JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY

NOIDA, SECTOR-62

SOFTWARE DEVELOPMENT LAB-2

(15B17CI271)

MINI PROJECT

REPORT FILE

DELHI METRO ROUTE FINDER

Team Members:

1) Mukund Sarda (21103105)

2) Akash Sharma (21103093)

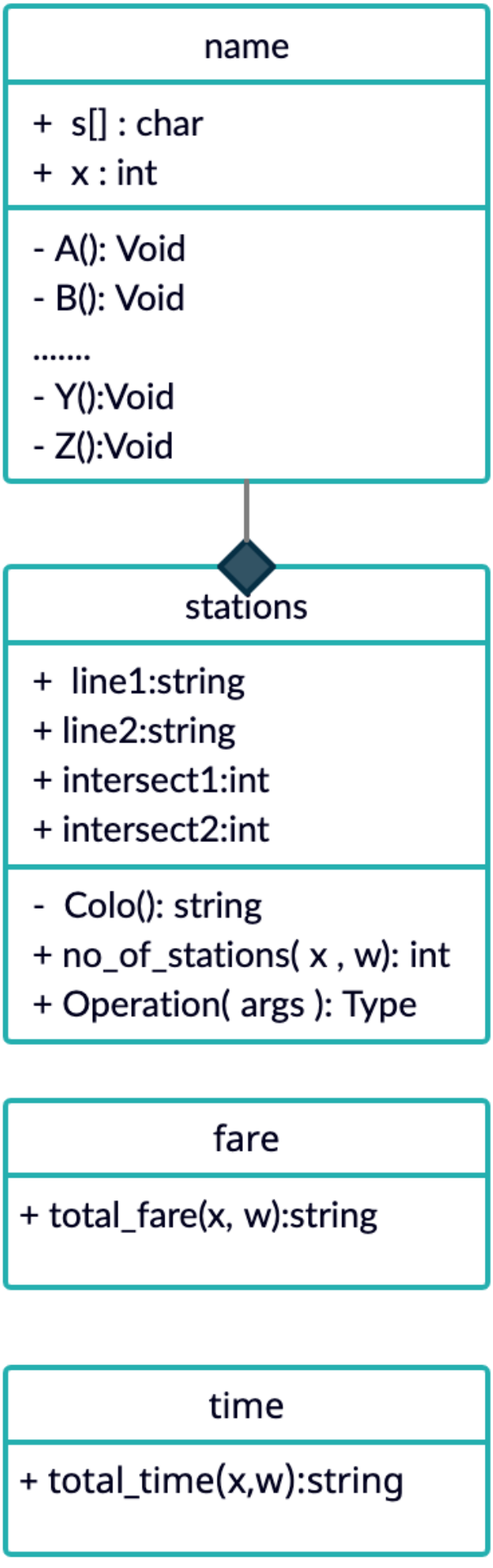
3) Karanjot Singh (21103096)

ABSTRACT:

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 kilometres. 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 number of stations in the journey. It uses concepts of OOPs, STL, and Data File Handling for its implementation.

UML DIAGRAM:


name
+ s[] : char
+ x : int
- A(): Void
- B(): Void
.....
- Y():Void
- Z():Void
stations
+ line1:string
+ line2:string
+ intersect1:int
+ intersect2:int
- Colo(): string
+ no_of_stations(x , w): int
+ Operation(args): Type
fare
+ total_fare(x, w):string
time
+ total_time(x,w):string

