**Software Requirements Specification**

For

**THE STATE EXCUSRION PLANNER**

22-10-2021

**Prepared by**

|  |  |  |
| --- | --- | --- |
| **Specialization** | **SAP ID** | **Name** |
| CSE-BIGDATA | 500077419 | BHOOMIKA SINGH |
| CSE-BIGDATA | 500077188 | MOHD AHZAM AUSAF |
| CSE-BIGDATA | 500077450 | PRABHJOT SINGH |
| CSE-BIGDATA | 500077183 | SHRUTI SHARMA |



**Department of Informatics**

**School Of Computer Science**

**UNIVERSITY OF PETROLEUM & ENERGY STUDIES,**

**DEHRADUN- 248007. Uttarakhand**

Table of Contents

|  |  |  |
| --- | --- | --- |
| **Topic** | | **Page No** |
| Table of Content | |  |
| Revision History | |  |
| 1 | Introduction |  |
|  | 1.1 Purpose of the Project |  |
|  | 1.2 Target Beneficiary |  |
|  | 1.3 Project Scope |  |
|  | 1.4 References |  |
| 2 | Project Description |  |
|  | 2.1 Reference Algorithm |  |
|  | 2.2 Data/ Data structure |  |
|  | 2.3 SWOT Analysis |  |
|  | 2.4 Project Features |  |
|  | 2.5 User Classes and Characteristics |  |
|  | 2.6 Design and Implementation Constraints |  |
|  | 2.7 Design diagrams |  |
|  | 2.8 Assumption and Dependencies |  |
| 3 | System Requirements |  |
|  | 3.1 User Interface |  |
|  | 3.2 Software Interface |  |
|  | 3.3 Database Interface |  |
|  | 3.4 Protocols |  |
| 4 | Non-functional Requirements |  |
|  | 4.1 Performance requirements |  |
|  | 4.2 Security requirements |  |
|  | 4.3 Software Quality Attributes |  |
| 5 | Other Requirements |  |
| Appendix A: Glossary | |  |
| Appendix B: Analysis Model | |  |
| Appendix C: Issues List | |  |

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Change** | **Reason for Changes** | **Mentor Signature** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

1. **INTRODUCTION:**
   1. PURPOSE OF THE PROJECT:

The State Excursion Planner is a route-map-like system that targets reducing the travel time of the client by demonstrating the briefest course and in this manner enhancing their experience. The objective is to track down the briefest way covering all places the traveller has decided to visit. The course is given by remembering the beginning stage of the excursion.

The idea is to implement the Nearest Neighbor algorithm (NN) and improve the results using Dynamic Programming (DP). The algorithm starts with the user selecting the journey's starting location and then moves on to the next spot closest to the starting place. This iterative loop continues until all of the nodes have been covered and then ends when it returns to the start point. The algorithm will explore all possible routes from the source to the destinations, compute the shortest among them, and give this result as output to the user. The Nearest-Neighbor algorithm has many variants. In this project, The Bellman-Held-Karp algorithm will be used as a dynamic programming variant to find the most optimized route.

* 1. TARGET BENEFICIARY:
* Tourists wanting to explore and visit the beautiful tourist attractions are the target beneficiaries of this project.
* Along with the tourists, it will be beneficial for the tourist guides and tourist management systems working for enhancing the state’s travel experience.
* The businessmen visiting Uttarakhand from different parts of the country, to save their valuable time as well as cost expenditure can use The State Excursion Planner to increase the quality and equity of their business trips.
* The destinations in the project can easily be modified to cater to everyone’s needs.
* As claimed by the researchers studying the Travelling Salesman Problem and the usage of Bellman Held Karp Algorithm in several systems like E-commerce Delivery Routing Systems, a person can save approximately 80% of the travelling time.
  1. PROJECT SCOPE:

Mastering the process of Route Optimization will help calculate the best-optimized route in seconds and thus to save time, money as well as improve customer services in various fields of work like delivery services, cab services, tourist management systems, etc.

Over the projected period of 2020-2025, the route optimization software market is expected to develop at a compound annual growth rate (CAGR) of 10.9 percent. With automated multi-stop route planning capability, route optimization helps drivers fulfill day-to-day activities in less time and removes daily manual hours of preparation.

* 1. REFERENCES:

Lenstra, J. K., & Kan, A. R. (1975). Some simple applications of the travelling salesman problem. *Journal of the Operational Research Society*, *26*(4), 717-733.

Rai, K., Madan, L., & Anand, K. (2014). Research paper on travelling salesman problem and it’s solution using genetic algorithm. *International Journal of Innovative Research in Technology*, *1*(11), 103-114.

Lai, F., Szczecinski, M., So, W., & Fernandez, M. Improving Operations with Route Optimization.

Held, M., & Karp, R. M. (1962). A dynamic programming approach to sequencing problems. *Journal of the Society for Industrial and Applied mathematics*, *10*(1), 196-210.

