CS 241, Exam 2, Spring 2014 Exam #: _____ Name: Solution
Prof. David Nassimi (Time: 1.5 hours) First Last

1. Prove the following polynomial is $\Theta(n^3)$.

$$P(n) = 5n^3 + 10n^2 - 20n - 50$$

(a) Prove $O(n^3)$:

Discard Neg Terms

$$P(n) \le 5n^3 + 10n^2, \quad n \ge 0$$
$$\le 5n^3 + 10n^2\left(\frac{n}{100}\right), \quad n \ge 100$$
$$\le 5n^3 + 0.1n^3, \quad n \ge 100$$
$$\le 5.1n^3, \quad n \ge 100$$

PROBLEM AVERAGE

	GRADE	
1	18.4	/20
2	14.5	/20
3	12.7	/20
4	16.8	/20
5	10.3	/20
SUM	72.7	/100

DIST:

Tot: 33
90's 7
80's 8
70's 4
60's 8
50's 3
40's 2
30's 1

MEDIAN = 73
AVERAGE = 72.7
High = 95
Low = 33

(b) Prove $\Omega(n^3)$:

$$P(n) = 5n^3 + 10n^2 - 20n - 50$$

Discard Pos TERMS

$$\ge 5n^3 - 20n - 50$$
$$\ge 5n^3 - 20n\left(\frac{n}{10}\right)^2 - 50\left(\frac{n}{10}\right)^3 \quad n \ge 10$$
$$\ge 5n^3 - 0.2n^3 - 0.050n^3$$
$$\ge 4.75n^3, \quad n \ge 10$$

CS 241, Exam 2, Spring 2014 Name: Solution

2. Consider the relation $R = \{(1,1),(1,3),(2,2),(3,1),(3,2),(3,3)\}$.

(a) Does the relation satisfy each of following properties? Explain.
(12 points)

- Reflexive?

YES, $(i,i) \in R$, $\forall i$.

- Symmetric?

No, $(3,2) \in R$ but $(2,3) \notin R$

- Antisymmetric?

No, $(1,3) \in R$ and $(3,1) \in R$.

- Transitive? (Decide this by directly examining the ordered pairs. Do NOT use matrix multiplication for this part.)

No, $(1,3) \in R$ and $(3,2) \in R$, but $(1,2) \notin R$

- Partial Order?

No, Not antisym and not trans.

- Equivalence Relation?

No, Not symmetric and not Trans.

(b) Show the matrix of this relation. Use matrix multiplication to decide if the relation is transitive. Explain.
(8 points)

$$A = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

$$A^2 = \begin{bmatrix} 1 & \boxed{0} & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & \boxed{1} & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

Not Transitive: $A[1,2] = 0$ and $A^2[1,2] = 1$.

CS 241, Exam 2, Spring 2014 Name: Solution

3. The following program computes $(\log_2 n)$, assuming n is an integer power of 2.

```
int LOG (int n){
int m, k;
m = n;    k = 0;
while (m > 1)
  {m = m/2;    k = k + 1  }
ruturn (k)
}
```

(a) Write a recursive version of this algorithm.

```
int LOG (int n)
{ if (n == 1) return 0;
  T = LOG (n/2)
  return (1 + T);
}
```

(b) Let $f(n)$ be the number of division operations performed by your recursive algorithm to compute $(\log_2 n)$. Write a recurrence equation for $f(n)$. Find the solution by repeated substitution.

$$f(n) = \begin{cases} 0, & n = 1 \\ 1 + f\left(\frac{n}{2}\right), & n \geq 2 \end{cases}$$

$$\begin{aligned} f(n) &= 1 + f\left(\tfrac{n}{2}\right) \\ &= 1 + 1 + f\left(\tfrac{n}{4}\right) \\ &= 1 + 1 + 1 + f\left(n/8\right) \\ &\vdots \\ &= K + f\left(\tfrac{n}{2^K}\right), \quad \text{where } n = 2^K, \quad K = \log n \\ &= K + f(1) = K + 0 \end{aligned}$$

$$f(n) = \log n$$

4. Consider the following recurrence equation, where n is a power of 2.

$$T(n) = \begin{cases} 4T(n/2) + n, & n \geq 2 \\ 0, & n = 1. \end{cases}$$

Prove by induction the solution is $T(n) = An^2 + Bn$, and find the constants A, B.

CLAIM: $T(n) = An^2 + Bn$

Base, $n = 1$: $T(1) = 0$
$= A + B$ $\boxed{A + B = 0}$

To prove the CLAIM for any $n \geq 2$, suppose the CLAIM is true for $n/2$:

$$T\left(\frac{n}{2}\right) = A\left(\frac{n}{2}\right)^2 + B\left(\frac{n}{2}\right)$$

Then,

$$T(n) = 4T\left(\frac{n}{2}\right) + n$$
$$= 4\left[A\left(\frac{n^2}{4}\right) + B\left(\frac{n}{2}\right)\right] + n$$
$$= An^2 + (2B+1)n$$
$$? = An^2 + Bn$$

To make the latter equality, need $\boxed{2B+1 = B}$

So, we need: $\boxed{\begin{array}{l} A + B = 0 \\ 2B + 1 = B \end{array}}$

This gives $B = -1$, $A = +1$.

So, $\boxed{T(n) = n^2 - n}$

CS 241, Exam 2, Spring 2014 Name: Solution

5. Consider the following divide-and-conquer recursive algorithm. Parameter i is the starting index of the array, and j is the ending index. The initial call for an array of size n is FIND$(A, 0, n-1)$.

```
Boolean FIND (int A[ ], int i, int j) {
if (i == j) return FALSE;
m = ⌊(i + j)/2⌋;
f(⌊n/2⌋)  C1 = FIND (A, i, m);          // Left half
f(⌈n/2⌉)  C2 = FIND (A, m + 1, j);      // Right half
1         C3 = (A[i] ≠ A[j])
return (C1 ∨ C2 ∨ C3)
}
```

10 PTS

(a) Figure out what the algorithm does. When does the algorithm return TRUE? Be precise. Explain briefly how the algorithm works.

This algorithm returns true iff there exists any pair i, j, where $A[i] \neq A[j]$.

If $n = 1$ it returns false;

For $n \geq 2$, it makes two recursive calls and returns true iff

(1) There exist a pair in left half that are not eq.

OR (2) " " " " right " "

OR (3) An element from first half $\neq$ An element in 2nd half.

10 PTS

(b) Let $f(n)$ be the number of key comparisons performed by this algorithm for an array of size n. Assuming n is a power of 2, write a recurrence equation for $f(n)$. Guess the solution of the recurrence and prove the correctness of it by induction.

$$f(n) = \begin{cases} 0, & n = 1 \\ 2f(\frac{n}{2}) + 1, & n \geq 2 \end{cases}$$

CLAIM: $\boxed{f(n) = n - 1}$

Base, $f(1) = 0$, $f(1) = 1 - 1 = 0$, so base is correct.

To prove for any $n \geq 2$, suppose true for $n/2$:

$$f(\tfrac{n}{2}) = \tfrac{n}{2} - 1$$

Then,

$$f(n) = 2f(\tfrac{n}{2}) + 1 = 2(\tfrac{n}{2} - 1) + 1 = n - 2 + 1 = n - 1.$$