Analysis Report — HeapSort

Student: Ерасыл Ғинаят
Assignment 2 — Design and Analysis of Algorithms
Algorithm: HeapSort (In-place, O(n log n))

1. Experiment Setup

- Input sizes: 100, 1000, 5000, 10000
- Collected metrics: time (ns), comparisons, swaps, allocations
- System: Standard PC (Intel CPU, Windows)
- Average of 5 runs for each test

2. Results Summary

n	Time (ns)	Comparisons	Swaps	Allocations
100	438700	1024	580	0
1000	358100	16892	9097	0
5000	1237800	107715	57145	0
10000	1974500	235271	124129	0

3. Performance Analysis

- Time grows roughly as **O(n log n)**, confirming theoretical complexity.
- **Comparisons** and **swaps** also increase in a logarithmic pattern.
- HeapSort is efficient and memory-friendly (no extra allocations).

4. Plots

- `heapsort_time_vs_n.png` — runtime growth
- `heapsort_comparisons_vs_n.png` — comparison scaling
- `heapsort_swaps_vs_n.png` — swap scaling

5. Conclusion

HeapSort shows expected performance with stable efficiency for large inputs. Its in-place design makes it well-suited for memory-limited environments.