CS 3343.001 (Spring 2017) Exam 1
Feb 23, 2017
11:30am - 12:50pm (80 minutes)

Name: ___________________________ ID: _______________________

• Don’t forget to put your name and ID on the cover page
• This exam is closed-book, but you can have a one-page cheat sheet.
• If you have a question, **stay seated** and raise your hand.
• Please try to write legibly – if I cannot read it, you may not get credit.
• Do not waste time – if you cannot solve a question immediately, skip it and return to it later.
• Try your best to answer each question. Partial credits will be given if you show that you have some ideas – but not according to the length of your answer.

<table>
<thead>
<tr>
<th></th>
<th>Question</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Comparing functions</td>
<td>13</td>
</tr>
<tr>
<td>2</td>
<td>Proof of asymptotic notations</td>
<td>7 + 5*</td>
</tr>
<tr>
<td>3</td>
<td>Analyzing an iterative algorithm</td>
<td>16</td>
</tr>
<tr>
<td>4</td>
<td>Master theorem</td>
<td>30 + 5*</td>
</tr>
<tr>
<td>5</td>
<td>Recursion tree method</td>
<td>10</td>
</tr>
<tr>
<td>6</td>
<td>Substitution method</td>
<td>10</td>
</tr>
<tr>
<td>7</td>
<td>Analyzing recursive algorithms</td>
<td>14 + 5*</td>
</tr>
<tr>
<td>8</td>
<td>Substitution method (Extra credit)</td>
<td>10*</td>
</tr>
<tr>
<td></td>
<td><strong>Total</strong></td>
<td><strong>125</strong></td>
</tr>
</tbody>
</table>

* Extra credit.
1. Comparing functions (13 points)
   (a) (8 points) Order the following functions according to their order of growth from the lowest to the highest. Put functions that are of the same order (i.e. \( f(n) \in \Theta(g(n)) \)) in the same group.
   \( n^{1.9}, \frac{n}{\log n}, 2^{2n}, \log(n!), n^2 + n \log n, 5^{n/2}, 10n + n^2 \log n, n^3 - 2n + 5 \).

   (b) (5 points) For each pair of functions in the table below, determine whether \( f(n) \in O(g(n)) \), \( f(n) \in \Omega(g(n)) \), \( f(n) \in \Theta(g(n)) \), or all of them. It is NOT necessary to justify your answer.

<table>
<thead>
<tr>
<th></th>
<th>( f(n) )</th>
<th>( O )</th>
<th>( \Omega )</th>
<th>( \Theta )</th>
<th>( g(n) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( n^2 \log n )</td>
<td></td>
<td></td>
<td>( n\sqrt{n} )</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>((n^2 + 5) \log n )</td>
<td></td>
<td>( 3n^2 + \log n )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>( \log_5(n^n) )</td>
<td></td>
<td>( n \log_2 n )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>( n^3 - \log n )</td>
<td></td>
<td>( n^3 + \sqrt{n} )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>( \frac{n^2}{\log n} )</td>
<td></td>
<td>( 5n^2 + n \log n )</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2. (7 + 5 points) Proof of asymptotic notations
   (a) (7 points) Use the basic definition of \( O \), prove that \( n^2 + 4n + 10 \in O(n^2) \).

   (b) (Extra credit - 5 points) Given three asymptotically positive functions \( f(n) \), \( g(n) \), and \( h(n) \) that satisfy the following conditions: \( f(n) \in O(g(n)) \) and \( g(n) \in \Theta(h(n)) \), prove or disprove the following statement:
   1) \( f(n) + g(n) \in O(g(n) + h(n)) \);
   2) \( g(n) + h(n) \in O(f(n) + g(n)) \).

   To prove, use the basic definition of \( O \). To disprove, provide a counterexample.
3. (16 points) You are given the following iterative algorithm:

```plaintext
Enigma(A[1..n])
   //Input: An array A[1..n] of n real numbers
   x = A[1];
   y = A[1];
   for (i = 2; i <= n; i++)
      if (A[i] < x)
         x = A[i];
      if (A[i] > y)
         y = A[i];
   return y - x;
```

a. Which line is the algorithm’s basic operation (i.e., the most executed line)?
   (a) Line 3
   (b) Line 5
   (c) Line 7
   (d) Line 9

b. What is the time complexity of the algorithm, as a function of n?
   (a) $\Theta(n^2)$
   (b) $\Theta(n)$
   (c) $\Theta(1)$
   (d) $\Theta(n \log n)$

c. What does this algorithm compute?
   (a) The maximum number of times that any element is repeated in the array;
   (b) The number of elements in the array;
   (c) The difference between the largest value and the smallest value in the array;
   (d) None of the above.

