Name_____

Midterm 1 (75 minutes)

CSc 345 – Summer 2014

Instructor: Qiyam Tung

Instructions

1. You must do your own work. You may not use notes, computers (cell phones are computers), calculators, or help from your neighbor.
2. If you are having difficulty and need to ask a question you can raise your hand and wait for me to come to you.
3. Unless otherwise noted, give tight upper bounds when asked for run-time or space requirements.

Online students

- If you are not sure what the question is asking, state your assumption before attempting to write the most likely answer.
- The number of points on a question is usually a good indicator of what amount of work is expected.

Non-online students

- If you have a question, raise your hand and I will come to you.

Rules and Theorems

1. Master Theorem:
   For any recurrence relation of the form

   $$T(n) = \begin{cases} T(n) = aT(\frac{n}{b}) + c_1 n^k \\ T(1) = c_2 \end{cases}$$

   where
   a is real >= 1,
   b is integer > 1,
   c_1, c_2 and k are reals > 0,

   the following relationship hold

   $$T(n) = \begin{cases} \Theta(n^{\log_b a}) & \text{if } a > b^k \\ \Theta(n^k \log n) & \text{if } a = b^k \\ \Theta(n^k) & \text{if } a < b^k \end{cases}$$

2. $\sum_{i=1}^{n} \frac{1}{2^i} = 1 - \frac{1}{2^n}$
3. $\sum_{i=0}^{n} 2^i = 2^{n+1} - 1$
4. $\sum_{i=0}^{n-1} ar^k = a\frac{1-r^n}{1-r}$

1 Questions (74 pts total)

1. (8)
A **lower triangular** matrix is a matrix whose entries above the diagonal are zero. Consider writing a data structure such to represent lower triangular matrix, seen visually below.

a			
b	c		
d	e	f	
g	h	i	j

Assume the memory is actually laid out contiguously, like so.

a	b	c	d	e	f	g	h	i	j

(a) (5) Assuming valid input, compute the address of the $n \times n$ lower triangular matrix `array` for row `r` and column `c` (i.e. the address of `array[r][c]`). The starting address is stored in `start_addr`. The size of the element is stored in `elmt_size`.

The calculation is

```
address = start_addr + (r*(r+1)/2 + c)*elmt_size;
```

(b) (3) Calculate the amount of space used for a $n \times n$ lower-triangular matrix. Give the exact solution and explain your reasoning.

The number of columns in successive rows increases by one and starts with one. Hence the total number of elements is $1 + 2 + 3 + \cdots + n = \frac{n(n+1)}{2}$

2. (10) **True or False**. Use the definitions of O, Ω, Θ, o, and ω to answer the following. (I am not concerned about the distinction between bounds and tight bounds for this question).

   Circle **T** or **F**:

   T F $nlogn \in o(logn^n)$: False

   T F $n^{\frac{3}{2}} \in O(\sqrt{5n^3 + 35})$: True

   T F $n^3 + n^2 + 2^n \in O(n^3)$: False

   T F $log_2 n \in O(log_3 n)$: True

   T F $n^4 \in \Theta(4^n)$: False

   T F $2^n \in \omega(n^{100})$: True

   T F $25 \in o(1)$: False

   T F $kf(n) \in O(f(n))$: True

   T F if $f(n) * g(n) \in O(n^3)$, then $f(n) = g(n) = n^{1.5}$: False

   T F if $f(n) \in o(g(n))$, then $f(n) + g(n) \in O(g(n))$: True

3. (5) Step-count the following function and calculate the total amount of work for the *worst case*.

```
// Assume n is the length of the array
function sum_above_128(array, n)
{
    sum = 0                              o++
                                         o++
    for (i = 0; i < n; i++)
    {
                                       o++
                                       o++
      for (j = 0; j < i; j++)
      {
                                    o++
       val = array[j]               o += 3
        if val > 128                o++
          sum +=  val               o+=2 OR o++
                                    o++
      }
                                      o++
    }
                                        o++
    print sum                           o++
}
Both
4 + 3n + 8n(n+1)/2  = 4n^2 + 7n + 4
and
4 + 3n + 8n(n-1)/2
are acceptable responses
```

4. (6) Find the closed form for the following recurrence relation. Assume $n = 7^b$ where $b \in \mathbb{Z}^+$.

$$T(n) = 7T(\frac{n}{7}) + 2n \tag{1}$$

$$T(1) = C_0 \tag{2}$$

$$T(n) = 7T(\frac{n}{7}) + 2n$$

$$T(\frac{n}{7}) = 7T(\frac{n}{7^2}) + 2\frac{n}{7}$$

$$T(\frac{n}{7^2}) = 7T(\frac{n}{7^3}) + 2\frac{n}{7^2}$$

$$T(\frac{n}{7}) = 7(7T(\frac{n}{7^3}) + 2\frac{n}{7^2}) + 2\frac{n}{7}$$

$$T(\frac{n}{7}) = 7^2T(\frac{n}{7^3}) + 2\frac{n}{7} + 2\frac{n}{7}$$

$$T(n) = 7(7^2T(\frac{n}{7^3}) + 2\frac{n}{7} + 2\frac{n}{7}) + 2n$$

$$T(n) = 7^3T(\frac{n}{7^3}) + 2n + 2n + 2n$$

$$T(n) = 7^kT(\frac{n}{7^k}) + 2nk$$

$$log_7 n = k \tag{3}$$

$$T(n) = C_0 n + 2nlog_7 n$$

5. (6) Prove by induction that your answer to question 4 is correct OR prove using induction that

$$T(n) = \begin{cases} 3T(\frac{n}{3}) + n & \text{if } n > 1 \\ C_0 & \text{if } n = 1 \end{cases}$$

has a closed form $C_0 n + nlog_3 n$, where $n = 3^b, b \in \mathbb{Z}^+$

Proof of $T(n) = C_0 n + 2nlog_7 n$

Proof (Inductive):

