Q1 (20 points): Asymptotic Notations

(a) (10 points) Try to prove $2n^2 - 100 = \Theta(n^2)$ using the basic definition of $\Theta$ notation. That is, to find positive constants $c_1, c_2, n_0$ such that $c_1 n^2 \leq 2n^2 - 100 \leq c_2 n^2, \forall n \geq n_0$.

(b) (10 points) Let $f(n)$ and $g(n)$ be asymptotically positive functions. Please answer “true” or “false” for the following statements.

1. $f(n) = \Omega(g(n))$ implies $g(n) = o(f(n))$
2. $f(n) + g(n) = \Omega(min(f(n), g(n)))$
3. $n^2 + \log(n!) + 10^6 n = \Theta(\log(n!))$
4. If $f(n) = O(g(n))$ and $g(n) = \Omega(h(n))$, then $f(n) = \Theta(h(n))$
5. $O(f(n)) + \Omega(f(n)) = \Theta(f(n))$
• Q2 (15 points): Recurrences

A ternary search algorithm is to search an element within a sorted array.

Divide: cut the sorted array into three (about) equal-sized sorted subarrays.

Conquer: search the element in only one of the three subarrays recursively.

Combine: no effort.

(a) (10 points) Please write a recursive pseudocode for this algorithm.

Ternary-Search(A, p, r, x) // A: the input array, p: starting index, 
    // r: ending index, x: element to search. 
    // Return true if found, otherwise, false.

{ 

(b) (5 points) Please write down the recurrence for an unsuccessful search using this ternary search algorithm.
• Q3 (10 points): Divide-and-Conquer

Given an array of \( n \) numbers, we would like to find the maximum. Please write down the divide, conquer and combine steps for solving this problem.

• Q4 (30 points): Solving Recurrences

(a) (10 points) Please use the master method to solve the recurrence \( T(n) = 2T(n - 2) + 1 \) by transforming it to an appropriate form. (hint: let \( T(n) = S(2^n) \) and let \( m = 2^n \))
(b)(10 points) Try to use a recursion tree to solve the same recurrence $T(n) = 2T(n-2)+1$.

(c)(10 points) For the recurrence $T(n) = T(n-1) + 1$, try to use the substitution method to verify the upper bound for $T(n)$ is $O(n)$. (suppose that $T(1) = 1$)
• Q5 (15 points): Sorting

For a given input array $A: <5, 8, 3, 2, 4, 1, 9, 7, 6> \$

(a) (5 points) For the Heap-Sort algorithm, what is the sequence of numbers in $A$ after calling Build-Max-Heap($A$) ?

(b) (5 points) For the Quick-Sort algorithm, what is the sequence of numbers in $A$ after the first Partition?
(c)(5 points) For the Quick-Sort algorithm, we know that if the partition on each recursive step always produces unbalanced two subarrays with the size ratio 99 to 1, then the running time for this quick sort is still $\Theta(n \log n)$. What is the running time for a quick sort if the partition on each recursive step always produces two subarrays with size 2 and $n - 2$? Justify your answer.

• Q6(10 points): Running Time

Please indicate the running time of each sorting algorithm in the table below (for best-case and worst-case, you need to answer the tightest upper and lower bounds respectively).

<table>
<thead>
<tr>
<th></th>
<th>Bubble-Sort</th>
<th>Insertion-sort</th>
<th>Merge-Sort</th>
<th>Heap-Sort</th>
<th>Quick-Sort</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best-case</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Average-case</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Worst-case</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>