1. 28 points, 2 point for each correct response and 2 points for working.

   (a) Using the Master Theorem (Case II): $n^{log_9 3} = n^{1/2}$ is theta of $f(n) = n^{1/2}$. Therefore $T(n) \in \Theta(\sqrt{n} \log n)$.

   (b) Using the Master Theorem (Case I): $n^{log_9 3} = n^{1/2}$, however, $f(n)$ grows polynomially slower than $n^{1/2}$, for, say, $\epsilon = 0.5$. Therefore $T(n) \in \Theta(n^{\log_9 3}) = \Theta(n^{0.5})$.

   (c) Using the Master Theorem (Case III): $n^{log_9 3} = n^{1/2}$, however, $f(n) = n^3$ grows polynomially faster than $n^{1/2}$.
   $n^{0.5} \in O(n^3)$, for, say, $\epsilon = 0.5$.
   and $af(n/b) \leq cf(n)$, i.e., $3 \cdot (n/9)^3 \leq cn^3$ for $c > 1/243$.
   Therefore $T(n) \in \Theta(n^3)$.

   (d) We use recursion tree

   $$\begin{aligned} T(n) &= 4T(n/2) + n^2 \log n \\ &= 16T(n/4) + 4(\frac{n}{2})^2 \log n/2 + n^2 \log n \\ &= 16T(n/4) + n^2 \log n/2 + n^2 \log n \\ &= \cdots \end{aligned}$$

   $$\begin{aligned} T(n) &= n^2 \log n + n^2 \log n/2 + n^2 \log n/4 + \cdots + n^2 \log n/(2^{\log n}) \\ &= n^2(\log n + \log n/2 + logn/4 + \cdots) \\ &= n^2(\log n \cdot n/2 \cdot n/4 \cdots) \\ &= n^2(\log 2^{logn}) \\ &= n^2 \log n \end{aligned}$$

   Therefore, $T(n) \in \Theta(n^2 \log n)$

   (e) Using the Master Theorem (Case I): $n^{log_2 7} = n^{2.807}$, however, $f(n) = n^2 \in O(n^{2.807})$, for, say, $\epsilon = 0.5$. Therefore $T(n) \in \Theta(n^{\log_2 7}) = \Theta(n^{2.807})$

   (f) Using the recursion tree. Assume that the runtime is $O(2^n)$ and $\Omega(n^2)$. Proving this using substitution method:

   - $T(n) \leq c2^n - 4n$
     $T(n) \leq c2^{n-1} - 4(n-1) + c2^{n/2} - 4(n/2) + n$
     $= c(2^{n-1} + 2^{n/2}) - 5n + 4$
     $\leq c(2^{n-1} + 2^{n/2}) - 4n \qquad (n \geq 1/4)$
     $= c(2^{n-1} + 2^{n-1}) - 4n \qquad (n \geq 2)$

$\leq c2^n - 4n$
$= O(2^n)$

- $T(n) \geq cn^2$
  $T(n) \geq c(n-1)^2 + c(n/2)^2 + n$
  $= cn^2 - 2cn + c + cn^2/4 + n$
  $= (5/4)cn^2 + (1-2c)n + c$
  $\geq cn^2 + (1-2c)n + c \qquad (c \leq 1/2)$
  $\geq cn^2$
  $= \Omega(n^2)$

