MODULE 3 (B17)

Implementation of Floyd Warshall algorithm in

Route Map Navigation

# Description of the module :

The third module of my project focuses on developing the route map navigation system using the Floyd-Warshall algorithm implementation from Module 2. This module encompasses all the components and functionalities required to make the navigation system user-friendly and efficient. It integrates the algorithm's results with various features, such as user input, map visualization, landmark restriction within your campus, and shortest distance calculation.

The purpose of this module is to enhance the usability and practicality of the route map navigation system. While the implementation of the Floyd-Warshall algorithm in Module 2 lays the foundation for finding the shortest distance between any two points, Module 3 expands upon it to create a comprehensive application. This module is necessary to address the following requirements:

1. User Interaction: Module 3 enables users to interact with the navigation system. Users can input their current location and desired destination, providing the system with the necessary information to calculate the shortest route.
2. Map Visualization: It is essential to have a visual representation of the campus map to aid users in understanding the routes and landmarks. Module 3 should include functionality to display the map, highlighting the paths and landmarks. This visual representation will greatly assist users in identifying their current location, target destination, and the route they need to follow.
3. Landmark Restriction: Within campus, certain areas or landmarks may have restrictions or be inaccessible. Module 3 should incorporate these restrictions into the navigation system to ensure that the calculated routes are feasible and abide by the campus rules. This feature will help users avoid restricted areas and navigate safely within the campus.

# Motivation for the module :

The motivation behind Module 3 of our project lies in the need to create a comprehensive route map navigation system that goes beyond the implementation of the Floyd-Warshall algorithm. This module is crucial in transforming the algorithm's raw output into a user-friendly and practical application that caters specifically to the navigation needs within campus.

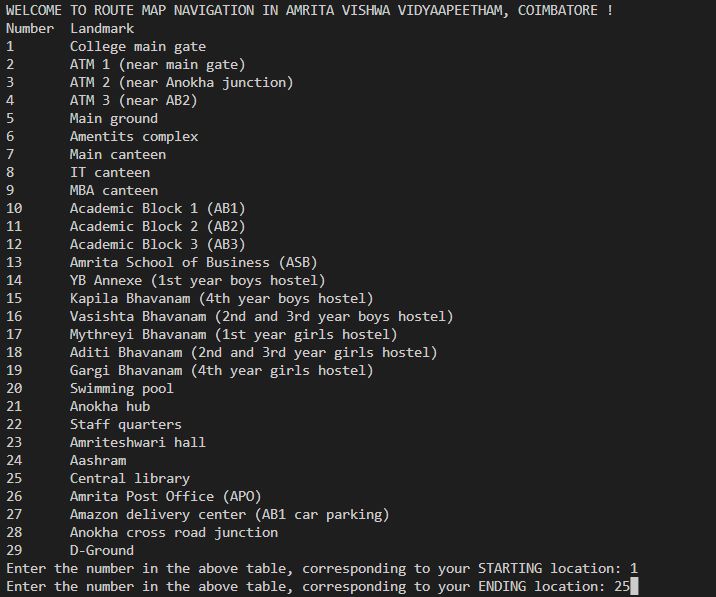
1. User-Friendly Interface: The implementation of the Floyd-Warshall algorithm in Module 2 provides the shortest distance between any two points within the campus. However, users may not be comfortable working directly with the algorithm or understanding the raw results it produces. Module 3 aims to bridge this gap by designing an intuitive user interface that allows users to input their current location and desired destination easily. This interface will provide a more user-friendly and accessible experience for individuals navigating through the campus.
2. Map Visualization: Understanding the routes and landmarks visually is essential for efficient navigation. Module 3 takes into account the importance of map visualization by integrating features that allow users to view the campus map with highlighted routes and landmarks. By providing a graphical representation of the map, users can easily comprehend their current location, the path they need to follow, and the landmarks they should encounter along the way. This visual aid significantly enhances the navigation experience, making it easier for users to traverse the campus.
3. In conclusion, the motivation for Module 3 stems from the desire to create a user-friendly, campus-specific route map navigation system that goes beyond the raw results of the Floyd-Warshall algorithm. By addressing user interface design, campus-specific features, map visualization, enhanced user experience, and practical application, Module 3 ensures that the navigation system caters to the unique needs of your campus community, facilitating efficient and reliable navigation.

