Interesting Questions to consider before the exam

Consider the graph below.
It shows many of the functions we use to define complexity of algorithms - not a topics we have cover in depth, but a topic this course is preparing you for. It is also an interesting overview of important functions.

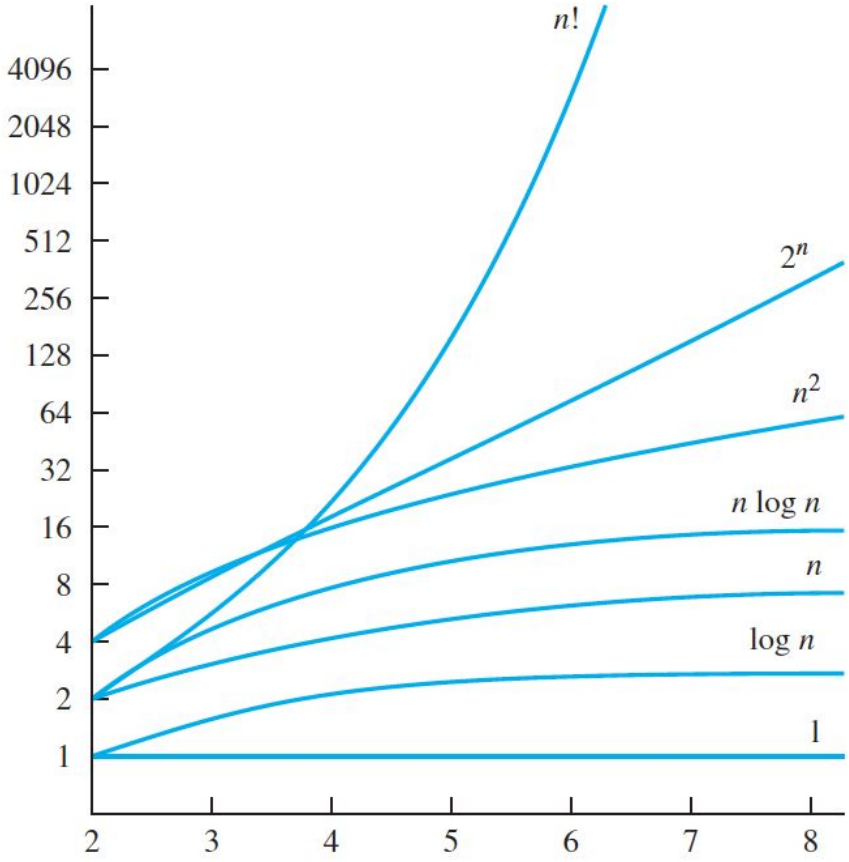
- Notice how fast **n!** grows. We used factorials in counting, and our counts got large quickly, so this makes sense.
- **2^n** could remind you of our proofs of the rules of exponents - and remembering the rules of exponents!
- **n^2** goes all the the way back to proving why FOIL works.
- **n log n** and **log n** should remind you about proving and remembering log rules.
- **n** is something that we have used again and again. Interesting that **n** fits in between **log n** and **n log n**.

The context of this graph is that the y-axis represents (roughly) time of an algorithm, while the x-axis represents (roughly) the amount of data being processed. When considering complexity of algorithms we want to put the complexity of a particular algorithm into one of these categories.

As we saw in hw5 #5, even if **100 log n** is slower than **n** in the short term, long term, **log n** is the better choice (with more data).

Here are some questions about this graph and similar ideas:

1. **Know the order of these - could you put them in order if asked?**
2. **Where does $\sqrt{n}$ fit in ? Use Desmos to play around. Does it have the quality that we saw in hw5 #5 where the coefficients don't really matter?**
3. **Can you answer simple questions about this graph and/or a question like hw5 #5 ?**



Here is another great final exam question.

Consider the set A of all polynomials. Define a relation from A to A as follows -

R = { (a, b) | a and b have the same highest power leading term ignoring the coefficient }

more formally we can define it this way:

> Let $f(x) = a_n x^n + a_{n-1}x^{n-1} + \cdots + a_1 x + a_0$, where $a_0, a_1, \ldots, a_n$ are real numbers with $a_n \neq 0$. Then $f(x)$ is of order x^n.

4. Show R is an equivalence relation.

Understanding and being comfortable with log functions is another goal of this class.

Look at **log n** again.

In terms of complexity if you have an algorithm that is **log n** that is awesome because it grows so slowly.

5. Does the function f(x) = log (x) level off at some point? Or does it grow to infinity? How can we tell?

Logs.

Seriously, understand logs. Know log rules. Be able to prove log rules. Simple slide rules concepts.