Basis: n=1

$C_0 = C_0 \cdot 1 + 2 \cdot log_7(1) = Cn_0$

Inductive: if $T(n) = Cn + 2nlog_7 n$, then $T(7n) = 7Cn + 2 \cdot 7nlog_7 7n$

Going from the definition:

$T(7n) = 7T(n) + 2 \cdot 7n$

$= 7(Cn + 2nlog_7 n) + 2 \cdot 7n$, by IH

$= 7Cn + 2 \cdot 7nlog_7 n + 2 \cdot 7n$

$= 7Cn + 2 \cdot 7n(log_7 n + 1)$

$= 7Cn + 2 \cdot 7n(log_7 n + log_7 7)$

$= 7Cn + 2 \cdot 7n(log_7 7n)$

$= 7Cn + 2 \cdot 7nlog_7 7n$

QED

Proof of $T(n) = C_0 n + nlog_3 n$

Proof (Inductive):

Basis: n=1

$C_0 = C_0 \cdot 1 + 1 \cdot log_3(1) = C_0$

Inductive: if $T(n) = C_0 n + nlog_3 n$, then $T(3n) = 3C_0 n + 3nlog_3 3n$

Going from the definition:

$T(3n) = 3T(n) + 3n$

$= 3(C_0 n + nlog_3 n) + 3n$, by IH

$= 3C_0 n + 3nlog_3 n + 3n$

$= 3C_0 n + 3n(log_3 n + 1)$

$= 3C_0 n + 3n(log_3 n + log_3 3)$

$= 3C_0 n + 3n(log_3 3n)$

$= 3C_0 n + 3nlog_3 3n$

QED

6. (2) Derive an asymptotic upper bound for the recurrence relation in question 4 using the Master Theorem.

$a = 7$

$b = 7$

$c_1 = 2$

$k = 1$

$7 = 7^1$, therefore

$T(n) = \Theta(nlogn)$

7. (6) Prove that $5n^3 + 3n^2 \in O(n^3)$. Do not use the Master Theorem.

You can prove it with (1) induction, (2) limits, or a direct proof (3), or (4) cite the polynomial theorem. The following is a direct proof

We need to find a c such that $5n^3 + 3n^2 \leq cn^3$

Proof (Direct):

$5n^3 + n^2 \leq cn^3$

$\frac{5n^3}{n^3} + \frac{n^2}{n^3} \leq c$

$5 + \frac{1}{n} \leq c$

We can pick any arbitrary $c \geq 5$, say $c = 10$.

$5 + \frac{1}{n} \leq 10$

Since $\frac{1}{n}$ always decreases and $n_0 = 1$ makes the left hand side 6, then we have found a c and n_0 such that this condition holds.

QED

8. (16) **Short Answers**

(a) (4) Suppose your task was to write a program to sort a bunch of records. These are records of items sold in a store. Each item has a 4-digit ID. Which sort would you use and why?

Use counting sort. Counting sort can sort a list in $O(n + k)$ time, where n is the number of elements and k is the range of values. Since we know the range is from 0-9999, or 10000 possible values, this then becomes $O(n + 10000) = O(n)$

This is much faster than any comparison sort such as Quicksort.

(b) (4) The e-mails from a user's folder (e.g. "csc345") has been downloaded to a local drive as a single `.mbox` file. Most of the e-mails are sorted by date except a few out-of-order files that were added manually. Would you use Selection Sort or Insertion Sort and why?

Use Insertion sort. Insertion sort is much faster on lists that are mostly sorted. In particular, the number of comparisons would be roughly $O(n)$ whereas for selection sort it is always $O(n^2)$.

(c) (8) Consider the following list:

44 53 14 98 39 191 16 183 95 61

i. (4) Use Radix sort to show the results of the list after the first two passes. Assume the base is 10.

After pass 1: 191 61 53 183 44 14 95 16 98 39

After pass 2: 14 16 39 44 53 61 183 191 95 98

ii. (4) Using the same list above, use Quicksort to sort the list for two passes. Assume for the first pass the pivot is 61 and for the second the pivots are 39 and 191 for the left and right partitions, respectively. Show the list after the pivot has been swapped back after the first and second pass.

After pass 1: 44 53 14 16 39 61 98 183 95 191

After pass 2: 16 14 39 44 53 61 98 183 95 191

9. (5) Give an example of a list (with at least 5 elements) that exposes the weakness of Counting Sort. That is, give a list that would be sorted more quickly by Insertion Sort than by Counting Sort, no matter what the initial order of the elements.

1 100 1000 10000 100000

10. (4) The minimum number of comparisons required to sort three numbers (say, x_1, x_2, and x_3) is three in the worst case, as seen in class.

    Give the minimum number of comparisons required to sort each of the following lists in the worst case: a list of five numbers, and a list of seven numbers.

    Your answer must be a single integer for each input array, but you can leave your answer as an expression. (e.g. $\lceil 2^5 + \pi \rceil$ is an expression that evaluates to an integer).

$\lceil log5! \rceil$ and $\lceil log7! \rceil$, respectively.

We know from the decision tree we came up with in the proof of the worst case lower bound that the height of the tree (and thus comparisons) is $logn!$. Since $logn!$ isn't always an integer, you need to take the ceiling. $logn!$ isn't always an integer because not numbers (all leaves) are powers of two.

11. (6) Suppose merge sort were modified so that it split the list into 3 separate lists (as opposed to 2).

    (a) (4) Give the recurrence relation for the worst case

$T(n) = 3T(\frac{n}{3}) + n$

    (b) (2) Would this change the asymptotic memory requirements?

No.