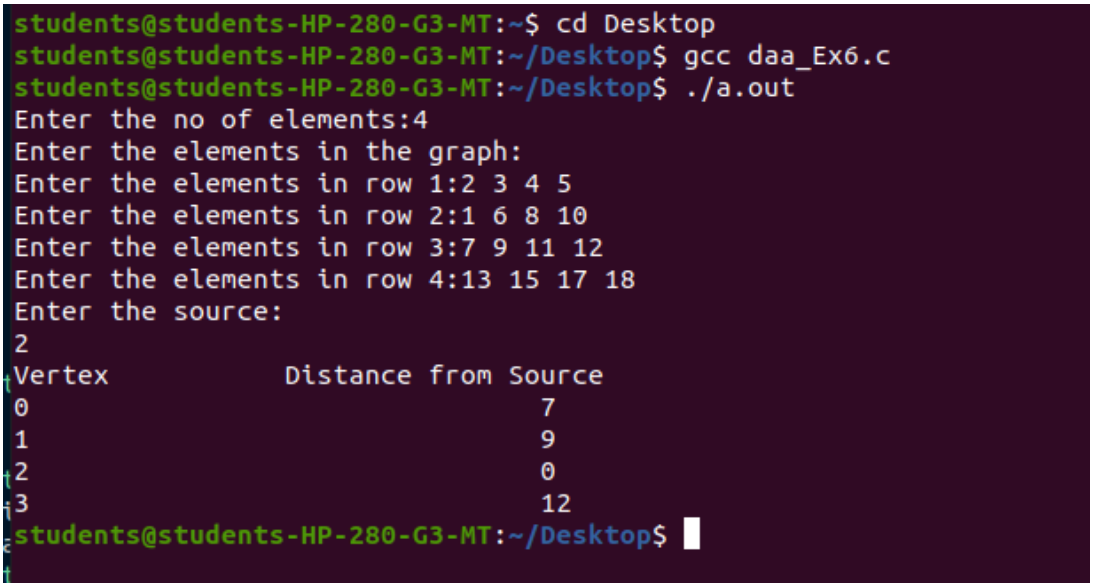
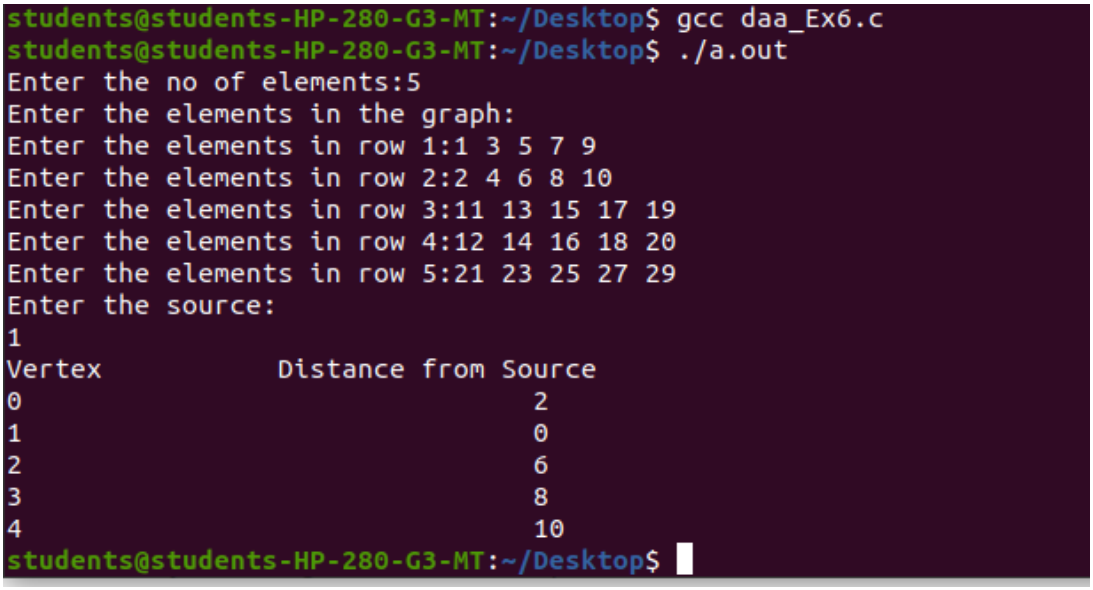
NAME:	Sanskar Kamble
UID:	2021300054
SUBJECT	Design and Analysis of Algorithm
EXPERIMENT NO :	06
DATE OF PERFORMANCE	27/03/2023
DATE OF SUBMISSION	03/04/2023
AIM:	To find the shortest possible path using Dijkstra algorithm.
ALGORITHM and THEORY:	function dijkstra(G, S) for each vertex V in G distance[V] <- infinite previous[V] <- NULL If V != S, add V to Priority Queue Q distance[S] <- 0 while Q IS NOT EMPTY U <- Extract MIN from Q for each unvisited neighbour V of U tempDistance <- distance[U] + edge_weight(U, V) if tempDistance < distance[V] distance[V] <- tempDistance previous[V] <- U return distance[], previous[]

PROGRAM:

```
#include <limits.h>
#include <stdbool.h>
#include <stdio.h>
int V;
int minDistance(int dist[], bool sptSet[])
{
        // Initialize min value
        int min = INT_MAX, min_index;

        for (int v = 0; v < V; v++)
                if (sptSet[v] == false && dist[v] <= min)
                        min = dist[v], min_index = v;

        return min_index;
}

// A utility function to print the constructed distance
void printSolution(int dist[])
{
        printf("Vertex \t\t Distance from Source\n");
        for (int i = 0; i < V; i++)
        printf("%d \t\t\t\t %d\n",i,dist[i]);
}

void dijkstra(int graph[V][V], int src)
{
        int dist[V];
        // The output array. dist[i] will hold the shortest distance from src
to i
   bool sptSet[V];
        for (int i = 0; i < V; i++)
                dist[i] = INT_MAX, sptSet[i] = false;

        // Distance of source vertex from itself is always 0
```

```
        dist[src] = 0;

        for (int count = 0; count < V - 1; count++)
        {
                int u = minDistance(dist, sptSet);
                sptSet[u] = true;
        for (int v = 0; v < V; v++)
                        if (!sptSet[v] && graph[u][v]
                                && dist[u] != INT_MAX
                                && dist[u] + graph[u][v] < dist[v])
                                dist[v] = dist[u] + graph[u][v];
        }
        printSolution(dist);
}

int main()
{

int s;
printf("Enter the no of elements:");
scanf("%d",&V);
int graph[V][V];
printf("Enter the elements in the graph:\n");
for(int i=0;i<V;i++)
{
printf("Enter the elements in row %d:",(i+1));
for(int j=0;j<V;j++)
{
scanf("%d",&graph[i][j]);
}
}
printf("Enter the source:\n");
scanf("%d",&s);
        dijkstra(graph,s);
```

	return 0; }
OUTPUT:	

	
CONCLUSION:	I have successfully understood Dijkstra's Algorithm and also found the shortest possible path using it.