JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY

NOIDA, SECTOR-62

DATA STRUCTURES LAB

(15B17CI371)

DS PROJECT

REPORT FILE

DELHI METRO ROUTE FINDER

Team Members:

1) Mukund Sarda (21103105)

2) Akash Sharma (21103093)

3) Karanjot Singh (21103096)

4) Priyanshi Gaur (21103097)

BRIEF INTRODUCTION ABOUT PROJECT:

The Delhi Metro is a mass rapid transit system serving Delhi and its satellite cities. It is by far the largest and busiest metro rail system in India. The network consists of 11 colour coded lines serving 255 stations in Delhi with a total length of 348 kilometers. Since it is widely spread, a person constantly needs to change metro lines to reach the destination station. Our project helps a person find the route required to reach the destination. DMRF is a program developed in C++ that helps a person to find a route to the destination entered. It also finds the total fare, average time and numberof stations in the journey. It uses concepts of graphs, vectors, file handling and graphics (UI).

FEATURES OF THE PROJECT:

- Shortest route between two stations,
- Stations in between the two stations,
- Changes in line shown using colours,
- Average fare,
- Average time,
- Number of stations,
- Route with an intermediate station

CONCEPTS USED:

-DMRF is a program developed in C++ in which a user enters the starting station and end station.
-The program uses various concepts of DSA with an interactive UI in C++.
-The program is made by applying the concepts of graphs, vectors, file handling, etc.