Karkory, F. A., & Abudalmola, A. A. (2013). Implementation of heuristics for solving travelling salesman problem using nearest neighbour and minimum spanning tree algorithms. *International Journal of Computer and Information Engineering*, *7*(10), 1524-1534.

1. **PROJECT DESCRIPTION:**
   1. REFERENCE ALGORITHM:

**Bellman-Held-Karp-Algorithm:**

1. Select any starting vertex s ∈ V .
2. For each S ⊆ V − s and v ∈ S, let:

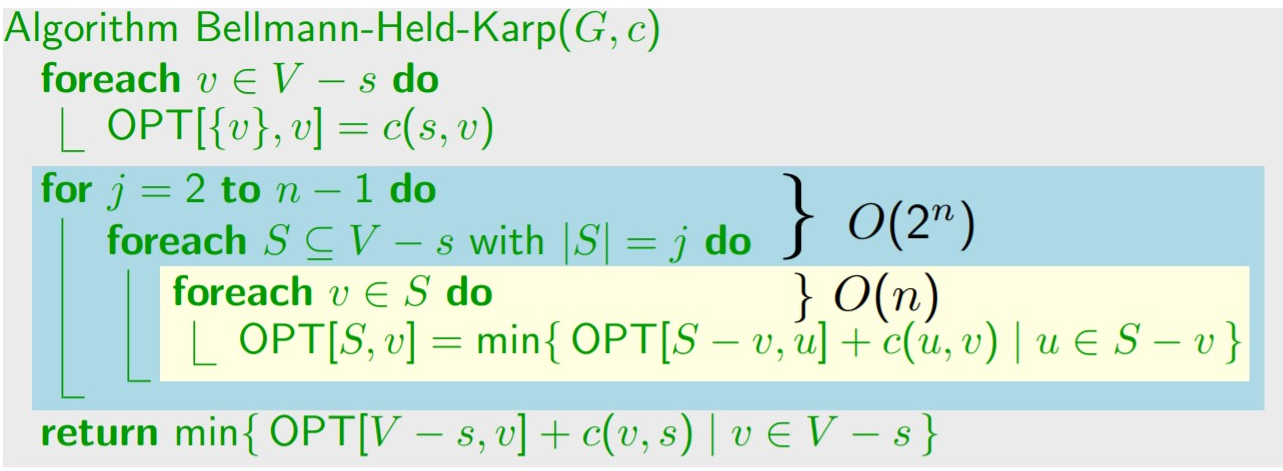
OPT[S, v] = length of a shortest s-v-path that visits precisely the vertices of S ∪ {s}.

1. The base case: S = {v}, is easy: OPT[{v}, v] = c(s, v).
2. When |S| ≥ 2, we compute OPT[S, v] recursively:

OPT[S, v] = min{ OPT[S − v, u] + c(u, v) | u ∈ S − v }

1. After computing OPT[S, v] for each S ⊆ V − s, the optimal

solution is easily obtained as follows: OPT= min{ OPT[V − s, v] + c(v, s) | v ∈ V − s }



2.2 CHARACTERISTIC OF DATA

**Dataset as adjacency matrix:**

**List of Tourist Places:**

{Starting Location, Valley of Flowers, Rajaji National Park, Kedarnath, Laxman Jhula, FRI, Mussoorie, Beatles Ashram, Bhimtal Lake, Gangotri, Har Ki Pauri, Jim Corbett, Auli, Chakrata, Nainital Lake}

**Distance Dataset:**

**float DDN\_Railway\_Station[15][15] =**

**{**

0.0, 306.8, 57.5, 254.0, 45.2, 5.7, 34.2, 53.2, 287.5, 242.0, 51.9, 177.7, 305.0, 87.1, 273.8,

306.8, 0.0, 284.1, 231.0, 262.8, 312.6, 315.2, 265.5, 286.7, 398.5, 291.7, 271.9, 457.2, 394.6, 287.7,

57.5, 284.1, 0.0, 238.0, 22.6, 63.1, 89.5, 22.2, 257.5, 280.8, 23.3, 147.7, 207.9, 145.4, 243.8,

254.0, 231.0, 238.0, 0.0, 216.0, 254.0, 260.0, 219.0, 295.0, 408.0, 122.0, 293.0, 187.0, 114.0, 154.0,

45.2, 262.8, 22.6, 216.0, 0.0, 50.7, 77.1, 11.1, 265.6, 263.9, 30.0, 155.8, 195.6, 133.0, 251.9,

5.7, 312.6, 63.1, 254.0, 50.7, 0.0, 35.0, 59.0, 293.3, 242.7, 57.7, 183.5, 145.8, 83.3, 279.5,

34.2, 315.2, 89.5, 260.0, 77.1, 35.0, 0.0, 83.9, 318.1, 213.7, 82.5, 208.3, 167.8, 81.1, 304.4,

53.2, 265.5, 22.2, 219.0, 11.1, 59.0, 83.9, 0.0, 267.0, 273.8, 32.4, 157.2, 203.5, 141.0, 253.3,

