```
//Implement Traveling Salesman Problem using Local Search Algorithm.
//AI Travelling Salesman Problem using GA
//Submitted by: Sanskar Sharma
//Date: 20/09/2020
//PRN: 0120180381
//Roll numbetr: 090

#include<iostream>
#include<string.h>
#include <cstdlib>
#include <ctime>

using namespace std;

class graphnode
{
    public:
        int city_index;
        char city_name[50];
};


class TSP
{

    public:
        void input();
        void display_graph(int n, graphnode graph[],int **distance_graph);
        void all_routes(int choice,int n, graphnode graph[],int **distance_graph);
        void operations(int choice,int n, graphnode graph[],int **distance_graph);

};

int factorial(int val)
{
    int mul=val;
    while(val!=1)
    {
        mul=mul*(--val);
    }
    return mul;
}

int check_visited_full(int n,int visited[])
{
    int flag=1;
    for(int i=0;i<n;++i)
    {
        if(visited[i]==0)
        {
            flag=0;
            break;
        }
    }
    return flag;
}

void print_currentPath(int n,graphnode graph[],int current_path[])
{

    for(int i=0;i<n;++i)
    {
        if(i==n-1)
        {
            cout<<graph[current_path[i]].city_name;
            cout<<" --> "<<graph[current_path[0]].city_name;
        }
        else
        {
            cout<<graph[current_path[i]].city_name<<" --> ";
        }

    }
}

int print_currentCost(int n,int current_path[],int **distance_graph)
{
    int cost=0;
    for(int i=0;i<n;++i)
    {
        if(i==n-1)
        {
            cost+=distance_graph[current_path[i]][current_path[0]];
        }
        else
        {
            cost+=distance_graph[current_path[i]][current_path[i+1]];
        }

    }
    cout<<"\n\t\tCost of current route is: "<<cost;
    return cost;

}

void TSP::all_routes(int choice,int n, graphnode graph[],int **distance_graph)
{
    int n_minus_1fact=factorial(n-1);
    int total_possible_routes[n_minus_1fact][n+1];
    display_graph(n,graph,distance_graph);
    cout<<"\n\tEnter the initial node index: ";
    int index;
    cin>>index;
    while(index<1||index>n)
    {
        cout<<"\n\tRe-enter initial node index (between 1 and "<<n<<"): ";
        cin>>index;
    }
    --index;
    cout<<"\n\tInitial node is: "<<graph[index].city_name;
    if(choice==1)
    {
        int Iteration=0;
        cout<<"\n\tThe graph is Asymmetrical/ Uni-directional.";
        cout<<" Therefore, finding the optimum solution for our TSP.";
        cout<<"\n\t\t(NOTE: May not be least!)";
        int current_path[n];
        int optimal_path[n];
        current_path[0]=index;//starting with initial city
        optimal_path[0]=index;
        //Current route
        int incrementor=1;
        for(int i=0;i<n;++i)
        {
            if(i!=index)
            {
                current_path[incrementor]=i;
                optimal_path[incrementor]=i;
                incrementor++;
            }
        }

        //Printing current path
        cout<<"\n\n\t\t0 Iteration: ";
        cout<<"\n\t\tCurrent route is: ";
        print_currentPath(n,graph,current_path);
        int curr_cost=print_currentCost(n,current_path,distance_graph);
        int min_cost=curr_cost;

        int rand1,rand2,temp;
        cout<<"\n\n\t\tLet's start swapping randomnly and get optimal route.";
        for(int i=0;i<10*n;++i)
        {
            cout<<"\n\n\t\t"<<i+1<<" Iteration: ";
            rand1= rand()%((n-1)-1 + 1) + 1;
            rand2=rand()%((n-1)-1 + 1) + 1;
            while(rand2==rand1)
            {
                rand2=rand()%((n-1)-1 + 1) + 1;
            }
            temp=current_path[rand1];
            current_path[rand1]=current_path[rand2];
            current_path[rand2]=temp;
            cout<<"\n\t\tCurrent route is: ";
            print_currentPath(n,graph,current_path);
            curr_cost=print_currentCost(n,current_path,distance_graph);
            if(curr_cost<min_cost)
            {
                //make current path as optimal path
                for(int j=0;j<n;++j)
                {
                    optimal_path[i]=current_path[i];
                }
                //and current cost as minimum cost
                min_cost=curr_cost;
                Iteration=i+1;
            }


        }
        cout<<"\n\t\tEnd of "<<10*n<<" Iterations";
