

# CIS 121 - Fall 2009

## Midterm Review 1

### 1

Order the following functions by asymptotic growth rate indicating when two or more are big-Theta of each other:  $n^2\sqrt{n}$ ,  $2^{2n-3}$ ,  $\sqrt{n^3}$ ,  $4n \log n + 2n$ ,  $3^{n+3}$ ,  $1 + 2^2 + \dots + n^2$ ,  $\log n^n$ ,  $\frac{\log n^2}{\log n}$

### 2

For each statement below, decide whether it is true or false. In each case attach a *very brief* explanation of your answer.

1. Suppose that the worst-case of method `qq` is  $\Theta(n \log n)$  and the worst case running time of method `uu` is  $\Theta(n^2)$ . Then there is no input for which `uu` runs faster than `qq`, true or false?
2. Suppose that  $f(n)$  is  $O(1)$ . Then, there exists a constant  $c'$  such that  $f(n) \leq c'$  for all  $n \geq 1$ , true or false?
3. Suppose we decide to change our model of computation (step-counting) by counting instructions that involve the creation of an array `new int[n]` as taking  $10n$  steps. In this new model, a program can have a different asymptotic complexity than in the original model, true or false?

### 3

In this problem you are NOT allowed to use the theorems about Big-Oh stated in the lecture notes. Your proof should follow just from the definition of Big-Oh.

Prove that  $n\sqrt{n}$  is *not*  $O(n)$ .

### 4

Let `A` be an integer array of length  $n \geq 3$ . Give an  $O(n)$  algorithm to compute another array `B` of length  $n - 1$  such that for  $i = 0, \dots, n - 2$  we have  $B[i] = \max(\sum_{j=0}^i A[j], \sum_{j=i+1}^{n-1} A[j])$ .

You can use pseudocode to describe the algorithm. (You do *not* need to justify that your algorithm works, just give it.)

## 5

Consider the following statement:

“ if  $f(n)$  and  $g(n)$  are positive functions and  $f(n)$  is  $O(g(n))$  then  $n^{f(n)}$  is  $O(n^{g(n)})$  “.

If the statement is true, prove it. If the statement is false, give a counterexample.

## 6

By adding to the code fragment below, describe in Java a method for multiplying an  $n \times m$  matrix  $A$  and an  $m \times p$  matrix  $B$ . Then, give a Big-oh upper bound on the running time of your method in terms of  $n, m, p$ .

```
double[] matmult(int n, int m, int p, double A[][], double B[][]) {  
    ...  
}
```

## 7

Define the function  $f(n)$  as follows:

$$f(n) = 2^n \text{ when } n \text{ is odd}$$
$$f(n) = 2^{n/2} \text{ when } n \text{ is even}$$

$2^n$  is  $O(f(n))$ , true or false. Prove your answer.

## 8

For each statement below, decide whether it is **true** or **false**. Attach a *very brief* explanation.

1. Suppose program  $A$  runs in time  $\Theta(n^2)$  and program  $B$  runs in time  $\Theta(n^3)$ . Then there is no input for which program  $B$  runs faster than program  $A$ , true or false?
2. Suppose we decide to make our JAVA model of computation more realistic by dividing JAVA instructions into *easy* and *hard* instructions where the easy ones cost 1 step and the hard ones cost 10 steps. A program can have a different asymptotic complexity in this new model of computation from its complexity in the original model, true or false?
3.  $\gcd(x, y) = \gcd(x - y, y)$  for all pairs of integers  $x$  and  $y$ , true or false?

4. Theoretical analysis of running time is useful in distinguishing between programs that are similar in their efficiency while empirical measurements can be used to distinguish between programs with widely different running times, true or false?

## 9

Prove that if  $f(n)$  is  $O(g(n))$  then  $1000f(n)$  is  $O(0.001g(n))$ .

## 10

Consider the following Java code fragment

```
static void swap( int[] a, int i, int j) {
    int temp = a[i];
    a[i] = a[j];
    a[j] = temp;
}
```

```
static int foo( int j, int i) {
    int r = 1;
    for (int k = j; k < i; k++)
        r = (r * k) % 100 ;
    return r;
}
```

```
static void bar(int n) {
    int[] a = new int[n];
    for (int i = 0; i < n; i++)
        a[ i ] = i+1;
    for (int i = 1; i < n; i++ )
        swap( a, i, foo( 0, i) );
}
```

Analyze the running time of bar in Big-Oh notation.