SOURCE CODE:

```
/*
SOFTWARE DEVELOPMENT LAB-2 (15B17CI271)
MINI PROJECT

Title: DELHI METRO ROUTE FINDER (DMRF)

Team Members:
1) Akash Sharma (21103093)
2) Mukund Sarda (21103105)
3) Karanjot Singh (21103096)
*/

#include <iostream>
#include <string>
#include <vector>
#include <map>
#include <algorithm>
#include <fstream>
#include <dos.h>
#include <sstream>
#include <ctime>
#include <math.h>

using namespace std;

static vector<string> blue1, blue2, yellow, red, magenta, green1, green2,
orange, violet, pink, grey;
static int st;

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;
        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";
        }
    }

};

string intersection_point(vector<string> vector1, vector<string> vector2)
{

    string *s;
    s = new string[10];

    sort(vector1.begin(), vector1.end());
    sort(vector2.begin(), vector2.end());

    vector<string> v(vector1.size() + vector2.size());
    vector<string>::iterator it, st;
    it = set_intersection(vector1.begin(), vector1.end(), vector2.begin(),
vector2.end(), v.begin());

    int i = 0;
    for (st = v.begin(); st != it; ++st)
    {
        s[i] = *st;
        i++;
    }

    if (i != 0)
        return s[0];

    else
        return "NULL";

}

void minsec(int Time)
{

    int minutes = 0, seconds = 0;
    minutes = Time / 60;
    seconds = Time % 60;

    if (seconds == 0)
        cout << "Expected Time: " << minutes << " minutes" << endl
             << endl;

    else
        cout << "Expected Time: " << minutes << " minutes " << seconds << "
seconds " << endl
             << endl;

}
```

```
string capitalize(string text)
{

    for (int x = 0; x < text.length(); x++)
    {
        text[x] = toupper(text[x]);
    }
    return text;

}

int neg_to_pos(int a)
{

    int b= abs(a);
    return b;

}

vector<string> convert(string s)
{

    if (s == "blue1")
        return blue1;

    else if (s == "blue2")
        return blue2;

    else if (s == "green1")
        return green2;

    else if (s == "green2")
        return green2;

    else if (s == "grey")
        return grey;

    else if (s == "orange")
        return orange;

    else if (s == "red")
        return red;

    else if (s == "pink")
        return pink;

    else if (s == "magenta")
        return magenta;

    else if (s == "yellow")
        return yellow;

    else if (s == "violet")
        return violet;

}

void delay(int msecs)
{
```

```
    clock_t endTime;
    endTime = clock() + msecs * CLOCKS_PER_SEC/1000;
    while(clock() < endTime) { }

}

int get_index(vector<string> v, string s)
{

    int i = 0;
    int x = v.size();

    for (int i = 0; i < x; i++)
    {
        if (v[i] == s)
        {
            return (i + 1);
        }
    }
    return 0;

}

class stations
{

public:

    stations()
    {

        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 0D");

    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 0F");

    for(int i=0;i<10000;i++)
        {

            for(int j=0;j<10000;j++)
            {

            }
        }

    string s;
    ifstream f;

    /*blue1*/
    f.open("blue1.txt", ios::in);

    while (getline(f, s))
    {
        blue1.push_back(s);
    }
    f.close();

    /*blue2*/
    f.open("blue2.txt", ios::in);

    while (getline(f, s))
    {
```

```
        blue2.push_back(s);
    }
    f.close();

    /*yellow*/
    f.open("yellow.txt", ios::in);

    while (getline(f, s))
    {
        yellow.push_back(s);
    }
    f.close();

    /*green1*/
    f.open("green1.txt", ios::in);

    while (getline(f, s))
    {
        green1.push_back(s);
    }
    f.close();

    /*green2*/
    f.open("green2.txt", ios::in);

    while (getline(f, s))
    {
        green2.push_back(s);
    }
    f.close();

    /*red*/
    f.open("red.txt", ios::in);

    while (getline(f, s))
    {
        red.push_back(s);
    }
    f.close();

    /*magenta*/
    f.open("magenta.txt", ios::in);

    while (getline(f, s))
    {
        magenta.push_back(s);
    }
    f.close();

    /*orange*/
    f.open("orange.txt", ios::in);

    while (getline(f, s))
    {
