# Report B: Analysis of Kadane's Algorithm

Author: Student A (peer reviewer)  
Partner: Student B (implementation author)

## 1. Algorithm Overview

Kadane's algorithm finds the maximum subarray sum in O(n) time by maintaining local (max\_ending\_here) and global (max\_so\_far) maxima. Java implementations in the project include a variant with position tracking (`KadanesAlgorithm.java` at root) and a namespaced implementation `com.daa.assignment2.algorithms.KadanesAlgorithm` returning only the sum.

## 2. Complexity Analysis

Time Complexity:  
- Best/Average/Worst: Θ(n) — single pass over array.  
  
Space Complexity:  
- O(1) auxiliary space (a few integer variables).  
  
Positions tracking adds constant overhead to record start/end indices.

## 3. Code Review & Optimization Suggestions

Findings:  
- The root-level implementation returns MaxSubarrayResult with start/end indices; good for diagnostics.  
- The packaged implementation is minimal and returns sum only.  
  
Suggestions:  
- Ensure consistent API across package: either always return a result object or provide utility wrappers.  
- Add handling for all-negative arrays explicitly if desired (current code handles it correctly).  
- Add unit tests verifying indices correctness on various inputs (tie cases, multiple equal maxima).

## 4. Empirical Results

Benchmark setup: replicated algorithm in Python. Each size run 3 trials; reported values are averages.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| n | time\_ms\_avg | time\_ms\_std | comparisons\_avg | array\_accesses\_avg |
| 100 | 0.0168 | 0.0031 | 199.0 | 200.0 |
| 1000 | 0.1682 | 0.0020 | 1999.0 | 2000.0 |
| 10000 | 1.7355 | 0.0215 | 19999.0 | 20000.0 |
| 100000 | 17.4051 | 0.2639 | 199999.0 | 200000.0 |

Additional run with all-negative inputs confirmed correct behavior (returns largest negative).

## 5. Conclusion

Kadane's algorithm implementation is correct and efficient. Position-tracking variant in repo is valuable. No major algorithmic optimizations are possible beyond micro-optimizations; focus on testing and consistent API.