287.5, 286.7, 261.2, 295.0, 269.4, 296.9, 323.0, 270.7, 0.0, 526.6, 242.8, 77.5, 441.5, 378.9, 21.4,

242.0, 398.5, 281.0, 408.0, 264.2, 241.9, 214.1, 274.1, 523.1, 0.0, 287.5, 413.3, 313.8, 296.0, 509.4,

51.9, 291.7, 21.8, 122.0, 27.3, 54.8, 80.9, 29.7, 239.3, 284.5, 0.0, 129.5, 199.3, 136.8, 225.6,

177.7, 271.9, 151.4, 293.0, 159.6, 187.0, 213.2, 160.9, 48.5, 416.8, 133.0, 0.0, 331.6, 269.1, 63.7,

305.0, 457.2, 209.7, 186.0, 195.6, 145.8, 169.8, 203.6, 437.9, 313.8, 202.3, 344.0, 0.0, 97.9, 424.1,

87.1, 394.6, 145.3, 114.0, 133.0, 83.2, 84.9, 141.0, 375.3, 296.0, 139.7, 265.5, 97.9, 0.0, 361.6,

273.8, 287.7, 251.7, 154.0, 259.9, 287.4, 313.5, 261.2, 21.4, 517.1, 233.3, 68.0, 431.9, 369.4, 0.0

**}**

**float DDN\_Airport[15][15] =**

**{**

0.0,284.3, 35.0, 254.0, 22.6, 32.2, 59.3, 30.7, 275.5, 265.4, 39.8, 165.6, 176.8, 114.3, 261.7,

284.3, 0.0, 284.1, 231.0, 262.8, 312.6, 315.2, 265.5, 286.7, 398.5, 291.7, 271.9, 457.2, 394.6, 287.7,

35.0, 284.1, 0.0, 238.0, 22.6, 63.1, 89.5, 22.2, 257.5, 280.8, 23.3, 147.7, 207.9, 145.4, 243.8,

254.0, 231.0, 238.0, 0.0, 216.0, 254.0, 260.0, 219.0, 295.0, 408.0, 122.0, 293.0, 187.0, 114.0, 154.0,

22.6, 262.8, 22.6, 216.0, 0.0, 50.7, 77.1, 11.1, 265.6, 263.9, 30.0, 155.8, 195.6, 133.0, 251.9,

32.2, 312.6, 63.1, 254.0, 50.7, 0.0, 35.0, 59.0, 293.3, 242.7, 57.7, 183.5, 145.8, 83.3, 279.5,

59.3, 315.2, 89.5, 260.0, 77.1, 35.0, 0.0, 83.9, 318.1, 213.7, 82.5, 208.3, 167.8, 81.1, 304.4,

30.7, 265.5, 22.2, 219.0, 11.1, 59.0, 83.9, 0.0, 267.0, 273.8, 32.4, 157.2, 203.5, 141.0, 253.3,

275.5, 286.7, 261.2, 295.0, 269.4, 296.9, 323.0, 270.7, 0.0, 526.6, 242.8, 77.5, 441.5, 378.9, 21.4,

265.4, 398.5, 281.0, 408.0, 264.2, 241.9, 214.1, 274.1, 523.1, 0.0, 287.5, 413.3, 313.8, 296.0, 509.4,

39.8, 291.7, 21.8, 122.0, 27.3, 54.8, 80.9, 29.7, 239.3, 284.5, 0.0, 129.5, 199.3, 136.8, 225.6,

165.6, 271.9, 151.4, 293.0, 159.6, 187.0, 213.2, 160.9, 48.5, 416.8, 133.0, 0.0, 331.6, 269.1, 63.7,

176.8, 457.2, 209.7, 186.0, 195.6, 145.8, 169.8, 203.6, 437.9, 313.8, 202.3, 344.0, 0.0, 97.9, 424.1,

114.3, 394.6, 145.3, 114.0, 133.0, 83.2, 84.9, 141.0, 375.3, 296.0, 139.7, 265.5, 97.9, 0.0, 361.6,

261.7, 287.7, 251.7, 154.0, 259.9, 287.4, 313.5, 261.2, 21.4, 517.1, 233.3, 68.0, 431.9, 369.4, 0.0

**}**

**float DDN\_ISBT [15][15] =**

**{**

0.0, 309.4, 60.1, 254.0, 47.8, 8.2, 39.3, 55.8, 290.1, 247.1, 54.5, 180.3, 151.2, 88.7, 276.4

309.4, 0.0, 284.1, 231.0, 262.8, 312.6, 315.2, 265.5, 286.7, 398.5, 291.7, 271.9, 457.2, 394.6, 287.7,

60.1, 284.1, 0.0, 238.0, 22.6, 63.1, 89.5, 22.2, 257.5, 280.8, 23.3, 147.7, 207.9, 145.4, 243.8,