        orange.push_back(s);
    }
    f.close();

    /*violet*/
    f.open("violet.txt", ios::in);
```

```
    while (getline(f, s))
    {
        violet.push_back(s);
    }
    f.close();

    /*pink*/
    f.open("pink.txt", ios::in);

    while (getline(f, s))
    {
        pink.push_back(s);
    }
    f.close();

    /*grey*/
    f.open("grey.txt", ios::in);

    while (getline(f, s))
    {
        grey.push_back(s);
    }
    f.close();
}

void colo(string s)
{
    if (s == "blue1" || s == "blue2")
    {
        system("color 0B");
    }

    else if (s == "green1" || s == "green2")
    {
        system("color 0A");
    }

    else if (s == "red" || s == "orange")
    {
        system("color 0C");
    }

    else if (s == "violet")
    {
        system("color 05");
    }

    else if (s == "magenta" || s == "pink")
    {
        system("color 0D");
    }

    else if (s == "grey")
    {
        system("color 08");
    }

    else if (s == "yellow")
    {
        system("color 0E");
    }
```

```
}

int no_of_stations(string x, string w)
{
    int ind1[11];
    string c1[11];

    c1[0] = "blue1";
    c1[1] = "blue2";
    c1[2] = "green1";
    c1[3] = "green2";
    c1[4] = "yellow";
    c1[5] = "red";
    c1[6] = "magenta";
    c1[7] = "orange";
    c1[8] = "violet";
    c1[9] = "grey";
    c1[10] = "pink";

    int y;
    int j = 0;

    y = get_index(blue1, x);
    ind1[j] = y;
    j++;

    y = get_index(blue2, x);
    ind1[j] = y;
    j++;

    y = get_index(green1, x);
    ind1[j] = y;
    j++;

    y = get_index(green2, x);
    ind1[j] = y;
    j++;

    y = get_index(yellow, x);
    ind1[j] = y;
    j++;

    y = get_index(red, x);
    ind1[j] = y;
    j++;

    y = get_index(magenta, x);
    ind1[j] = y;
    j++;

    y = get_index(orange, x);
    ind1[j] = y;
    j++;

    y = get_index(violet, x);
    ind1[j] = y;
    j++;

    y = get_index(grey, x);
    ind1[j] = y;
    j++;
```

```
y = get_index(pink, x);
ind1[j] = y;
j++;

int ind2[11];
j = 0;

y = get_index(blue1, w);
ind2[j] = y;
j++;

y = get_index(blue2, w);
ind2[j] = y;
j++;

y = get_index(green1, w);
ind2[j] = y;
j++;

y = get_index(green2, w);
ind2[j] = y;
j++;

y = get_index(yellow, w);
ind2[j] = y;
j++;

y = get_index(red, w);
ind2[j] = y;
j++;

y = get_index(magenta, w);
ind2[j] = y;
j++;

y = get_index(orange, w);
ind2[j] = y;
j++;

y = get_index(violet, w);
ind2[j] = y;
j++;

y = get_index(grey, w);
ind2[j] = y;
j++;

y = get_index(pink, w);
ind2[j] = y;
j++;

int stations;
for (int i = 0; i < 11; i++)
{

    if ((ind1[i] != 0) && (ind2[i] != 0))
    {
        vector<string> v1;
        v1 = convert(c1[i]);
```

```
            colo(c1[i]);

            if (ind1[i] > ind2[i])
            {
                stations = ind1[i] - ind2[i];

                for (int k = ind1[i]; k > ind2[i]; k--)
                {

                    for (int k = 0; k < 10000; k++)
                    {

                        for (int j = 0; j < 10000; j++)
                        {
                        }
                    }
                    cout << v1[k - 1] << " -> ";
                    delay(500);
                }
                cout << v1[ind2[i] - 1] << endl;
                delay(500);
            }

            else
            {
                stations = ind2[i] - ind1[i];

                for (int k = ind1[i]; k < ind2[i]; k++)
                {

                    for (int k = 0; k < 10000; k++)
                    {

                        for (int j = 0; j < 10000; j++)
                        {
                        }
                    }
                    cout << v1[k - 1] << " -> ";
                    delay(500);
                }
                cout << v1[ind2[i] - 1] << endl;
                delay(500);
            }

            st = stations;
            stations=neg_to_pos(stations);
            system("color 0F");
            return stations;
        }
    }

    string line1[11], line2[11];
    int l = 0;
    int k = 0;
    int intersect1, intersect2;

    for (int i = 0; i < 11; i++)
    {

        if (ind1[i] != 0)
        {
```

