JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY, NOIDA

ALGORITHMS AND PROBLEM SOLVING PROJECT

TripTrekker

SUBMITTED BY:

RIYA SUMRA (21103155)

PRERNA BADLANI (21103164)

RAJAT KHANDELWAL (21103167)

RHYTHM (21103168)

BATCH- B6

SUBMITTED TO:

MS. SHERRY GARG

DR. DHANALEKSHMI G

DR. BHARAT GUPTA

ABSTRACT

Map applications, as the name suggests, are implementations of maps in a virtual form with an ever increasing number of features that go beyond just showing directions. Some most commonly used navigation apps include Google Maps, Waze, Apple Maps etc. Map applications, now more than ever, have become nothing short of a necessity for most people. In order to plan trips or even daily commutes, it becomes important to know what route would suit one the most so they can use their time and money efficiently. There's also the modern problem of traffic which can cause undesirable delays if the route is not chosen properly.

The purpose of this project is to design a one-day itinerary for a traveller who wants to explore a specific location. The itinerary will include a variety of activities and destinations, based on the traveller's interests and preferences. The itinerary will also be designed to optimise the traveller's time and budget, while providing a memorable and enjoyable experience.

GOALS OF THE PROJECT

- Planning a Full day tour: Planning a Full day visits of the city in a minimum time which can help in saving time and avoid confusion (Gives an efficient path) .User are requested to enter the source destination, Then Travelling Salesman problem / Hamiltonian cycle is applied to give a path (having a full route with same starting point and ending point) with minimum time travel and shows the time travel between the places.

- Checking Traffic Conditions: It helps to know the traffic conditions which can help in saving time.You can avoid congested areas and find alternative routes, ultimately saving you time. By avoiding congested routes, you can save money on gas and reduce wear and tear on your vehicle, ultimately saving you money in the long run.

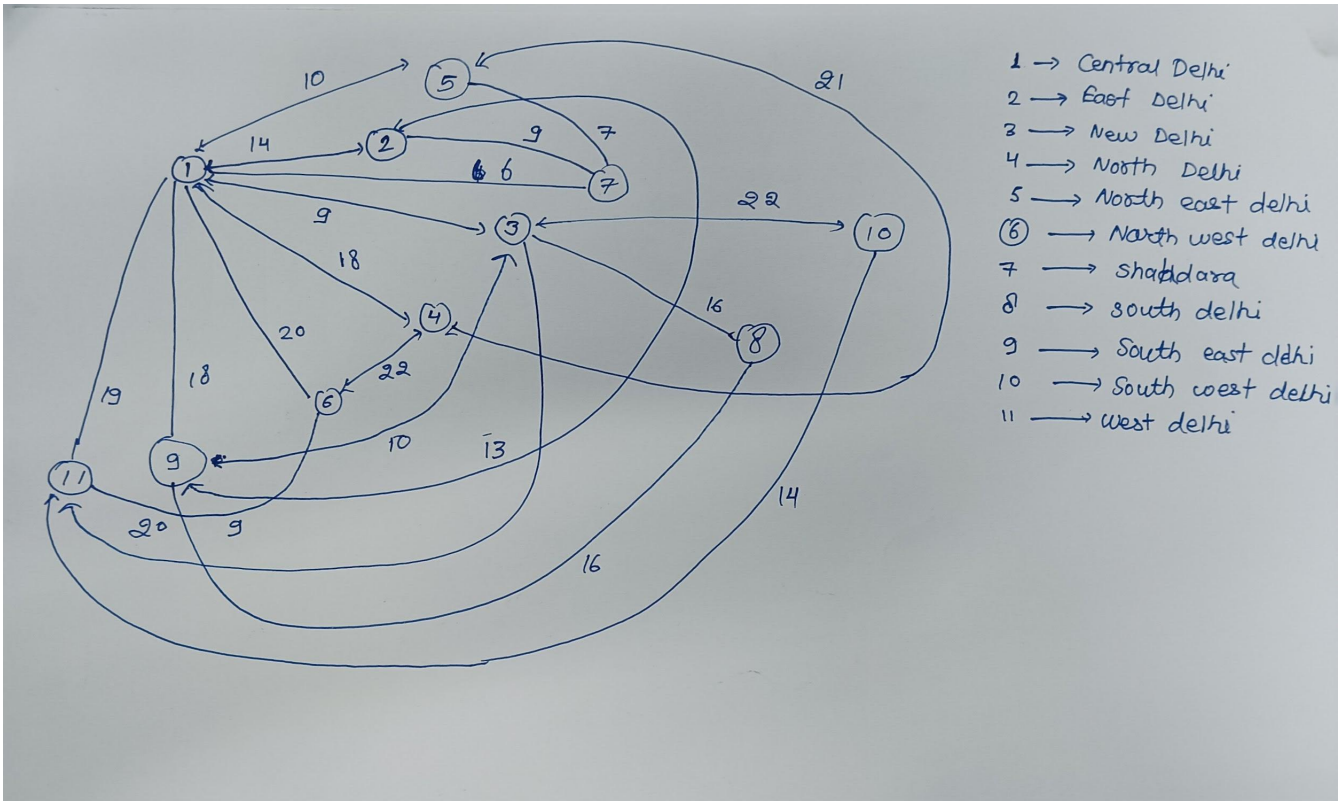
- Full day Activities Planner: It provides you with the list of activities that can be performed at the particular location. The aim is that the user can participate in

