1 Which recurrence best describes the running time of the *Karatsuba* algorithm for integer multiplication?

1. $T(n) \leq 2 \cdot T(\frac{n}{2}) + O(n^2)$
2. $3 \cdot T(\frac{n}{2}) + O(n)$
3. $3 \cdot T(\frac{n}{2}) + O(n^2)$
4. $4 \cdot T(\frac{n}{2}) + O(n)$

(2) is correct. The algorithm does three recursive calls and $O(n)$ additional work.

2 Use the *Master Method* to prove that the *MergeSort* is $O(n \cdot log_2 n)$.

The recurrence of *MergeSort* is $T(n) = 2 \cdot T(\frac{n}{2}) + O(n)$. Recall for the *Master Theorem*

$$T(n) \quad = \quad \begin{cases} O(n^d logn) & if\, a = b^d \\ O(n^d) & if\, a < b^d \\ O(n^{log_b a}) & if\, a > b^d \end{cases}$$

where the parameters a, b and d are obtained from the form $a \cdot T(\frac{n}{b}) + O(n^d)$. Here we see that $a = 2, b = 2$ and $d = 1$. Hence $2 = 2^1$ as in the first case and we $T(n) = O(n \cdot logn)$.

3 What are the respective values of a, b, and d for the binary search algorithm?

1. 1,2,0
2. 1,2,1
3. 2,2,0
4. 2,2,1

The binary search algorithm recurrence is of the form $T(n) = T(\frac{n}{2}) + O(1)$. Hence $a = 0, b = 2$ and $d = 0$. Thus (1) is correct.

4 Use the *Master Theorem* to determine the closed form solution for the *RecIntMult* recurrence.

Recall the recurrence for the *RecIntMult* algorithm is of the form $T(n) = 4 \cdot T(\frac{n}{2}) + O(n)$. $a = 4, b = 2$ and $d = 1.4 > 2^1$ hence $O(n^{log_2 4}) = O(n^2)$.

5 Use the *Master Theorem* to determine the closed form solution for the *Karatsuba* algorithm recurrence

Recall that the recurrence is of the form $T(n) = 3 \cdot T(\frac{n}{2}) + O(n)$. Here $a = 3, b = 2$ and $d = 1$, hence $a > 2^1$ and we obtain $T(n) = O(n^{log_2 3})$.

6 What running time bounds does the master method provide for the $RecMatMult$ and $Strassen$ algorithm, respectively?

1. $O(n^3)$ and $O(n^2)$
2. $O(n^3)$ and $O(n^{log_2 7})$
3. $O(n^3)$ and $O(n^3)$
4. $O(n^3 logn)$ and $O(n^3)$

The $RecMatMult$ recurrence is of the form $T(n) = 8 \cdot T(\frac{n}{2}) + O(1)$. $8 > 2^1$. Here $a = 8, b = 2$ and $d = 0$, hence $T(n) = O(n^{log_2 8}) = O(n^3)$. The *Strassen* matrix multiplication algorithm does $T(n) = 7 \cdot T(\frac{n}{2}) + O(1)$. Hence $T(n) = O(n^{log_2 7})$.

7 What running time bound does the master method provide for the recurrence $T(n) \leq 2 \cdot T(\frac{n}{2}) + O(n^2)$.

$2 < 2^2$ hence $T(n) = n^2$.

8 What is the pattern? Fill in the blanks in the following statement: at each level $j = 0, 1, 2, \ldots$ of the recursion tree, there are [blank] subproblems, each operation on a subarray of length [blank].

1. a^j and $\frac{n}{a^j}$, respectively
2. a^j and $\frac{n}{b^j}$, respectively
3. b^j and $\frac{n}{a^j}$, respectively
4. b^j and $\frac{n}{b^j}$, respectively

(2) is correct.

9 Which of the following statements are true?

1. If RSP < RWS then the amount of work performed is decreasing with the recursion level j.
2. If RSP > RWS then the amount of work performed is increasing with the recursion level j.

3. No conclusion can be drawn about how the amount of work varies with the recursion level j unless RSP=RWS.

4. If RSP=RWS then the amount of work performed is the same at every recursion level.

(1), (2) and (4) are correct.

10 Recall the master method and its three parameters a, b, and d. Which of the following is the best interpretation of b^d?

1. The rate at which the total work is growing
2. The rate at which the number of subproblems is growing
3. The rate at which the subproblem size is shrinking
4. The rate at which the work-per-subproblem is shrinking

(4) is the correct answer.

11 This and the next two questions will give you further practice with the master method. Suppose the running time $T(n)$ of an algorithm is bounded by a standard recurrence with $T(n) \leq 7 \cdot T(\frac{n}{3}) + O(n^2)$. Which of the following is the smallest correct upper bound on the asymptotic running time of the algorithm?

1. $O(n \cdot logn)$
2. $O(n^2)$
3. $O(n^2 \cdot logn)$
4. $O(n^{2.81})$

(2) is correct. Here $a = 7, b = 3$ and $d = 2.7 < 3^2$ hence $T(n) = O(n^2)$.

12 Suppose the running time $T(n)$ of an algorithm is bounded by a standard recurrence with $T(n) \leq 9 \cdot T(\frac{n}{3}) + O(n^2)$. Which of the following is the smallest correct upper bound on the asymptotic running time of the algorithm?

1. $O(n \cdot logn)$
2. $O(n^2)$
3. $O(n^2 \cdot logn)$
4. $O(n^{3.17})$

(3) is correct. Here $a = 9, b = 3$ and $d = 2.a = b^d$ thus $T(n) = O(n^2 \cdot logn)$.

13 Suppose the running time $T(n)$ of an algorithm is bounded by a standard recurrence with $T(n) \leq 5 \cdot T(\frac{n}{3}) + O(n)$. Which of the following is the smallest correct upper bound on the asymptotic running time of the algorithm?

1. $O(n^{log_5 3})$
2. $O(n \cdot logn)$
3. $O(n^{log_3 5})$
4. $O(n^2)$
5. $O(n^{2.59})$

Here $a = 5, b = 3$ and $d = 1. a > b^d$ thus $T(n) = O(n^{log_3 5})$.

14 Suppose the running time $T(n)$ of an algorihm is bounded by the recurrence with $T(1) = 1$ and $T(n) \leq T(\sqrt{n}) + 1$ for $n > 1$. Which of the following is the smallest correct upper bound on the asymptotic running time of the algorithm?

1. $O(1)$
2. $O(loglogn)$
3. $O(logn)$
4. $O(\sqrt{n})$

(2) is correct. Let $m = log_2 n$ and $2^m = n$. We have that $T(2^m) = T(2^{\frac{m}{2}}) + 1$. Let $S(m) = T(2^m)$ then $S(m) = S(\frac{m}{2}) + 1$. We have $T(n) = S(m) = log_2 m$. Which becomes $log_2 log_2 n$.