254.0, 231.0, 238.0, 0.0, 216.0, 254.0, 260.0, 219.0, 295.0, 408.0, 122.0, 293.0, 187.0, 114.0, 154.0,

47.8, 262.8, 22.6, 216.0, 0.0, 50.7, 77.1, 11.1, 265.6, 263.9, 30.0, 155.8, 195.6, 133.0, 251.9,

8.2, 312.6, 63.1, 254.0, 50.7, 0.0, 35.0, 59.0, 293.3, 242.7, 57.7, 183.5, 145.8, 83.3, 279.5,

39.3, 315.2, 89.5, 260.0, 77.1, 35.0, 0.0, 83.9, 318.1, 213.7, 82.5, 208.3, 167.8, 81.1, 304.4,

55.8, 265.5, 22.2, 219.0, 11.1, 59.0, 83.9, 0.0, 267.0, 273.8, 32.4, 157.2, 203.5, 141.0, 253.3,

290.1, 286.7, 261.2, 295.0, 269.4, 296.9, 323.0, 270.7, 0.0, 526.6, 242.8, 77.5, 441.5, 378.9, 21.4,

247.1, 398.5, 281.0, 408.0, 264.2, 241.9, 214.1, 274.1, 523.1, 0.0, 287.5, 413.3, 313.8, 296.0, 509.4,

54.5, 291.7, 21.8, 122.0, 27.3, 54.8, 80.9, 29.7, 239.3, 284.5, 0.0, 129.5, 199.3, 136.8, 225.6,

180.3, 271.9, 151.4, 293.0, 159.6, 187.0, 213.2, 160.9, 48.5, 416.8, 133.0, 0.0, 331.6, 269.1, 63.7,

151.2, 457.2, 209.7, 186.0, 195.6, 145.8, 169.8, 203.6, 437.9, 313.8, 202.3, 344.0, 0.0, 97.9, 424.1,

88.7, 394.6, 145.3, 114.0, 133.0, 83.2, 84.9, 141.0, 375.3, 296.0, 139.7, 265.5, 97.9, 0.0, 361.6,

276.4, 287.7, 251.7, 154.0, 259.9, 287.4, 313.5, 261.2, 21.4, 517.1, 233.3, 68.0, 431.9, 369.4, 0.0

**}**

2.3 SWOT ANALYSIS FOR THE STATE EXCURSION PLANNER:

**STRENGTHS:**

1. Time saver: Providing travellers with the most optimal option for their journey.
2. Fuel Cost:

Fuel consumption can be reduced by using highway miles instead of surface streets with stop and starts, and by using an algorithm rather than guessing to identify the most efficient route.

1. Vehicle Maintenance Expenses:

Effective route planning can lower vehicle maintenance costs.

Vehicles are driven for fewer hours and waste less mileage. As a result, there is less wear and tear on the vehicle, as well as cheaper fuel expenses.

**WEAKNESS:**

1. Due to the several constraints like constructions, roads blocked as well as the continuous development of new highways and expressways, it is impossible to trace the ideal path in real-time, online.
2. Covering all the Tourism places in a state like Uttarakhand would be quite difficult since there are several hidden and unexplored places yet to be discovered. Thus, the project includes the destinations that have been visited the maximum number of times by the travellers.

**OPPORTUNITIES:**

1. Visualization of a route plan

Route planning is a very visual process. We can expand on this project in the future. It's a lot easier to plan a delivery route when you can see exactly where each stop is, which driver will serve it, and how changes to a stop would affect the whole route.

1. Working as a tourist guide and responding to customer requests to create tour packages using your services.

**THREATS:**

1. Saturated Market-As the number of travellers around the world increases, more tour optimizers are developing.
   1. PROJECT FEATURES:
2. The user must choose one of three reference locations: Dehradun train station, Jolly Grant, or Dehradun ISBT.
3. He has the option of entering as many of the top 15 tourist attractions as he wishes.
4. The State Excursion Planner will present him with the best optimal routing that includes all of his visiting sites while saving his time.
5. Dynamic programming with optimal substructures is used.

The concept is to simply save the results of subproblems so that we don't have to recalculate them later.

1. The runtime complexity of this simple optimization is reduced from exponential to polynomial.
2. Given the current stage of dynamic programming, future decisions will not be influenced by previous decisions or states.
   1. USER CLASSES AND CHARACTERISTICS:
3. Users of the software system include researchers, practicing engineers, and software developers.
4. Data structures and algorithms should be familiar to all users.
5. Users must understand the concepts of time and space complexity.
6. Users should be familiar with the tasks, processes, and algorithms involved in dynamic programming.
   1. DESIGN AND IMPLEMENTATION CONSTRAINTS:

Installation and configuration of any open-source IDE for the C programming language are included in the setup and maintenance. Other than those defined for an application that executes according to the standard documentation, the IDEs have no extra requirements.

Unless an IDE is cloud-based, a developer's operating system will limit which IDEs are accessible, and if the application is meant for an end-user with a certain operating system (such as Android or iOS), this may be an additional constraint.

