Test 1

First Name: ______________________________

Last Name: ______________________________

Student Number: ______________________

- *This test lasts* 75 minutes. *No aids allowed. Write legibly.*
- *Make sure your test has 6 pages, including this cover page.*
- *Answer in the space provided. (If you need more space, use page 6 and indicate clearly that your answer is continued there.)*
- *You may use any algorithm that was covered in class or the readings without explaining how it works. You may use any result that was proved in class or the readings without reproving it.*

Master Theorem: *Let $a \geq 1$ and $b > 1$ be constants, let $f(n)$ be a function and let $T(n)$ be defined on the nonnegative integers by the recurrence $T(n) = aT(\frac{n}{b}) + f(n)$ (where we interpret $\frac{n}{b}$ to mean either $\lfloor \frac{n}{b} \rfloor$ or $\lceil \frac{n}{b} \rceil$).*
1. If $f(n)$ is $O(n^{\log_b a - \varepsilon})$ for some constant $\varepsilon > 0$, then $T(n)$ is $\Theta(n^{\log_b a})$.
2. If $f(n)$ is $\Theta(n^{\log_b a})$ then $T(n)$ is $\Theta(n^{\log_b a} \log n)$.
3. If $f(n)$ is $\Omega(n^{\log_b a + \varepsilon})$ for some constant $\varepsilon > 0$ and if $af(\frac{n}{b}) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n)$ is $\Theta(f(n))$.

Question 1	/3
Question 2	/3
Question 3	/3
Question 4	/5
Question 5	/4
Question 6	/3
Total	/21

[3] **1.** Briefly explain what it means to say the worst-case running time of an algorithm is $O(f(n))$.

[3] **2.** Consider the following recursive algorithm.

```
FOO(n)
    if n > 1 then
        printline("Still going")
        FOO(⌊n/3⌋)
        FOO(⌊n/3⌋)
    end if
end FOO
```

Use Θ notation to describe the number of lines printed by FOO(n).
Briefly explain why your answer is correct.

[3] **3.** What are the last two digits of 3^{96}? Show your work.

[5] **4.** The following algorithm computes $\lfloor\sqrt{n}\rfloor$ by executing a binary search.

SQRT(n) % precondition: n is a positive integer
  $lo = 1$; $hi = n + 1$
  loop
    invariants: (1) lo and hi are positive integers and (2) $lo \leq \sqrt{n} < hi$
    exit when $hi == lo + 1$
    $m = \left\lfloor \frac{lo+hi}{2} \right\rfloor$
    if $m * m \leq n$ then $lo = m$
    else $hi = m$
  end loop
  return lo
end SQRT

(a) Prove invariant (2). (You may assume invariant (1) is already proved.)

(b) Prove that if the loop terminates, $lo = \lfloor\sqrt{n}\rfloor$.

[4] **5.** Consider the following recursive algorithm. We can merge two sorted arrays A and B into a single sorted array C by calling $\textsc{Merge}(A[0..n-1], B[0..m-1], C[0..n+m-1], 0, 0)$.

$\textsc{Merge}(A[0..n-1], B[0..m-1], C[0..n+m-1], i, j)$
  preconditions: (1) $0 \le i \le n$ and $0 \le j \le m$, and (2) A and B are sorted
  if $i < n$ and ($j == m$ or $A[i] \le B[j]$) then
    $C[i+j] = A[i]$
    $\textsc{Merge}(A, B, C, i+1, j)$
  else if $j < m$
    $C[i+j] = B[j]$
    $\textsc{Merge}(A, B, C, i, j+1)$
  end if
end Merge

Prove that the algorithm terminates. Do *not* prove that it satisfies postconditions. If you use a proof by induction, you must carefully state the claim you are proving and what quantity you are using for the size of the input.

[3] **6.** The least common multiple of two positive integers x and y is denoted $\text{lcm}(x, y)$. It is the smallest positive integer z such that z is a multiple of x and a multiple of y.

(a) Give a good algorithm to compute $\text{lcm}(x, y)$. You do *not* have to prove your algorithm is correct. Hint: You can use any algorithm covered in class as a subroutine.

(b) Use big-O notation to state the worst-case running time of your algorithm if it is given two numbers with at most n bits. You may assume the inputs x and y each fit into a single word of memory and arithmetic on single words can be done in constant time.

This nearly blank page is just for additional workspace, if you need it.