LEARNING JOURNAL: UNIT 3 - LISTS, STACKS, QUEUES, AND DICTIONARIES

Introduction

This unit has significantly deepened my understanding of fundamental data structures in computer science. I've explored linked lists, stacks, queues, and their various implementations. The most compelling learning came from understanding the stack data structure and its Last-In-First-Out (LIFO) principle, which forms the backbone of many computer systems. As Cormen et al. (2009) explain, stacks are particularly valuable for managing function calls, handling recursive algorithms, and implementing undo functionality in applications. The assembly line stack implementation assignment demonstrated how the LIFO principle works practically in a manufacturing context, where inspection statuses are processed in reverse order of their entry. I also gained insights into queues and their First-In-First-Out (FIFO) model, which contrasts with stacks and serves different computational purposes such as scheduling and buffering operations (Goodrich et al., 2014).

Difficulties Faced

The most challenging aspect was visualizing the dynamic nature of linked list implementations, especially in stacks. During the assembly line assignment, I initially struggled to understand how the nodes connect and how the "top" pointer moves during push and pop operations. The concept of maintaining references correctly during these operations required careful attention to avoid memory leaks or dangling pointers. Another difficulty was grasping the efficiency differences between array-based and linked list implementations of these data

structures. The big-O analysis in the assignment helped clarify this, showing that while both implementations can achieve O(1) time complexity for core operations, they differ in memory allocation strategies and flexibility.

Activities Performed

The discussion allowed me to compare stacks and queues, focusing on their operational principles and applications. I explored how stacks follow the LIFO principle while queues implement FIFO behavior, and how these characteristics make them suitable for different problem domains.

The programming assignment involved implementing a stack using a linked list to simulate an assembly line inspection process. I created a LinkedStack class with nodes storing inspection statuses (0, 1, 2) and implemented push(), pop(), isEmpty(), and peek() operations. The simulation demonstrated how vehicles move through three inspection stations, with each station popping the relevant status from the stack. This implementation provided hands-on experience with dynamic memory allocation, as each node was created and linked appropriately during push operations and properly disconnected during pop operations.

Additionally, I completed a self-quiz that tested my understanding of various data structure operations and their time complexities. This helped reinforce concepts about array versus linked list implementations, particularly how "the linked list implementation is particularly well-suited for this application because the number of elements is small (3 inspections)" and operations are straightforward (Cormen et al., 2009).

Conclusion

Unit 3 has provided me with crucial knowledge about fundamental data structures that form the building blocks of more complex algorithms and systems. Understanding these structures' behaviors, implementations, and appropriate use cases will be invaluable in my future programming endeavors. The practical implementation of the stack for the assembly line simulation demonstrated how theoretical concepts translate into practical solutions. Moving forward, I aim to explore how these basic structures can be combined and extended to solve more complex computational problems, particularly in areas like graph algorithms and tree implementations that build upon these foundational concepts.

Wordcount: 529

References:

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). *Introduction to Algorithms* (3rd ed.). MIT Press. https://mitpress.mit.edu/9780262533058/introduction-to-algorithms/

Goodrich, M. T., Tamassia, R., & Goldwasser, M. H. (2014). *Data Structures and Algorithms in Java* (6th ed.). Wiley. https://www.wiley.com/en-us/Data+Structures+and+Algorithms+in+Java%2C+6th+Edition-p-9781118771334