maximum activities therefore we have used activity selection using Greedy approach.The users then choose their desired activities, and we have used quicksort to sort the activities on the basis of the end time, therefore avoiding the clash of time.

- Fastest Path recongnization : By finding the fastest path, you can arrive at your destination more quickly and efficiently, ultimately saving you time. Users are requested to enter their source and destination locations where they want to go. Then we applied a Brute Force method to recognise the fastest path. Shorter travel times can also reduce the risk of accidents and other incidents on the road, as you spend less time in traffic and on the road in general.

- Shortest path between two cities: It helps in finding the shortest path, users are requested to enter the district codes of the source and destination locations where they want to trace the path. Then Dijkstra's Algorithm is applied for the entered source and destination giving the shortest route which will help save your time and the hustle to find the ideal route.

GRAPH OF THE PROJECT :-



ALGORITHM USED :-

- **Activity Time Scheduling** - Using Greedy Algorithm - Greedy is an algorithmic paradigm that builds up a solution piece by piece, always choosing the next piece that offers the most obvious and immediate benefit. Greedy algorithms are used for optimization problems.The greedy choice is to always pick the next activity whose finish time is the least among the remaining activities and the start time is more than or equal to the finish time of the previously selected activity. We can sort the activities according to their finishing time so that we always consider the next activity as the minimum finishing time activity

  Time Complexity-**O(n log n)**

- **Dijkstra algorithm** - Dijkstra's algorithm is an algorithm for finding the shortest paths between nodes in a weighted graph, which may represent, for example, road networks. It can also be used for finding the shortest paths from a single node to a single destination node by stopping the algorithm once the shortest path to the destination node has been determined. Dijkstra's original algorithm does not use a min-priority queue and runs in time **Θ(|V|^{2})**(where |V| is the number of nodes).

- **Travelling Salesman Problem Branch and Bound** - The Travelling salesman problem (TSP) is a classic optimization problem in computer science and operations research. It involves finding the shortest possible route that can be taken to visit a set of cities exactly once and return to the starting city with the minimum cost. The TSP is an NP-hard problem, which means that finding the exact optimal solution is computationally infeasible for large problem sizes. One popular approach for solving the TSP is the branch and bound algorithm. The basic idea behind this algorithm is to recursively divide the problem into smaller subproblems, and solve each subproblem separately. The algorithm keeps track of the current best solution found so far, and uses this to guide the search for better solutions.
  Time complexity **-O(n2*2n)**

- **Travelling Salesman Problem / Hamiltonian Cycle (finding minimum cost on basis of time)** - This states that, in a cycle, the same point or vertex should never be revisited. Given several points (cities) to be visited, the objective of the problem is to find the shortest possible route (called Hamiltonian Path) that visits each point exactly once and returns back to the starting point. This algorithm was then used at the core of the web-based tool (a practical use case) developed for release in public domain which helps users find an optimal round-trip route (i.e. Hamiltonian Path) among the points marked on the map.
  Time complexity **-O(n2*2n)**