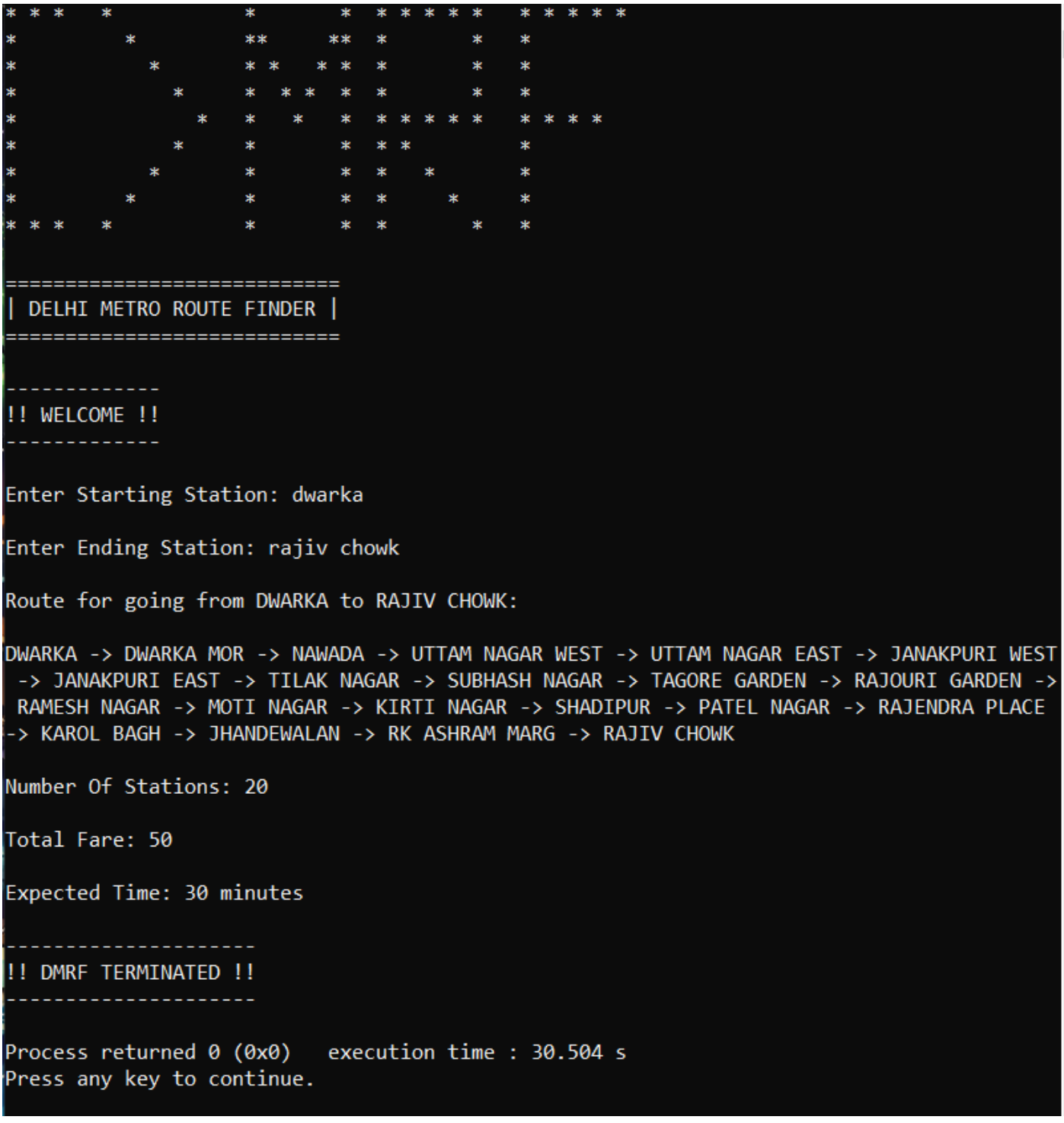
```
                line1[l] = c1[i];
                l++;
            }

            if (ind2[i] != 0)
            {
                line2[k] = c1[i];
                k++;
            }
        }

        for (int i = 0; i < l; i++)
        {

            for (int j = 0; j < k; j++)
            {
                vector<string> v1, v2;
                v1 = convert(line1[i]);
                v2 = convert(line2[j]);

                if (intersection_point(v1, v2) != "NULL")
                {
                    intersect1 = get_index(v1, intersection_point(v1, v2));
                    intersect2 = get_index(v2, intersection_point(v1, v2));
                    cout << "Intersection Point: " << intersection_point(v1,
v2) << endl;
                    cout<<endl;

                    int p1, p2;
                    int a, b;
                    p1 = get_index(v1, x);
                    p2 = get_index(v2, w);

                    if (p1 > intersect1)
                    {
                        a = p1 - intersect1;
                    }

                    else
                    {
                        a = intersect1 - p1;
                    }

                    if (p2 > intersect2)
                    {
                        b = p2 - intersect2;
                    }

                    else
                    {
                        b = intersect2 - p2;
                    }

                    colo(line1[j]);

                    if (p1 < intersect1)
                    {

                        for (int i = p1; i < intersect1; i++)
                        {
```

```
                for (int k = 0; k < 10000; k++)
                {

                    for (int j = 0; j < 10000; j++)
                    {
                    }
                }
                cout << v1[i - 1] << " -> ";
                delay(500);
            }
        }

        else
        {

            for (int i = p1; i > intersect1; i--)
            {

                for (int k = 0; k < 10000; k++)
                {

                    for (int j = 0; j < 10000; j++)
                    {
                    }
                }
                cout << v1[i - 1] << " -> ";
                delay(500);
            }
        }

        colo(line2[j]);

        if (p2 > intersect2)
        {

            for (int i = intersect2 - 1; i < p2 - 1; i++)
            {

                for (int k = 0; k < 10000; k++)
                {

                    for (int j = 0; j < 10000; j++)
                    {
                    }
                }
                cout << v2[i] << " -> ";
                delay(500);
            }
        }

        else
        {

            for (int i = intersect2 - 1; i > p2 + 1; i--)
            {

                for (int k = 0; k < 10000; k++)
                {

                    for (int j = 0; j < 10000; j++)
                    {
```

