Name	Anand Tiwari
UID no.	2021700068
Experiment No.	6

AIM:	Greedy Approach- Single Source Shortest path-Dijkstra's Algorithm
Programs	
PROBLEM STATEMENT :	Find the shortest path from a single source vertex to all other vertices in a weighted graph using Dijkstra's algorithm
THEORY:	Dijkstra's algorithm allows us to find the shortest path between any two vertices of a graph. It differs from the minimum spanning tree because the shortest distance between two vertices might not include all the vertices of the graph. Dijkstra's Algorithm works on the basis that any subpath `B -> D` of the shortest path `A -> D` between vertices A and D is also the shortest path between vertices B and D. Djikstra used this property in the opposite direction i.e we overestimate the distance of each vertex from the starting vertex. Then we visit each node and its neighbors to find the shortest subpath to those neighbors. The algorithm uses a greedy approach in the sense that we find the next best solution hoping that the end result is the best solution for the whole problem.

	.

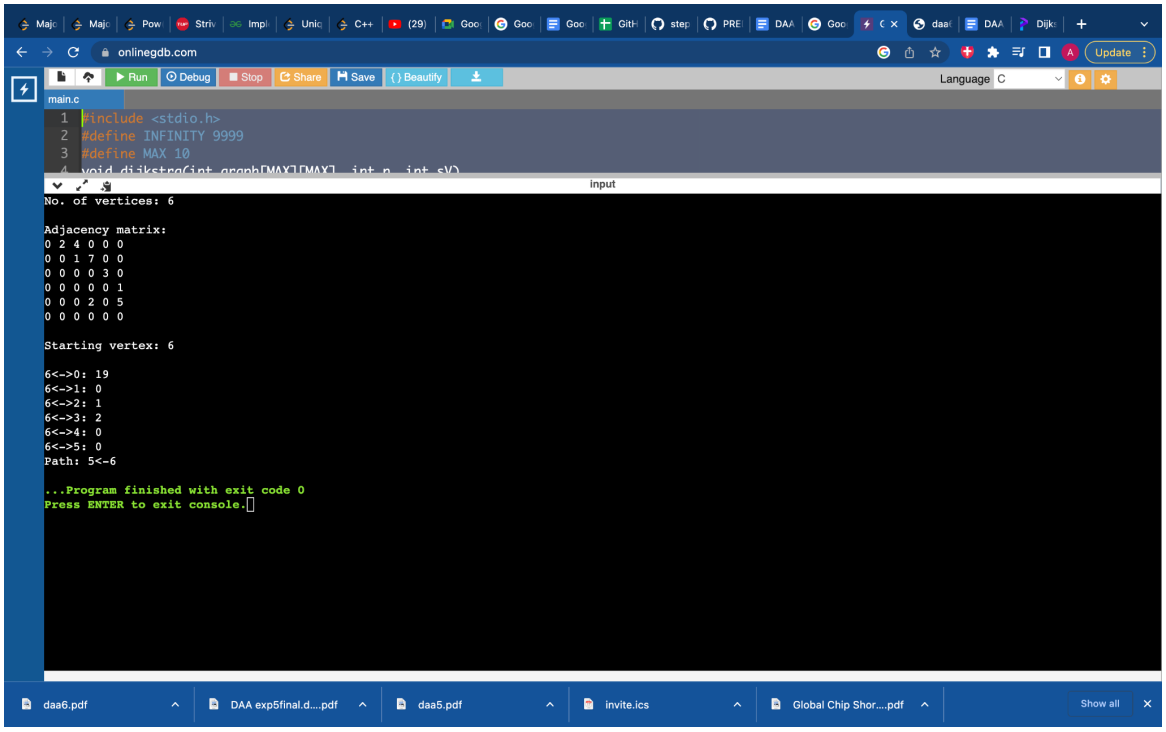
PROGRAM:

```
#include <stdio.h>
#define INFINITY 9999
#define MAX 10
void dijkstra(int graph[MAX][MAX], int n, int sV)
{
int cost[MAX][MAX], distance[MAX], pred[MAX];
int visited[MAX], count, mDis, nV, i, j;
int t = 0;
for (i = 0; i < n; i++)
for (j = 0; j < n; j++)
if (graph[i][j] == 0)
cost[i][j] = INFINITY;
else
cost[i][j] = graph[i][j];
for (i = 0; i < n; i++)
{
distance[i] = cost[sV][i];
pred[i] = sV;
visited[i] = 0;
}
distance[sV] = 0;
visited[sV] = 1;
count = 1;
while (count < n - 1)
{
mDis = INFINITY;
for (i = 0; i < n; i++)
if (distance[i] < mDis && !visited[i])
{
mDis = distance[i];
nV = i;
}
visited[nV] = 1;
for (i = 0; i < n; i++)
if (!visited[i])
```

```
if (mDis + cost[nV][i] < distance[i])
{
distance[i] = mDis + cost[nV][i];
pred[i] = nV;
}
count++;
}
for (int i = 0; i < n; i++)
{
if (i != sV)
{
printf("\n%d<->%d: %d", sV, i, distance[i]);
if (i == n - 1)
{
printf("\nPath: %d", i);
j = i;
while (j != sV)
{
j = pred[j];
printf("<-%d", j);
}
}
}
}
}
int main()
{
int graph[MAX][MAX], i, j, n, u;
printf("No. of vertices: ");
scanf("%d", &n);
printf("\nAdjacency matrix: \n");
for (i = 0; i < n; i++)
for (j = 0; j < n; j++)
scanf("%d", &graph[i][j]);
printf("\nStarting vertex: ");
```

```
scanf("%d", &u);
dijkstra(graph, n, u);
return 0;
}
```

RESULT



CONCLUSION:Successfully understood and implemented Dijkstra's algorithm in C