Implement the Dijkstra Algorithm used for Link State Routing

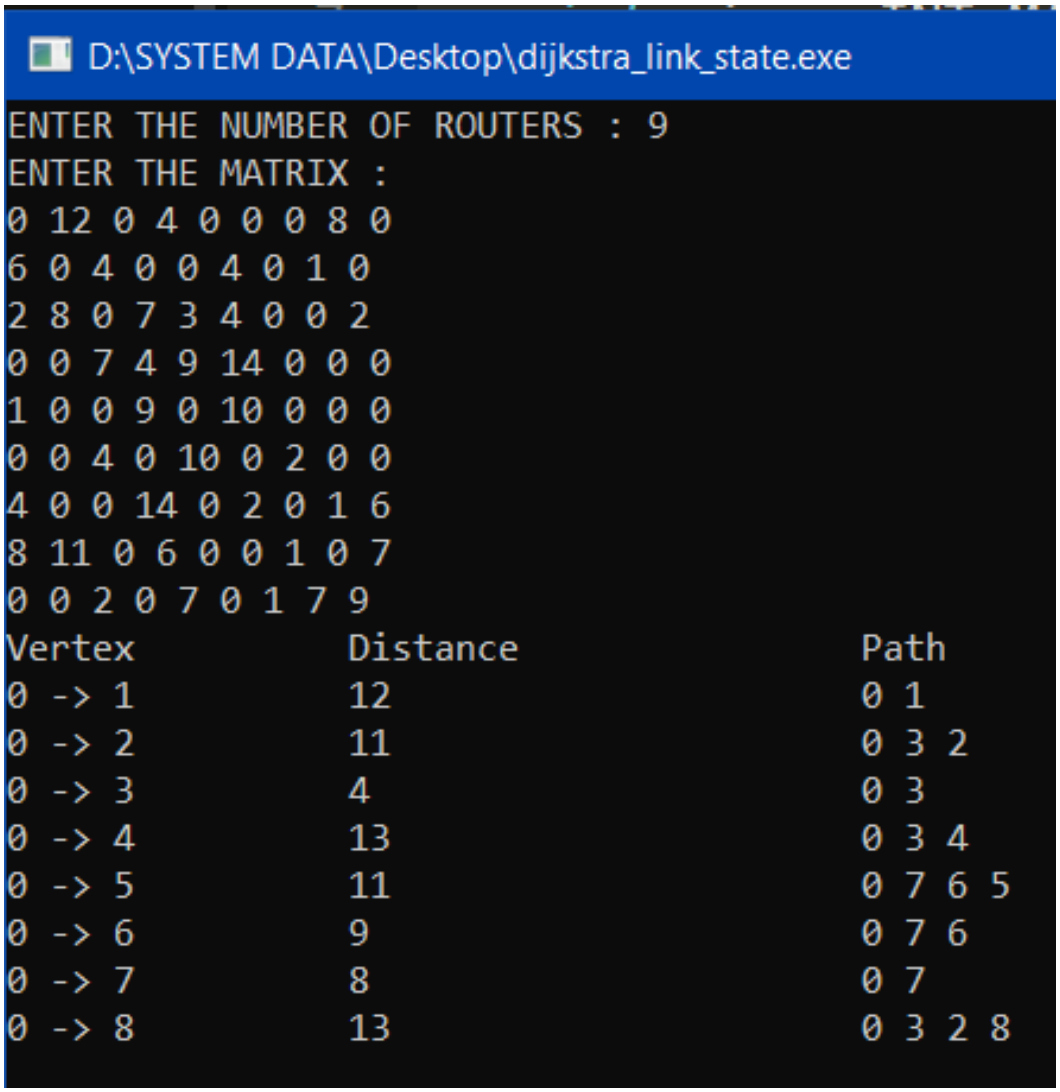
Code:

```
#include<bits/stdc++.h>
using namespace std;
int len;

int mindis(int distance[], bool ggset[])
{
   int min = INT_MAX, min_index;

   for (int v = 0; v < len; v++)
      if (ggset[v] == false && distance[v] <= min)
         min = distance[v], min_index = v;

   return min_index;
}

void print(int parent[], int j)
{
   if (parent[j] == - 1)
      return;

   print(parent, parent[j]);
```

```
    cout<<" "<<j;
}



int main()
{
        cout<<"ENTER THE NUMBER OF ROUTERS : ";
    cin>>len;
    vector< vector<int> > route(len,vector<int>(len,0));
    cout<<"ENTER THE MATRIX : \n";
    for(int i=0;i<len;i++){
        for(int j=0;j<len;j++){
            cin>>route[i][j];
        }
    }
    int src=0;
    int distance[len];
    bool ggset[len];
    int parent[len];
    for (int i = 0; i < len; i++)
    {
        parent[0] = -1;
        distance[i] = INT_MAX;
        ggset[i] = false;
```

```
    }

    distance[src] = 0;

    for (int count = 0; count < len - 1; count++)
    {
        int u = mindis(distance, ggset);
        ggset[u] = true;
        for (int v = 0; v < len; v++)
            if (!ggset[v] && route[u][v] &&
                distance[u] + route[u][v] < distance[v])
            {
                parent[v] = u;
                distance[v] = distance[u] + route[u][v];
            }
    }
    int st = 0;
    cout<<"Vertex\t\tDistance\t\tPath\n";
    for (int i = 1; i < len; i++)
    {
        cout<<st<<" -> "<<i<<"\t\t"<< distance[i]<<"\t\t\t"<< st;
        print(parent, i);
        cout<<"\n";
    }
    return 0;
}
```

Result:


D:\SYSTEM DATA\Desktop\dijkstra_link_state.exe
ENTER THE NUMBER OF ROUTERS : 9
ENTER THE MATRIX :
0 12 0 4 0 0 0 8 0
6 0 4 0 0 4 0 1 0
2 8 0 7 3 4 0 0 2
0 0 7 4 9 14 0 0 0
1 0 0 9 0 10 0 0 0
0 0 4 0 10 0 2 0 0
4 0 0 14 0 2 0 1 6
8 11 0 6 0 0 1 0 7
0 0 2 0 7 0 1 7 9
Vertex Distance Path
0 -> 1 12 0 1
0 -> 2 11 0 3 2
0 -> 3 4 0 3
0 -> 4 13 0 3 4
0 -> 5 11 0 7 6 5
0 -> 6 9 0 7 6
0 -> 7 8 0 7
0 -> 8 13 0 3 2 8