It is recommended that you read through the exam before you begin. Answer all questions in the space provided.

Name: _____

Answer whether the following statements are TRUE or FALSE and support your answer with a proof sketch or counter-example in the space provided.

1. [TRUE / FALSE] $\log(n!) \in \Theta(n \log n)$ [5 pts]

2. [TRUE / FALSE] When comparing two algorithms, the one with the higher average asymptotic complexity will perform worse in all cases. [5 pts]

3. [TRUE / FALSE] If the pivot is chosen uniformly at random, then there is no input to QuickSort that will guarantee $\Theta(n^2)$ performance. [5 pts]

4. [TRUE / FALSE] Every binary tree with n leaves has height $O(\log n)$. [5 pts]

5. [TRUE / FALSE] Breadth First Search is guaranteed to find the shortest path between two nodes in any undirected graph. [5 pts]

6. What is the maximum number of edges that a simple graph with $|V|$ vertices can contain? [5 pts]

7. Prove that $f(n) = 2^{n+3} + 15$ is an element of $O(2^n)$. [5 pts]

8. Prove that $f(n) = 14n^3 - 5n^2$ is an element of $\Omega(n^3)$. [5 pts]

9. Let n be some large integer and $k < \lg n$. For each of the following arrays explain which sorting algorithm you would use to sort the array the fastest and why you chose that algorithm. Make sure you state any assumtions you make about the implementation of your chosen sorting algorithm. Specify the big-O running time for each in terms of k and n.

   (a) An array of n elements where all of the elements are integers between 0 and k. [5 pts]

   (b) An array of n comparable elements in a completely random order. [5 pts]

   (c) An array of n comparable elements that is sorted except for k randomly located elements that are out of place (that is, the list without those k elements would be completely sorted). [5 pts]

   (d) An array of n real numbers randomly distributed between 0 and 1 [5 pts]

10. For the following algorithm represent the value of T as a summation and find a closed form for the summation in terms of n. [10 pts]

```
FOO(n)
1  T = 0
2  k = n
3  while k > 1
4        k = k/2
5        for i = 3 to n
6              for j = 1 to i
7                    T = T + 1
8  return T
```

11. The following algorithm recusively tests whether a character array is a palindrome. (A palindrome is a word that is the same spelled forwards or backwards, example: racecar)

```
IsPalindrome(word, start, end)
1  if end − start < 2
2        return TRUE
3  if word[start] ≠ word[end]
4        return FALSE
5  return IsPalindrome(word, start + 1, end − 1)
```

(a) What is the basic operation for this algorithm and what parameter(s) should be used to measure the size of the input? [5 pts]

(b) Give a recurrence relation for the number of times the basic operation is performed. [5 pts]

(c) Find the closed form and asymptotic complexity of your recurrence relation. [10 pts]

12. Consider the following buggy version of MaxHeapify, the function used to repair a single misplaced element in a heap. What is the bug here? Give an example of a situation in which the bug would happen. [10 pts]

```
BuggyMaxHeapify(A, i)
1  l = Left(i)
2  r = Right(i)
3  if l ≤ A.heapsize and A[l] > A[i]
4        Exchange A[i] and A[l]
5        BuggyMaxHeapify(A, l)
6  elseif r ≤ A.heapsize and A[r] > A[i]
7        Exchange A[i] and A[r]
8        BuggyMaxHeapify(A, r)
```