Name ______________________________ ID No. ______________

There are 200 points possible.

1. (30 pts) Suppose you have a computer that requires, on average, one second to solve problem instances of size $n = 100$. Assuming memory and storage are not a problem, how long would it take, on average, to solve a problem ten times as large ($n = 1,000$) in each of the following situations? (Recall that $A(n)$ is the average-case time for a problem of size n.)

   (a) $A(n) = Cn$ for some constant C. Answer ____________

   (b) $A(n) = Cn^2$ for some constant C. Answer ____________

   (c) $A(n) = C\,2^n$ for some constant C. Answer ____________

2. (30 pts) We have two algorithms for a problem.

   - The average run time for Algorithm A is **much better** than the average run time for Algorithm B.
   - The worst-case run time for Algorithm A is **much worse** than the worst-case run time for Algorithm B.
   - The two algorithms require the same amount of storage and are equally difficult to program correctly.

   The following examples should be rather specific, as in "reordering student names according to GPAs" **but not** "sorting a list."

   (a) Give an example of a situation where **Algorithm A** should be used rather than Algorithm B. Explain BRIEFLY why it should be used.

   (b) Give an example of a situation where **Algorithm B** should be used rather than Algorithm A. Explain BRIEFLY why it should be used.

3. (30 pts.) I have found two divide and conquer algorithms for a problem I want to solve. I tell you that all running times increase with n, the problem size, and also:
   - the average time for Algorithm 1 satisfies $A_1(n) = 2A_1(n/2) + 3n$ when n is a power of 2 and
   - the worst time for Algorithm 2 satisfies $W_2(n) = 5W_2(n/3) + n$ when n is a power of 3.

   (a) Determine the complexity categories of A_1 and W_2.

   (b) I ask you which algorithm is better for large problems. What is your answer? Why?

   (c) A few minutes later, I return and apologize because **I gave you the wrong equations**. I had reversed average case and worst case. The correct recursions are

   $$W_1(n) = 2W_1(n/2) + 3n \qquad \text{and} \qquad A_2(n) = 5A_2(n/3) + n.$$

   What is your answer to (b) now? Why?

4. (20 pts.) Complete the following sentences with a word or brief phrase.

   (a) If it is possible to design a divide and conquer algorithm for a problem, ONE important factor in whether or not the running time will grow at a reasonable rate as the problem size grows is

   ____.

   (b) Suppose it is possible to design a backtracking algorithm for a problem. There are usually various choices to be made when setting up the algorithm. ONE choice that can significantly affect the running time is

   ____.

5. (30 pts) Consider the following algorithm:

```
TRANS(lo, hi) {
    if (1 == hi-lo) return;
    mid = (hi+lo)/2;
    TRANS(lo, mid);
    TRANS(mid, hi);
    for (i=lo; i<mid; i=i+1) {
        t = w[i] + w[i+mid];
        w[i+mid] = w[i] - w[i+mid];
        w[i] = t;
    }
}
```

The algorithm is used when `n` is a power of 2. One invokes the algorithm by `TRANS(0,n)`. It uses an `n`-long external array of numbers `w`. Assume that executing one step of the `for` loop is a basic operation.

(a) What algorithm category(e.g., backtracking) does it belong in and why?

(b) Using induction on m prove that the number of basic operations in `TRANS(0,m)` is the same as the number in `TRANS(j,m+j)` for all j. (You may assume that m is a power of 2.)

(c) Write a recursion for the every-case time complexity of the algorithm. *Do NOT solve the recursion.*
HINT: Use the result from (b). You can do this even if you have not done (b).

6. (60 pts.) Indicate whether true or false. Beware of guessing:

correct answer +4pts. incorrect answer −2pts. no answer 0pts

(a) ____ If $f(n) \in \Theta(g(n))$, then $g(n) \in \Theta(f(n))$.

(b) ____ If $f(n) \in o(g(n))$, then $g(n) \in o(f(n))$.

(c) ____ If $f(n) \in o(g(n))$, then $g(n) \notin o(f(n))$.

(d) ____ If $f(n) \in O(g(n))$, then $g(n) \notin O(f(n))$.

(e) ____ Divide and conquer algorithms use a bottom up approach.

(f) ____ If a divide an conquer algorithm requires recomputing the same quantity many times, it is a good idea to look for a dynamic programming algorithm.

(g) ____ Quicksort has a good average run time and a poor worst-case run time.

(h) ____ Although it requires more complicated data structures, Prim's algorithm for a minimum spanning tree is better than Kruskal's when the graph has a large number of vertices.

(i) ____ Monte Carlo algorithms can be used to estimate the run times for some backtracking programs.

(j) ____ The complexity category of a backtracking program such as n-queens can be determined by a Monte Carlo algorithm.

(k) ____ If you can devise a simple backtracking algorithm for a problem, you should use it since no other algorithm is likely to be faster.

(l) ____ It is impossible to design a sorting algorithm based on comparison of keys whose worst-case run time is in $\Theta(n)$.

(m) ____ It is impossible to design a search algorithm based on comparison of keys of items in a sorted list such that the worst-case run time requires at most $\log_{10} n$ comparisons for large n.

(n) ____ For most problems, it is fairly easy to obtain lower bounds for run-time complexity that are close to the times of the best known algorithms for the problems.

(o) ____ For many problems, the best known algorithms require keeping track of data that was not asked for in the problem.