* 1. DESIGN DIAGRAMS:
* USE –Case, Class Diagram
* Activity Diagram
* Sequence Diagram
* Data Flow diagram
* State Diagram.
* Use Case Diagram for the State Excursion Pl Fig.2.7.1

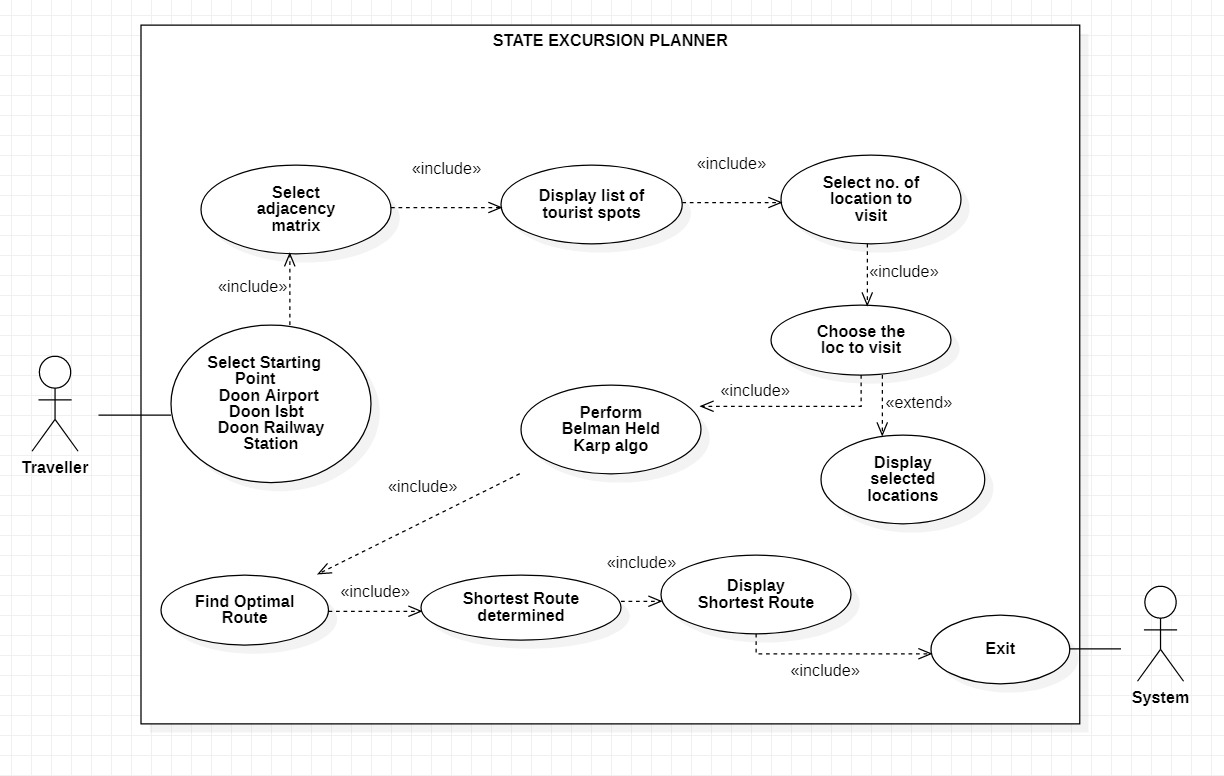


Fig.2.7.1

* Activity Diagram for the State Excursion Planner

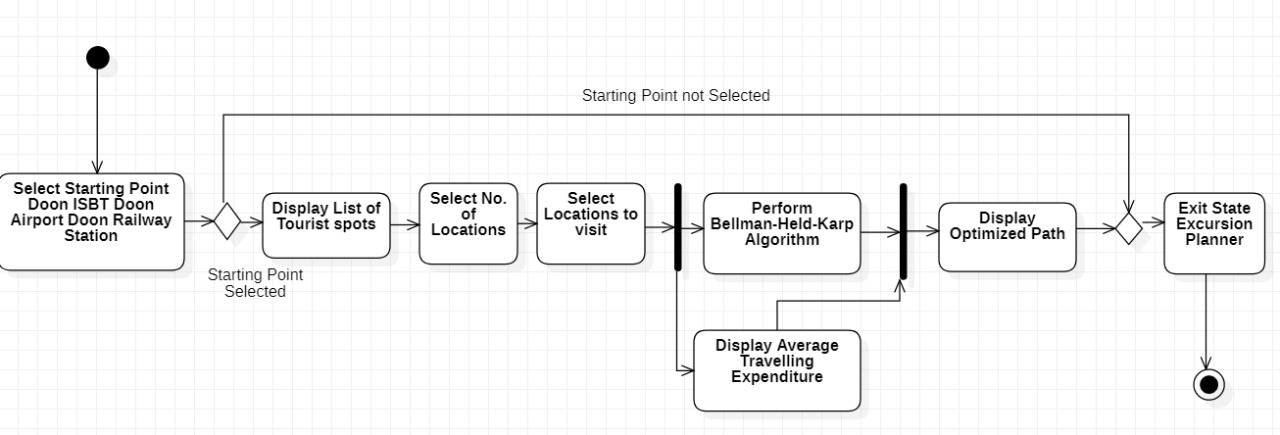


Fig.2.7.2

* Sequence Diagram for the State Excursion Planner

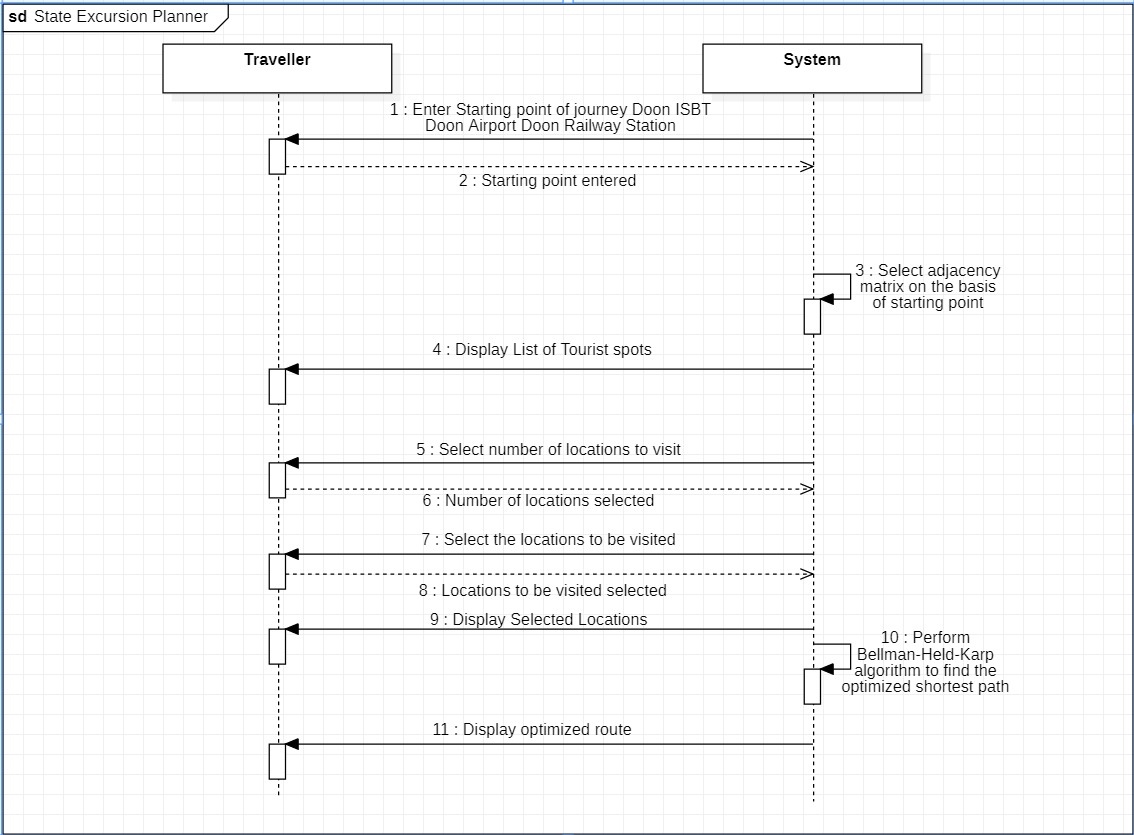


Fig.2.7.3

* Data Flow Diagram for the State Excursion Planner

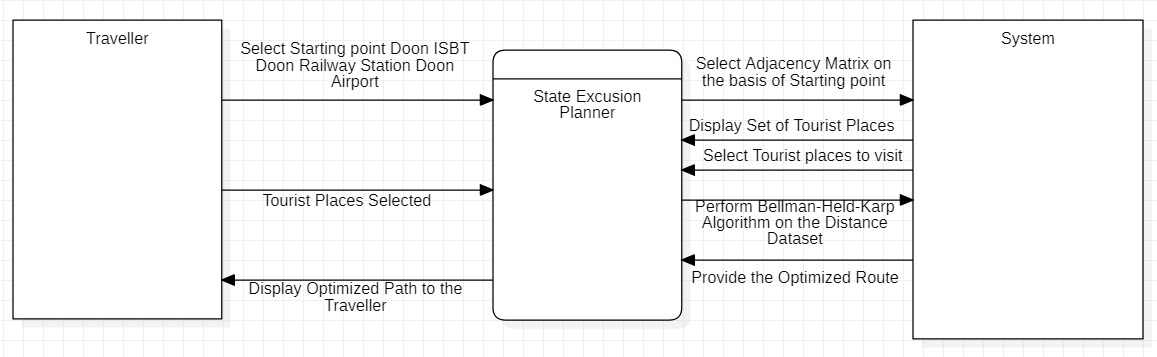


Fig.2.7.4

* State Diagram for the State Excursion Planner

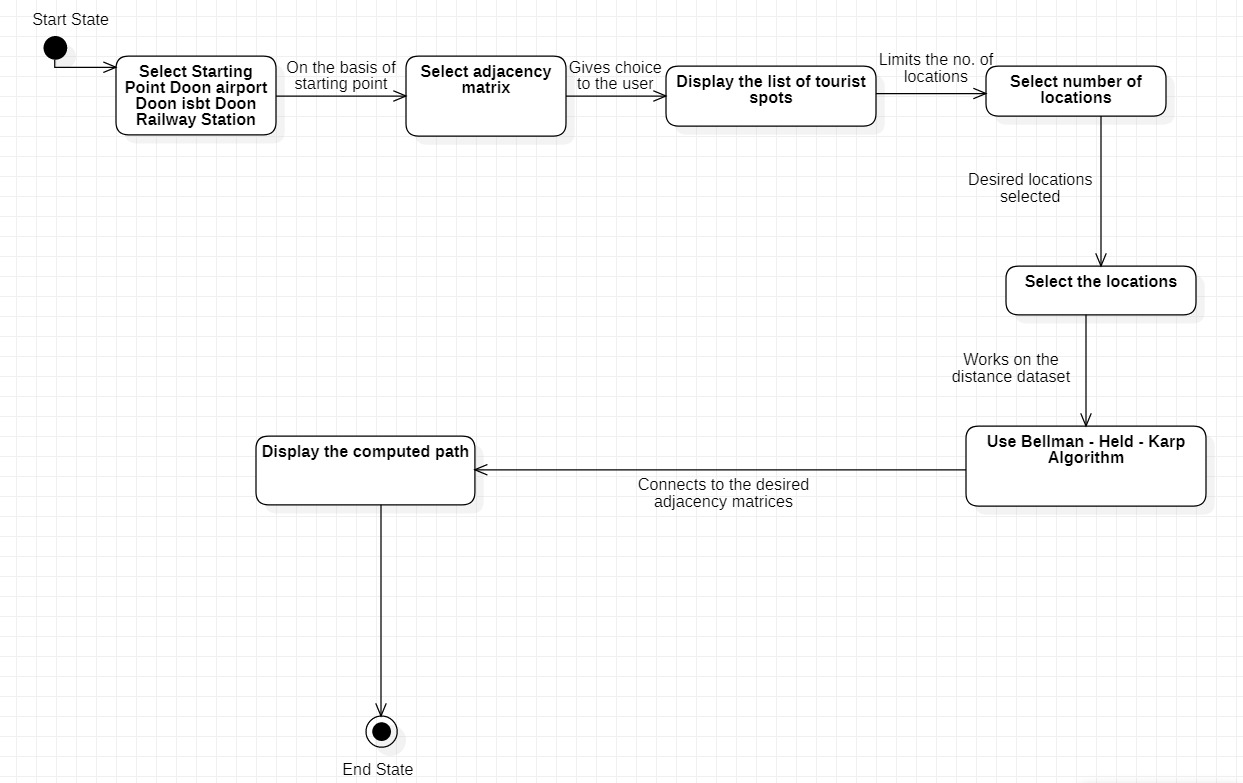


Fig.2.7.5

* 1. ASSUMPTION AND DEPENDENCIES:

At this time, no particular assumptions or dependencies are being considered.

