COP 3530
Section U03
Spring 2017

MIDTERM EXAM

Due Date: March 20th

Question 1 (15 points)

The program below uses the divide and conquer method to minimize the number of multiplications needed to compute x to the power n. The standard approach is to compute $x^n = x * x \ldots * x$, where the multiplication $*$ is performed $n-1$ times.

```
// assume that x != 0 and n >= 0
public static double xTon(double x, int n)
{
    // base cases
    if (n == 0)
        return 1.0;
    if (n == 1)
        return x;
    int half = n / 2; // get the half
    double y = xToN(x,half); // get x to half of x
    if (n % 2 == 0)
        return y * y ;
    else // n is odd
        return y * y * x;
}
```

Let $T(n)$ be the largest number of multiplications performed by this method. Write the boundary value $T(1)$ and the recurrence relation for $T(n)$. Then solve the recurrence relation.

$T(1) = \ldots, T(n) = \ldots$

Question 2. (20 points)

The recurrence $T(n)$ is defined as follows:

$T(n) = c$ if $n \leq 4$

$T(n) = T(n/2) + T(n/3) + T(n/4) + n^2$ if $n > 4$

Find $f(n)$ such that $T(n) = \Theta(f(n))$. Show your work.

Question 3. (25 points)

Solve the recurrence relation

$T(0) = 0$

$T(n) = (1/n)(T(0) + T(1) + \ldots + T(n-1)) + cn$ if $n > 0$

Here c is a positive constant.

Show your derivation.

Question 4. (35 points)

Prove Theorem 10.7, pp 450, from Mark Weiss' book which says,

The solution to the equation $T(n) = aT(n/b) + \Theta(n^k log^p n)$ where $a \geq 1, b > 1, p \geq 0$ is

$T(n) = O(n^{log_b a})$ if $a > b^k$

$T(n) = O(n^k log^{p+1}(n))$ if $a = b^k$

$T(n) = O(n^k log^p(n))$ if $a < b^k$

This a generalization of the Master Theorem.

Question 5. (20 points)

Solve the recurrence relation $T(n) = 7T(n-1) - 16T(n-2) + 12T(n-3)$ with the boundary conditions $T(0) = 2, T(1) = 7, T(2) = 19$.

Show your work.

Question 6. (25 points)

Solve the recurrence $T(n) = T(n-1) + n^4$ by finding a particular solution for the equation. Use $T(1) = 1$ as the boundary condition.

Show your work.

Question 7. (20 Points)

Prove by induction that that $T(n) = \lceil log(n+1) \rceil$, where $T(n)$ is the largest number of comparisons needed to search an array with n items sorted in increasing order using the binary search method. $\lceil x \rceil$, called the ceiling of x, is the smallest integer greater than or equal to x. For example, $\lceil 4.5 \rceil = 5$. Here log is the logarithm in base 2.