CS 3303-01 Data Structures

Instructor: Professor Vandelun Amokaha

Name: Anonymos

Stack Assignment

**Introduction**

In this assignment, we implement a stack data structure using the Java programming language to track the progress of three inspection stations in an automobile manufacturing line. A stack is a Last In First Out (LIFO) data structure that is very useful when a program needs to manage data. In this case, the stack is used to pop entries as the vehicle passes through each of the three inspection stations, recording the progress of the inspections.

**Algorithm Design and Implementation**

In this algorithm, the stack is used to record the inspection status of a vehicle at each inspection station. Three entries (2, 1, 0) are pushed onto the stack in sequence, and a pop operation is performed at each station. The popped entries represent the ID of the inspection station.

**Implementation Details**

* push: The push operation adds elements to the stack. Before each inspection is performed, the vehicle’s progress is pushed onto the stack. The `push()` method is called in the program based on the state of the stack and automatically adds appropriate values (2, 1, 0). If the stack is full, an error message is displayed.
* pop: The pop operation removes elements from the stack. Once each inspection station completes the inspection, the station ID is popped from the stack and displayed in the console. If the stack is empty, an error message is displayed.

**Stack Implementation Details**

In this example, the stack is implemented using an array. While Java provides a built-in `Stack` class, we create the `ArrayStack` class to better understand how the internal stack operations work with an array. This class includes methods to push, pop, and check if the stack is empty.

