MAP NAVIGATION SYSTEM

A PROJECT REPORT

Submitted by

S. YAFFIN [Reg No: RA2211032010053]

Under the guidance of

Dr. PRAVEENA AKKI

(Assistant Professor, Department of Networking and Communications, School of Computing)

in partial fulfillment of the requirements for the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

with specialization in INTERNET OF THINGS

TITUTE OF

AND TEC

SCIENCE

OSRM

INSTITUTE **OF** SCIENCE **& TECHNOLOGY**
Deemed to *be* **University** *u/s* 3 *of UGC*
Act, 1956

DEPARTMENT OF NETWORKING AND

COMMUNICATIONS

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY,

KATTANKULATHUR- 603 203

November 2023

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR - 603203

(Under Section **3 of** UGC Act, 1956)

BONAFIDE **CERTIFICATE**

Certified **that 21CSC203P** project report titled **"MAP NAVIGATION SYSTEM"** is the bonafide work of "S. **YAFFIN** [Reg **No: RA2211032010053],** who **carried** out the **project**

work under my supervision. Certified further, **that to the** bestof my knowledge **the** work reported

herein does not form **any** other project report or dissertation on **the** basis **of which** a degree or

award was conferred on an earlieroccasion on this **or** any other candidate.

16/4/21

Dr. PRAVEENA AKKI

GUIDE

Assistant Professor

Netw

orkin

g and

Comm

unic

Kattankulather 603
20:

Networking and Communications

6. Jay

3

DR. **ANNAPURANI PANAIYAPPAN** K

HEAD OF THE **DEPARTMENT**

Professor

Networking and Communications

INTERNAL EXAMINER

EXTERNAL EXAMINER

SRM

INSTITUTE OF SCIENCE & TECHNOLOGY
(Dermed to be University / of LC **AL**
1956)

Annexure II

Department of Networking and Communications

SRM Institute of Science & Technology

OWN WORK DECLARATION

Degree/ Course: B.Tech/Computer Science Engineering with specialization in Internet of Things

Student Name: S. YAFFIN

Registration Number: RA2211032010053

Title of Work: Map Navigation System

We hereby certify that this assessment compiles with the University's Rules and Regulations relating to Academic misconduct and plagiarism, as listed in the University Website, Regulations, and the Education Committee guidelines.

We confirm that all the work contained in this assessment is our own except where indicated, and that We have met the following conditions:

Clearly references / listed all sources as appropriate

- Referenced and put in inverted commas all quoted text (from books, web, etc.)

- Given the sources of all pictures, data etc. that are not my own

Not made any use of the report(s) or essay(s) of any other student(s) either past or present

Acknowledged in appropriate places any help that I have received from others (e.g., fellow students, technicians, statisticians, external sources).

Compiled with any other plagiarism criteria specified in the Course handbook / University website

I understand that any false claim for this work will be penalized in accordancewith the University policies and regulations.

DECLARATION:

I am aware of and understand the University's policy on Academic misconduct and plagiarism and I certify that this assessment is my own work, except where indicated by referring, and that I have followed the good academic practices noted above.

5. Yaffin

S.YAFFIN

ACKNOWLEDGEMENT

We express our humble gratitude to **Dr C. Muthamizhchelvan**, Vice-Chancellor, SRM Institute of Science and Technology, for the facilities extended for the project work and his continued support. We extend our sincere thanks to Dean-CET, SRM Institute of Science and Technology, **Dr T.V.Gopal**, for his invaluable support. We wish to thank **Dr Revathi Venkataraman,** Professor & Chairperson, School of Computing, SRM Institute of Science and Technology, for her support throughout the project work. We are incredibly grateful to our Head of the Department, **Dr K. Annapurani Panaiyappan,** Professor, Department of Networking and Communications, SRM Institute of Science and Technology, for her suggestions and encouragement at all the stages of the project work.

We want to convey our thanks to our program coordinators **Dr. A. Suresh,** Associate Professor, Department of Networking and Communications, SRM Institute of Science and Technology, for their inputs during the project reviews and support. We register our immeasurable thanks to our Faculty Advisor, **Dr. Praveena Akki**, Assistant Professor, Networking & Communications, SRM Institute of Science and Technology, for leading and helping us to complete our course. Our inexpressible respect and thanks to my guide, **Dr. Praveena Akki**, Assistant Professor, Networking & Communications, SRM IST, for providing me with an opportunity to pursue my project under his mentorship. He provided me with the freedom and support to explore the research topics of my interest.