SOURCE CODE

```
/*
DATA STRUCTURES LAB (15B17CI371)
DS PROJECT
Title: DELHI METRO ROUTE FINDER (DMRF)
Version 2.0
Team Members:
1) Akash Sharma (21103093)
2) Mukund Sarda (21103105)
3) Priyanshi Gaur (21103097)
4) Karanjot Singh (21103096)
*/
#include<iostream>
#include<string>
#include<math.h>
#include<cstdlib>
#include<vector>
#include<fstream>
using namespace std;
const int V=248;
void delay(unsigned int ms);
void hourmin(int x);
string makecapital(string str);
void timetaken(float dist);
int money(float dist);
void Path(float dist,int e,int st,int inter);
int minDistance(float dist[], bool sptSet[]);
void dijkstra(float graph[V][V], int src,int targ);
int printSolution(float dist[], int n,int src,int temp);
void take_input();
void secondWindow();
int sameMatch(string s);
void firstWindow();
void writetext();
void writecsv();
class name
{
    void A(int i)
    {
        int j;
        for (j = 1; j <= 5; j++)
        {
            if ((i <= 3 && i + j == 4) || (i >= 3 && j == 1) || (i >= 3 && j == 5) || (i <= 3 && j == i + 2)
|| i == 5)
                cout << "* ";
            else
                cout << "  ";
        }
    }
    void B(int i)
    {
        int j;
        for (j = 1; j <= 5; j++)
        {
            if (j == 1 || (j <= 3 && (i == 1 || i == 5 || i == 9)) || (j >= 3 && ((i <= 3 && j == i + 2) ||
(i >= 3 && i <= 5 && j + i == 8) || (i <= 7 && i >= 5 && j == i - 2) || (i >= 7 && i + j == 12))))
                cout << "* ";
            else
                cout << "  ";
        }
    }
    void C(int i)
    {
        int j;
        for (j = 1; j <= 5; j++)
        {
            if (i == 1 || i == 9 || j == 1)
                cout << "* ";
            else
                cout << "  ";
        }
    }
    void D(int i)
    {
        int j;
        cout<<"\t";
        for (j = 1; j <= 9; j++)
        {
            if ((j <= 3 && (i == 1 || i == 9)) || j == 1 || (j >= 5 && ((i <= 5 && j == i + 4) || (i >= 5 &&
i + j == 14))))
                cout << "* ";
            else
```

```
                cout << "  ";
        }
    }
    void E(int i)
    {
        int j;
        for (j = 1; j <= 5; j++)
        {
            if (j == 1 || i == 1 || (i == 5 && j <= 4) || i == 9)
                cout << "* ";
            else
                cout << "  ";
        }
    }
    void F(int i)
    {
        int j;
        for (j = 1; j <= 5; j++)
        {
            if (j == 1 || i == 1 || (i == 5 && j <= 4))
                cout << "* ";
            else
                cout << "  ";
        }
    }
    void G(int i)
    {
        int j;
        for (j = 1; j <= 9; j++)
        {
            if ((j == 1 && i >= 4 && i <= 6) || ((i == 1 || i == 9) && j >= 4 && j <= 6) || i + j == 5 || i +
j == 15 || i == j + 5 || (i == 5 && j >= 5))
                cout << "*";
            else
                cout << " ";
        }
    }
    void H(int i)
    {
        int j;
        for (j = 1; j <= 5; j++)
        {
            if (j == 1 || i == 5 || j == 5)
                cout << "* ";
            else
                cout << "  ";
        }
    }
    void I(int i)
    {
        int j;
        for (j = 1; j <= 5; j++)
        {
            if (j == 3 || i == 1 || i == 9)
                cout << "* ";
            else
                cout << "  ";
        }
    }
    void J(int i)
    {
        int j;
        for (j = 1; j <= 5; j++)
        {
            if ((j == 5 && i <= 7) || (i >= 7 && ((j <= 3 && i == j + 6) || (j >= 3 && i + j == 12))))
                cout << "* ";
            else
                cout << "  ";
        }
    }
    void K(int i)
    {
        int j;
        for (j = 1; j <= 5; j++)
        {
            if (j == 1 || (i <= 5 && i + j == 6) || (i >= 5 && i == j + 4))
                cout << "* ";
            else
                cout << "  ";
        }
    }
    void L(int i)
    {
        int j;
```

```
        for (j = 1; j <= 5; j++)
        {
            if (j == 1 || i == 9)
                cout << "* ";
            else
                cout << "  ";
        }
    }
    void M(int i)
    {
        int j;
        for (j = 1; j <= 9; j++)
        {
            if (j == 1 || j == 9 || (j <= 5 && i == j) || (j >= 5 && i + j == 10))
                cout << "*";
            else
                cout << " ";
        }
    }
    void N(int i)
    {
        int j;
        for (j = 1; j <= 9; j++)
        {
            if (j == 1 || j == 9 || i == j)
                cout << "*";
            else
                cout << " ";
        }
    }
    void O(int i)
    {
        int j;
        for (j = 1; j <= 9; j++)
        {
            if (((j == 1 || j == 9) && i >= 4 && i <= 6) || ((i == 1 || i == 9) && j >= 4 && j <= 6) || i + j
== 5 || i + j == 15 || i == j + 5 || j == i + 5)
                cout << "*";
            else
                cout << " ";
        }
    }
    void P(int i)
    {
        int j;
        for (j = 1; j <= 5; j++)
