NAME: Haresh Kumar N L (192425009)

COURSE NAME: DATA STRUCTURES FOR MODERN COMPUTING SYSTEMS

COURSE CODE: CSA0302

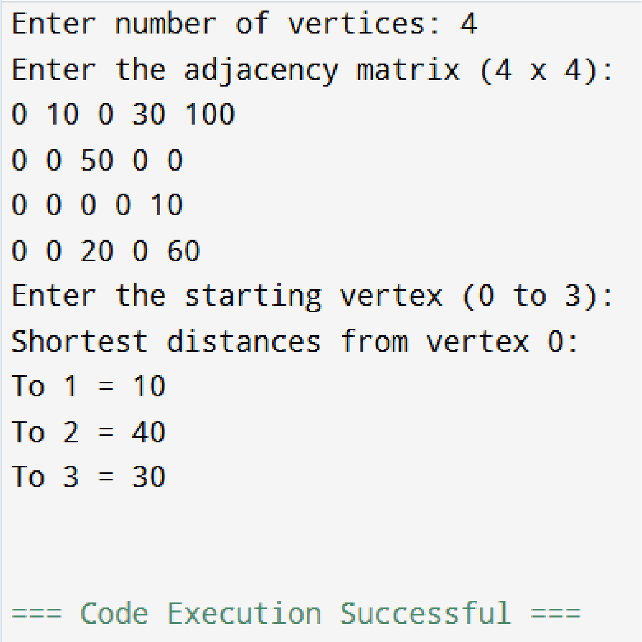
Experiment 36: Dijkstra Algorithm

Code:

```
#include <stdio.h>

#define INF 9999

#define MAX 20

void Dijkstra(int graph[MAX][MAX], int n, int start) {

   int cost[MAX][MAX], distance[MAX], visited[MAX], pred[MAX];

   int count, minDistance, nextNode, i, j;

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

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

         cost[i][j] = (graph[i][j] == 0) ? INF : graph[i][j];

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

      distance[i] = cost[start][i];

      pred[i] = start;

      visited[i] = 0;

   }

   distance[start] = 0;

   visited[start] = 1;

   count = 1;

   while (count < n - 1) {

      minDistance = INF;

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

         if (distance[i] < minDistance && !visited[i]) {

            minDistance = distance[i];

            nextNode = i;

         }

      visited[nextNode] = 1;
```

```
        for (i = 0; i < n; i++)
            if (!visited[i])
                if (minDistance + cost[nextNode][i] < distance[i]) {
                    distance[i] = minDistance + cost[nextNode][i];
                    pred[i] = nextNode;
                }

        count++;
    }
    printf("\nShortest distances from vertex %d:\n", start);
    for (i = 0; i < n; i++)
        if (i != start)
            printf("To %d = %d\n", i, distance[i]);
}
int main() {
    int graph[MAX][MAX];
    int n, start;
    printf("Enter number of vertices: ");
    scanf("%d", &n);
    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", &graph[i][j]);
    printf("Enter the starting vertex (0 to %d): ", n - 1);
    scanf("%d", &start);
    Dijkstra(graph, n, start);
    return 0;
}
```

Output:



```
Enter number of vertices: 4
Enter the adjacency matrix (4 x 4):
0 10 0 30 100
0 0 50 0 0
0 0 0 0 10
0 0 20 0 60
Enter the starting vertex (0 to 3):
Shortest distances from vertex 0:
To 1 = 10
To 2 = 40
To 3 = 30


=== Code Execution Successful ===
```