His passion for solving problems and making a difference in the world has always been inspiring. We sincerely thank the Networking and Communications Department staff and students, SRM Institute of Science and Technology, for their help during our project. Finally, we would like to thank parents, family members, and friends for their unconditional love, constant support, and encouragement.

5. Yaffin

S. YAFFIN

ABSTRACT

The Map Navigation System project is a sophisticated application of data structures, notably Dijkstra's algorithm, aimed at revolutionizing route planning and navigation. By efficiently representing locations and their connections as a graph, this system optimizes pathfinding

within complex networks. The core of the system relies on Dijkstra's algorithm, which systematically explores adjacent nodes, calculates cumulative distances, and selects the shortest path, all of which is enhanced by the use of efficient data structures such as priority queues or min-heaps. This results in the rapid identification of the most efficient routes. The user interface is designed for ease of use, allowing users to input their starting and ending points. The system then processes this input and displays the optimal route on a map, taking into account various factors like distance and travel time. Furthermore, the system can incorporate real-time data, such as traffic conditions and road closures, to ensure the most up-to-date navigation instructions. By integrating with GPS services and other map-related data sources, this Map Navigation System offers a powerful tool for commuters, travelers, and logistics experts alike, saving time and improving overall travel experiences.

TABLE OF CONTENTS

CHAPTER NUMBER	**CONTENTS**	**PAGE NO**
1	INTRODUCTION	ix
2	APPLICATIONS	x
3		xi

Dijkstra's Algorithm

4

FLOW CHART FOR

xii

THE PROJECT

5

7

8

Program

OUTPUT SCREENSHOTS

REFERENCES

xiii

xx

xxii

viii

ix

CHAPTER 1

INTRODUCTION

1.1 EFFORTLESS NOTE-TAKING WITH NOTES MANAGER:

The Notes Manager is a dynamic and user-friendly mobile application designed to revolutionize your note-taking experience. It offers a seamless interface with dedicated activities for creating accounts, logging in, managing notes, and customizing your preferences. From secure account creation to a powerful notes management system, this application streamlines the process of creating, editing, and organizing notes, ensuring that your information is always at your fingertips. Whether you're a student, professional, or anyone seeking an efficient digital note-taking solution, the Notes Manager is your all-in-one tool for staying organized and productive.

The Notes Manager is a mobile app that simplifies your note-taking. It provides account creation, secure login, and efficient notes management, helping you stay organized and productive. Whether you're a student or professional, it's the ideal tool for streamlining your digital note-taking process.

1.2 NEED FOR NOTES MANGER

A Notes Manager is a vital tool in today's digital age due to the need for efficient information organization, productivity, and accessibility. It offers users a streamlined method for organizing thoughts, ideas, and essential data, making it easily accessible from various devices and locations. Additionally, it enhances efficiency by simplifying note creation and management, ensures data security, provides quick search and retrieval options, and often supports collaborative efforts. Users can also customize their Notes Manager to align with their specific preferences, enhancing the overall note-taking experience. In essence, the Notes Manager addresses the fundamental need for effective, organized, and secure digital note-taking, serving both personal and professional requirements.

CHAPTER 2
APPLICATIONS

1. **GPS Navigation:** It can serve as the foundation for GPS navigation systems in vehicles and mobile applications, ensuring drivers find the most efficient routes to their destinations while accounting for real-time traffic conditions.

2. **Logistics and Supply Chain Management**: Logistics companies can use this system to optimize delivery routes, minimize transportation costs, and improve the overall efficiency of their operations.

3. **Emergency Services**: Emergency response teams can utilize this system to plan the fastest routes to accident scenes or medical facilities, potentially saving lives in critical situations.

4. **Tourism and Travel Planning:** Tourists can use this application to explore unfamiliar cities and find the best paths to their desired attractions, enhancing their travel experience.

5. **Public Transportation**: Public transportation systems can implement this technology to provide passengers with real-time information on bus or train routes and schedules, making public transit more user-friendly.

6. **Smart Cities**: In smart city initiatives, this system can help manage traffic flow, reduce congestion, and enhance overall urban mobility.

