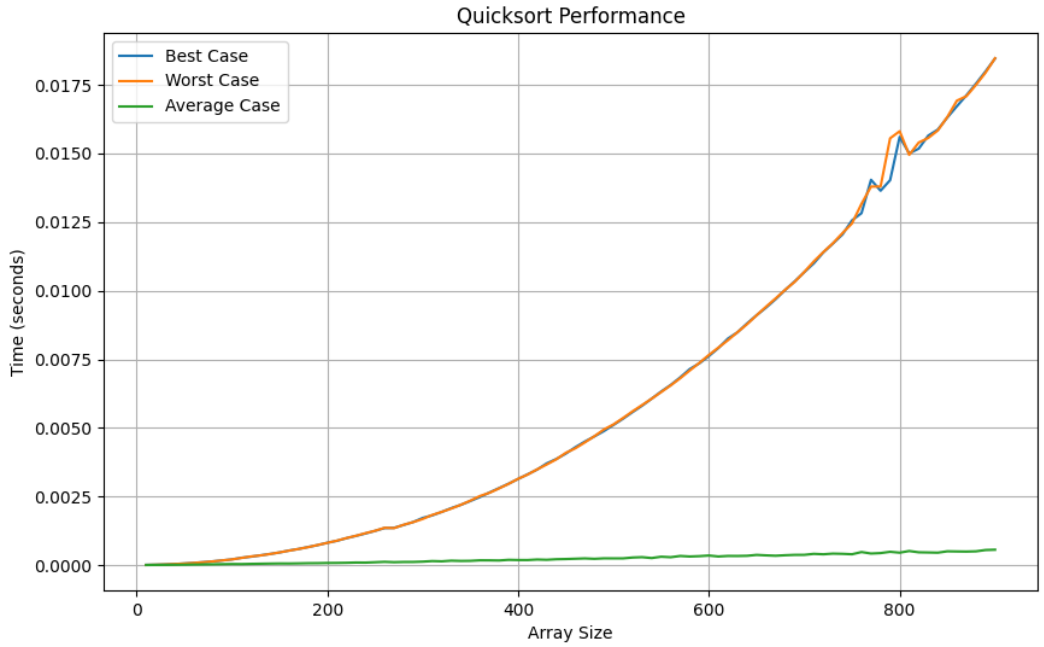
Q.2 For the non-random pivot version of quicksort, show the following benchmarks on the same graph:

2a) best case (generate a set of inputs that will always be the best case, repeat for multiple array input sizes "n").

2b) worst case (generate a set of inputs that will always be the worst case, repeat for multiple array input sizes "n").

2c) average case (generate a set of inputs from a uniform distribution, repeat for multiple array input sizes "n").



Q. 3 - Mathematically derive the average runtime complexity of the non-random pivot version of quicksort.

Initial Observations

1. Partitioning: Each step of Quicksort partitions the array into two segments based on the pivot - elements less than the pivot and elements greater than the pivot. The efficiency of Quicksort hinges on the sizes of these partitions. In the average case, we assume that partitions are reasonably balanced over many recursive calls, even though individual calls might not be perfectly balanced.

2. Recurrence Relation: The time complexity of Quicksort can be described by a recurrence relation. Suppose T(n) is the time to sort an array of n elements. In that

case, the partition operation (which scans the entire array) contributes a linear term, O(n), and the two recursive calls on the partitions contribute T(k) + T(n-k-1), where k is the number of elements in one partition and n-k-1 in the other.

Average Case Analysis

For the average case analysis, the pivot splits the array into two parts, and we assume all splits are equally likely. The recurrence relation for the average case is:

$$T(n) = 1\ n \sum (T(i) + T\ (n - i - 1) + O\ (n)$$

The O(n) term accounts for the partitioning work, and the summation averages the cost of sorting two partitions over all possible pivot positions (k).

The recurrence relation describing the time complexity of Quicksort, particularly with a non-random pivot selection, cannot be directly solved using the Master Theorem because the theorem requires the subproblems to be of equal size. In Quicksort, the sizes of subproblems can vary because they depend on the pivot's position after each partition. While the Master Theorem offers a straightforward way to determine the complexity of algorithms with subproblems of uniform size, it's less applicable to Quicksort's varied partition sizes.

Quicksort's average-case time complexity is considered O(n log n). However, this estimation is based on certain assumptions about how evenly the pivot divides the array. The precise derivation of this complexity, especially for versions of Quicksort that do not randomize pivot selection, may necessitate a probabilistic approach to account for the variability in pivot placement. Such an analysis often involves a deeper, more detailed examination to accurately assess the algorithm's performance across all possible inputs.