<v; i++)
    cin>> graph [i][j];
}
Cout << "Enter the Source: " << endl;
  int src;
dijkstra (graph, src);
  return 0;
}.
```