7. **Environmental Conservation**: Researchers and conservationists can use it to plan routes for data collection, wildlife monitoring, and environmental research expeditions in the most ecologically sensitive way.

8. **Location-Based Services**: Businesses can offer location-based services and promotions to users based on their current or intended destinations, creating

targeted marketing opportunities.

9. **Geographic Information Systems (GIS)**: In GIS applications, this system can assist in various spatial analysis tasks, including network analysis, facility location, and urban planning.

CHAPTER 3 Dijkstra's Algorithm

Dijkstra's algorithm is a popular algorithms for solving many single-source shortest path problems having non-negative edge weight in the graphs i.e., it is to find the shortest distance between two vertices on a graph. It was conceived by Dutch computer scientist Edsger W. Dijkstra in 1956.

The algorithm maintains a set of visited vertices and a set of unvisited vertices. It starts at the source vertex and iteratively selects the unvisited vertex with the smallest tentative distance from the source. It then visits the neighbors of this vertex and updates their tentative distances if a shorter path is found. This process continues until the destination vertex is reached, or all reachable vertices have been visited.

Basics requirements for Implementation of Dijkstra's Algorithm

1. Graph: Dijkstra's Algorithm can be implemented on any graph but it works best with a weighted Directed Graph with non-negative edge weights and the graph should be represented as a set of vertices and edges.

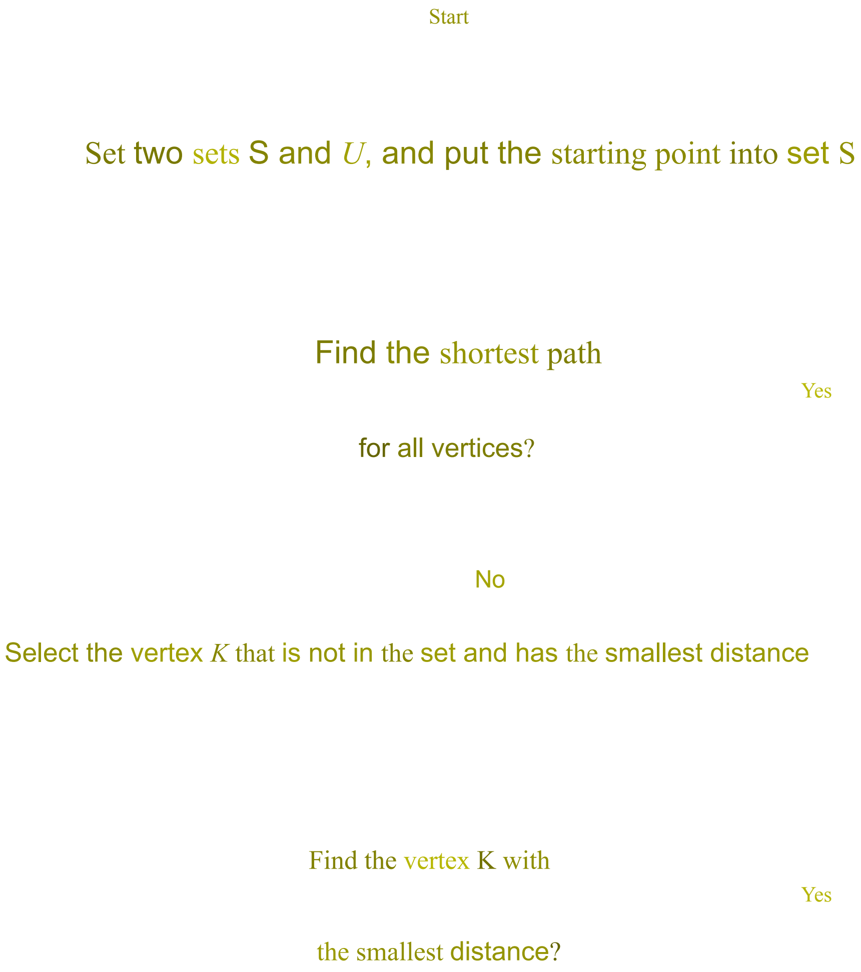
2. Source Vertex: Dijkstra's Algorithm requires a source node which is starting point for the search.

3. Destination vertex: Dijkstra's algorithm may be modified to terminate the search once a specific destination vertex is reached.

4. Non-Negative Edges: Dijkstra's algorithm works only on graphs that have positive weights this is because during the process the weights of the edge have to be added to find the shortest path. If there is a negative weight in the graph then the algorithm will not work correctly. Once a node has been marked as visited the current path to that node is marked as the shortest path to reach that node.

CHAPTER 4

FLOW CHART FOR THE PROJECT


Start
Set two sets S and *U*, and put the starting point into set S
Find the shortest path
Yes
for all vertices?
No
Select the vertex *K* that is not in the set and has the smallest distance
Find the vertex K with
Yes
the smallest distance?

No

Modify the distance of vertices not in set S

Add vertex K to set S

End

CHAPTER 6
PROGRAM

CODE:

```
#include<graphics.h>
#include<bits/stdc++.h>
#define V 15
using namespace std;
#include <stdio.h>
#include <limits.h>
#include <iostream>
int drline[100];
int k=0;

struct node
 {
     int data:
     struct node *left;
     struct node *right;
     struct node *up;
     struct node *down;

 }*n[16];

int ind=0;
int sol[15]={0};
int minDistance(int dist[], bool sptSet[])
 {
    int min = INT_MAX, min_index=0;

    for (int v = 0; v < V; v++)







}
     {
        if (sptSet[v]
                  ==
                             false && dist[v] <= min)
        {
            min = dist[v], min_index = v;



    return min_index;


void printPath(int parent[], int j)
{
    if(parent[j] -1)







}
        return;

    printPath(parent, parent[j]);
    drline[k]=parent[j];
    k++;
    cout << -> " << j;


void printSolution(int dist[], int n, int parent[], int src,int i)
{
    int min = INT_MAX;
    int nodo = -1:
    cout << "Vertex
        //int i=8;
                                        Distance from Source
                                                                                  Path" << endl;

        cout << i <<<<"\t\t\t" << dist[i] << "\t\t" << src;
        printPath(parent, i);
        drline[k]=i;
        cout << endl;

        if(dist[i] < min && dist[i] != 0)
        {
            min =
                    dist[i];

        }
            nodo = i;




}
    cout << endl;
    cout<<"shortest path: " << min << endl;


void dijkstra(int graph[V][V], int src, int dest)
{
    int dist[V];
    bool sptSet[V];
    int parent[V];
```

```
            parent[src] = -1;
            for (int i = 0; i < V; i++)
            {
                dist[i] = INT_MAX;
                sptSet[i] = false;
            }

            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]+graph[u][v] < dist[v])
                    {
                        dist[v] = dist[u] + graph[u][v];
                        parent[v] = u;
                    }
                }



        }
            }

            printSolution(dist, V, parent, src, dest);

        int main()
        {
            int src,dest;
            char cont;
            int gd=0,gm;
int x0=20+40,y0=420-20; int
                x1=100+40,y1=380-20; int
                x2=180+40,y2=340-20; int
                x3=260+40,y3-300-20; int
                x4-340+40,y4-240-20;
int x5=420+40,y5=180-20; int
                x6=340+40,y6=140-20; int
                x7-260+40,y7=200-20; int
                x8-240,y8=220; int
                x9=180,y9=260;
                int x10-140,y10=200;
                int x11=100,y11=120;
int            x12=110,y12=290;            int