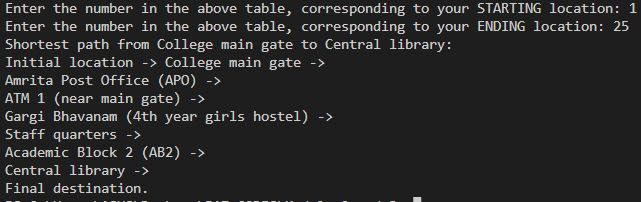
# Relevance of the module in entire system :

Module 3 plays a crucial role in the overall route map navigation system, complementing the implementation of the Floyd-Warshall algorithm in Module 2. Its relevance can be understood in the context of the entire system in the following ways:

1. Landmark Restriction Integration: Within the context of your campus, there may be specific landmarks or areas that have restrictions or limitations. Module 3 incorporates these landmark restrictions into the navigation system, ensuring that generated routes adhere to the rules and regulations of your campus. By integrating these restrictions, the module makes the system relevant and practical for navigation within the campus environment.
2. Integration with Module 2: Module 3 acts as the bridge between the core algorithm implementation in Module 2 and the user-facing functionalities of the navigation system. It takes the outputs from the Floyd-Warshall algorithm, such as the shortest distances between locations, and utilizes this information to generate user-friendly routes. By integrating with Module 2, Module 3 ensures that the algorithm's results are presented in a relevant and practical manner for users.
3. In summary, Module 3 relevance in the entire route map navigation system lies in its contribution to user interaction, incorporating landmark restrictions, providing visualization and contextual awareness, integrating with the algorithmic outputs from Module 2, and completing the system by including all necessary components.

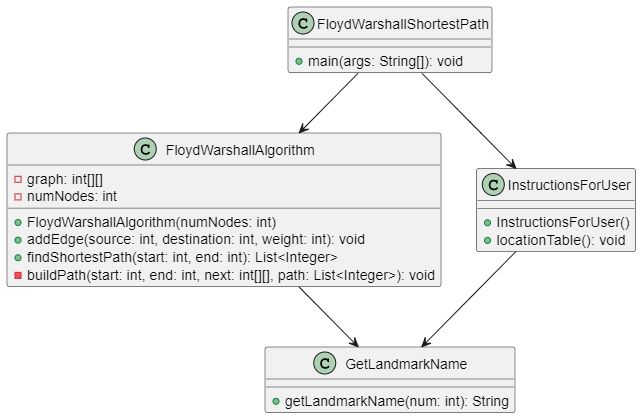
# Input and Output :





# Appendix :

## UML Diagram :



## *Breakdown of methods in all the classes :*

1. *FloydWarshallAlgorithm -> FloydWarshallAlgorithm :*

This is a constructor which creates the graph (tree) on which we are going to apply the algorithm, every time an object of this class is created.

1. *FloydWarshallAlgorithm -> addEdge :*

Using this method, We can add nodes to the tree. We will be using a total of 29 nodes and 36 connections between them in this navigation considering the landmarks in our campus. We have to specify the number corresponding to the starting and ending nodes of the road, and the weight (Which is calculated based on the real time distance [scale : 1 unit = 25 metres]), Based on which we can create nodes as required in the contrast of the university map.

1. *FloydWarshallAlgorithm -> findShortestPath*

This is the main method, Which takes the numbers corresponding to the starting and ending locations and gives the shortest path between them using the logic behind Floyd Warshall algorithm.

1. *FloydWarshallAlgorithm -> bulidPath*

The method takes four parameters: start and end represent the starting and ending vertices of the path, next is the matrix that stores the next vertex in the shortest path, and path is a list to store the path.The recursive nature of the buildPath method allows it to traverse the "next" matrix in reverse, building the path from the destination vertex (end) back to the starting vertex (start).

### *InstructionsForUser -> InstructionsForUser :*

This is a constructor used to print the welcome message every time an object is created for this class. It also pauses the whole compilation process for 1 second so that the user experience will be more immersive.

1. *InstructionsForUser -> LocationTable :*

This method gets the real time locations mapped with the temporary numbers and helps in displaying them in form of a table so that the user gets to know what numbers to enter for getting the path between his/her desired locations.

1. *GetLandMarkName -> getLandmarkName :*

This method has the huge switch case with 29 conditions which basically maps between the numbers assigned to nodes and the locations corresponding to those numbers.

### *FloydWarshallShortesPath -> main function :*

In this main function, we organize everything as discussed above, To justify the working mechanism on which the whole code relies.

MODULE PRESENTED BY : ANNA V VISWANADH[CB.EN.U4AIE22105]