        {
            if (i == 1 || i == 5 || j == 1 || (j == 5 && i <= 5))
                cout << "* ";
            else
                cout << "  ";
        }
    }
    void Q(int i)
    {
        int j;
        for (j = 1; j <= 9; j++)
        {
            if (((j == 1 || j == 9) && i >= 4 && i <= 6) || ((i == 1 || i == 9) && j >= 4 && j <= 6) || i + j
== 5 || i + j == 15 || i == j + 5 || j == i + 5 || (i >= 6 && i == j))
                cout << "*";
            else
                cout << " ";
        }
    }
    void R(int i)
    {
        int j;
        for (j = 1; j <= 5; j++)
        {
            if (i == 1 || i == 5 || j == 1 || (j == 5 && i <= 5) || (i >= 5 && j == i - 4))
                cout << "* ";

            else
                cout << "  ";
        }
    }
    void S(int i)
    {
        int j;
        for (j = 1; j <= 5; j++)
        {
            if (i == 1 || i == 5 || i == 9 || (j == 1 && i <= 5) || (j == 5 && i >= 5))
                cout << "* ";
```

```
            else
                cout << "  ";
        }
    }
    void T(int i)
    {
        int j;
        for (j = 1; j <= 5; j++)
        {
            if (i == 1 || j == 3)
                cout << "* ";
            else
                cout << "  ";
        }
    }
    void U(int i)
    {
        int j;
        for (j = 1; j <= 5; j++)
        {
            if (j == 1 || j == 5 || i == 9)
                cout << "* ";
            else
                cout << "  ";
        }
    }
    void V(int i)
    {
        int j;
        for (j = 1; j <= 9; j++)
        {
            if ((i <= 5 && (j == 1 || j == 9)) || (i >= 5 && (i == j + 4 || i + j == 14)))
                cout << "*";
            else
                cout << " ";
        }
    }
    void W(int i)
    {
        int j;
        for (j = 1; j <= 9; j++)
        {
            if ((i <= 7 && (j == 1 || j == 9)) || (i >= 7 && (i == j + 6 || i + j == 12 || i == j + 2 || i +
j == 16)))
                cout << "*";
            else
                cout << " ";
        }
    }
    void X(int i)
    {
        int j;
        for (j = 1; j <= 9; j++)
        {
            if (i == j || i + j == 10)
                cout << "*";
            else
                cout << " ";
        }
    }
    void Y(int i)
    {
        int j;
        for (j = 1; j <= 9; j++)
        {
            if ((j == 5 && i >= 5) || (i <= 5 && (i == j || i + j == 10)))
                cout << "*";
            else
                cout << " ";
        }
    }
    void Z(int i)
    {
        int j;
        for (j = 1; j <= 9; j++)
        {
            if (i == 1 || i == 9 || j + i == 10)
                cout << "*";
            else
                cout << " ";
        }
    }
public:
    name() {}
```

```
name(char s[])
{
    int x;
    for (int a = 1; a <= 9; a++)
    {
        for(int i=0; i<10000; i++)
        {
            for(int j=0; j<10000; j++)
            {
            }
        }
        x = 0;
        while (s[x] != '\0')
        {
            switch (s[x])
            {
            case 'A':
                A(a);
                break;
            case 'B':
                B(a);
                break;
            case 'C':
                C(a);
                break;
            case 'D':
                D(a);
                break;
            case 'E':
                E(a);
                break;
            case 'F':
                F(a);
                break;
            case 'G':
                G(a);
                break;
            case 'H':
                H(a);
                break;
            case 'I':
                I(a);
                break;
            case 'J':
                J(a);
                break;
            case 'K':
                K(a);
                break;
            case 'L':
                L(a);
                break;
            case 'M':
                M(a);
                break;
            case 'N':
                N(a);
                break;
            case 'O':
                O(a);
                break;
            case 'P':
                P(a);
                break;
            case 'Q':
                Q(a);
                break;
            case 'R':
                R(a);
                break;
            case 'S':
                S(a);
                break;
            case 'T':
                T(a);
                break;
            case 'U':
                U(a);
                break;
            case 'V':
                V(a);
                break;
            case 'W':
                W(a);
```

```
                    break;
                case 'X':
                    X(a);
                    break;
                case 'Y':
                    Y(a);
                    break;
                case 'Z':
                    Z(a);
                    break;
                case ' ':
                    cout << "      ";
                    break;
