Question 1: Show O() by mathematical analysis - Show all work/algebraic summative derivations.

Given: Binary and Linear Search Algorithms (See GitHub repository).

Linear Search Approach: With typical BigO() analysis, we look to find the worst case computation scenario for the algorithm. From the given algorithm within the repository, the most important line to consider is "if (val == a[indx]) return index;" inside the for loop. The worst case scenario for this line of code is either the target num is the last element of the array, or simply not present. Regardless, it iterates n times.

Let's let T equal the summation of the iterations within the function, so T(n) = $\sum_{i=0}^{n-1}(i+1)/(1/n) + (1-p) * n$, where p is just the worst case odds of the algorithm.

This summation simplifies down to $n(1-p/2)+p/2$, and for the worse case scenario (element not found), we're left with $1+(n+1)+n+n+1$, or $3n+3$. This answer comes from $T(n) = 1 + \sum_{i=0}^{n} 1 + \sum_{i=0}^{n-1} 1 + \sum_{i=0}^{n-1} 1 + 1 = 1 + (n+1) + n + n + 1 = 3n + 3$

Where summation i = 0 to n accounts for condition checks for the loop, summation i = 0 to n - 1 accounts for n comparisons, and summation i = 0 to n - 1 accounts for n increments.

Binary Search Approach: Binary Search differs only in formula derived to terminate the algorithm, which is taken by solving for n in $\frac{n}{2^k}$, which yields $\log_2(n)$. Plug this into the summation derivation, and you'll use $T(n) = \sum_{i=1}^{n} 4 * \log_2(n) + 2$ to solve for the worst case BigO() complexity.

Linear Search Conclusion: The summation derivations confirms that O(n) is the time complexity of the worst case BigO() timing. Where the best case is O(1), and the average

is typically also O(n).

Binary Search Conclusion: The summation derivations confirm that O(log n) is the worst case time complexity O() timing. Where the best case is when the element is found before the recursive call at O(1). The typical and case to prepare for is O(log n), however, when considering the different summations derived.