        cout<<"\n\tOptimal solution found at "<<Iteration<<" iteration.";
        cout<<"\n\tShortest/Optimal route achieved is: ";
        print_currentPath(n,graph,optimal_path);
        cout<<"\n\tMinimum cost for the route is: "<<min_cost;

    }
    else
    {
        int visited[n];
        for(int i=0;i<n;++i)
        {
            visited[i]=0;
        }
        visited[index]=1;
        int min_cost_path[n+1];
        min_cost_path[0]=index;
        min_cost_path[n]=index;
        int min=10000;
        int total_cost=0;
        int temp;
        for(int j=1;j<n;++j)
        {
            for(int i=0;i<n;++i)
            {
                if(visited[i]!=1)
                {
                    if(distance_graph[index][i]<min)
                    {
                        min=distance_graph[index][i];
                        temp=i;
                    }
                }
            }
            total_cost+=min;
            min=10000;
            min_cost_path[j]=temp;
            visited[temp]=1;
            index=temp;
        }
        total_cost+=distance_graph[temp][min_cost_path[0]];

        cout<<"\n\t\tShortest Path possible for Travelling Salesman: ";
        for(int i=0;i<n+1;++i)
        {
            if(i==n)
            {
                cout<<graph[min_cost_path[i]].city_name;
            }
            else
            {
                cout<<graph[min_cost_path[i]].city_name<<" --> ";
            }

        }
        cout<<"\n\t\tMin Cost: "<<total_cost;
    }




}

void TSP::display_graph(int n, graphnode graph[],int **distance_graph)
{
    cout<<"\n\tList of cities with their indexes: ";
    for(int i=0;i<n;++i)
    {
        cout<<"\n\t\t"<<i+1<<". "<<graph[i].city_name;
    }
    for(int i=0;i<n;++i)
    {
        cout<<"\n\t\t| ";
        for(int j=0;j<n;++j)
        {
            cout<<distance_graph[i][j];
            if(distance_graph[i][j]<10)
            {
                cout<<"    ";
            }
            else if(distance_graph[i][j]<100)
            {
                cout<<"   ";
            }
            else if(distance_graph[i][j]<1000)
            {
                cout<<"  ";
            }
            else
            {
                cout<<" ";
            }
        }
        cout<<"|";
    }
}

void TSP::operations(int choice,int n, graphnode graph[],int **distance_graph)
{
    int ch;
    cout<<"\n\tEnter your choice.";
    cout<<"\n\t\t1. Display city distance graph.";
    cout<<"\n\t\t2. Display all possible routes and cost with Optimal solution.";
    cout<<"\n\t\t3. Exit.\n\tEnter: ";
    cin>>ch;
    switch(ch)
    {
        case 1:
            display_graph(n,graph,distance_graph);
            operations(choice,n,graph,distance_graph);
            break;
        case 2:
            all_routes(choice,n,graph,distance_graph);
            operations(choice,n,graph,distance_graph);
            break;
        default:
            exit(0);
    }
}

int main()
{
    cout<<"\n\t\t\tAI Travelling Salesman Problem using GA";
    cout<<"\n\t\t\t (Mutation method used: Swap mutation)";
    cout<<"\n\t\t\t         By Sanskar Sharma";
    cout<<"\n\t\t\t             0120180381\n";
    cout<<"\n\t\tEnter the initial state of system. ";
    int n;
    cout<<"\n\tEnter the number of cities: ";
    cin>>n;
    graphnode graph[n];
    cout<<"\n\tEnter the names of all the cities.";
    for(int i=0;i<n;++i)
    {
        cout<<"\n\t\tName of "<<i+1<<" city: ";
        cin>>graph[i].city_name;
        graph[i].city_index=i;

    }
    //int distance_graph[n][n];
    int** distance_graph = new int*[n];
    for(int i=0; i<n; i++)
    {
        distance_graph[i] = new int[n];
    }
    cout<<"\n\tEnter the distance graph.";
    for(int i=0;i<n;++i)
    {
        for(int j=0;j<n;++j)
        {
            distance_graph[i][j]=0;
        }
    }
    cout<<"\n\tEnter your choice.\n\t\t1. Asymmetric/ Uni-directional.";
    cout<<"\n\t\t2. Symmetric/ Bi-directional.\n\tEnter: ";
    int choice;
    cin>>choice;
    if(choice==1)
    {
        cout<<"\n\t\tAssumption made, All cities have a direct link to all cities.";
        int d;
        for(int i=0;i<n;++i)
        {
            for(int j=0;j<n;++j)
            {
                if(i==j)
                {
                    distance_graph[i][j]=0;
