Name _____

Quiz - Module 1: Sorting

1. [6 points] For each of the following sorting algorithms, state the runtimes (average-case and worst-case), and whether or not the algorithm is stable and in-place. **This time, you **should consider stack space** when judging if an algorithm is in-place/stable.*

Algorithm	**Average-Case**	**Worst-Case**	**Stable?**	**In-Place?**
Insertion Sort				
MergeSort				
QuickSort				

2. [4 points] Answer the following True/False questions regarding *sorting algorithms*

Let's suppose I want to sort a list of students by age (in years), but sort alphabetically among students with the same age. I can do this by running an unstable sort by age first, and then running a *stable sort* on the resulting list to sort alphabetically by name only.	**True**	**False**
Suppose *Lomuto's partition* puts items with values equal to the pivot on either side of the partition randomly. *Quicksort* would still work correctly.	**True**	**False**
The *merge* method that *Mergesort* uses typically declares an extra array within which the merging can be done.	**True**	**False**
Insertion sort runs in $\Theta(n)$ time when given an array filled completed with the same element (e.g., a list of all 5's)	**True**	**False**

3. [2 points] When implementing *quicksort*, the *partition* method usually returns an *integer*. Briefly explain what this value represents and why it is necessary to return it from partition.

In class, we saw a lower-bounds proof that general *comparison sorts* are always $\Omega(n \log n)$. Answer the following questions about the *decision tree proof* that we did.

4. [2 points] What did the *internal nodes* in the decision tree represent?

5. [2 points] What did *leaf nodes* of the decision tree represent?

6. [1 points] How many leaf nodes must be there be in this tree and why?

7. [1 points] Using the answer above, what is the *minimum height* of this tree and why (Write your answer in terms of n, the number of nodes in the tree)?

NOTHING BELOW THIS POINT WILL BE GRADED. USE THE SPACE BELOW FOR SCRATCH WORK

Name _____

Quiz - Module 2: Recurrence Relations

1. [5 points] For each of the following *recurrence relations*, circle the correct runtime according to the master theorem (or other method if you prefer).

$T(n) = 4T(\frac{n}{4}) + 7n - 1$	$\Theta(n)$	$\Theta(n \log n)$	$\Theta(n^2)$	Does Not Apply
$T(n) = 4T(\frac{n}{2}) + n^3$	$\Theta(n^2)$	$\Theta(n^3)$	$\Theta(n^2 \log n)$	Does Not Apply
$T(n) = 2T(\frac{n}{4}) + 10$	$\Theta(n^2)$	$\Theta(n^{0.5})$	$\Theta(n^{0.5} \log n)$	Does Not Apply
$T(n) = 3T(\frac{n}{4}) + n \log n$	$\Theta(\log n)$	$\Theta(n)$	$\Theta(n \log n)$	Does Not Apply
$T(n) = 9T(\frac{n}{3}) + n^2$	$\Theta(n)$	$\Theta(n^2)$	$\Theta(n^2 \log n)$	Does Not Apply

2. [2 points] Your co-worker creates a modified version of mergesort that recursively solves four subproblems that are size $n/4$. It requires $2n$ operations to split the list of size n into the four subproblems. After the recursive calls, the work to merge takes $3n$ operations.

   (A) Write the recurrence relation for this algorithm. (Assume $T(1) = 0$.)

   (B) Compare this new algorithm's time-complexity order class (i.e. its Θ) to mergesort's.
   Circle one: it's the **SAME** / it's **BETTER** / it's **WORSE**

3. [2 points] Answer the following True/False questions regarding *recurrence relations*

When using the substitution method to show $T(n) \in O(n)$, if our guess requires us to prove $T(n) \leq cn$, then our induction proof succeeds even if this relation is not true for small values of c.	**True**	**False**
When using the substitution method to show $T(n) \in O(n^2)$, if our guess requires us to prove $T(n) \leq cn^2$, then our induction proof succeeds if it shows $T(n) \leq cn^2 + 1$ because the two formulas grow at the same rate. (i.e. same Θ)	**True**	**False**

4. [3 points] Consider the recurrence $T(n) = 2T(\frac{n}{2}) + n \log n$. If using the *master theorem* to discover the complexity of this recurrence, Case 3 appears to apply. Describe in your own words whether or not case 3 can actually be applied. Explain your answer either way. *You may assume the regularity condition holds. (It does, trust us.)*

5. [4 points] In the space below on this page, reduce the following recurrence to its closed form (non-recursive) using the "unrolling" method: $T(n) = 3T(\frac{n}{3}) + n$ and $T(1) = 1$. Then, to make it easier for us to grade, copy the appropriate parts of your work to questions A, B and C, putting your answers right by each question A, B, or C so we can easily see your results.