(g) We prove this by induction on n

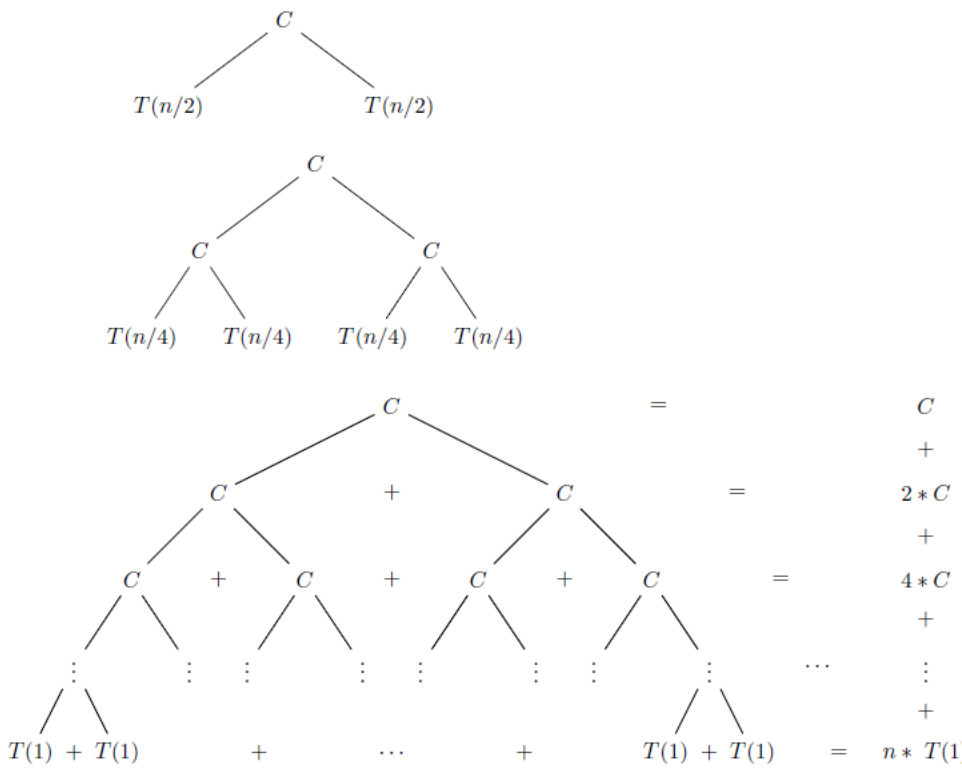
- Base case: $T(2) = 2lg2 = 2$
- Inductive hypothesis: For $n = 2^k$
  $T(2^k) = 2T(\frac{2^k}{2}) + 2^k$
- Inductive step: Prove for 2^{k+1} for $k > 1$
  $T(2^k) = 2T(\frac{2^k}{2}) + 2^k$
  $T(2(2^k)) = 2T(\frac{2 \cdot 2^k}{2}) + 2 \cdot 2^k$
  $T(2^{k+1}) = 2T(\frac{2^{k+1}}{2}) + 2^{k+1}$
  $= 2T(2^k) + 2^{k+1}$
  $= 2(2^k lg(2^k)) + 2^{k+1}$
  $= 2^{k+1} lg(2^k) + 2^{k+1}$
  $= 2^{k+1}(lg(2^k) + 1)$
  $= 2^{k+1}(lg(2^k) + lg(2))$
  $T(2^{k+1}) = 2^{k+1}(lg(2^k))$
  Therefore $nlgn$ is the solution.

2. 20 points, 2 points for each correct recurrence or closed form expression and 3 points for the correct asymptotic bounds.

(a) No Recursion here. The worst case run time of *double for* loop is n^2.

(b) $T(n) = T(n/20) + 1$.
Using the Master Theorem (Case II): $n^{\log_{20} 1} = n^0 = 1$ and $f(n) = 1$. $f(n)O(n^{\log_{20} 1})$. Therefore $T(n) \in \Theta(n^{\log_{20} 1} \cdot \log n) = \Theta(n^0 \cdot \log n) = \Theta(\log n)$.

(c) $T(n) = 2T(n/2) + 2$.
Note that the function $C(n)$ is called twice recursively and there are two *if* statements.



Height of the tree $= \log n$
Number of leaves $= 2^h = 2^{\log n} = n$
$T(n) = C + 2 \cdot C + 4 \cdot C + 8 \cdot C + \cdots$
Therefore, $T(n) \in \Theta(n)$.

(d) $T(n) = T(n/2) + n$.
Using the Master Theorem (Case III): $n^{\log_2 1} = n^0 = 1$ and $f(n) = n$.
$f(n)$ grows polynomially faster than $n^{\log_2 1}$, for, say, $\epsilon = 0.5$.
and $af(n/b) \le cf(n)$, i.e., $1 \cdot (n/2)^1 = n/2 \le cn$ for $c > 1/2$.
$n^0 \in O(n)$
Therefore $T(n) \in \Theta(n)$.