Intro to Algorithms assignment 1

1. Reagan Esteres

(a). Is $2^{n+1} \in O(2^n)$? Why or why not?

- yes $2^{n+1} \in O(2^n)$ because

$2^{n+1} = 2 \cdot 2^n$, thus you can multiply the constant on the other side because constants do not change the result. (as they can be removed).

Therefore, $2(2^n) = O(2^n)$, since they are equal, $2^n \in 2^n$.

(b). Is $2^{2n} \in O(2^n)$? Why or why not?

No, because if we suppose $2^{2n} = O(2^n)$ we can divide both sides by 2^n which leads to $2^n < C$, which is false.
also, for example, let $n = 2$. Thus,
$2^{2 \cdot 2} < 2^2 = 2^4 < 2^2$ which is incorrect
thus it is false.

1c) Is $n^{1.01} \in O(\log^2 n)$?

NO because if you apply L'Hôpital's rule you get:

$$\lim_{x \to c} \frac{f(x)}{g(x)} = \lim_{n \to \infty} \frac{n^{1.01}}{\log^2 n} = \frac{1.01 n^{0.01}}{2(\frac{1}{\ln 10} \cdot \frac{1}{n})}$$

$$\frac{1.01 n^{0.01}}{2 n^{-1} (\ln(10))^{-1}} = \lim_{n \to \infty} \frac{1.01 n^{1.01}}{2 \ln(10)^{-1}} = \boxed{\infty}$$

If it equaled 0 it would be true, but it resulted in ∞ instead making it false.

1d. Is $n^{1.01} \in \Omega(\log^2 n)^2$

Yes because, as proved above, the result from L'Hôpital's rule is ∞ which is to be found in Ω, so it is true since.

$$n^{1.01} = \lim_{n \to \infty} \frac{n^{1.01}}{\log^2 n} = \infty$$

1e). Is $\sqrt{n} \in O((\log n)^3)$?

No because L'Hôpital's rule, when applied, gives.

$$\lim_{n\to\infty} \frac{\sqrt{n}}{(\log_{10} n)^3} = \frac{\frac{1}{2}n^{\frac{1}{2}-1}}{3(\log n)^{3-1}\cdot\frac{1}{n\ln 10}} = \frac{n\ln(10)}{3(\log_{10} n)^2 2n^{\frac{1}{2}}} = \text{(repeat)}$$

$$\lim_{n\to\infty} \frac{\ln(10)}{3(\log_{10} n)^2 2n} = \boxed{\infty}$$

Since it does not equal 0, it cannot be in $O((\log n)^3)$. thus it is false.

1f). As proved in the previous question.

$$\lim_{n\to\infty} \frac{\sqrt{n}}{(\log n)^3} = \boxed{\infty}$$

So it is true because Ω allows for L'Hôpital's rule to yield ∞.

2b). defFoo(x) takes in an integer and sees if it is less than or equal to one. If it is, it returns that number. If not then then given number is subtracted twice by one and then 2 to give an index to two numbers that result from the subtraction, and this add them together. For example if $x=9$. then $9-1=8$ + $9-2=7$, so the 8th + 7th term are added together, $21+13=34$ (which is the 9th term

3b).
The work and span for the iterative version of longest run is O(n). This is because it does not execute several instructions simultaneously and only iterates through the given list once.

3d). Since this is a real dominated problem, the number of recursive calls gives us a height of $\lg n$, giving us the equation $2W(\frac{n}{2})+1$. Therefore, the work and span are the same due to it going in order, making this algorithm have a work and span of $O(n)$.

3e). If we parallelize this code, it will still have the same work, however for the same reason, span is given to us by $W(\frac{n}{2})+1$, making it balanced. Therefore, the worst-case scenario is found at the base case, thus $O(\lg n)$ is our span.