Design and Analysis of Algorithms

Assignment 2

Yerassyl Jetibay & Adilet Kabiyev

Joint comparison summary of Min Heap and Max Heap algorithms

Introduction

Both reports describe heap data structures implemented in Java — for Min Heap and for Max Heap . Both use an array to store elements and follow the heap property to keep their elements in order.

Main Idea

The Min Heap keeps the smallest element at the top (root), while the Max Heap keeps the largest element at the top. In both implementations, the children of an element are found at positions 2i+1 and 2i+2 in the array.

Operations

- Insert: O(log n)
- Extract (Min/Max): O(log n)
- Build Heap: Min Heap uses O(n) (Floyd's algorithm), Max Heap uses O(n log n) (inserting one by one)
- Space: O(n)

Both algorithms have the same speed for inserting and extracting elements, but the Min Heap builds faster because it uses the more efficient Floyd's build algorithm.

Code Design

The Min Heap uses a dynamic array that resizes automatically when needed. It includes a performance tracker, test files, and has a clear structure.
The Max Heap uses a fixed-size array and tracks detailed metrics such as comparisons and swaps. However, it has a small bug in the increaseKey() method and could be improved by adding automatic resizing and using Floyd's build method.

Results

Both heaps work correctly and follow the expected heap rules. The Min Heap is slightly more efficient when working with large data sets because of its optimized build process. The Max Heap is also correct and well-organized but can be improved for better performance.

Conclusion

Both implementations are correct and have the expected time complexities. However, Min Heap is more optimized and practical for large input sizes. Max Heap is also well-designed but would benefit from a few improvements, such as dynamic resizing and a faster build method. Overall, the Min Heap performs better in efficiency, while both show solid understanding of heap algorithms.