

1 Warm-up

Determine whether $f(n)$ is in $O(g(n))$, $\Omega(g(n))$ or $\Theta(g(n))$

$f(n) = 100$	$g(n) = 1$	_____
$f(n) = n^2 + n$	$g(n) = 0.1n^2$	_____
$f(n) = 2^n$	$g(n) = 2^{2n} + 100$	_____
$f(n) = n^{100}$	$g(n) = 2^n + n \log n$	_____
$f(n) = 3 \log n$	$g(n) = n^2 + n + \log n$	_____
$f(n) = n \log n$	$g(n) = (\log n)^2$	_____

2 Analyzing Runtime

Give the worst case runtime in $\Theta(\cdot)$ notation. Extra: Give the best case runtime in $\Theta(\cdot)$.

A. Give the runtime in terms of M and N . Assume that $\text{bump}() \in \Theta(1)$ and returns a boolean.

```

1 public void wug() {
2     int j = 0;
3     for (int i = 0; i < N; i += 1) {
4         for (; j < M; j += 1) {
5             if (bump(i, j))
6                 break;
7         }
8     }
9 }

1 public int weirdFib(int N) {
2     if (N <= 1) {
3         return N;
4     }
5     return weirdFib(N - 1) + weirdFib(N - 1);
6 }

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

B. Give the runtime in terms of N , where N is the length of the input array.

```
1 public static boolean mystery(int[] arr) {  
2     arr = mergesort(arr); // Runs in  $\Theta(N \log N)$ , where  $N$  is the length arr  
3     int N = arr.length;  
4     for (int i = 0; i < N; i += 1) {  
5         boolean x = false;  
6         for (int j = 0; j < N; j += 1) {  
7             if (i != j && arr[i] == arr[j])  
8                 x = true;  
9         }  
10        if (!x)  
11            return false;  
12    }  
13    return true;  
14 }
```

C. Give the runtime in terms of N , where N is the length of `arr`. Assume `arr` is a sorted array of unique elements and that we initially call `mystery2(arr, 0, arr.length)`.

```
1 public static int mystery2(int[] arr, int low, int high) {  
2     if (high <= low)  
3         return -1;  
4     int mid = (low + high) / 2; // (later, see http://goo.gl/gQI0FN)  
5     if (arr[mid] == mid)  
6         return mid;  
7     else if (mid > arr[mid])  
8         return mystery2(arr, mid + 1, high);  
9     else  
10        return mystery2(arr, low, mid);  
11 }
```

What are `mystery()` and `mystery2()` doing? Assuming an int can appear in `arr` at most twice, can you rewrite `mystery()` with a better runtime?