

Ghulam Ishaque Khan Institute of Engineering Sciences and Technology

Semester Project

Data Structures and Algorithms (ES221)

Report: **Student Helper Program**

|  |  |  |
| --- | --- | --- |
| Student Names | Reg # | Degree |
| **Ayaan Azam** | **2023596** | **ES** |
| **Raahim Ali** | **2023585** | **ES** |
| **Anas Ahmed** | **2023114** | **ES** |

**Introduction**

This report details the analysis of the C++ source code files provided in the Budget-Manager-Batch folder. The codebase implements a multi-functional student helper program featuring a Budget Manager, a Time Manager (including task scheduling and dependency management), and mentions a Plagiarism Detector (though its implementation is not fully provided in the analyzed files). The analysis focuses on identifying key programming concepts, data structures, and algorithms employed throughout the modules.

**1. Objective**

The goal of this project is to implement core data structures covered in the course and demonstrate their practical use in a functional application. The Student Helper Program achieves this by offering modules such as a Budget Manager, with future placeholders for Plagiarism Detection and Time Management. This report presents the functionality, structure, and efficiency of each implemented data structure.

**Note:** *For the deliverable 2, we have implemented only budget manager because it only includes all the concepts studied till now in the classroom.*

**2. Requirements Overview**

1. Basic Working Code – Budget Manager supports preset and custom budget input with priority rankings and history tracking.

2. Functional Demonstration – Demonstrates usage of Stack, Doubly Linked List, and Arrays via interactive console application.

3. Efficiency Analysis – Detailed time and space complexities for each structure are provided in the section (Efficiency Analysis).

4. Application of Class Concepts – All structures reflect OOP principles including encapsulation and dynamic memory usage.

**3. Core Data Structures Used**

1. Stack: Maintains history of accessed priority nodes using singly linked list.

2. Doubly Linked List: Links priority nodes in the Budget Manager for traversal and manipulation.

3. Arrays: Used for static storage of priority names and ranks.

**4. Algorithms Applied**

- Recursion: Used for recursive budget display.

- Linear Search: Used to locate a specific priority.

- Division Algorithms: Used to calculate division and subdivision budgets.

**5. Efficiency Analysis**

This section outlines the time and space complexities of key operations:

|  |  |  |  |
| --- | --- | --- | --- |
| Data Structure | Operation | Time Complexity | Space Complexity |
| Stack | push / pop | O(1) | O(n) |
| Stack | display | O(n) | O(n) |
| Doubly Linked List | insert at head | O(1) | O(n) |
| Doubly Linked List | search/remove | O(n) | O(n) |
| Arrays | access by index | O(1) | O(n) |
| Arrays | search | O(n) | O(n) |
| Budget Algorithms | division/subdivision | O(1) | O(1) |
| Recursion | DisplayAllBudget | O(n) | O(n) |

**6. Program Flow Summary**

- User selects Budget Manager.

- Inputs total budget and chooses preset or custom allocation.

- Enters priority name, division style, subdivisions, and rank.

- Data is stored in a doubly linked list, with access history tracked via stack.

- User can display all budgets, search by name, or view history.

- Menu-driven loop continues until user exits.

**7. Conclusion**

The Student Helper Program successfully demonstrates the implementation of key data structures and OOP concepts. The Budget Manager module is fully functional, with a modular structure and efficient use of memory and time. This project is extensible and lays the groundwork for future additions like the Time Manager and Plagiarism Detector.