CS5006 HW 1 7/9/19 Zach Rooney

1) 1) $f1(n) = n^2$

2) $f2(n) = 2n + 20$

1) $g(n) = n^2$, $c = 1$, $n_0 = 1$

2) $g(n) = n$, $c = 2$, $n_0 = 22$

$2n + 20 \leq n^2$ for $n \geq 6$

Equation $f2(n) = 2n + 20$ is better because the Big-O is an order of magnitude less (n vs n^2), it will be a faster algorithm for all $n > 5$

2) A function shows the largest and smallest values in a unsorted Array. Show $f(n)$ for this function and use Big-O def to show $f(n)$ is $O(n)$

Thm: $f(n) = 4 + n * (4 + 3 + 3 + 3 + 3) = 4 + 16n$

Choose: $g(n) = n$, $c = 20$, $n_0 = 5$

$4 + 16n < 20n$ for $n \geq 5$

$\therefore f(n) = 4 + 16n$ is $O(n)$

b would be the stop variable

3) $f(n) = 0+1+2+\ldots+((n-1)-b)$

↳ $= ((n*((n-1)-b))/2 = (n^2-n)-b, /2$

$g(n) = n^2$, $c = b/2$

Big(O) without stop variable = n^2

Big(O) w/ stop variable

Best case: If given a sorted array the Big(O) would be n

Worst case: If given a reverse order array The Big(O) would still be n^2

Since we do not now how the data is sorted, we assume the worst The Big(O) would still be n^2

So adding a stop variable does still results in Big(O) of n^2

4) $n!$: $fact(0) = 1$

$fact(n) = n * fact(n-1)$ $n > 0$

$fact(n) = n \times (n-1) \times (n-2) \times \ldots \times 2 \times 1 \times 1$

Since "n" is the largest number in the equation for factorial, the Big (O) is n. Also, since this is a recursive function it will be called "n" times.