Reflection for Data Structures Assignment 1

Challenge 1: The Array Artifact

Reflection:

In this challenge, I implemented an array-based storage system for artifacts. I learned how to maintain a sorted array for binary search operations and handle dynamic operations like add and remove.

Difficulty Encountered:

Handling array removal was slightly challenging, as shifting elements required careful consideration to avoid gaps.

Challenge 2: The Linked List Labyrinth

Reflection:

The linked list labyrinth was a good practice to understand singly linked list operations. Implementing loop detection using a set made it simpler to understand the logic.

Difficulty Encountered:

I initially faced issues in implementing the loop detection efficiently.

Challenge 3: The Stack of Ancient Texts

Reflection:

I used Java's built-in stack class to manage ancient scrolls. This helped reinforce my understanding of stack operations such as push and pop.

Difficulty Encountered:

The main challenge was understanding the practical use cases of stack operations in managing elements.

Challenge 4: The Queue of Explorers

Reflection:

Implementing a circular queue to manage explorers was an interesting challenge. I learned how to handle index wrapping and queue overflow scenarios.

Difficulty Encountered:

Index wrapping in the circular queue was a bit complex at first, but using modulo operations helped overcome this.

Challenge 5: The Binary Tree of Clues

Reflection:

This challenge involved creating a binary search tree. It was great practice for recursive methods and understanding tree traversal techniques.

Difficulty Encountered:

Ensuring that each node is placed correctly and handling edge cases for empty or single-node trees required careful consideration.

Ideas for Improvement

- Implement additional traversal methods (e.g., level-order traversal).
- Implement an AVL tree for self-balancing properties.
- Optimize array operations for better performance.