Analysis Assignment 1 Report

Question 1:

Part a: Code in repository

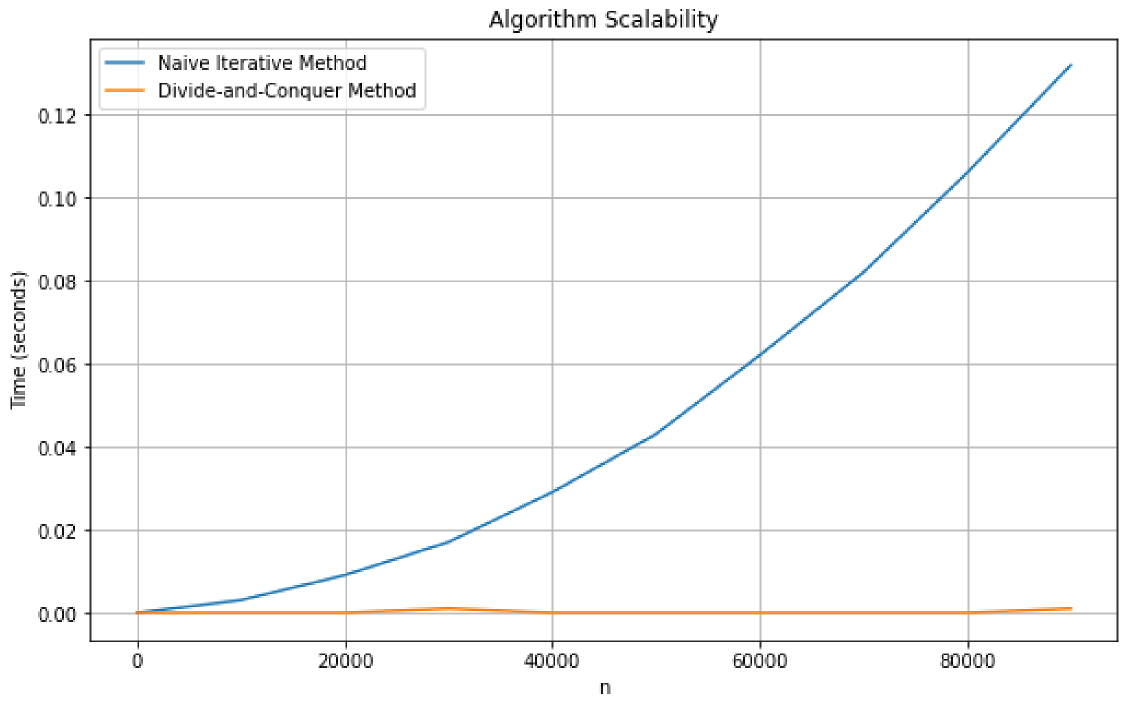
Part b:

The naive iterative method's time complexity is O(n) because it involves n number of multiplications

The divide and conquer method's time complexity is expressed using a recurrence relation T(n) = T(n/2) + O(1). This falls in the master theory under the category of T(n) = aT(n/b) + f(n), in which a = 1, b = 2, and f(n) = O(1). Since this is case 2 of the master theorem, the time complexity is O(log n).

Part c:

To determine the scalability of each algorithm experimentally, we can measure the running times for different values of n and plot the results. After running the code that is in the repository, following is the outputted graph:



Part d:

We can compare the running times with the expected time complexities from part b to determine whether experimental results confirm the theoretical analysis or not. In this question, it is worth noting that the iterative method has linear complexity while the divide and conquer method has logarithmic complexity, and the graphs above align correctly with these findings and confirm them.

Question 2:

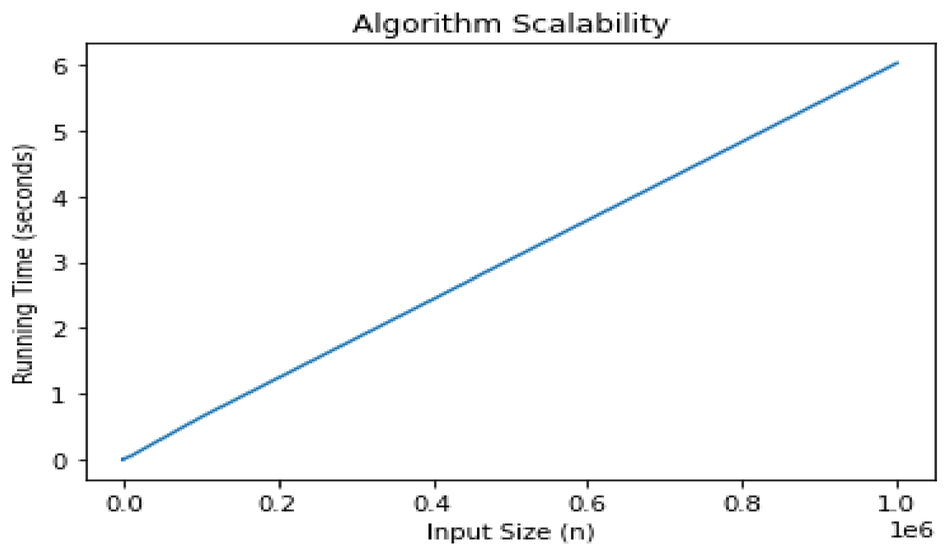
Part a: Code in repository

Part b:

The merge sort's time complexity is O(n log n), and binary search has a time complexity of O(log n). Since in the function I created in the code I use merge sort for sorting the array and I perform a binary search for each element, then the overall time complexity is O(n log^2 n)

Part c:

To determine the scalability of the algorithm experimentally, we can measure the running times for different values of n and plot the results. After running the code that is in the repository, following is the outputted graph:


Algorithm Scalability
Running Time (seconds)
1e6
Input Size (n)