Reflection

Name: Tsering Khando Lama

Student Id: C0916140

GitHub: https://github.com/Tseringkhando/DataStructures-Assignment1-TseringLama

1. The Array Artifact

Reflection:

In the ArtifactVault challenge, I was able to practice my knowledge of different search algorithms, specifically linear and binary search. Implementing binary search reinforced the importance of maintaining a sorted array and understanding how this data structure can optimize search operations. By the end of the challenge, I felt more confident in my ability to manage arrays and implement sorting algorithms effectively.

Difficulties Encountered:

Initially, I struggled with sorting arrays, but then, with some research, I was able to use Lamba expression to sort arrays quickly.

Future Improvements:

In the future, I would like to explore dynamic array implementations, such as ArrayLists, to handle cases where the array needs to grow without losing existing artifacts.

2. The Linked List Labyrinth

Reflection:

Creating the Labyrinth Path class introduced me to linked lists and their advantages over arrays, particularly in dynamic memory allocation. I learned how to effectively traverse a linked list and implement a loop detection algorithm, which was particularly interesting and applicable in real-world scenarios.

Difficulties Encountered:

The challenge for me was to be able to understand and play around with a linked list in a short time, and I was able to achieve this with the help of further research. The other challenge was to remove the last location. It was solved by implementing a method that traverses the linked list to find the second-to-last node. Once located, I set its next pointer to null, removing the last node from the list. This involved careful consideration of edge cases, such as when the list only contains one node or is empty.

Future Improvements:

I want to add functionality for reversing the path or retrieving a specific location based on criteria, such as distance from the start, which would enhance the usability of the labyrinth path.

3. The Stack of Ancient Texts

Reflection:

The ScrollStack challenge solidified my understanding of stack data structures and the Last-In-First-Out (LIFO) principle. Implementing push and pop operations helped me appreciate how stacks can be used to manage function calls in programming languages and help with backtracking algorithms.

Difficulties Encountered:

I encountered difficulties with checking for the existence of a Scroll title in the stack, as it required iterating through the stack elements. I resolved this by clearly understanding how to traverse the array representation of the stack.

Future Improvements:

In the future, I would like to implement a maximum size limit for the stack and add error handling to ensure that operations such as pop and peek are only called when the stack is not empty.

4. The Queue of Explorers

Reflection:

Implementing the ExplorerQueue using a circular queue was a valuable experience that deepened my understanding of queue operations and their importance in managing real-time data. This challenge highlighted how circular queues can effectively utilize memory and provide efficient enqueue and dequeue operations.

Difficulties Encountered:

One difficulty I faced was managing the pointers for the front and rear of the queue. I overcame this by carefully tracking their positions during each enqueue and dequeue operation and thoroughly testing the queue with various scenarios.

Future Improvements:

I would like to add a feature to expand the size of the queue dynamically.

5. The Binary Tree of Clues

Reflection:

Creating the ClueTree class was enlightening as I learned about binary trees and the importance of traversals (in-order, pre-order, post-order). Recursive functions were exquisite for handling tree operations, making the code more concise and readable.

Difficulties Encountered:

Implementing the various tree traversal methods posed a challenge, especially when it came to ensuring I followed the correct order. I solved this by visualizing the tree structure and writing out the traversal logic step by step.

Future Improvements:

I want to extend the binary tree to implement functionality to delete specific clues from the tree.

Conclusion

Through this assignment, I have deepened my understanding of fundamental data structures and algorithms in Java. Each challenge pushed me to think critically about data management and problem-solving. I got the opportunity to revise my knowledge of data structures, hone my skills and learn more about them. I encountered several challenges along the way, which required me to research and seek help, enhancing my learning experience. I look forward to applying these concepts in future projects and further expanding my skills in data structures.