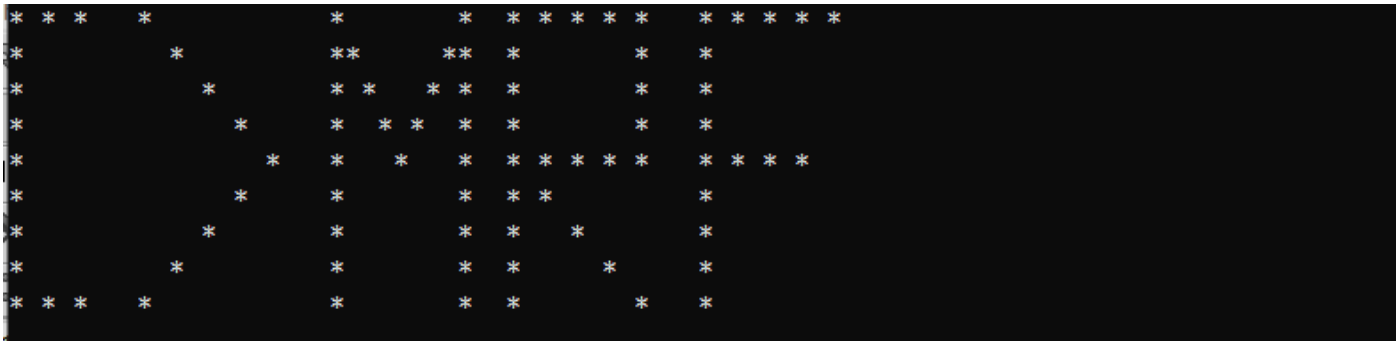
```
                                }
                            }
                            cout << v2[i] << " -> ";
                            delay(500);
                        }
                    }

                    cout << v2[p2 - 1];
                    delay(500);
                    cout << endl;

                    st = a + b;
                    system("color 0F");
                    return a + b;
                }
            }
        }
    }

};

class time
{

public:
    int total_time(string x, string w)
    {
        int p = st;
        return (90 * p);
    }

};

class fare
{

public:
    int total_fare(string x, string w)
    {
        int p = st;
        int a = p / 5;
        return ((a + 1) * 10);
    }

};

int main()
{

    class stations s;

    cout<<endl;
    cout<<"============================"<<endl;
    cout<<"| DELHI METRO ROUTE FINDER |"<<endl;
    cout<<"============================"<<endl;
    cout<<endl;

    cout<<"-------------"<<endl;
    cout<<"!! WELCOME !!"<<endl;
    cout<<"-------------"<<endl;
    cout<<endl;
```

```
    string a, b, x, w;

    cout<<"Enter Starting Station: ";
    getline(cin>>ws,a);
    cout<<endl;

    cout<<"Enter Ending Station: ";
    getline(cin>>ws,b);
    cout<<endl;

    x=capitalize(a);
    w=capitalize(b);

    class time t;
    class fare f;

    cout<<"Route for going from "<<x<<" to "<<w<<": "<<endl;
    cout<<endl;

    int st = s.no_of_stations(x, w);
    st=neg_to_pos(st);

    cout<<endl;
    cout<<"Number Of Stations: "<<st<<endl;
    cout<<endl;

    int Fare = f.total_fare(x, w);

    cout<<"Total Fare: "<<Fare<<endl;
    cout<<endl;

    int Time = t.total_time(x, w);
    minsec(Time);

    cout<<"----------------------"<<endl;
    cout<<"!! DMRF TERMINATED !!"<<endl;
    cout<<"----------------------"<<endl;

    return 0;

}
```