                default:
                    cout << "invalid input";
                }
                x++;
                cout << "  ";
            }
            cout << "\n";
        }
    }
};
struct station_code
{
    string name;
    int code;
    string color;
};
struct station_code station[V];
float graph[V][V];
struct node1
{
    vector <float>p;
} P[V];
string makecapital(string str)
{
    for(int i=0; i<str.length(); i++)
    {
        if(str[i]>96&&str[i]<123)
            str[i]=str[i]-32;
    }
    return str;
}
void delay(unsigned int ms)
{
    clock_t goal=ms+clock();
    while(goal>clock());
}
void hourmin(int x)
{
    int hours=0, minutes=0;
    hours=x/60;
    minutes=x%60;
    if (x<60)
    {
        cout<<x<<" minutes";
    }
    else if(x>=60)
    {
        cout<<hours<<" hours "<<minutes<<" minutes";
    }
}
void timetaken(float dist)
{
    float speed=0.55;
    int y=ceil(dist/speed);
    hourmin(y);
}
int money(float dist)
{
    if(dist<=2)
        return 10;
    else if(dist>2&&dist<=5)
        return 20;
    else if(dist>5&&dist<=12)
        return 30;
    else if(dist>12&&dist<=21)
        return 40;
    else if(dist>21&&dist<=32)
        return 50;
    else
        return 60;
}
```

```
void Path(float d,int e,int st,int inter)
{
    int t=e,s;
    static float dist=0;
    dist+=d;
    cout<<endl;
    cout<<"\tTHE SHORTEST PATH IS: ";
    vector <int> path;
    path.push_back(t);
    while(t!=st)
    {
        s=P[t].p.size();
        t=P[t].p[s-1];
        path.push_back(t);
    }
    vector <int>::iterator i=path.end();
    string str;
    string color;
    vector<int>::iterator i2=path.end();
    i2--;
    int n_of_stations=0;
    while(i!=path.begin())
    {
        i--;
        color=station[*i].color;
        if(color=="BLUE")
            system("color 09");
        else if(color=="YELLOW")
            system("color 06");
        else if(color=="PINK")
            system("color 0D");
        else if(color=="RED")
            system("color 04");
        else if(color=="MAGENTA")
            system("color 05");
        else if(color=="VOILET")
            system("color 01");
        else if(color=="GREEN")
            system("color 02");
        else if(color=="AQUA")
            system("color 03");
        else if(color=="ORANGE")
            system("color 0C");
        if(i!=i2)
        {
            if(station[*(i2)].color!=station[*(i)].color)
            {
                cout<<endl;
                cout<<"\t{change from "<<station[*i2].color<<" to "<<station[*i].color<<"}"<<endl;
                cout<<endl<<"\t";
            }
            i2--;
        }
        str=station[*i].name;
        cout<<" ->> "<<str;
        if (n_of_stations%5==0)
        {
            cout<<endl<<"\t";
        }
        n_of_stations++;
        delay(700);
    }
    if(!inter)
    {
        cout<<" {INTERMEDIATE STATION}";
    }
    if(inter)
    {
        cout<<endl;
        delay(1000);
        cout<<endl;
        cout<<"\tPATH LENGTH IS: ";
        cout<<d<<" KM";
        cout<<endl;
        cout<<endl;
        delay(1000);
        cout<<"\tAVERAGE TIME: ";
        timetaken(d);
        cout<<endl;
        cout<<endl;
        delay(1000);
        cout<<"\tAVERAGE FARE: ";
        cout<<" Rs. "<<money(d);
        cout<<endl;
```

```
        cout<<endl;
        delay(1000);
        cout<<"\tNO OF STATIONS: ";
        cout<<n_of_stations-1;
        cout<<endl;
    }
    else
    {
        cout<<endl;
        delay(1000);
        cout<<endl;
        cout<<"\tPATH LENGTH IS: ";
        cout<<d<<" KM";
        cout<<endl;
        cout<<endl;
        delay(1000);
        cout<<"\tAVERAGE TIME: ";
        timetaken(d);
        cout<<endl;
        cout<<endl;
        delay(1000);
        cout<<"\tAVERAGE FARE: Rs. ";
        cout<<money(d);
        cout<<endl;
        cout<<endl;
        delay(1000);
        cout<<"\tNO OF STATIONS: ";
        cout<<n_of_stations-1;
        cout<<endl;
    }
    if(inter)
    {
        cout<<endl;
        delay(1000);
        cout<<"\tWANT TO SEARCH AGAIN?: ";
        string choice;
        cin>>choice;
        fflush(stdin);
        choice=makecapital(choice);
        if(choice=="Y"||choice=="YES")
            secondWindow();
        cout<<endl;
    }
    return;
}
int minDistance(float dist[], bool sptSet[])
{
    float min = INT_MAX;
    int 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;
}
int printSolution(float dist[], int n,int src,int temp,int inter)
{
    Path(dist[temp],temp,src,inter);
}
void dijkstra(float graph[V][V], int src,int targ,int inter)
{
    float dist[V];
    bool sptSet[V];
    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] != INT_MAX
                    && dist[u]+graph[u][v] < dist[v])
            {
                dist[v] = dist[u] + graph[u][v];
                P[v].p.push_back(u);
            }
    }
    printSolution(dist, V,src,targ, inter);
}
