/*dijkastra*/

```c
#include <stdio.h>

void dijkstra(int G[100][100], int n, int source);
void relax(int v, int u, int w[100][100], int dist[]);
int extract_min(int q[], int dist[], int n);

void dijkstra(int G[100][100], int n, int source) {
    int distance[100];
    int visited[100];

    for (int i = 0; i < n; i++) {
        distance[i] = 99999;
        visited[i] = 0;
    }

    distance[source] = 0;

    for (int i = 0; i < n; i++) {
        int u = extract_min(visited, distance, n);
        visited[u] = 1;

        for (int v = 0; v < n; v++) {
            if (!visited[v] && G[u][v] != -1) {
                relax(v, u, G, distance);
            }
        }
    }

    printf("Vertex \t Distance from Source\n");
    for (int i = 0; i < n; i++) {
```

```c
        printf("%d \t %d\n", i, distance[i]);
    }
}


void relax(int v, int u, int w[100][100], int dist[]) {
    if (w[u][v] + dist[u] < dist[v]) {
        dist[v] = w[u][v] + dist[u];
    }
}


int extract_min(int visited[], int dist[], int n) {
    int min = 99999;
    int min_index = -1;


    for (int v = 0; v < n; v++) {
        if (!visited[v] && dist[v] < min) {
            min = dist[v];
            min_index = v;
        }
    }


    return min_index;
}


int main() {
    int G[100][100];
    int n, source;


    printf("Enter the number of vertices: ");
    scanf("%d", &n);
```

```c
    printf("Enter the adjacency matrix (%d x %d):\n", n, n);

    for (int i = 0; i < n; i++) {

        for (int j = 0; j < n; j++) {

            scanf("%d", &G[i][j]);

        }

    }


    printf("Enter the source vertex: ");

    scanf("%d", &source);


    dijkstra(G, n, source);


    return 0;

}
```
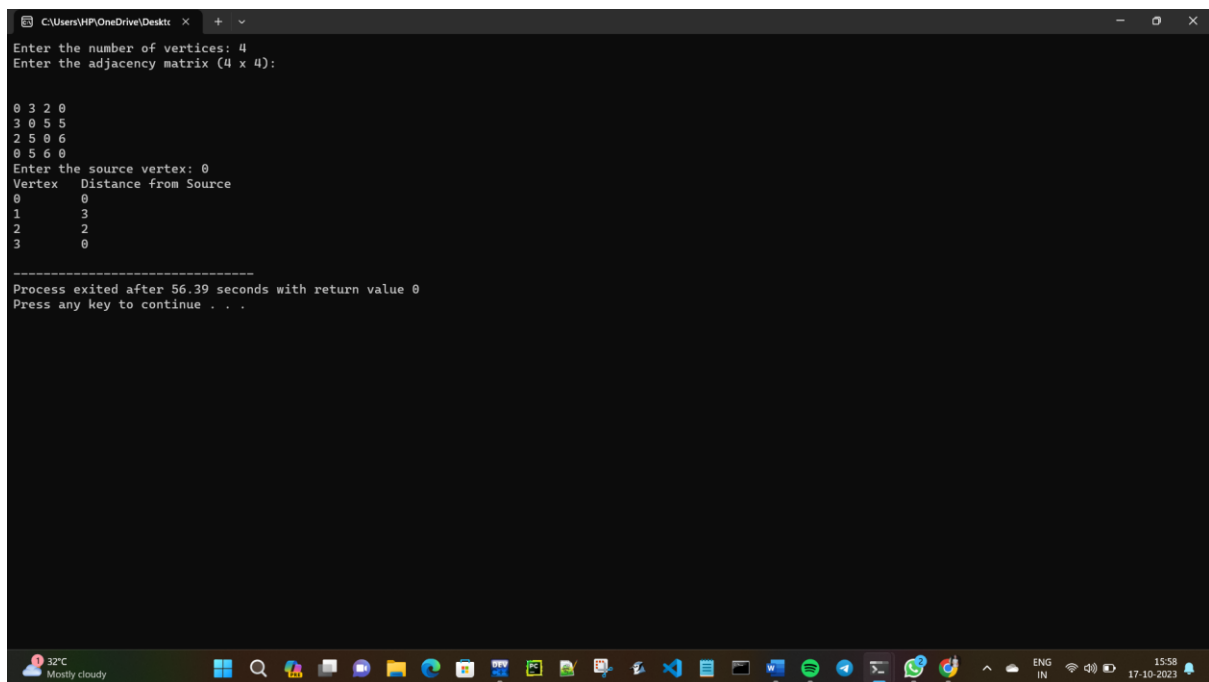
```
Enter the number of vertices: 4
Enter the adjacency matrix (4 x 4):

0 3 2 0
3 0 5 5
2 5 0 6
0 5 6 0
Enter the source vertex: 0
Vertex    Distance from Source
0         0
1         3
2         2
3         0

--------------------------------
Process exited after 56.39 seconds with return value 0
Press any key to continue . . .
```