A. Write down the general form of the recurrence in terms of d where d reflects how many times you've "unrolled."

B. Write down the value of d where the recurrence reaches the base case.

C. Write down the closed form, i.e. the formula for $T(n)$ without a recursive term.

MASTER THEOREM: FOR YOUR REFERENCE

For a recurrence of form $T(n) = aT(\frac{n}{b}) + f(n)$

- Case 1: if $f(n) = O(n^{\log_b a - \varepsilon})$ for some constant $\varepsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$
- Case 2: if $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \log n)$
- Case 3: if $f(n) = \Omega(n^{\log_b a + \varepsilon})$ for some constant $\varepsilon > 0$, and if $af(\frac{n}{b}) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$

Name _____

Quiz - Module 3: Advanced Div. and Con.

1. [3 points] For each of the following questions, circle the **True** or **False**.

Strassen's Algorithm is more efficient than the naive divide and conquer approach because it solves fewer subproblems.	**True**	**False**
Strassen's Algorithm has been proved to be an optimal algorithm for the matrix multiplication problem.	**True**	**False**
The brute-force solution to the *Closest Pair of Points problem* has time-complexity of $\Theta(n^3)$.	**True**	**False**

2. [3 points] For each of the following questions, circle the **True** or **False**.

The time-complexity of *Quickselect* depends on the efficiency of the *Partition* algorithm that it uses.	**True**	**False**
In the *median-of-medians* strategy we studied, the median of each sublist of size 5 is determined by calling *Quickselect* recursively on each of those "size 5" sublists.	**True**	**False**
When using the *median-of-medians* strategy to find a pivot for *Quickselect* on an input of size n, the maximum size of the subproblem to be solved recursively will be roughly $\frac{7n}{10}$.	**True**	**False**

3. [1 points] Your friend Kim has developed a new partition algorithm that works correctly and always does $2n$ comparisons. For some inputs it places the pivot value in the last position of the list, but 90 percent of the time the pivot value is placed in the middle 20% of the list. Which statement about *Quickselect* would be true if Kim's algorithm is used to do partitioning? (Choose one and circle the letter of your choice.)

A. Quickselect still works correctly and has worst-case time-complexity thats $\Theta(n)$.

B. Quickselect still works correctly and has worst-case time-complexity thats $\Theta(n^2)$.

C. Quickselect would no longer work correctly on some inputs.

4. [2 points] Suppose we implement the *Closest Pair of Points (CPP)* algorithm with the following change. In **each recursive call** to CPP, after getting the delta-values for the left and right subproblems for a given input, **we next sort the points** in the "strip" by y-value in order to find the smallest "crossing" delta.

A. Write the recurrence relation for this version of the CPP algorithm:

B. Do you think the overall time-complexity for this is $\Theta(n \log n)$ or $\omega(n \log n)$? Answer by circling the one you think is the best answer.

5. [6 points] The following questions are about the *combine step* of the *closest pair of points* algorithm studied in class.

A. True or false? When processing points in the strip, it's important to check the distance for two points in the strip that are on the same side of the midpoint. Circle one: **True** **False**

B. Your friend Jai produces a variation of our CPP algorithm that needs to check the next 15 points with a higher y-value for each point in the strip. (Our algorithm only needs to checks the next 7 such points at most.) She has proved her approach produces correct results, but is its time-complexity in the same Θ class as our algorithm? Circle one: it's the **SAME** / it's **WORSE**

C. For the algorithm we studied, when proving that we only needed to check each point against the next 7 points in the strip with larger y-values, we based this argument on a "box" of certain dimensions. Briefly explain what the size or dimensions of this box would be. (Write a sentence or two at most, or draw a picture.)

D. Consider the box described in the last question. It's not possible for there to be 9 or more points in the box, because then two of these points would cause "an issue" or "a problem". Briefly explan what issue or problem would occur because of two of these points. (Answer is one or two sentences at most. Think about your answer first, and then write something concise and clear, please!)

NOTHING BELOW THIS POINT WILL BE GRADED. USE THE SPACE BELOW FOR SCRATCH WORK