OUTPUT:-

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

                                TripTrekker

    ********************************************************************************
Choose an option

1. Plan a full city tour

2. Update speed (current speed = 32 km/h)

3. View current traffic conditions

4. Shortest distance between 2 districts

5. Minimum distance to travel efficiently across the city

6. Fastest Path

7. Full day planner

8. Exit
```

```
0. Central Delhi
1. East Delhi
2. New Delhi
3. North Delhi
4. North East Delhi
5. North West Delhi
6. Shahdara
7. South Delhi
8. South East Delhi
9. South West Delhi
10. West Delhi


Enter your source district (no.): 2
```

```
Here is the path we recommend you take to tour the entire city in minimum time. Happy journey!

New Delhi
  |
  |  0.340426 hrs
  v
South Delhi
  |
  |  0.340426 hrs
  v
South East Delhi
  |
  |  0.276596 hrs
  v
East Delhi
  |
  |  0.28125 hrs
  v
Shahdara
  |
  |  0.148936 hrs
  v
North East Delhi
  |
  |  0.212766 hrs
  v
Central Delhi
  |
  |  0.5625 hrs
  v
North Delhi
  |
  |  0.6875 hrs
  v
North West Delhi
  |
  |  0.28125 hrs
  v
West Delhi
  |
  |  0.297872 hrs
  v
South West Delhi
  |
  |  0.6875 hrs
  v
New Delhi

Total time taken = 4.11702 hours
```

```
 Enter new speed in km/h: 14
 Enter valid speed in km/h

 Enter new speed in km/h: 18


 Speed has been updated.
```

```
From Central Delhi to Shahdara
DISTANCE: 6 km
TRAFFIC: LOW
TIME TO REACH WITH CURRENT SPEED: 0.181818 hrs

From Shahdara to Central Delhi
DISTANCE: 6 km
TRAFFIC: NORMAL
TIME TO REACH WITH CURRENT SPEED: 0.333333 hrs
```

```
0. Central Delhi
1. East Delhi
2. New Delhi
3. North Delhi
4. North East Delhi
5. North West Delhi
6. Shahdara
7. South Delhi
8. South East Delhi
9. South West Delhi
10. West Delhi


Enter your source district (no.): 4


Enter your destination district (no.): 2
```

```
Shortest distance from source vertex to destination vertex is :
19
distance of all vertices from destination vertex  :
0                       10
1                       16
2                       19
3                       21
4                       0
5                       30
6                       7
7                       35
8                       28
9                       41
10                      29
```

```
          0. Central Delhi
          1. East Delhi
          2. New Delhi
          3. North Delhi
          4. North East Delhi
          5. North West Delhi
          6. Shahdara
          7. South Delhi
          8. South East Delhi
          9. South West Delhi
          10. West Delhi
Minimum distance required to travel : 156 km
Path Taken : 0 3 5 10 9 2 7 8 1 6 4 0
```

```
    Available activities in East Delhi: (Enter 1 if you want to participate and 0 if not)

    AB Exhibition : 10:00 to 12:00
    0
    Monuments Tour: 11:00 to 14:00
    1
    Water show    : 10:00 to 11:00
    1
    Birdwatching  : 5:00 to 7:00
    1
    Art Gallery   : 16:00 to 17:00
    1

    For participating in max. no. of desired activities,
    we recommend following the following schedule:

    Name                  Start time             End time
    Birdwatching                 5:00           7:00
    Water show                   10:00          11:00
    Monuments Tour               11:00          14:00
    Art Gallery                  16:00          17:00

    Max. no. of activities: 4
```

```
          0. Central Delhi
          1. East Delhi
          2. New Delhi
          3. North Delhi
          4. North East Delhi
          5. North West Delhi
          6. Shahdara
          7. South Delhi
          8. South East Delhi
          9. South West Delhi
          10. West Delhi


          Enter your source district (no.): 0


          Enter your destination district (no.): 2
0 1 2
0 1 3 2
0 2
0 3 1 2
0 3 2
fastest path=0 2

 do you want to continue: 1 to continue and 0 otherwise
```

REFERENCES

- https://www.geeksforgeeks.org/
- https://www.tutorialspoint.com/index.htm
- https://www.w3schools.com/cpp/default.asp
- A. Levitin “Introduction to design and analysis of algorithms”, Second edition.