

Name:

ID Number:

Exam Type:

...A...

Exam Number:

.....

Grade: 1: ... 2: ... 3: ... 4: ... 5: ... 6: ... Total:

SOLVE all the problems IN THE SPACE PROVIDED

THERE ARE 3 (THREE) PAGES THIS PAGE INCLUDED
Read and Sign the statement below at the end of the exam
Unsigned exams will be marked with grade 0 (zero).

STATEMENT

On my honor, I pledge that I have not violated the provision of the NJIT Student Honor Code.

Sign below at the end of the exam

Signature

.....

List of useful formulae

$$\sum_{i=1}^n i = \frac{n(n+1)}{2}, \quad \sum_{i=1}^n i^2 = \frac{n(n+1)(2n+1)}{6}, \quad \sum_{i=1}^n i^3 = \frac{n^2(n+1)^2}{4}, \quad n! \approx \left(\frac{n}{e}\right)^n, \quad a^{\log_b n} = n^{\log_b a},$$

B1. $f(n) = \Theta(g(n))$ iff $\exists$ positive constants $c_1, c_2, n_0 : 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \quad \forall n \geq n_0$.

B2. $f(n) = \Omega(g(n))$ iff $\exists$ positive constants $c_1, n_0 : 0 \leq c_1 g(n) \leq f(n) \quad \forall n \geq n_0$.

B3. $f(n) = O(g(n))$ iff $\exists$ positive constants $c_2, n_0 : 0 \leq f(n) \leq c_2 g(n) \quad \forall n \geq n_0$.

Master Method. $T(n) = aT(n/b) + f(n)$, such that $a > 0, b > 1, f(n) > 0$.

M1 If $f(n) = O(n^{\lg_b a - \epsilon})$ for some constant $\epsilon > 0$, then $T(n) = \Theta(n^{\lg_b a})$.

M2 If $f(n) = \Theta(n^{\lg_b a} \lg^k n)$, then $T(n) = \Theta(n^{\lg_b a} \lg^{k+1} n)$, where $k \geq 0$ is a non-negative constant.

M3 If $f(n) = \Omega(n^{\lg_b a + \epsilon})$ for some constant $\epsilon > 0$, and if $af(n/b) \leq cf(n)$ for some constant $c < 1$ and for large n , then $T(n) = \Theta(f(n))$.

NOTE: Give an algorithm means, present the algorithm, show that it works as claimed, and analyze its running time by providing an asymptotically tight bound. **Efficient** means an algorithm whose performance (space or time or both) cannot be improved upon asymptotically.

Problem 1. (35 POINTS)

Write a **TRUE** or **FALSE** ; **anything else will be considered wrong**. All algorithms are the class description/implementations. Sequence **DECR** is the decreasing arithmetic sequence $\langle n, n-1, n-2, \dots, 3, 2, 1 \rangle$ of n keys and **INCR** is the increasing sequences $\langle 1, 2, 3, \dots, n-2, n-1, n \rangle$ of n keys.

- (1) SelectionSort is a divide-and-conquer algorithm.
- (2) MergeSort is an incremental algorithm.
- (3) The worst case of InsertionSort is $\Theta(n^2)$.
- (4) MergeSort sorts in-place.
- (5) On input sequence **DECR** InsertionSort is asymptotically faster than MergeSort.
- (6) On input sequence **INCR** MergeSort is asymptotically faster than InsertionSort.
- (7) The solution of $T(n) = T(n-1) + n$ is $T(n) = \Theta(n)$.

Problem 2. (40 POINTS)

Replace ? with the tightest of o, ω, Θ that best describes the relationship between the two sides.

- (1) $2^{4 \lg n} = ? (n^2)$.
- (2) $\lg(n!) = ? (n)$.
- (3) $2^{2 \lg n} = ? (4^{\lg n})$.
- (4) $2^n = ? (n^{\lg n})$.

Problem 3. (25 POINTS)

(Read this Problem very carefully.) Rank the following functions by order of growth; that is, find an arrangement $g_1, g_2, g_3, \dots, g_4$ of the functions satisfying $g_1 = \Omega(g_2)$, $g_2 = \Omega(g_3)$, $g_3 = \Omega(g_4)$, $g_4 = \Omega(g_5)$, . . . Partition your list in equivalence classes such that $f(n)$ and $h(n)$ are in the same class if and only if $f(n) = \Theta(h(n))$. For example for functions $\lg n, n, n^2$, and $2^{\lg n}$ you could write: $n^2, \{n, 2^{\lg n}\}, \lg n$.

$$2^n, \quad n!, \quad n^{\lg n} \quad \lg(n!)$$

Problem 4. (40 POINTS)

The Fibonacci sequence is defined as follows $F_0 = 0$ and $F_1 = 1$ with $F_n = F_{n-1} + F_{n-2}$ for $n > 1$. Give the first five terms $F_0 - F_4$ of the sequence. Then show how to find F_n in constant space $\Theta(1)$ and $O(n)$ time. Justify your claims.

Problem 5. (30 POINTS)

Show that $n^3 - 4n^2 + n = \Theta(n^3)$ by providing the n_0, c_1, c_2 of the definition of Θ .

Problem 6. (30 POINTS)

Solve the following recurrence relations by providing asymptotically tight bounds. You only need to provide the bound, intermediate derivations are not required. If no boundary case is given, the choice of the constants is yours. You may assume that $T(n)$ is positive and monotonically increasing, if you need to do so.

- (1) $T(n) = 9T(n/3) + \sqrt{n}.$
- (2) $T(n) = T(n/3) + n^3 \lg n.$
- (3) $T(n) = 9T(n/3) + n^4.$