
INSTRUCTIONS
Students should feel free to discuss these problems.
KEY CONCEPTS Sorting, Asymptotic notation, loop invariants.

---

1. For each of the following claims, say whether it is true or false, and give a short explanation (2 points correct answer, 2 points explanation).
   (a) $2^n \in O(n!)$
   (b) $2^n \in o(n!)$
   (c) $2^{2n} \in O(2^n)$
   (d) $(n^2+n+3)^3 \in \Theta(n^6)$.
   (e) $\sum_{i=1}^{\lceil \log_2 n \rceil} n/2^i \in \Theta(n)$.

2. As part of selection sort, we saw how to find the minimum element of an array $A[1..n]$ of distinct integers using $n-1$ comparisons. We could also find the maximum element of the array using the same number of comparisons.
   (a) (10 points) Give an algorithm that finds BOTH the maximum and minimum value in the array using exactly $3n/2-2$ comparisons for a list with n elements. (assume that n is even.).
   (b) State and prove a loop invariant for this algorithm, and use it to conclude your algorithm is correct.
   (c) Explain carefully the number of comparisons this algorithm uses (exact, not just order)
   (d) Design a sorting algorithm (related to SelectionSort) that uses the algorithm from part a as a subroutine.
   (e) Recall that SelectionSort does $n(n-1)/2$ comparisons for any input of size n. How many comparisons does your algorithm from part b do for an input of size n? (You can assume that n is an even number.)

3. For each algorithm, compute the exact number of times the algorithm prints in terms of n and compute the runtime in terms of Θ, where $n \geq 3$. (show your work.)
   (a)
   ```
   for i = 1 to n:
       for j = 1 to i − 1:
           print (i, j)
   ```
   (b)
   ```
   i = 0
   j = 0
   while i < n:
       print i.
       i = i + 2j + 1
       j = j + 1
   ```
   (c)
   ```
   for i = 1 to n − 2:
       for j = i + 1 to n − 1:
           for k = j + 1 to n:
               print (i, j, k)
   ```

4. State if each statement is true or false and give a justification either way. (you may use any of the methods used in class including the limit rules.)

(a) $n/2 + n/3 + n/4 + \ldots + n/n \in O(n)$

Hint: You can use the fact that the harmonic series is divergent in your proof.

(b) $2^{log_2(n)} = O(2^{log_4(n)})$

(c) $\left(\sqrt{n} + \sqrt[3]{n}\right)^2 \in O(n \log n)$

(d) $n! \in O(n^n)$

(e) $1 + 4 + 4^2 + 4^3 + \ldots + 4^n \in \Theta(4^n)$

(f) $1^2 + 2^2 + ..n^2 \in= O(n^2)$.

5. Design a "binary search style" algorithm for the following problem: The input is an array of integers $A[1..n]$ so that $A[1] > A[n]$. The problem is to find an integer I so that $1 \leq I \leq n-1$ and $A[I] > A[I+1]$, i.e., a particular place where the list is not sorted. Use loop invariants to show that your algorithm correctly solves this problem, and that it takes $O(logn)$ time.