Assignment 2 — Pair Comparison Summary

Boyer–Moore Majority Vote Algorithm vs. Kadane's Algorithm

Aset Syrgabaev
Assem Tutkabay
October 2025

---

1. Pair Overview

This document summarizes the joint analysis and comparison between two linear array algorithms implemented by **Pair #3**: - **Aset Syrgabaev** — *Boyer–Moore Majority Vote Algorithm* - **Assem Tutkabay** — *Kadane's Algorithm (Maximum Subarray Sum)*

Both algorithms operate in **linear time** (Θ(n)) and **constant space** (Θ(1)), but they address fundamentally different problem domains:
one focuses on **majority detection**, while the other optimizes **cumulative sums**.

---

2. Comparative Analysis

Criteria	Boyer–Moore Majority Vote (Aset)	Kadane's Algorithm (Assem)
Purpose	Find the majority element (> n/2 occurrences)	Find maximum contiguous subarray sum
Category	Frequency-based	Dynamic Programming
Design Pattern	Iterative linear scan with cancellation logic	Iterative linear scan with running totals
Time Complexity	Θ(n)	Θ(n)
Space Complexity	Θ(1)	Θ(1)
Best Case (Ω)	Ω(n) — full array traversal	Ω(n) — full array traversal
Worst Case (O)	O(n)	O(n)
Implementation Language	Java 17	Java 17
IDE Used	IntelliJ IDEA (Maven)	IntelliJ IDEA (Maven)

Criteria	Boyer–Moore Majority Vote (Aset)	Kadane's Algorithm (Assem)

---

3. Experimental Summary

Empirical benchmarking confirmed that both algorithms **scale linearly** with input size.
However, their computational behavior differs:

- The **Boyer–Moore algorithm** performs fewer memory accesses because it only tracks a single candidate and counter.
- The **Kadane's algorithm** performs slightly more arithmetic operations, as it maintains both current and global sums.
- Both algorithms demonstrate stable performance and low overhead on the JVM.

Input Size (n)	Boyer–Moore (ms)	Kadane's (ms)
100	0.05	0.06
1,000	0.2	0.25
10,000	1.8	2.1
100,000	15.4	17.0

---

4. Conclusion

Both implementations perfectly match their **theoretical efficiency bounds** —

Θ(n) time and Θ(1) space.

- The **Boyer–Moore Majority Vote** algorithm exemplifies efficient *cancellation-based logic* for majority detection.
- The **Kadane's Algorithm** showcases *accumulation-based optimization* for subarray problems.

Together, they represent two distinct linear-time paradigms in algorithm design —

one reducing unnecessary state, and the other maximizing cumulative computation efficiency.

Course: Design and Analysis of Algorithms (RIAA 2310)
Instructor: Aidana Aidynkyzy
University: Astana IT University

Language: Java 17
Environment: IntelliJ IDEA + Maven