**3. SYSTEM REQUIREMENTS:**

3.1 USER INTERFACE

Any IDE is necessary for writing source code in C, for example CodeBlocks, VScode etc. and a C-compiler is required to turn source code into an executable file.

3.2 SOFTWARE INTERFACE

**Software Used:**

Windows OS

**Description:**

We selected Windows as our operating system because of its excellent support and user-friendliness.

3.3 DATABASE INTERFACE

Instead of using a database, we used an adjacency matrix as a dataset.

3.4 PROTOCOLS

TLS: Stands for Transport Layer Security, a cryptographic protocol that provides communications security over a computer network.

**4. NON- FUNCTIONAL REQUIREMENTS:**

4.1 PERFORMANCE REQUIREMENTS:

The application should be able to react to the traveller’s questions in a timely manner. The software should not take long to display the results when a user searches for the places he wishes to visit.

4.2 SECURITY REQUIREMENTS:

Since the tourist or guide decides to utilize the STATE EXCURSION PLANNER, he or she will do it on his or her own system, ensuring the authenticity and privacy of their personal information.

There are no external threats for any other external party.

4.3 SOFTWARE QUALITY ATTRIBUTES:

**Adaptability:** The project's design in terms of creating tour plans for adaptively guiding a visitor and ensuring safe journeys.

**Availability:** Reasonable efforts should be made to guarantee that the source code works properly in all IDEs and compilers. Optimized path should be available to users as quickly as possible.

