| **STUDENT PORTFOLIO** | |
| --- | --- |
|  | **Name: Kaluru Anirudh**  **Register Number: RA2311056010001**  **Mail ID: ka9890@srmist.edu.in**  **Department: DATA SCIENCE AND BUSINESS SYSTEMS**  **Semester: 3**  **Work Experience: JAVA Project Intern at Infosys** |
| **Subject Title: 21CSC201J Data Structures and Algorithm**  **Handled By: Dr. Rajkumar K.** | |
| **ELAB Completion Status Completed 4-5 questions from all the topics** | |
| **Lab Experiment Completion Status**  **All the 15 experiments were solved, run and executed successfully and took the completed sign from Faculty.** | |
| **REAL WORLD APPLICATION IN DSA PPT VR/SIMULATION DEMO**  **Navigation System Using Dijkstra’s Algorithm**  **The Navigation System project showcases an efficient pathfinding solution for calculating the shortest route between locations on a map. By implementing Dijkstra's algorithm, the system efficiently manages navigation data, demonstrating the application of graph theory and algorithmic optimization in real-world scenarios. The project leverages key data structures, such as adjacency lists and priority queues, to ensure optimized performance, especially in large and complex road networks.**  **The program is designed with two primary components to handle navigation data effectively:**  **1. Graph Representation Using Adjacency List:**  - This structure represents each location as a node, with edges indicating the paths between locations, storing the distances or weights. The adjacency list is optimal for large, sparse networks, allowing for efficient edge storage without excessive memory use.  - The adjacency list format is ideal for graph traversal algorithms like Dijkstra's, enabling easy access to neighboring nodes, reducing overhead in scenarios with vast numbers of nodes and selective connectivity.  **2. Priority Queue (Min-Heap):**  - The priority queue is essential in Dijkstra's algorithm for efficiently retrieving the next closest node based on the minimum distance. By using a min-heap-based priority queue, the algorithm ensures that each extraction and insertion is handled in **O(log V)** time.  - This data structure supports real-time priority updates, necessary for recalculating paths dynamically as traffic data changes, making it suitable for navigation applications that need frequent, low-latency route adjustments.  **This project showcases the strategic combination of graph representation and efficient data handling techniques to manage and optimize real-time navigation in large-scale systems.**  **Report:**[**https://docs.google.com/document/d/1ttmT\_j\_JF1OjHDYN1fz8hQNwrmtmpFZ6/edit?usp=sharing&ouid=102559905727520417170&rtpof=true&sd=true**](https://docs.google.com/document/d/1ttmT_j_JF1OjHDYN1fz8hQNwrmtmpFZ6/edit?usp=sharing&ouid=102559905727520417170&rtpof=true&sd=true) **OUTPUT:** | |
| **GATE Questions Solution.** | |
| <https://drive.google.com/file/d/1ktTfjAYu1FjwcX4U216Pr1lrdoPqphQo/view?usp=sharing>  **Any other**  **(Write if you registered or practise apart from Hackerrank, Leetcode, Github.etc**  **Eg: Certification Programs related to DSA )  Hackerrank and CodeChef** | |

Kaluru Anirudh

Signature

****