int sameMatch(string s)
{
    for(int i=0; i<248; i++)
```

```
    {
        int x=s.compare(station[i].name);
        if(x==0)
        {
            return i;
        }
    }
    return -1;
}
void take_input()
{
    char ch;
    string start_s,end_s,inter_s,line,col;
    cout<<endl;
    cout<<endl;
    cout<<endl;
    cout<<endl;
    cout<<"\tENTER THE STARTING STATION: ";
    getline(cin,start_s);
    cout<<endl;
    cout<<"\tENTER THE DESTINATION STATION: ";
    getline(cin,end_s);
    cout<<endl;
    cout<<"\tENTER THE INTERMEDIATE STATION: ";
    getline(cin,inter_s);
    start_s=makecapital(start_s);
    end_s=makecapital(end_s);
    inter_s=makecapital(inter_s);
    int startcode,endcode,intercode=-1;
    int startflag=0,endflag=0,interflag1=0,interflag2=0;
    if(!(inter_s=="NO"||inter_s=="N"))
        interflag1=1;
    startcode=sameMatch(start_s);
    endcode=sameMatch(end_s);
    intercode=sameMatch(inter_s);
    if(startcode==-1)
        startflag=1;
    if(endcode==-1)
        endflag=1;
    if(endcode==-1)
        interflag2=1;
    int fault=0;
    if(startflag==1)
    {
        cout<<endl;
        cout<<"WRONG STARTING STATION NAME ENTERED !";
        fault=1;
    }
    if(endflag==1)
    {
        cout<<endl;
        cout<<"WRONG DESTINATION STATION NAME ENTERED !";
        fault=1;
    }
    if(interflag1&&interflag2==1)
    {
        cout<<endl;
        cout<<"WRONG INTERMEDIATE STATION NAME ENTERED !";
        fault=1;
    }
    if(fault)
    {
        secondWindow();
        return;
    }
    if(interflag1)
    {
        dijkstra(graph,startcode,intercode,0);
        dijkstra(graph,intercode,endcode,1);
    }
    else
        dijkstra(graph, startcode,endcode,1);
}
void secondWindow()
{
    fflush(stdin);
    system("cls");
    delay(90);
    take_input();
}
void firstWindow()
{
    cout<<endl;
    cout<<endl;
```

```
    cout<<endl;
    name n("DMRF");
    delay(300);
    system("color 0A");
    for(int i=0; i<10000; i++)
    {
        for(int j=0; j<10000; j++)
        {
        }
    }
    delay(300);
    system("color 0B");
    for(int i=0; i<10000; i++)
    {
        for(int j=0; j<10000; j++)
        {
        }
    }
    delay(300);
    system("color 0C");
    for(int i=0; i<10000; i++)
    {
        for(int j=0; j<10000; j++)
        {
        }
    }
    delay(300);
    system("color 0F");
    for(int i=0; i<10000; i++)
    {
        for(int j=0; j<10000; j++)
        {
        }
    }
    delay(300);
    system("color 0E");
    for(int i=0; i<10000; i++)
    {
        for(int j=0; j<10000; j++)
        {
        }
    }
    delay(300);
    system("color 0D");
    for(int i=0; i<10000; i++)
    {
        for(int j=0; j<10000; j++)
        {
        }
    }
    system("color 07");
    cout<<endl;
    cout<<endl;
    cout<<"\t============================"<<endl;
    cout<<"\t| DELHI METRO ROUTE FINDER |"<<endl;
    cout<<"\t============================"<<endl;
    cout<<endl;
    cout<<"\t-------------"<<endl;
    cout<<"\t!! WELCOME !!"<<endl;
    cout<<"\t-------------"<<endl;
    cout<<endl;
    cout<<"\t-- DEVELOPED BY:"<<endl;
    cout<<"\t1) Mukund Sarda (21103105)"<<endl;
    cout<<"\t2) Akash Sharma (21103093)"<<endl;
    cout<<"\t3) Priyanshi Gaur (21103097)"<<endl;
    cout<<"\t4) Karanjot Singh (21103096)"<<endl;
}
// Writing the matrix in text file
void writetext()
{
    ofstream matrix;
    matrix.open("matrix.txt");
    matrix<<"  ";
    for(int i=0; i<V; i++)
        matrix<<i<<" ";
    matrix<<endl;
    for(int i=0; i<V; i++)
    {
        for(int j=0; j<V; j++)
        {
            if(j==0)
                matrix<<i<<" ";
            matrix<<graph[i][j]<<" ";
        }
```

```
        matrix<<endl;
    }
}
// Writing the matrix in csv file
void writecsv()
{
    ofstream matrix;
    matrix.open("test.csv",ios::out|ios::app);
    matrix<<",";
    for(int i=0; i<V; i++)
        matrix<<i<<",";
    matrix<<endl;
    for(int i=0; i<V; i++)
    {
        for(int j=0; j<V; j++)
        {
            if(j==0)
                matrix<<i<<",";
            matrix<<graph[i][j]<<",";
        }
        matrix<<endl;
    }
}
int main()
{
    int temp,n1,n2;
    float dis;
    ifstream fin;
    fin.open("node_values_new.txt");
    for(int i=0; i<248; i++)
    {
        for(int j=0; j<248; j++)
            graph[i][j]=0;
    }
    for(int i=1; i<=V; i++)
    {
        fin>>temp;
        fin>>n1;
        for( int j=0; j<temp; j++)
        {
            fin>>n2;
            fin>>dis;
            graph[n1-1][n2-1]=dis;
        }
    }
    string line,col;
    ifstream code;
    ifstream color;
    code.open("stationcodes.txt");
    color.open("stationcolorcodes.txt");
    for(int i=0; i<V; i++)
    {
        getline(code, line);
        station[i].name=line;
        station[i].code=i;
        getline(color,col);
        station[i].color=col;
    }
    firstWindow();
    char ch;
    scanf("%c",&ch);
    secondWindow();
}
```

