[**Closed book and notes**]Show all of your work clearly in the space provided. Be sure to **read each problem carefully**. You should answer all 4 questions. Note that the exam is double sided.

$$f(n) = \Theta(g(n)) \text{ and } g(n) = \Theta(h(n)) \quad \Rightarrow \quad f(n) = \Theta(h(n)) \tag{1}$$
$$f(n) = O(g(n)) \text{ and } g(n) = O(h(n)) \quad \Rightarrow \quad f(n) = O(h(n)) \tag{2}$$
$$f(n) = \Omega(g(n)) \text{ and } g(n) = \Omega(h(n)) \quad \Rightarrow \quad f(n) = \Omega(h(n)) \tag{3}$$

$$f(n) = \Theta(g(n)) \iff g(n) = \Theta(f(n)) \tag{4}$$

$$x - 1 < \lfloor x \rfloor \leq x \leq \lceil x \rceil < x + 1 \tag{5}$$

$$p(n) = \sum_{i=0}^{d} a_i n^i \tag{6}$$

represents a **polynomial in n of degree d** where d is a positive integer and $a_0, a_1, \ldots, a_d$ are the coefficients of the polynomial ($a_d \neq 0$)

$$\lg n = \log_2 n \tag{7}$$
$$\ln n = \log_e n \tag{8}$$
$$a = b^{\log_b a} \tag{9}$$
$$\log_c(ab) = \log_c a + \log_c b \tag{10}$$
$$\log_b a^n = n \log_b a \tag{11}$$
$$\log_b a = \frac{\log_c a}{\log_c b} \tag{12}$$

$$\sum_{k=1}^{n} k = 1 + 2 + \cdots + n = \frac{n(n+1)}{2} = \Theta(n^2) \tag{13}$$

$$\sum_{k=0}^{n} x^k = 1 + x + \cdots + x^n = \frac{x^{n+1} - 1}{x - 1} = \Theta(x^n), \qquad x \neq 1 \tag{14}$$

$$\sum_{k=1}^{n} \frac{1}{k} = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n} \approx \ln n + .577 = \Theta(\log n) \tag{15}$$

Given positive functions $f(n)$ and $g(n)$ such that

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} = c$$

for some constant c.

1. If $0 < c < \infty$, then $f(n) = \Theta(g(n))$
2. If $0 \leq c < \infty$, then $f(n) = O(g(n))$
3. If $0 < c \leq \infty$, then $f(n) = \Omega(g(n))$

Master Method: Let $a \geq 1, b > 1$ be constants and let $T(n)$ be the following recurrence:

$$T(n) = aT(n/b) + cn^k$$

defined for $n \geq 0$.

1. If $a > b^k$, then $T(n) = \Theta(n^{\log_b a})$
2. If $a = b^k$, then $T(n) = \Theta(n^k \log n)$
3. If $a < b^k$, then $T(n) = \Theta(n^k)$

1. During the running of the procedure RANDOMIZED-QUICKSORT, how many calls are made to the random-number generator RANDOM…

(a) (7 points) … in the worst case? Justify your answer.

(b) (7 points) … in the best case? Justify your answer.

2. For each summation, show that it is either bounded or not bounded above by a constant.

(a) (7 points)

$$\sum_{k=1}^{n} \frac{1}{k^2}$$

(b) (7 points)

$$\sum_{k=-1}^{n+5} \frac{1}{k^2}$$

(c) (7 points)

$$\sum_{k=-4}^{n-4} \frac{5}{(5+k)^2}$$

3. Although merge sort runs in $\Theta(n \lg n)$ worst-case time and insertion sort runs in $\Theta(n^2)$ worst-case time, the constant factors in insertion sort make it faster for small n. Thus, it makes sense to use insertion sort within merge sort when subproblems become sufficiently small. Consider a modification to merge sort in which n/k sublists of length k are sorted using insertion sort and then merged using the standard merging mechanism, where k is a value to be determined.

(a) (10 points) Show that the n/k sublists, each of length k, can be sorted by insertion sort in $\Theta(nk)$ worst-case time.

(b) (10 points) Show that the sublists can be merged in $\Theta(n \lg(n/k))$ worst-case time.

(c) (10 points) Given that the modified algorithm runs in $\Theta(nk + n\lg(n/k))$ worst-case time, what is the largest asymptotic (Θ-notation) value of k as a function of n for which the modified algorithm has the same asymptotic running time as standard merge sort?

(d) (5 points) How should k be chosen in practice?

4. An element x of an array A is the *majority* element of A if at least 50% of the array elements have the value x (Note: it is not neccessary for an array to have a majority element.)

(a) (20 points) Suppose you can perform comparisons on the elements of the array. The comparison test can give three possible outcomes: "=", "$<$", and "$>$". Describe in C++, pseudocode, or unambiguous English an algorithm to determine whether or not a majority element exists. You may make use of algorithms we have discussed in class (e.g., if you decide it would be useful to use insertion sort, you do not need to describe how insertion sort works). Points will be awarded based on the correctness and efficiency of your solution.

(b) (10 points) What is the $\Theta()$ time complexity for your algorithm? Be sure to explain your answer.