                }
                else
                {
                    cout<<"\n\t\tEnter the distance from "<<graph[i].city_name<<" to ";
                    cout<<graph[j].city_name<<": ";
                    cin>>d;
                    while(d<=0||d>9999)
                    {
                        cout<<"\n\t\tDistance cannot be negative, 0 or more than 9999km";
                        cout<<"\n\t\tRe-enter distance from "<<graph[i].city_name<<" to ";
                        cout<<graph[j].city_name<<": ";
                        cin>>d;
                    }
                    distance_graph[i][j]=d;
                }
            }
        }
    }
    else
    {
        int d;
        for(int i=0;i<n;++i)
        {
            for(int j=0;j<n;++j)
            {
                if(distance_graph[i][j]==0)
                {
                    if(i==j)
                    {
                        distance_graph[i][j]=0;
                    }
                    else
                    {
                        cout<<"\n\t\tEnter the distance from "<<graph[i].city_name<<" to ";
                        cout<<graph[j].city_name<<": ";
                        cin>>d;
                        while(d<=0||d>9999)
                        {
                            cout<<"\n\t\tDistance cannot be negative, 0 or more than 9999km";
                            cout<<"\n\t\tRe-enter distance from "<<graph[i].city_name<<" to ";
                            cout<<graph[j].city_name<<": ";
                            cin>>d;
                        }
                        distance_graph[i][j]=d;
                        distance_graph[j][i]=distance_graph[i][j];
                    }
                }

            }
        }
    }


    TSP tsp;
    tsp.operations(choice,n,graph,distance_graph);

    return 0;
}

/*
Output 1: For Symmetric/Bi-directional graph, that is distance from A to B = B to A


                        AI Travelling Salesman Problem using GA
                         (Mutation method used: Swap mutation)
                                 By Sanskar Sharma
                                     0120180381

                Enter the initial state of system.
        Enter the number of cities: 5

        Enter the names of all the cities.
                Name of 1 city: Delhi

                Name of 2 city: Mumbai

                Name of 3 city: Kolkata

                Name of 4 city: Pune

                Name of 5 city: Neemuch

        Enter the distance graph.
        Enter your choice.
                1. Asymmetric/ Uni-directional.
                2. Symmetric/ Bi-directional.
        Enter: 2

                Enter the distance from Delhi to Mumbai: 2555

                Enter the distance from Delhi to Kolkata: 6223

                Enter the distance from Delhi to Pune: 5667

                Enter the distance from Delhi to Neemuch: 5322

                Enter the distance from Mumbai to Kolkata: 52145

                Distance cannot be negative, 0 or more than 9999km
                Re-enter distance from Mumbai to Kolkata: -5225

                Distance cannot be negative, 0 or more than 9999km
                Re-enter distance from Mumbai to Kolkata: 0

                Distance cannot be negative, 0 or more than 9999km
                Re-enter distance from Mumbai to Kolkata: 2355

                Enter the distance from Mumbai to Pune: 2555

                Enter the distance from Mumbai to Neemuch: 6332

                Enter the distance from Kolkata to Pune: 5456
                Enter the distance from Kolkata to Neemuch: 2355

                Enter the distance from Pune to Neemuch: 2345

        Enter your choice:
                1. Display city distance graph.
                2. Display all possible routes and cost with Optimal solution.
                3. Exit.1

        List of cities with their indexes:
                1. Delhi
                2. Mumbai
                3. Kolkata
                4. Pune
                5. Neemuch
                | 0    2555 6223 5667 5322 |
                | 2555 0    2355 2555 6332 |
                | 6223 2355 0    5456 2355 |
                | 5667 2555 5456 0    2345 |
                | 5322 6332 2355 2345 0    |
        Enter your choice:
                1. Display city distance graph.
                2. Display all possible routes and cost with Optimal solution.
                3. Exit.2

        List of cities with their indexes:
                1. Delhi
                2. Mumbai
                3. Kolkata
                4. Pune
                5. Neemuch
                | 0    2555 6223 5667 5322 |
                | 2555 0    2355 2555 6332 |
                | 6223 2355 0    5456 2355 |
                | 5667 2555 5456 0    2345 |
                | 5322 6332 2355 2345 0    |
        Enter the initial node index: 3

        Initial node is: Kolkata
                Shortest Path possible for Travelling Salesman:
                Kolkata --> Mumbai --> Delhi --> Neemuch --> Pune --> Kolkata
                Min Cost: 18033
        Enter your choice:
                1. Display city distance graph.
                2. Display all possible routes and cost with Optimal solution.
                3. Exit.
```

Output 2: For Asymmetric, that is distance from A to B (not =) B to A

```
                        AI Travelling Salesman Problem using GA
                         (Mutation method used: Swap mutation)
                                 By Sanskar Sharma
                                     0120180381

                Enter the initial state of system.
        Enter the number of cities: 5

        Enter the names of all the cities.
                Name of 1 city: Vishranthwadi

                Name of 2 city: Alandi

                Name of 3 city: Dighi

                Name of 4 city: Moshi

                Name of 5 city: Hinjewadi

        Enter the distance graph.
        Enter your choice.
                1. Asymmetric/ Uni-directional.
                2. Symmetric/ Bi-directional.
        Enter: 1

                Assumption made, All cities have a direct link to all cities.
                Enter the distance from Vishranthwadi to Alandi: 30

                Enter the distance from Vishranthwadi to Dighi: 20

                Enter the distance from Vishranthwadi to Moshi: 40

                Enter the distance from Vishranthwadi to Hinjewadi: 50

                Enter the distance from Alandi to Vishranthwadi: 35

                Enter the distance from Alandi to Dighi: 24

                Enter the distance from Alandi to Moshi: 27
                Enter the distance from Alandi to Hinjewadi: 70

                Enter the distance from Dighi to Vishranthwadi: 34

                Enter the distance from Dighi to Alandi: 26

                Enter the distance from Dighi to Moshi: 10

                Enter the distance from Dighi to Hinjewadi: 67

                Enter the distance from Moshi to Vishranthwadi: 80

                Enter the distance from Moshi to Alandi: 25

                Enter the distance from Moshi to Dighi: 13

                Enter the distance from Moshi to Hinjewadi: 82

                Enter the distance from Hinjewadi to Vishranthwadi: 45

                Enter the distance from Hinjewadi to Alandi: 65

                Enter the distance from Hinjewadi to Dighi: 57

                Enter the distance from Hinjewadi to Moshi: 74

        Enter your choice:
                1. Display city distance graph.
                2. Display all possible routes and cost with Optimal solution.
                3. Exit.1

        List of cities with their indexes:
                1. Vishranthwadi
                2. Alandi
                3. Dighi
                4. Moshi
                5. Hinjewadi
                | 0    30   20   40   50   |
                | 35   0    24   27   70   |
                | 34   26   0    10   67   |
                | 80   25   13   0    82   |
                | 45   65   57   74   0    |
        Enter your choice:
                1. Display city distance graph.
                2. Display all possible routes and cost with Optimal solution.
                3. Exit.2

        List of cities with their indexes:
                1. Vishranthwadi
                2. Alandi
                3. Dighi
                4. Moshi
                5. Hinjewadi
                | 0    30   20   40   50   |
                | 35   0    24   27   70   |
                | 34   26   0    10   67   |
                | 80   25   13   0    82   |
                | 45   65   57   74   0    |
        Enter the initial node index: 2

        Initial node is: Alandi
        The graph is Asymmetrical/ Uni-directional. Therefore, finding the optimum solution for our TSP.
                (NOTE: May not be least!)

                0 Iteration:
                Current route is: Alandi --> Vishranthwadi --> Dighi --> Moshi --> Hinjewadi --> Alandi
                Cost of current route is: 212

                Let's start swapping randomnly and get optimal route.

