1. The Array Artifact

What I Learned: In this task, I learned how to manage an array's size and capacity effectively, ensuring it doesn't exceed its limits by tracking the number of artifacts with a size variable. I also learned to maintain the array's order using Arrays.sort() after each insertion or removal. Removing artifacts required finding the artifact, replacing it with the last one, setting the last slot to null, decrementing the size, and sorting the array again. Implementing both linear and binary search methods taught me the importance of keeping the array sorted for efficient searching. These challenges helped me understand the importance of efficient data management and sorting in arrays.

2. The Linked List Labyrinth

What I Learned: In this task, I learned how to manage a singly linked list effectively, including adding new locations, removing the last visited location, checking for loops, and printing the entire path. The challenge was to ensure that each operation worked correctly and efficiently. I approached this by implementing methods to traverse the list and handle edge cases, such as when the list is empty. Detecting loops using the Floyd's Cycle-Finding Algorithm was another challenge, which I addressed by using two pointers (slow and fast). Creating a user-friendly, menu-driven interface was also important, and I achieved this using a Scanner object for input and a while loop with a switch statement for menu navigation. These challenges helped me understand the importance of efficient data management and user interaction in linked list operations.

3. The Stack of Ancient Texts

What I Learned: In this task, I learned how to manage stack operations effectively, including pushing, popping, peeking, and checking for the existence of elements. The challenge was to handle edge cases, such as when the stack is empty, which I addressed by adding checks and appropriate messages. Creating a user-friendly, menu-driven interface was another challenge, which I tackled using a Scanner object for input and a while loop with a switch statement for menu navigation. Handling invalid input gracefully was achieved by adding a default case in the switch statement. These challenges helped me understand the importance of robust error handling and user interaction in stack management.

4. The Queue of Explorers

What I Learned: In this task, I learned how to manage a circular queue effectively, including enqueuing, dequeuing, peeking, and checking if the queue is full or empty. The challenge was to handle the circular nature of the queue and ensure that operations work correctly even when the queue wraps around. I approached this by using modular arithmetic to update the front and rear indices. Handling edge cases, such as when the queue is full or empty, was another challenge, which I addressed by adding checks and appropriate messages. Creating a user-friendly, menu-driven interface was also important, and I achieved this using a Scanner object for input and a while loop with a switch statement for menu navigation. These challenges helped me understand the importance of efficient data management and user interaction in queue operations.

5. The Binary Tree of Clues

What I Learned: In this task, I learned how to manage a binary tree effectively, including inserting new clues, performing in-order, pre-order, and post-order traversals, finding specific clues, and counting the total number of clues. The challenge was to ensure that each operation worked correctly and efficiently. I approached this by implementing recursive methods for insertion and traversal. Handling edge cases, such as when the tree is empty, was another challenge, which I addressed by adding checks and appropriate messages. Creating a user-friendly, menu-driven interface was also important, and I achieved this using a Scanner object for input and a while loop with a switch statement for menu navigation. These challenges helped me understand the importance of efficient data management and user interaction in binary tree operations.