Name __________________________
PID __________________________

Honor Pledge:
I have not given nor received unauthorized assistance in completing this exam.

Signature __________________________

Note:
(1) All “lg”s are based 2 if unspecified.

(2) Points of subproblems are evenly distributed in each problem.
1. (14’ True or false.
   ______(a) The sorting problem is of \( \Omega(n) \)
   ______(b) If the input list is already sorted, Quicksort runs in \( \Theta(n \log n) \) time
   ______(c) If the input list is already sorted, Randomized Quicksort runs in expected \( \Theta(n \log n) \) time
   ______(d) Mergesort runs in \( \Theta(n) \) best-case time.
   ______(e) \( f(n) = \Omega(g(n)) \) implies \( f(n) \neq o(g(n)) \).
   ______(f) \( f(n) \neq o(g(n)) \) implies \( f(n) = \Omega(g(n)) \).
   ______(g) \( f(n) = \Theta(g(n)) \) implies \( f(n) \neq o(g(n)) \).

2. (16’) For each problem, write ALL correct asymptotic relationships in \{O, \Theta, \Omega\} between the given functions. (Determine if any or some of the following relationships hold: \( f(x) = \mathcal{O}(g(x)) \), \( f(x) = \Theta(g(x)) \), \( f(x) = \Omega(g(x)) \))
   (a) \( f(x) = x^2 - x \), \( g(x) = 3x^2 + 20 \). _________________
   (b) \( f(x) = 6x + 1 \), \( g(x) = 3 \log^2 x + 2 \). _________________
   (c) \( f(x) = 1/(\log x) \), \( g(x) = (\log x)/x \). _________________
   (d) \( f(x) = x^{\log_7} \), \( g(x) = x^3 \). _________________
   You might need the following facts:
   \( \log^a b = (\log b)^a \); \( \log(ab) = \log a \times \log b \); \( \log_a b = \log_c b / \log_c a \) (a,b,c >0 & \neq 1)

3. (16’) Here is a recursive algorithm to solve the Tower of Hanoi problem:
   Algorithm TowerofHanoi(n, Source, Spare, Destination)
   Input : The n smallest rings are stacked on Source
   Output: The n smallest rings are stacked on Destination
   begin
   TowerofHanoi(n – 1, Source, Destination, Spare)
   Move the nth-largest ring directly from Source to Destination (takes time 1)
   TowerofHanoi(n – 1, Spare, Source, Destination)
   end algorithm

   (a) What’s the recurrence relation of the algorithm? _________________
   (b) How many levels are there in the recursion tree? _________________
   (c) Obtain an asymptotic expression for \( T(n) \)? _________________
   (d) Obtain an exact expression for \( T(n) \). _________________
4. (20’) Determine the asymptotic behavior of each of the following recurrence relations using the Master Method. Identify the values a,b, f(n), and the relationship between f(n) and n^{\log_b a}. If Master method is not applicable, write “Does not apply”.
   (a) T(n) = 2*T(n/3) + n
   (b) T(n) = 8*T(n/2)+3n^3+4n+10
   (c) T(n) = 2*T(2n^{1/3})+10n
   (d) T(n) = T(n/3)+\log n

5. (16’) A “double-1” fair die when tossed will give the value 1 to 5, where the probability of 1 is twice than any other side (two sides with “1”, and no “6”).
   (a) A “double-1” fair die is tossed, what is the expected value of the toss, under the condition that the value produced is odd? _____________
   (b) Suppose two “double-1” fair dice are tossed, what is the probability that they will all produce equal values? _________________
   (c) Suppose two “double-1” fair dice are tossed, what is the probability that the sum of their values will equal to 3? _________________
   (d) What is the expected value for the sum of two tosses of a “double-1” fair die? _________________
6. (8') The following algorithm prints all divisors of the number n

```java
x = 1;
while ( x <= n ) {            // Run x = 1, 2, ..., n
    if ( n % x == 0 ) {       // x is a divisor of n
        System.out.println(x); // Print x (because it's a divisor)
    }
    x++;
}
```

(a) State a loop invariant for the while loop.
   At the beginning of the ith loop, we have ...

(b) Assume the loop invariant in (a) holds, prove the correctness of the algorithm.

7. (10') You are given k nuts N and k bolts B, each of different diameter. For any pair (n,b) ∈ N*B, you will learn either n<b, n>b or they fit (n=b). Note that you cannot compare nut to nut or bolt to bolt directly. We'd like to find all pairs that fit.
   In the case that NOT every nut has a bolt that it is paired with and vice versa, prove that any algorithm must do Ω (k^2) tests in the worst case, for some input.