Name:-Dikesh Ganboi
Roll NO:- A36

```
#include<iostream>
#include<climits>
using namespace std;

// Function to find the vertex with minimum distance
int min_dist(int dist[], bool visited[]) {
    int minimum = INT_MAX, ind;
    for(int k = 0; k < 6; k++) {
        if(visited[k] == false && dist[k] <= minimum) {
            minimum = dist[k];
            ind = k;
        }
    }
    return ind;
}

// Function to find shortest paths from source using Dijkstra's algorithm
void greedy_dijsktra(int graph[6][6], int src) {
    int dist[6]; // Array to store distances from source
    bool visited[6]; // Array to track visited vertices

    // Initialize dist[] and visited[]
    for(int k = 0; k < 6; k++) {
        dist[k] = INT_MAX; // Initialize distances to infinity
        visited[k] = false; // Mark all vertices as not visited
    }

    dist[src] = 0; // Distance from source to itself is 0

    // Iterate over all vertices
    for(int k = 0; k < 6; k++) {
        // Find the vertex with the minimum distance among unvisited vertices
        int m = min_dist(dist, visited);
        visited[m] = true; // Mark the selected vertex as visited

        // Update distances of neighboring vertices
        for(int k = 0; k < 6; k++) {
            // Update the distance if the vertex is not visited,
            // there is an edge from m to k, and the new distance is smaller
            if(!visited[k] && graph[m][k] && dist[m] != INT_MAX && dist[m] +
graph[m][k] < dist[k])
```

```
                dist[k] = dist[m] + graph[m][k];
            }
        }

        // Print the shortest distances from the source vertex
        cout << "Vertex\t\tdist from source vertex" << endl;
        for(int k = 0; k < 6; k++) {
            char str = 65 + k; // Convert vertex index to corresponding character
(A, B, C, ...)
            cout << str << "\t\t\t" << dist[k] << endl;
        }
}

int main() {
    int graph[6][6] = {
        {0, 1, 2, 0, 0, 0},
        {1, 0, 0, 5, 1, 0},
        {2, 0, 0, 2, 3, 0},
        {0, 5, 2, 0, 2, 2},
        {0, 1, 3, 2, 0, 1},
        {0, 0, 0, 2, 1, 0}
    };

    // Find shortest paths from source vertex 0
    greedy_dijsktra(graph, 0);

    return 0;
}
```

OUTPUT:-

```
PS C:\Users\HP\Desktop\DAA EXperiment> cd "c:\Users\HP\Desktop\DAA
EXperiment\" ; if ($?) { g++ dijasktras.cpp -o dijasktras } ; if
($?) { .\dijasktras }
Vertex          dist from source vertex
A                       0
B                       1
C                       2
D                       4
E                       2
F                       3
```