**Correctness:** From the correct start terminal, the visitor should arrive at his correct destination.

**Flexibility:** The project demonstrates the feasibility and significance of the tour optimization method, and it can be used by transit planners to design a superior travel service further. Personalization can also be done.

**Interoperability:** Both the source code and the executable file should be saved in separate files and both would be separately operable in different software.

**Maintainability:** The administrators should maintain correct dataset.

**Portability:** The STATE EXCURSION PLANNER would be user-friendly.

The system should be able to be operated from any location without difficulty.

**Reliability:** Under normal circumstances, the application should work properly and efficiently and perform the required functions without any delay.

**Reusability:** A user interface for updating dataset and adding more tourist attractions.

**Robustness:** The project should be able to handle large datasets properly.

**Testability:** The project should run properly without any errors in compilation.

**Usability:** The STATE EXCUSRION PLANNER should be able to meet the needs of as many consumers as possible.

**5. OTHER REQUIREMENTS:**

**5.1 ER-Diagram:**

**APPENDIX A: GLOSSARY**

1. **UI:** The user interface (UI) is the space where interactions between humans and machines occur.
2. **OS:** An **operating system** is the **most important software** that runs on a computer. It manages the computer's **memory** and **processes**, as well as all of its **software** and **hardware.**
3. **IDE:** An integrated development environment (IDE) is a software application that provides full software development features to computer programmers. A source code editor, build automation tools, and a debugger are usually included in an IDE.
4. **C:** C is a general-purpose, procedural computer programming language supporting structured programming, lexical variable scope, and recursion, with a static type system.
5. **COMPILER:** The name "compiler" is primarily used for programs that translate source code from a high-level programming language to a lower-level language to create an executable program.
6. **Dataset:** A data set is a collection of data.
7. **Data Structure:** A data structure is a particular way of organizing data in a computer so that it can be used effectively.
8. **Graph**: A Graph is a non-linear data structure consisting of nodes and edges. A Graph consists of a finite set of vertices (or nodes) and set of Edges which connect a pair of nodes.
9. **Adjacency Matrix**: Adjacency Matrix is a 2D array of size V x V where V is the number of vertices in a graph.
10. **Dynamic Programming:** It is mainly an optimization over plain recursion. Wherever we see a recursive solution that has repeated calls for same inputs, we can optimize it using Dynamic Programming. The idea is to simply store the results of subproblems, so that we do not have to re-compute them when needed later. This simple optimization reduces time complexities from exponential to polynomial.
11. **Time Complexity:**  time complexity is the amount of time taken by an algorithm to run, as a function of the length of the input.

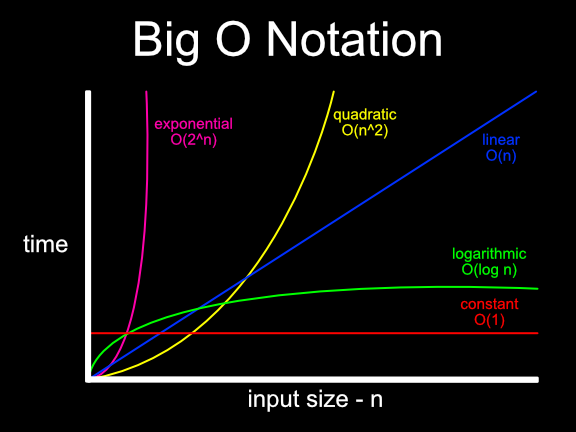


Fig AA.1

Big O Notation Time Complexity Graph

SOURCE: [medium.com](https://medium.com/@ashish_12343/big-o-notation-common-runtime-scenarios-f0cf714b18e3), Google

1. **MTBF:** Mean time between failures is the predicted elapsed time between inherent failures of a mechanical or electronic system, during normal system operation.
2. **MT Connect:** An open industry standard for enabling interoperability between controls, devices, and software in manufacturing systems.
3. **MTTF:** Mean time to failure is the average time between non-repairable failures of a technology product.
4. **Personalization:** A personalization is a set of changes made to your application for a specific group of visitors.
5. **TLS:** Stands for Transport Layer Security, a cryptographic protocol that provides communications security over a computer network.
6. **UUID:** Universally Unique Identifiers, or UUIDS, are 128-bit numbers, composed of 16 octets and represented as 32 base-16 characters, that can be used to identify information across a computer system.

**APPENDIX B: ANALYSIS MODEL**

**APPENDIX C: ISSUES LIST**

1. **System Failure:** A system failure can occur because of a hardware failure or a severe software issue, causing the system to freeze, reboot, or stop functioning altogether.
2. **Runtime-error:** Run-time errors are errors that occur during program execution (run-time) following successful compilation. These errors are difficult to locate since the compiler does not indicate the line where the error occurs.
3. **Compile-time error:** Compile-Time errors are errors that arise when you break the conventions of writing syntax. Before the code can be compiled, this compiler error indicates that something needs to be fixed. All of these problems are known as compile-time errors since they are identified by the compiler.