                x13=260,y13=120;          int
                x14-260,y14-400;

initgraph(&gd,&gm," ");
                /*setcolor(WHITE);
circle (x0,y0,20); circle (x1,y1,20);
                line(x0,y0,x1,y1);
                circle(x2,y2,20); line
                (x1,y1,x2,y2); circle
                (x3,y3,20); line
                (x2,y2,x3,y3);
                circle(x4,y4,20);
                line(x3,y3,x4,y4);
                circle(x5,y5,20);
```

```
                line (x4,y4,x5,y5);
                circle (x6,y6,20); line
                (x5,y5,x6,y6);
                circle(x7,y7,20);
                line(x6,y6,x7,y7);
                circle(x8,y8,20);
                line(x7,y7,x8,y8);
                circle(x9,y9,20);
                line(x8,y8,x9,y9);
                circle(x10,y10,20);
                line (x9,y9,x10,y10);
                circle(x11,y11,20);
                line(x10,y10,x11,y11);
                circle(x12,y12,20); line
```

```
(x9,y9,x12,y12); line
(x9,y9,x2,y2);
circle(x13,y13,20);
line(x13,y13,x7,y7);
```

```
circle(x14,y14,20);
line(x14,y14,x2,y2);
outtextxy(x0,y0,"0");
outtextxy(x1,y1,"1");
outtextxy(x2,y2,"2");
outtextxy(x3,y3,"3");
outtextxy(x4,y4,"4");
outtextxy(x5,y5,"5");
outtextxy(x6,y6,"6");
outtextxy(x7,y7,"7");
outtextxy(x8,y8,"8");
outtextxy(x9,y9,"9");
outtextxy(x10,y10,"10");
outtextxy(x11,y11,"11");
outtextxy(x12,y12,"12");
outtextxy(x13,y13,"13");
outtextxy(x14,y14,"14");*/
```

```
   int mat[15][15]={{0,8,0,0,0,0,0,0,0,0,0,0,0,0,0},
      {8,0,14,0,0,0,0,0,0,0,0,0,0,0,0},
      {0,14,0,7,0,0,0,0,0,9,0,0,0,0,21},
      {0,0,7,0,10,0,0,0,0,0,0,0,0,0,0},
      {0,0,0,10,0,7,0,13,0,0,0,0,0,0,0},
      {0,0,0,0,7,0,4,0,0,0,0,0,0,0,0},
      {0,0,0,0,0,4,0,5,0,0,0,0,0,0,0},
      {0,0,0,0,13,0,5,0,2,0,0,0,0,8,0},
      {0,0,0,0,0,0,0,2,0,19,0,0,0,0,0},
      {0,0,9,0,0,0,0,0,19,0,15,0,4,0,0},
      {0,0,0,0,0,0,0,0,0,15,0,5,0,0,0},
      {0,0,0,0,0,0,0,0,0,0,5,0,0,0,0},
      {0,0,0,0,0,0,0,0,0,4,0,0,0,0,0},
      {0,0,0,0,0,0,0,8,0,0,0,0,0,0,0},
      {0,0,21,0,0,0,0,0,0,0,0,0,0,0,0}};

for(int i=0;i<15;i++)
{
   n[i]=new node;
   n[i]->data=i;
   n[i]->left=NULL;
   n[i]->right=NULL;
   n[i]->up=NULL;
   n[i]->down=NULL;

}

n[0]->right=n[1];
n[1]->left=n[0];n[1]->right=n[2];
n[2]->right=n[3];n[2]->left=n[1];n[2]->down=n[14];n[2]->up=n[9];
n[3]->right=n[4];n[3]->left=n[2];
n[4]->right=n[5];n[4]->left=n[3];n[4]->up=n[7];
n[5]->left=n[4];
n[6]->left=n[7];
n[7]->right=n[6];n[7]->left=n[8];n[7]->down=n[4];n[7]->up=n[13];
n[8]->right=n[7];n[8]->left=n[9];
n[9]->right=n[8];n[9]->left=n[12];n[9]->down=n[2];n[9]->up=n[10];
```

```
n[10]->down=n[9];n[10]->up=n[11];
n[11]->down=n[10];
n[12]->right=n[9];
n[13]->down=n[7];
n[14]->up=n[2];
int choice:
cout<<"\t\t\t//***Welcome to our transportation system***//"<<endl<<endl;
g:
setcolor(WHITE);
circle (x0,y0,20);
    circle (x1,y1,20);
    line(x0,y0,x1,y1);
    circle(x2,y2,20); line
    (x1,y1,x2,y2); circle
    (x3,y3,20); line
    (x2,y2,x3,y3);
    circle(x4,y4,20);
    line(x3,y3,x4,y4);
    circle(x5,y5,20); line
    (x4,y4,x5,y5); circle
    (x6,y6,20); line
    (x5,y5,x6,y6);
    circle(x7,y7,20);
```

```
    line(x6,y6,x7,y7);
    circle(x8,y8,20);
    line(x7,y7,x8,y8);
```

```
    line(x7,y7,x4,y4);
    circle(x9,y9,20);
    line(x8,y8,x9,y9);
    circle(x10,y10,20);
    line (x9,y9,x10,y10);
    circle(x11,y11,20);
    line(x10,y10,x11,y11);
    circle(x12,y12,20); line
    (x9,y9,x12,y12); line
```

