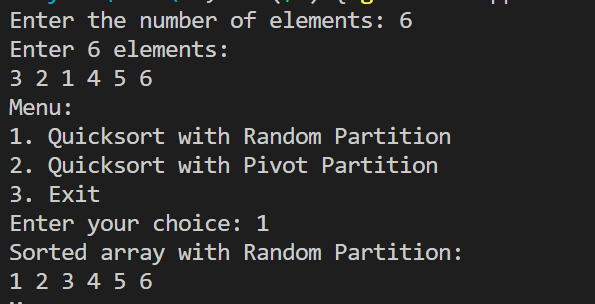
# Lab 4

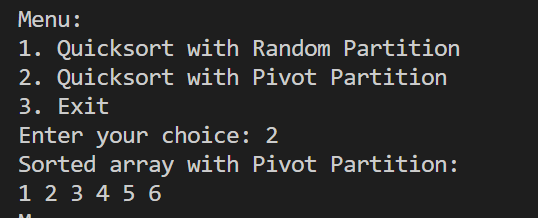
Assignment Based on analysis of quick sort consider deterministic and randomised approach also mention time and space complexity of both algorithms

# Source Code

# <https://github.com/OmkarMundlik/Complexity-Analysis/blob/master/Lab4/Lab4.cpp>

# Results





# Analysis:

Deterministic Quicksort and Randomized Quicksort share similar time and space complexity

characteristics on average and in the best case. In the worst case, both algorithms have a

time complexity of O(n^2), but Randomized Quicksort significantly mitigates the probability

of encountering such scenarios compared to its deterministic counterpart.

For Deterministic Quicksort:

- Worst-case time complexity: O(n^2)

- Average-case time complexity: O(n log n)

- Best-case time complexity: O (n log n), with optimal performance when partitions are

balanced.

- Space complexity: O (log n) for the recursive call stack in average and best cases.

For Randomized Quicksort:

- Worst-case time complexity: O(n^2) remains possible, but the likelihood is notably lower

due to randomized pivot selection.

- Average-case time complexity: O (n log n). The randomization ensures better average-case

performance by reducing the chances of consistently unbalanced partitions.

- Best-case time complexity: O (n log n), similar to the average-case, as balanced partitions

contribute to efficient sorting.

- Space complexity: O (log n) for the recursive call stack, comparable to the deterministic

version.

In summary, while both algorithms share comparable complexities, Randomized Quicksort

provides a probabilistic guarantee of superior average-case performance by minimizing the

occurrence of worst-case scenarios through random pivot selection