1 Warmup

Given the following method on a sorted array, what is the worst-case runtime? There is an approach to make this algorithm faster. What is that approach and what is the worst-case runtime of the faster algorithm?

```
1   public static int f1(int i, int[] numList) {
2       for (int j = 0; j < numList.length; j++) {
3           if (numList[j] == i) {
4               return j;
5           }
6       }
7       return -1;
8   }
```

$\Theta(n)$

binary search: $\Theta(\log n)$

2 You wanna hang out this Spring '15? Asymptotes!

For each of the pieces of code below, give the runtime in Θ notation as a function of the given parameters. Let $f(x)$ be a function that runs in time linear to the size of its input x.

```
1   public static void f1(int n) {
2       if (n == 0) {return;}
3       f1(n/2);
4       f(n);
5       f1(n/2);
6   }
```

n n

$f(n) + 2f(\frac{n}{2}) + 4f(\frac{n}{4}) + \ldots + \ldots$

$\Theta(n \log n)$ $\log n$.

```
1   public static void f2(int n) {
2       if (n == 0) {return;}
3       f2(n-1);
4       f(17);
5       f2(n-1);
6   }
```

$1 + 2 + 4 + 8$ $1 + 2^1 + 2^2 + \ldots + 2^n \in \Theta(2^n)$

n

```
1   public static void f3(int n, int m) {
2       if (m <= 0) {
3           return;
4       } else {
5           for (int i = 0; i < n; i +=1) {
6               f3(n, m - 1);
7           }
8       }
9   }
```

n

$\Theta(n^m)$

3   It's Fall '16 And I'm Still Doing Asymptotics

1. Give best- and worst-case runtime bounds for the call foo2(N,N) as a function of N. Assume that cnst() is some function that runs in constant time.

```
1    public static void foo2(int i, int N) {
2        if (i==0) {return;}
3        for (int j = 0; j < i; j = j+1) {
4            cnst();
5        }
6        if (i > N/2) {
7            foo2(i-1, N);
8        } else {
9            foo2(i/2, N) + foo2(i/2, N);
10       }
11   }
```

Best case: $\Theta($ N^2 $)$ Worst case: $\Theta($ N^2 $)$

2. True or false: if $f(N) \in O(N)$ and $g(N) \in O(N^2)$, and both functions are non-negative, then $|g(N) - f(N)| \in \Omega(N)$. if true, explain why; otherwise, give a counterexample.

   false $f(N) = N$ $g(N) = N$

3. True or false: if $f(N) \in \Theta(N)$ and $g(N) \in \Theta(N^2)$, and both functions are non-negative, then $|g(N) - f(N)| \in \Omega(N)$. If true, explain why; otherwise give a counterexample.

   $\in \Theta(N^2)$

4. What is a tight big-O bound for the worst case running time of the following algorithm, as a function of the parameter r?

```
1    /** Assumes that VALS is a square array, and that 0 <= R, C < vals.length. */
2    double best(double vals[][], int r, int c) {
3        if (r == 0) {
4            return vals[r][c];
5        }
6        double v = best(vals, r-1, c);
7        if (c > 0) {
8            v = Math.max(v, best(vals, r-1, c-1));
9        }
10       if (c < vals[r].length - 1) {
11           v = Math.max(v, best(vals, r-1, c+1));
12       }
13       return v + vals[r][c];
14   }
```

r
$\downarrow$
$3^{(r-1)}$
$\downarrow$
$9^{(r-2)}$

$1 + 3 + 3^2 + \cdots + 3^r$

Worst case: $O($ 3^r $)$

4   Fall '16: I'm more than just a runtime

1. Your friend, a budding politician, meets several hundred people a day and places their names onto the front of an ArrayList. Once there, he never removes a name, but he sometimes looks through the list to see the order in which he met people. At least one aspect of his procedures is slower than it could be. Describe and justify a small change in your friend's use of the data structure that would improve runtime without changing the data structure involved.

   add at the end

   Now describe and justify a change in the data structure that would improve runtime without requiring a change in actions taken.

   Linked List

5 $\left(\text{Most 61B Problems}\right) \in O\left(\text{These Problems}\right)$

1. Give a tight Θ bound on the running time.

```
1    public int f1(n):
2        if (n == 1){return 0;}
3        if (n is even){
4            return f3(n/2);
5        } else {
6            return f3(n+1);
7        }
```

$$\frac{n}{2} \log\left(\frac{n}{2}\right)$$

Answer: $\Theta($ *log N* $)$

2. Give a tight Θ bound on the running time, where process() is a method that runs in $\Theta(n \log n)$

```
1    function f2(n):
2        if (n = 1){return 1;}
3        int a = f2(n/2);
4        int b = f2(n/2);
5        x = process(a, b);
6        return x;
```

$n \to n\log(n)$

$\downarrow$

$2 + 2(n)2)$ $\dfrac{n\log(n)}{}$

$\downarrow$

Answer: $\Theta($ $n \log^2(n)$ $)$ $4 + 2(\sfrac{4}{n})$.

3. True or False: If $f(n) = O(g(n))$, then $2^{f(n)} = O(2^{g(n)})$

false

$f(n) = n$ 2^n

$g(n) = \dfrac{n}{2}$ $2^{\frac{n}{2}}$