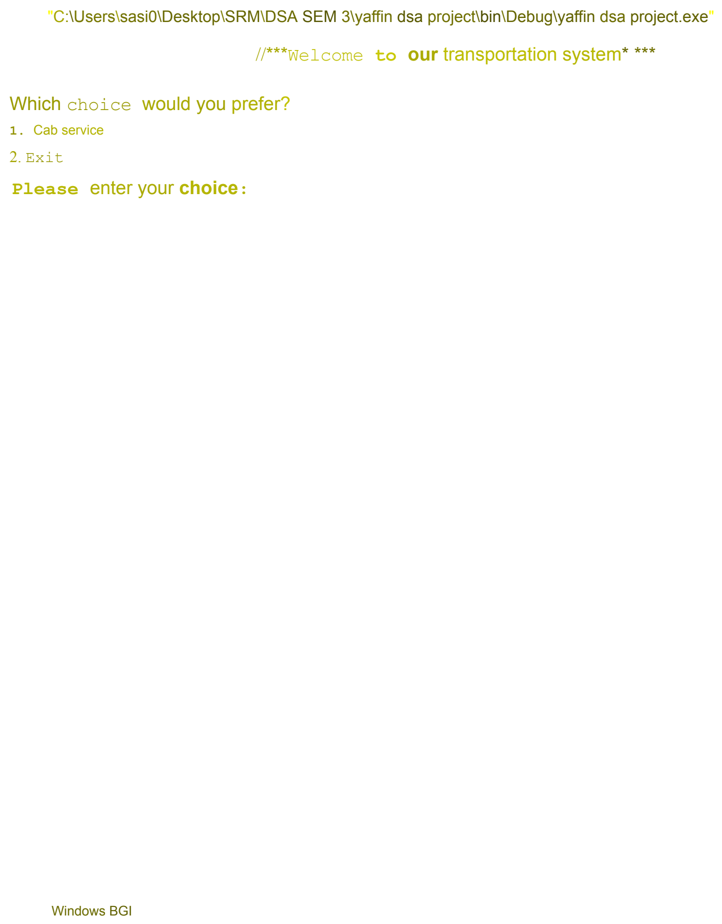
```
          (x9,y9,x2,y2);
          circle(x13,y13,20);
          line(x13,y13,x7,y7);
          circle(x14,y14,20);
          line(x14,y14,x2,y2);
          outtextxy(x0,y0,"0");
          outtextxy(x1,y1,"1");
          outtextxy(x2,y2,"2");
          outtextxy(x3,y3,"3");
          outtextxy(x4,y4,"4");
          outtextxy(x5,y5,"5");
          outtextxy(x6,y6,"6");
          outtextxy(x7,y7,"7");
          outtextxy(x8,y8,"8");
          outtextxy(x9,y9,"9");
          outtextxy(x10,y10,"10");
          outtextxy(x11,y11,"11");
          outtextxy(x12,y12,"12");
          outtextxy(x13,y13,"13");
          outtextxy(x14,y14,"14");
        cout<<"Which choice would you prefer?"<<endl;
        cout<<"1. Cab service "<<endl;
        cout<<"2. Exit"<<endl;
        //cout<<"Or press O to have a look at the map"<<endl;
        cout<<"Please enter your choice:"<<endl;
        cin>>choice;
        switch(choice)
        {
        case 1:
cout<<"Enter source you want to start from: "<<endl; cin>>src;
          cout<<"Enter destination you want to reach to: "<<endl; cin>>dest;
          dijkstra(mat,src,dest);
          setcolor(RED);
      for (int j=0;j<k;j++)
}
  if(drline[j]==0 && drline[j+1]==1 ||drline[j]==1 && drline[j+1]==0)
      line (x0,y0,x1,y1);
  if(drline[j]==1 && drline[j+1]==2 || drline[j]==2 && drline[j+1]==1)
      line (x1,y1,x2,y2);
  if(drline[j]==2 && drline[j+1]==3 || drline[j]==3 && drline[j+1]==2)
      line (x2,y2,x3,y3);
  if(drline[j]==3 && drline[j+1]==4 ||drline[j]==4 && drline[j+1]==3)
      line (x3,y3,x4,y4);
  if(drline[j]==4 && drline[j+1]==5 ||drline[j]==5 && drline[j+1]==4)
      line (x4,y4,x5,y5);
```

```
    if(drline[j]==5 && drline[j+1]==6 ||drline[j]==6 && drline[j+1]==5)
        line (x5,y5,x6,y6);
    if(drline[j]==6 && drline[j+1]==7 ||drline[j]==7 && drline[j+1]==6)
        line (x6,y6,x7,y7);
    if(drline[j]==7 && drline[j+1]==8 ||drline[j]==8 && drline[j+1]==7)
        line (x7,y7,x8,y8);
    if(drline[j]==7 && drline[j+1]==4 ||drline[j]==4 && drline[j+1]==7)
        line (x7,y7,x4,y4);
    if(drline[j]==8&& drline[j+1]==9 ||drline[j]==9 && drline[j+1]==8)
        line (x8,y8,x9,y9);
     if(drline[j]==9 && drline[j+1]==10 ||drline[j]==10 && drline[j+1]==9)
        line (x9,y9,x10,y10);
     if(drline[j]==10 && drline[j+1]==11 ||drline[j]==11 && drline[j+1]==10)
        line (x10,y10,x11,y11);
      if(drline[j]==9 && drline[j+1]==12 ||drline[j]==12 && drline[j+1]==9)
        line (x9,y9,x12,y12);
      if (drline[j]==9 && drline[j+1]==2 ||drline[j]==2 && drline[j+1]==9)
        line (x9,y9,x2,y2);
       if(drline[j]==2 && drline[j+1]==14 ||drline[j]==14 && drline[j+1]==2)
        line (x2,y2,x14,y14);
      if(drline[j]==7&& drline[j+1]==13 ||drline[j]==13 && drline[j+1]==7)
        line (x7,y7,x13,y13);

cout<<"press c to continue"<<endl;
cin>>cont;
if(cont=='c')
 {
    goto g;
 }
getch();
closegraph();
    case 2:
        exit(1);
    }
    return 0;
}
```