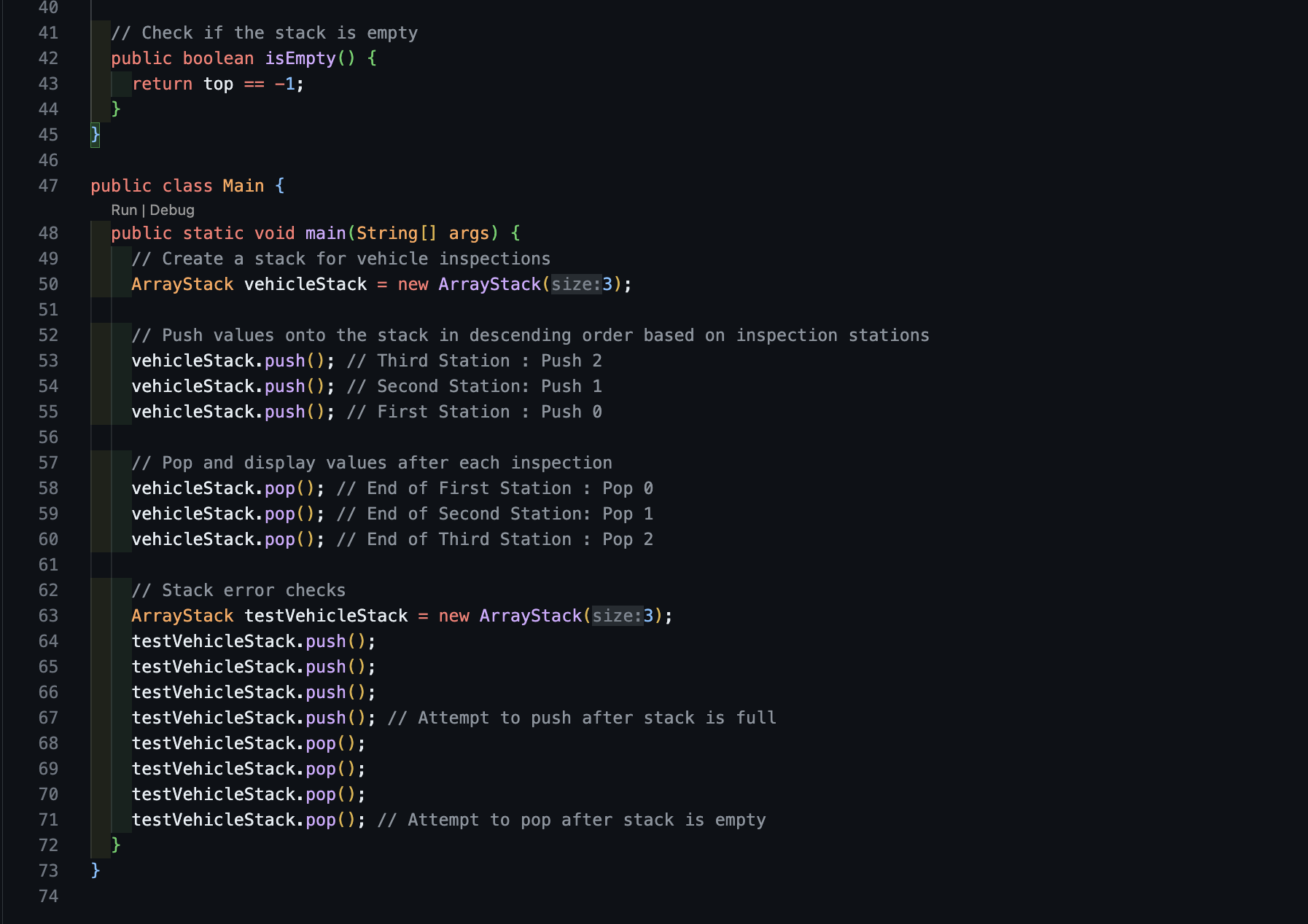
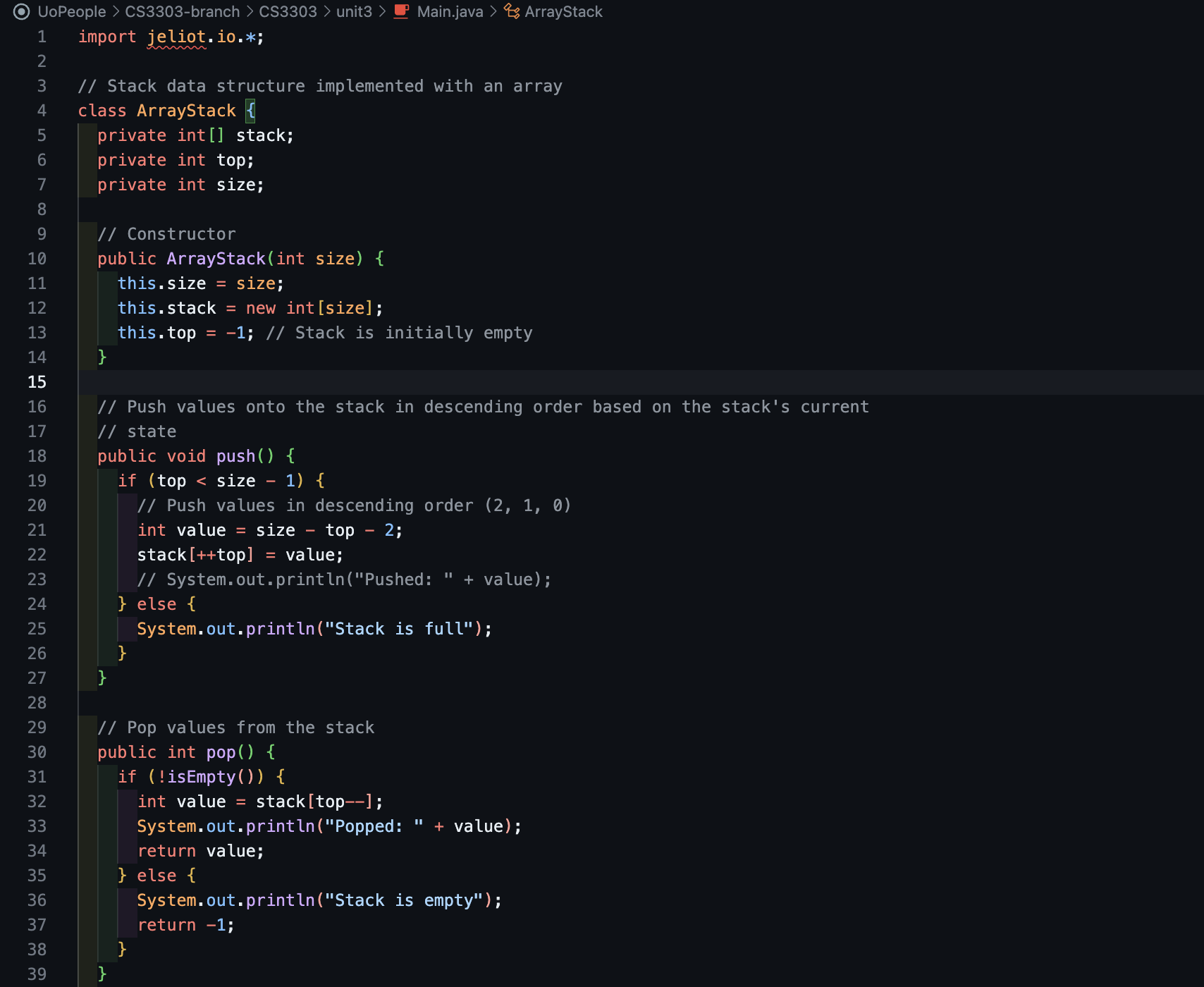


Figure 1 Stack algorithm: Main.java

**Algorithm Efficiency and Asymptotic Analysis**

The push and pop operations in this algorithm both access array elements via direct indexing. Since arrays are stored contiguously in memory, accessing specific indices always takes a constant amount of time regardless of the number of elements. Therefore, the time complexity for both push and pop operations is O(1).

**Conclusion**

In this assignment, we implemented a stack data structure using Java to track vehicle inspection progress on a manufacturing line. The algorithm provides correct output, with clear and efficient steps. The time complexity of the push and pop operations is O(1), and the LIFO properties of the stack effectively address the problem. Additionally, the results were verified, and the program works as expected.

Word Count: 402

References

1. Silberschatz, A., Korth, H.F., & Sudarshan, S. (2001). *Database System Concepts (4th ed.).* New York, NY: McGraw-Hill. Available at Database System Concepts 4th Edition By Silberschatz-Korth-Sudarshan.pdf