Jai Jain
1BM18CS040

Page No.
Date :

Dijkstra's Algoritham to compute shortest path.

```
class Dijkstra
{   void main ()
    {   print ("Enter the no. of vertices")
        n = scan input
        arr [ ][ ] ;
        print ("Enter adjacency matrix")
        for (int i=0 ; i<n ; i++)
            for (int j=0 ; j<n ; j++)
                arr [i][j] = scan input
                if (arr [i][j] == 0)
                    arr [i][j] = 999
        print ("Enter source vertex")
        start = scan input
        dijkistra (n, arr, start)
    }
void dijkistra ( int n, int arr [][], int start )
{   visited [ ], parent = new int [n],
    parent [ ]
    distance [ ]
    int count = 0
    distance [start ] = 0
    for (int i=0 ; i<n ; i++)
        visted [i] = ~~start~~ 0
        parent [i] = start
        if (i != start)
            distance [i] = arr [start ][i],
    parent [start ] = -1
    visited [start] = 1
```

```
while (count < n-1)
    int min = 999 , index = 0 , i .
    for (i=0 ; i<n , i++)
        if (visited [i] != 1 && distance[i] < min)
            min = dist [i]
            index = i ,
    visited [index] = 1
    for (int j=0 ; j<n ; j++)
        if (visited [j] != 1 && arr [index][j] != 999
            && (distance [index] + arr[ind][j]
            < distance [j])
            distance [j] = distace [index]
            + arr [index] [j] ;
            parent [j] = index ;
    Count ++
    print ("Distance & path :")
for (int i=0 ; i<n ; i++)
    print (i + ":" + distance [i] + " "+
        start + " — ") ;
    print Path (parent , i) ;
    print ("\n")

void printPath (int parent [] ; int j)
    if (parent [j] == -1)
        return ;
    printPath (parent , parent [j])
    print (j + " — ") ;
```