CHAPTER 7

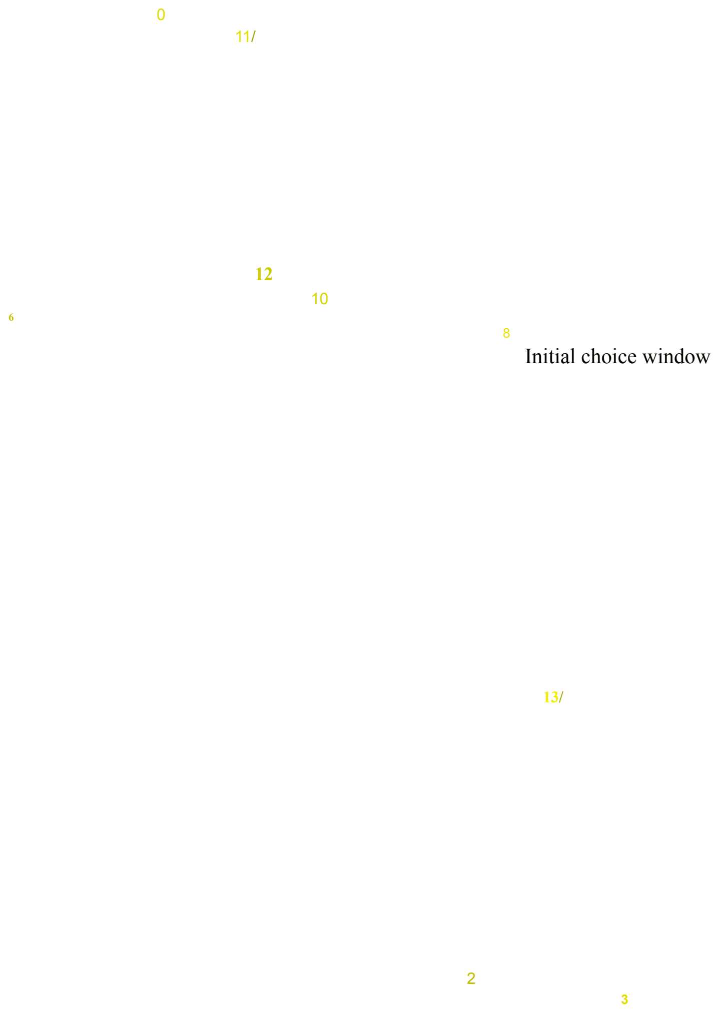
OUTPUT SCREENSHOTS



```
"C:\Users\sasi0\Desktop\SRM\DSA SEM 3\yaffin dsa project\bin\Debug\yaffin dsa project.exe"

                    //***Welcome to our transportation system* ***

Which choice would you prefer?
1. Cab service
2. Exit
Please enter your choice:
```