OUTPUT:

```
* * *   *          *       *  * * * * *   * * * * *
*         *        **     **  *       *   *
*           *      * *   * *  *       *   *
*             *    *  * *  *  *       *   *
*               *  *   *   *  * * * * *   * * * *
*             *    *       *  * *         *
*           *      *       *  *   *       *
*         *        *       *  *     *     *
* * *   *          *       *  *       *   *


============================
| DELHI METRO ROUTE FINDER |
============================

-------------
!! WELCOME !!
-------------

-- DEVELOPED BY:
1) Mukund Sarda (21103105)
2) Akash Sharma (21103093)
3) Priyanshi Gaur (21103097)
4) Karanjot Singh (21103096)
```

```
ENTER THE STARTING STATION: janakpuri west

ENTER THE DESTINATION STATION: saket

ENTER THE INTERMEDIATE STATION: no

THE SHORTEST PATH IS:  ->> JANAKPURI WEST

{change from BLUE to MAGENTA}

 ->> DABRI MOR ->> DASHRATH PURI ->> PALAM ->> SADAR BAZAR CANTONMENT ->> TERMINAL T1 IGI AIRPORT
 ->> SHANKAR VIHAR ->> VASANT VIHAR ->> MUNIRKA ->> R K PURAM ->> IIT
 ->> HAUZ KHAS
{change from MAGENTA to YELLOW}

 ->> MALVIYA NAGAR ->> SAKET

PATH LENGTH IS: 24.1 KM

AVERAGE TIME: 44 minutes

AVERAGE FARE:  Rs. 50

NO OF STATIONS: 13

WANT TO SEARCH AGAIN?:
```