**PROJECT OVERVIEW**

This project is designed to simulate a Canteen Management System, making it easier to organize and serve students based on their arrival order or wait time. This system is unique because students can input their unique IDs, names, and orders, which are then stored and managed using a Binary Search Tree (BST). In this system, each student is represented as a node in the tree. The Binary Search Tree helps keep everything organized by ensuring that students are stored based on their IDs. This makes it easy to find, add, or remove students as needed. The program is simple and user-interactive, allowing users to add new students, view the current queue, search for a specific student ID, and serve (remove) students once their orders are completed. It focuses on organizing the process, saving time, and ensuring everyone is served fairly. This project is not just a canteen application; it also showcases how a Binary Search Tree can solve real-life problems efficiently.

In order to use the code correctly the user should first enter his or her choice; there are choices above but the other choices will not work if the user will not input data first. So, the first thing that the user must do is to press number 1 and then then input the student id,student name and the student order. The order will now be listed to the canteen management of the Binary Search Tree.

The code provides several functionalities for managing a student order system using a binary search tree (BST). First, it allows adding a new student by entering their ID, name, and order details, inserting them into the BST while maintaining its sorted structure. It includes traversal options to view the students in different orders: **in-order traversal** lists students sorted by ID (indicative of their priority or wait time), **pre-order traversal** shows the order in which they were added (arrival order), and **post-order traversal** displays the order in which they would be served (served order). The system also supports searching for a student by ID, helping quickly retrieve their details if they exist in the tree. Additionally, there is functionality to serve a student by deleting them from the BST, with the program efficiently handling all deletion cases, including nodes with no children, one child, or two children (using the in-order successor for replacement). The program provides a menu-driven interface for user interaction and offers an exit option to terminate the application gracefully. These functionalities make the system efficient for managing and organizing student orders dynamically.