                1 Iteration:
                Current route is: Alandi --> Vishranthwadi --> Hinjewadi --> Moshi --> Dighi --> Alandi
                Cost of current route is: 198

                2 Iteration:
                Current route is: Alandi --> Moshi --> Hinjewadi --> Vishranthwadi --> Dighi --> Alandi
                Cost of current route is: 200

                3 Iteration:
                Current route is: Alandi --> Hinjewadi --> Moshi --> Vishranthwadi --> Dighi --> Alandi
                Cost of current route is: 270

                4 Iteration:
                Current route is: Alandi --> Vishranthwadi --> Moshi --> Hinjewadi --> Dighi --> Alandi
                Cost of current route is: 240

                5 Iteration:
                Current route is: Alandi --> Vishranthwadi --> Dighi --> Hinjewadi --> Moshi --> Alandi
                Cost of current route is: 221
```

```
6 Iteration:
Current route is: Alandi --> Vishranthwadi --> Moshi --> Hinjewadi --> Dighi --> Alandi
Cost of current route is: 240

7 Iteration:
Current route is: Alandi --> Vishranthwadi --> Moshi --> Dighi --> Hinjewadi --> Alandi
Cost of current route is: 220

8 Iteration:
Current route is: Alandi --> Hinjewadi --> Moshi --> Dighi --> Vishranthwadi --> Alandi
Cost of current route is: 221

9 Iteration:
Current route is: Alandi --> Vishranthwadi --> Moshi --> Dighi --> Hinjewadi --> Alandi
Cost of current route is: 220

10 Iteration:
Current route is: Alandi --> Vishranthwadi --> Dighi --> Moshi --> Hinjewadi --> Alandi
Cost of current route is: 212

11 Iteration:
Current route is: Alandi --> Moshi --> Dighi --> Vishranthwadi --> Hinjewadi --> Alandi
Cost of current route is: 189

12 Iteration:
Current route is: Alandi --> Vishranthwadi --> Moshi --> Vishranthwadi --> Hinjewadi --> Alandi
Cost of current route is: 270

13 Iteration:
Current route is: Alandi --> Vishranthwadi --> Moshi --> Hinjewadi --> Vishranthwadi --> Alandi
Cost of current route is: 232

14 Iteration:
Current route is: Alandi --> Vishranthwadi --> Moshi --> Vishranthwadi --> Hinjewadi --> Alandi
Cost of current route is: 270

15 Iteration:
Current route is: Alandi --> Vishranthwadi --> Moshi --> Hinjewadi --> Vishranthwadi --> Alandi
Cost of current route is: 232

16 Iteration:
Current route is: Alandi --> Moshi --> Vishranthwadi --> Hinjewadi --> Vishranthwadi --> Alandi
Cost of current route is: 232

17 Iteration:
Current route is: Alandi --> Moshi --> Vishranthwadi --> Vishranthwadi --> Hinjewadi --> Alandi
Cost of current route is: 222

18 Iteration:
Current route is: Alandi --> Moshi --> Vishranthwadi --> Hinjewadi --> Vishranthwadi --> Alandi
Cost of current route is: 232

19 Iteration:
Current route is: Alandi --> Vishranthwadi --> Moshi --> Hinjewadi --> Vishranthwadi --> Alandi
Cost of current route is: 232

20 Iteration:
Current route is: Alandi --> Vishranthwadi --> Vishranthwadi --> Hinjewadi --> Moshi --> Alandi
Cost of current route is: 184

21 Iteration:
Current route is: Alandi --> Vishranthwadi --> Vishranthwadi --> Hinjewadi --> Moshi --> Alandi
Cost of current route is: 184

22 Iteration:
Current route is: Alandi --> Moshi --> Vishranthwadi --> Hinjewadi --> Vishranthwadi --> Alandi
Cost of current route is: 232

23 Iteration:
Current route is: Alandi --> Moshi --> Hinjewadi --> Vishranthwadi --> Vishranthwadi --> Alandi
Cost of current route is: 184

24 Iteration:
Current route is: Alandi --> Moshi --> Vishranthwadi --> Vishranthwadi --> Hinjewadi --> Alandi
Cost of current route is: 222

25 Iteration:
Current route is: Alandi --> Moshi --> Hinjewadi --> Vishranthwadi --> Vishranthwadi --> Alandi
Cost of current route is: 184

26 Iteration:
Current route is: Alandi --> Moshi --> Vishranthwadi --> Hinjewadi --> Vishranthwadi --> Alandi
Cost of current route is: 232

27 Iteration:
Current route is: Alandi --> Vishranthwadi --> Vishranthwadi --> Hinjewadi --> Moshi --> Alandi
