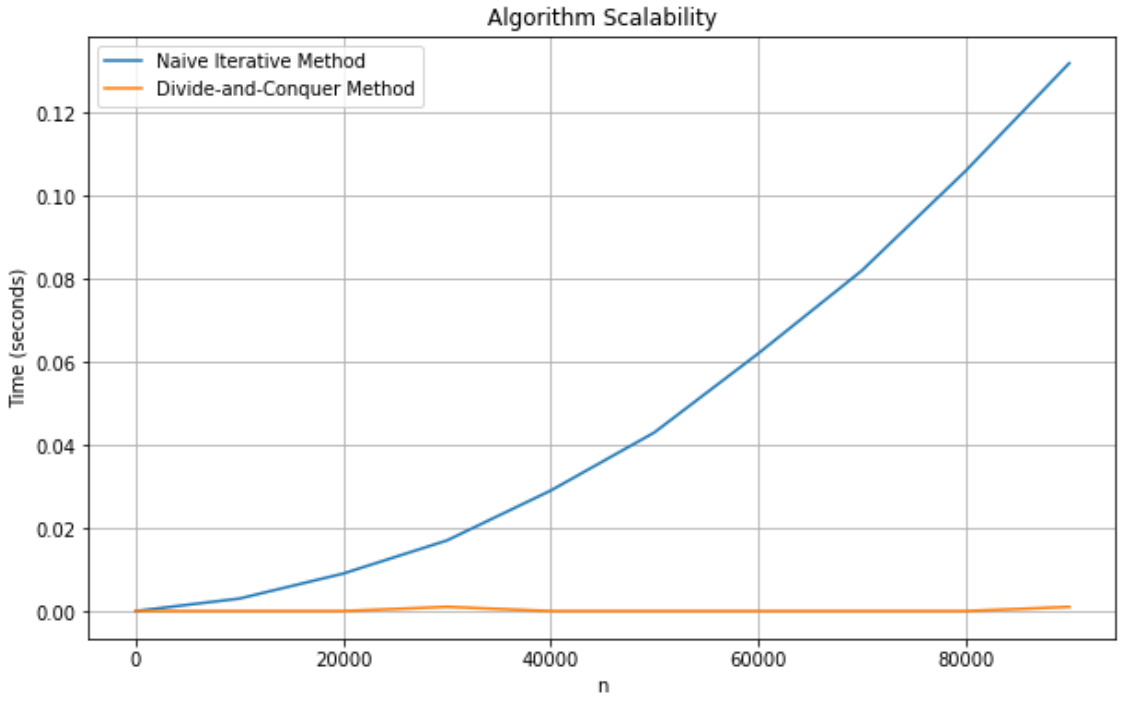
Question 1:

a. Code sent.

b. The time complexity of the straightforward iterative approach is O(n) since it performs n multiplications.

   For the divide-and-conquer method, its time complexity can be described through a recurrence relation: T(n) = T(n/2) + O(1). This recurrence corresponds to case 2 in the master theorem, where the parameters are f(n) = O(1), a = 1, and b = 2. Consequently, the time complexity is O(log n).

c. To assess the scalability of each algorithm through practical experimentation, we can measure the execution times across various n values and generate a graphical representation. Upon executing the code (available in the GitHub repo), the resulting graph is as follows:



d. We can compare the actual running times with the anticipated time complexities outlined in part b. In this context, as shown in the graph above, the iterative approach exhibits linear complexity, whereas the divide-and-conquer method demonstrates logarithmic complexity.

Question 2:

a. Code sent.

b. The time complexity of the Merge Sort algorithm is O(n log n), and Binary Search operates with a time complexity of O(log n). Since the code uses both sort algorithms, the resultant overall time complexity amounts to O(n log^2 n).

c. To assess the scalability of the algorithm through practical experimentation, we can measure the execution times across various n values and generate a graphical representation. Upon executing the code (available in the GitHub repo), the resulting graph is as follows: