1. Consider the **searching problem**:

Input: A sequence of $n$ numbers $A = \langle a_1, a_2, \ldots, a_n \rangle$ and a value $\nu$.

Output: An index $i$ such that $\nu = A[i]$ or the special value `nil` if $\nu$ does not appear in $A$.

Write pseudocode for **linear search**, which scans through the sequence, looking for $\nu$. Using a loop invariant, prove that your algorithm is correct. Make sure that your loop invariant fulfills the three necessary properties.

2. How many elements of the input sequence linear search needs to check on the average, assuming that the element being searched for is equally likely to be any element in the array? How about in the worst case? What are the average-case and worst-case running times of linear search in $\Theta$-notation? Justify your answers.

3. Observe that if in the searching problem the sequence $A$ is sorted, we can check the midpoint of the sequence against $\nu$ and eliminate half of the sequence from further consideration. The binary search algorithm repeats this procedure, halving the size of the remaining portion of the sequence each time. Write pseudocode, either iterative or recursive, for binary search. Argue that the worst-case running time of binary search is $\Theta(\lg n)$.

4. Consider sorting $n$ numbers stored in array $A$ by first finding the smallest element of $A$ and exchanging it with the element in $A[1]$. Then find the second smallest element of $A$, and exchange it with $A[2]$. Continue in this manner for the first $n-1$ elements of $A$. Write pseudocode for this algorithm, which is known as **selection sort**. What loop invariant does this algorithm maintain? Why does it need to run for only the first $n-1$ elements? Give the best-case and worst-case running times of selection sort in $\Theta$-notation.

5. Rewrite the **MERGE** procedure so that it does not use sentinels, instead stopping once either array $L$ or $R$ has had all its elements copied back to $A$ and then copying the remainder of the other array back into $A$.

6. Use mathematical induction to show that when $n$ is an exact power of 2, the solution of the recurrence

$$T(n) = \begin{cases} 
2 & \text{if } n = 2, \\
T(n/2) + n & \text{if } n = 2^k, \text{ for } k > 1.
\end{cases}$$

is $T(n) = \Theta(n \lg n)$. 