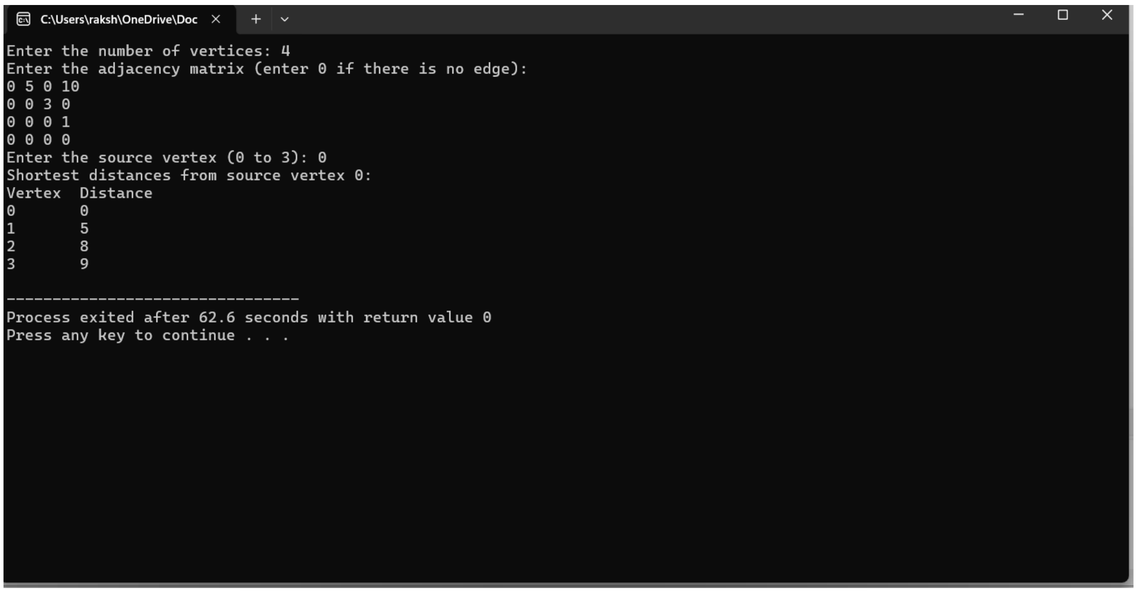
Dijkstra’s Algorithm

CODE:

```
#include <stdio.h>
#include <limits.h>
#define MAX 100
int minDistance(int dist[], int visited[], int V) {
    int min = INT_MAX, min_index;
    for (int v = 0; v < V; v++)
        if (!visited[v] && dist[v] <= min) {
            min = dist[v];
            min_index = v;
        }
    return min_index;
}
void printSolution(int dist[], int V, int src) {
    printf("Shortest distances from source vertex %d:\n", src);
    printf("Vertex\tDistance\n");
    for (int i = 0; i < V; i++)
        printf("%d\t%d\n", i, dist[i]);
}
void dijkstra(int graph[MAX][MAX], int V, int src) {
    int dist[MAX];
    int visited[MAX];
    for (int i = 0; i < V; i++) {
        dist[i] = INT_MAX;
        visited[i] = 0;
    }
    dist[src] = 0;
    for (int count = 0; count < V - 1; count++) {
        int u = minDistance(dist, visited, V);
        visited[u] = 1;
        for (int v = 0; v < V; v++)
            if (!visited[v] && graph[u][v] && dist[u] != INT_MAX
                && dist[u] + graph[u][v] < dist[v])
                dist[v] = dist[u] + graph[u][v];
    }
    printSolution(dist, V, src);
}
int main() {
    int V, graph[MAX][MAX], src;
    printf("Enter the number of vertices: ");
    scanf("%d", &V);
    printf("Enter the adjacency matrix (enter 0 if there is no edge):\n");
    for (int i = 0; i < V; i++)
        for (int j = 0; j < V; j++)
            scanf("%d", &graph[i][j]);
    printf("Enter the source vertex (0 to %d): ", V - 1);
    scanf("%d", &src);
    dijkstra(graph, V, src);
    return 0;
}
```

OUTPUT:



```
Enter the number of vertices: 4
Enter the adjacency matrix (enter 0 if there is no edge):
0 5 0 10
0 0 3 0
0 0 0 1
0 0 0 0
Enter the source vertex (0 to 3): 0
Shortest distances from source vertex 0:
Vertex  Distance
0       0
1       5
2       8
3       9

--------------------------------
Process exited after 62.6 seconds with return value 0
Press any key to continue . . .
```