1. Sorting and Searching

Give the number of comparisons that will be performed by each sorting algorithm if the input array of length $n$ happens to be of the form $[1, 2, ..., n - 3, n - 2, n, n - 1]$ (i.e., sorted except for the last two elements).

Note: On the real exam, you would be given pseudocode for the algorithms, though it is a very good idea to be comfortable with how the algorithms work to save time on the exam. For now, you can refer to the textbook and/or lecture slides for pseudocode.

(a) MinSort (SelectionSort)
(b) BubbleSort
(c) InsertionSort

2. Asymptotic Notation

For each part, answer True or False, and give a short explanation for your answer.

(a) $\sqrt{n^3} \in O(n^2)$.
(b) $8\log(n^2) \in \Theta(n^6)$.
(c) $\log(n) \in \Omega(\log(\log(n)))$.
(d) If $f$, $g$, and $h$ are functions from the natural numbers to the non-negative real numbers with $f(n) \geq g(n) \forall n \geq 1$, $f(n) \in \Theta(h(n))$, and $g(n) \in \Theta(h(n))$, then $(f - g)(n) \in \Theta(h(n))$.
(e) If $f$, $g$, and $h$ are functions from the natural numbers to the non-negative real numbers with $f(n) \in \Theta(h(n))$ and $g(n) \in \Theta(h(n))$, then $(f + g)(n) \in \Theta((h(n))^2)$.

3. Best and Worst Case

Suppose we are adding two $n$-digit integers, using the usual algorithm learned in grade school. The primary operation here is the number of single-digit additions that must be performed. For example, to add 48 plus 34, we would do three single-digit additions:

1. In the ones place, add $8 + 4 = 12$.
2. In the tens place, add $4 + 3 = 7$.
3. In the tens place, add $7 + 1 = 8$.

(a) If $n = 5$, give an example of two $n$-digit numbers that would be a best-case input to the addition algorithm, in the sense that they would cause the fewest single-digit additions possible.
(b) In the best case, how many single-digit additions does this algorithm make when adding two $n$-digit numbers?
(c) In the best case, when adding two $n$-digit numbers, describe the number of single-digit additions in $\Theta$ notation.
(d) If $n = 5$, give an example of two $n$-digit numbers that would be a worst-case input to the addition algorithm, in the sense that they would cause the most single-digit additions possible.
(e) In the worst case, how many single-digit additions does this algorithm make when adding two $n$-digit numbers?
(f) In the worst case, when adding two $n$-digit numbers, describe the number of single-digit additions in $\Theta$ notation.

4. Iterative Algorithms and Loop Invariants

In the following problem, we are given a list $A = a_1, \ldots, a_n$ of salaries of employees at our firm and two integers $L$ and $H$ with $0 \leq L \leq H$. We wish to compute the average salary of employees who earn between $L$ and $H$ (inclusive), and the number of such employees. If there are no employees in the range, we say that 0 is the average salary. This is an iterative algorithm which takes as input $A, L,$ and $H$ and returns an ordered pair $(avg, N)$ where $avg$ is the average salary of employees in the range, and $N$ is the number of employees in the range.
AverageInRange \( A : \text{list of } n \text{ integers}, L, H : \text{integers with } 0 \leq L \leq H \) 

1. \( \text{sum} := 0 \) 
2. \( N := 0 \) 
3. \text{for } i := 1 \text{ to } n 
4. \text{if } L \leq a_i \leq H \text{ then} 
5. \( \text{sum} := \text{sum} + a_i \) 
6. \( N + + \) 
7. \text{if } N = 0 \text{ then} 
8. \text{return } (0,0) 
9. \text{return } (\text{sum}/N, N) 

(a) State a loop invariant that can be used to show the algorithm AverageInRange is correct. 
(b) Prove your loop invariant from part (a). 
(c) Conclude from the loop invariant that the algorithm AverageInRange is correct. 
(d) Describe the running time of this algorithm in \( \Theta \) notation, assuming that comparisons and arithmetic operations take constant time. Justify your answer.

5. Recursive Algorithms 

In the following problem, we are given a list \( A = a_1, \ldots, a_n \) of salaries of employees at our firm and two integers \( L \) and \( H \) with \( 0 \leq L \leq H \). We wish to compute the average salary of employees who earn between \( L \) and \( H \) (inclusive), and the number of such employees. If there are no employees in the range, we say that 0 is the average salary. This is a recursive algorithm which takes as input \( A, L, \) and \( H \) and returns an ordered pair \((avg, N)\) where \( avg \) is the average salary of employees in the range, and \( N \) is the number of employees in the range.

RecAIR \( A : \text{list of } n \text{ integers}, L, H : \text{integers with } 0 \leq L \leq H \) 

1. \text{if } n = 0 \text{ then} 
2. \text{return } (0,0) 
3. \( B := a_1, a_2, \ldots, a_{n-1} \) 
4. \((avg, N) := \text{RecAIR}(B, L, H)\) 
5. \text{if } L \leq a_n \leq H \text{ then} 
6. \text{return } ((avg * N + a_n)/(N + 1), N + 1) 
7. \text{else} 
8. \text{return } (avg, N) 

(a) Prove by induction on \( n \) that for any input, the algorithm correctly returns the average salary and number of employees in the range. 
(b) Write down a recurrence for the time taken by this algorithm, assuming that comparisons and arithmetic operations take constant time. Assume also that removing an element from a list (line 3) takes constant time. 
(c) Use your answer from part (b) to determine the running time of this algorithm in \( \Theta \) notation. Justify your answer mathematically. 
(d) Write down a recurrence for the time taken by this algorithm, assuming that comparisons and arithmetic operations take constant time. Assume now that removing an element from a list (line 3) takes linear time. 
(e) Use your answer from part (d) to determine the running time of this algorithm in \( \Theta \) notation. Justify your answer mathematically.

6. Solving Recurrences 

Suppose a function \( f \) is defined by the following recursive formula, where \( n \) is a positive integer.

\[ f(n) = f(n-1) + 2n - 1, \quad f(1) = 6 \]

Use any method we learned in this class to get a closed-form formula for \( f(n) \).