0
11/
12
10
6
8
Initial choice window
13/
2
3

14

5

XX

X

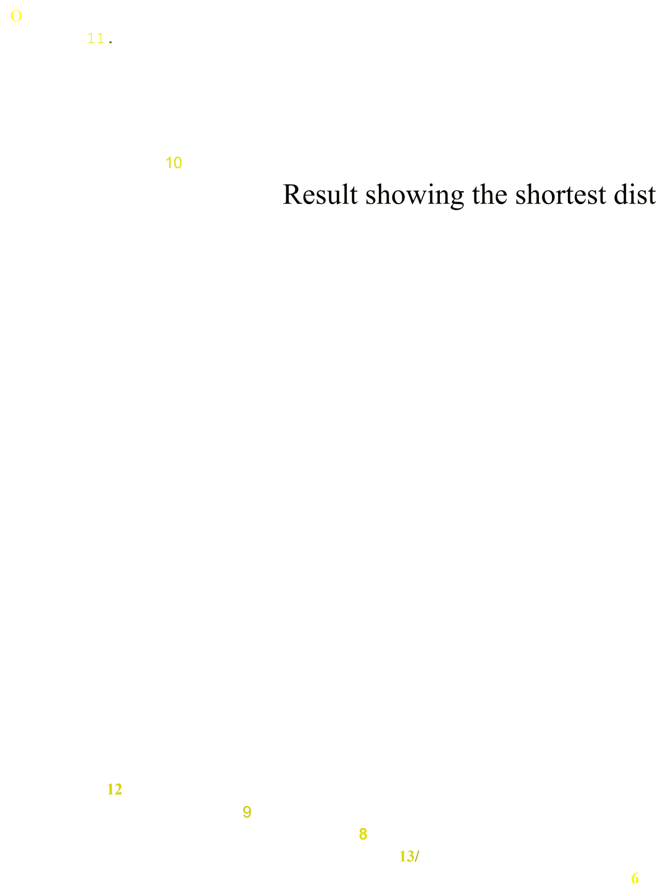
Nodes representing the locations

```
"C:\Users\sasi0\Desktop\SRM\DSA SEM 3\yaffin dsa project\bin\Debug\yaffin dsa project.exe"
                         //***Welcome to our transportation system***//

Which choice would you prefer?
1. Cab service
2. Exit
Please enter your choice:
1
Enter source you want to start from:
10
Enter destination you want to reach to:
3
Vertex
              Distance from Source
                                             Path
3
                          31
                                             10 -> 9 -> 2 -> 3

shortest path : 31
press c to continue
```

Windows BGI



Result showing the shortest distance

2

3

14

Showing result in the map

X

CHAPTER 8
REFERENCES

1. https://academictechnologies.it.miami.edu/explore-technologies/technology-summaries/note-taking applications/index.html

2. I. Culjak, D. Abram, T. Pribanic, H. Dzapo and M. Cifrek, "A brief introduction to OpenCV," 2012 Proceedings of the 35th International Convention MIPRO, 2012, pp. 1725-1730.

3. Xie, Guobo and Wen Lu. "Image Edge Detection Based On Opencv." International Journal of Electronics and Electrical Engineering 1 (2013): 104-106.

4. Ansari, Mohd & Kurchaniya, Diksha & Dixit, Manish. (2017). A Comprehensive Analysis of Image Edge Detection Techniques. International Journal of Multimedia and

Ubiquitous Engineering. 12. 1-12. 10.14257/ijmue.2017.12.11.01.

5. P. Ganesan and G. Sajiv, "A comprehensive study of edge detection for image processing applications," 2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS), 2017, pp. 1-6, doi: 10.1109/ICIIECS.2017.8275968.

6. P. Tirodkar, H. Malgundkar, S. Ghadi and A. Save, "Buddy Scanner - A Scanning Application," 2021 2nd International Conference for Emerging Technology (INCET), 2021, pp. 1-9, doi: 10.1109/INCET51464.2021.9456316.