d. Use Loop Invariant to prove that the answer you selected above in 4c is correct (Hint: before the start of the $i$-th iteration, what are the values stored in $x$ and $y$?)
4. (30 + 5 points) Assume that $T(1) \in \Theta(1)$. Solve the following recurrence functions using the \textbf{master method}. Mark the correct answer. You do NOT need to show all details but doing so could potentially help you earn partial credit.

a. $T(n) = 4T(n/2) + n$;
   
   (a) $\Theta(n)$
   (b) $\Theta(n \log n)$
   (c) $\Theta(n^2)$
   (d) Master Method cannot be applied.
   (e) None of the above.

b. $T(n) = 4T(n/3) + n^{3/4}$;
   
   (a) $\Theta(n^{\log_3 4})$
   (b) $\Theta(n^{3/4})$
   (c) $\Theta(n^{\log_4 3})$
   (d) Master Method cannot be applied.
   (e) None of the above.

c. $T(n) = 9T(n/3) + (n^2 + \log n)$;
   
   (a) $\Theta(n^2)$
   (b) $\Theta(n^2 \log n)$
   (c) $\Theta(n^3)$
   (d) Master Method cannot be applied.
   (e) None of the above.

d. $T(n) = T(n/3) + n \log n$;
   
   (a) $\Theta(n \log n)$
   (b) $\Theta(n \log \frac{n}{\log n})$
   (c) $\Theta(n)$
   (d) Master Method cannot be applied.
   (e) None of the above.

e. $T(n) = 4T(n/4) + n \log^2 n$;
   
   (a) $\Theta(\sqrt{n} \log^2 n)$
   (b) $\Theta(n \log^2 n)$
   (c) $\Theta(n \log^3 n)$
   (d) Master Method cannot be applied.
   (e) None of the above.
f. $T(n) = kT(n/2) + n$, where $k > 2$ is a constant.

(a) $\Theta(n)$
(b) $\Theta(n \log n)$
(c) $\Theta(n^{\log_2 k})$
(d) Master Method cannot be applied.
(e) None of the above.

(g. (Extra credit) $T(n) = 3T(n - 2) + n^3$; Show details if necessary.

(a) $\Theta(n^2 3^n)$
(b) $\Theta(n^3 2^n)$
(c) $\Theta(2^n)$
(d) $\Theta(\sqrt[3]{n})$
(e) $\Theta(3^n)$

5. (10 points) Assume that $T(1) \in \Theta(1)$. Solve the following recurrence function using the recursion tree method to get an asymptotically tight bound.

$$T(n) = 4T(n - 2) + 1$$
6. (10 points) Assume that $T(1) \in \Theta(1)$ and $T(n) = T(n/2) + T(n/3) + n$. Prove that $T(n) \in O(n)$ using the substitution method.

7. (14 + 5 points) Analysis of recursive algorithm. Consider the pseudocode of the following two algorithms. The input $A$ is an array of size $n$. In Alg1, $A$ is divided into two subarrays, and the algorithm is recursively applied to only one subarray. In Alg2, $A$ is divided into five subarrays, and the algorithm is recursively applied to two or three of them, depending on the value of the random number $x$. Assume that parameter passing takes constant time.

```
Alg1 (A[1..n])
    if (n <= 1) return;
    mid = floor (n/2);
    x = rand(); // 0 < x < 1
    if (x <= 0.5)
       Alg1 (A[1..mid]);
    else
       Alg1 (A[mid+1..n]);
end

Alg2 (A[1..n])
    if (n <= 1) return;
    s = floor(n/5);
    x = rand(); // 0 < x < 1
    if (x <= 0.5)
       Alg2 (A[1..s]);
       Alg2 (A[s+1..2s]);
    else
       Alg2 (A[2s+1..3s]);
       Alg2 (A[3s+1..4s]);
       Alg2 (A[4s+1..n]);
end
```
a. (4 points) What is the worst-case running time of Alg1? What about best-case? Write down the recurrence function for the worst- (best-) case running time of Alg1 and solve it.

b. (4 points) What is the worst-case running time of Alg2? What about best-case? Write down the recurrence function for the worst- (best-) case running time of Alg2 and solve it.
c. (6 points) We have assumed that parameter passing takes constant time. Now let’s say it actually takes $f(n) = \Theta(n)$ time to pass an array of size $n$. What are the worst/best-case running time of Alg1 and Alg2 respectively?

d. (Extra credit - 5 points) Assuming that parameter passing takes constant time. What is the average case running time of Alg2? What if it actually takes $f(n) = \Theta(n)$ time to pass an array of size $n$?
8. (Extra credit - 10 points) Using substitution method, prove that the solution you got for the recurrence in 4(g) is correct.
Scratch paper
Scratch paper