Cost of current route is: 184

28 Iteration:
```

```
Current route is: Alandi --> Hinjewadi --> Vishranthwadi --> Vishranthwadi --> Moshi --> Alandi
Cost of current route is: 180

29 Iteration:
Current route is: Alandi --> Vishranthwadi --> Vishranthwadi --> Hinjewadi --> Moshi --> Alandi
Cost of current route is: 184

30 Iteration:
Current route is: Alandi --> Hinjewadi --> Vishranthwadi --> Vishranthwadi --> Moshi --> Alandi
Cost of current route is: 180

31 Iteration:
Current route is: Alandi --> Vishranthwadi --> Vishranthwadi --> Hinjewadi --> Moshi --> Alandi
Cost of current route is: 184

32 Iteration:
Current route is: Alandi --> Vishranthwadi --> Hinjewadi --> Vishranthwadi --> Moshi --> Alandi
Cost of current route is: 195

33 Iteration:
Current route is: Alandi --> Vishranthwadi --> Vishranthwadi --> Hinjewadi --> Moshi --> Alandi
Cost of current route is: 184

34 Iteration:
Current route is: Alandi --> Vishranthwadi --> Hinjewadi --> Vishranthwadi --> Moshi --> Alandi
Cost of current route is: 195

35 Iteration:
Current route is: Alandi --> Hinjewadi --> Vishranthwadi --> Vishranthwadi --> Moshi --> Alandi
Cost of current route is: 180

36 Iteration:
Current route is: Alandi --> Hinjewadi --> Moshi --> Vishranthwadi --> Vishranthwadi --> Alandi
Cost of current route is: 254

37 Iteration:
Current route is: Alandi --> Vishranthwadi --> Moshi --> Hinjewadi --> Vishranthwadi --> Alandi
Cost of current route is: 232

38 Iteration:
Current route is: Alandi --> Hinjewadi --> Moshi --> Vishranthwadi --> Vishranthwadi --> Alandi
Cost of current route is: 254

39 Iteration:
Current route is: Alandi --> Vishranthwadi --> Moshi --> Vishranthwadi --> Hinjewadi --> Alandi
Cost of current route is: 270

40 Iteration:
Current route is: Alandi --> Hinjewadi --> Moshi --> Vishranthwadi --> Vishranthwadi --> Alandi
Cost of current route is: 254

41 Iteration:
Current route is: Alandi --> Hinjewadi --> Moshi --> Vishranthwadi --> Vishranthwadi --> Alandi
Cost of current route is: 254

42 Iteration:
Current route is: Alandi --> Hinjewadi --> Vishranthwadi --> Moshi --> Vishranthwadi --> Alandi
Cost of current route is: 265

43 Iteration:
Current route is: Alandi --> Hinjewadi --> Moshi --> Vishranthwadi --> Vishranthwadi --> Alandi
Cost of current route is: 254

44 Iteration:
Current route is: Alandi --> Hinjewadi --> Vishranthwadi --> Vishranthwadi --> Moshi --> Alandi
Cost of current route is: 180

45 Iteration:
Current route is: Alandi --> Hinjewadi --> Vishranthwadi --> Moshi --> Vishranthwadi --> Alandi
Cost of current route is: 265

46 Iteration:
Current route is: Alandi --> Moshi --> Vishranthwadi --> Hinjewadi --> Vishranthwadi --> Alandi
Cost of current route is: 232

47 Iteration:
Current route is: Alandi --> Moshi --> Hinjewadi --> Vishranthwadi --> Vishranthwadi --> Alandi
Cost of current route is: 184

48 Iteration:
Current route is: Alandi --> Moshi --> Vishranthwadi --> Hinjewadi --> Vishranthwadi --> Alandi
Cost of current route is: 232

49 Iteration:
Current route is: Alandi --> Vishranthwadi --> Moshi --> Hinjewadi --> Vishranthwadi --> Alandi
Cost of current route is: 232

50 Iteration:
Current route is: Alandi --> Hinjewadi --> Moshi --> Vishranthwadi --> Vishranthwadi --> Alandi
Cost of current route is: 254
```

```
                End of 50 Iterations
        Optimal solution found at 28 iteration.
        Shortest/Optimal route achieved is: Alandi --> Vishranthwadi --> Dighi --> Moshi --> Hinjewadi --> Alandi
        Minimum cost for the route is: 180
        Enter your choice:
                1. Display city distance graph.
                2. Display all possible routes and cost with Optimal solution.
                3. Exit.3

--------------------------------
Process exited after 222.3 seconds with return value 0
Press any key to continue . . .
*/
```