OUTPUT:




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

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

Enter Starting Station: dwarka

Enter Ending Station: rajiv chowk

Route for going from DWARKA to RAJIV CHOWK:

DWARKA -> DWARKA MOR -> NAWADA -> UTTAM NAGAR WEST -> UTTAM NAGAR EAST -> JANAKPURI WEST
 -> JANAKPURI EAST -> TILAK NAGAR -> SUBHASH NAGAR -> TAGORE GARDEN -> RAJOURI GARDEN ->
 RAMESH NAGAR -> MOTI NAGAR -> KIRTI NAGAR -> SHADIPUR -> PATEL NAGAR -> RAJENDRA PLACE
-> KAROL BAGH -> JHANDEWALAN -> RK ASHRAM MARG -> RAJIV CHOWK

Number Of Stations: 20

Total Fare: 50

Expected Time: 30 minutes

---------------------
!! DMRF TERMINATED !!
---------------------

Process returned 0 (0x0)   execution time : 30.504 s
Press any key to continue.
```



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

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

Enter Starting Station: huda city centre

Enter Ending Station: rithala

Route for going from HUDA CITY CENTRE to RITHALA:

Intersection Point: KASHMERE GATE

HUDA CITY CENTRE -> IFFCO CHOWK -> MG ROAD -> SIKANDERPUR -> GURU DRONACHARYA -> ARJAN
 GARH -> GHITORNI -> SULTANPUR -> CHHATARPUR -> QUTAB MINAR -> SAKET -> MALVIYA NAGAR
-> HAUZ KHAS -> GREEN PARK -> AIIMS -> DILLI HAAT -> JOR BAGH -> LOK KALYAN MARG -> UD
YOG BHAWAN -> CENTRAL SECRETARIAT -> PATEL CHOWK -> RAJIV CHOWK -> NEW DELHI -> CHAWRI
 BAZAR -> CHANDNI CHOWK -> KASHMERE GATE -> TIS HAZARI -> PUL BANGASH -> PRATAP NAGAR
-> SHASTRI NAGAR -> INDERLOK -> KANHAIYA NAGAR -> KESHAV PURAM -> NETAJI SUBHASH PLACE
 -> KOHAT ENCLAVE -> PITAMPURA -> ROHINI EAST -> ROHINI WEST -> RITHALA

Number Of Stations: 38

Total Fare: 80

Expected Time: 57 minutes

---------------------
!! DMRF TERMINATED !!
---------------------

Process returned 0 (0x0)   execution time : 56.942 s
Press any key to continue.
```