Introductions
Dr. | Prof | Professor
Dave | Kauchak
Pronouns: he/him/his

Meet your neighbors
What’s your name?
What year?
What has been your favorite CS class?
What’s been your least favorite CS class?
What do you hope to learn (or get out of) this class?

Algorithms
“For me, great algorithms are the poetry of computation. Just like verse, they can be terse, allusive, dense and even mysterious. But once unlocked, they cast a brilliant new light on some aspect of computing.”
– Francis Sullivan

What is an algorithm?
### Example algorithms
- Sort a list of numbers
- Find a route from one place to another (cars, packet routing, phone routing, ...)
- Find the longest common substring between two strings
- Add two numbers
- Microchip wiring/design (VLSI)
- Solve sudoku
- Cryptography
- Compression (file, audio, video)
- Spell checking
- Page rank
- Classify a web page

### Log properties
- \( x = a^b \)
- What is \( b \)?
- \( b = \log_a x \)
Log properties

\[ \log_a x = a^b \]

\( a \) raised to what exponent is \( x \)?

\[ \log_a a = 1 \]

\[ \log_a x > 1 \text{ if } x > a \]

\[ \log_a x < 1 \text{ if } x < a \]

\( a \) raised to what exponent is \( x \)?

greater than 1  less than 1  exactly 1

Log properties

Which is bigger?

1) \( \log_3 2 \)

2) \( \log_4 2 \)
Log properties

Which is bigger?

1) \( \log_3 2 \equiv x \quad \Rightarrow \quad 2 \equiv 3^x \)

2) \( \log_4 2 \equiv x \quad \Rightarrow \quad 2 \equiv 4^x \)

Log properties

\( \log (ab) = \log a + \log b \)

Which is bigger?

1) \( \log_3 27 = \log_3 3 + \log_3 3 + \log_3 3 \)

2) \( \log_4 36 = \log_4 4 + \log_4 3 + \log_4 3 \)

Log properties

\( \log (ab) = \log a + \log b \)

Which is bigger?

1) \( \log_3 27 = \log_3 3 + \log_3 3 + \log_3 3 \)

2) \( \log_4 36 = \log_4 4 + \log_4 3 + \log_4 3 \)

Log properties

\( \log (ab) = \log a + \log b \)

Which is bigger?

1) \( \log_3 27 = \log_3 3 + \log_3 3 + \log_3 3 \)

2) \( \log_4 36 = \log_4 4 + \log_4 3 + \log_4 3 \)
Log properties

\[ \log (a/b) = \log a - \log b \]

Which is bigger?

1) \( \log_3 4.5 \)
2) \( \log_4 8 \)

1) \( \log_3 4.5 = \log_3 9 - \log_3 2 \)
2) \( \log_4 8 = \log_4 16 - \log_4 2 \)
Log properties

\[ \log_b a = \frac{\log a}{\log b} \]
allows you to change bases!

Log properties

\[ \log a = ? \]
\[ \log x = ? \]
\[ \log x = ? \] if \( x > a \)
\[ \log x = ? \] if \( x < a \)
\[ \log a > 1 \] greater than 1
\[ \log a < 1 \] less than 1
\[ \log a = 1 \] exactly 1

Log properties

Which is bigger?

1) \( \log_3 2 \)
2) \( \log_4 2 \)

Log properties

\[ \log_a b = \frac{\log b}{\log a} \]
Log properties

Which is bigger?

- \( \log_2 2 = \frac{\log 2}{\log 2} \)

- \( \log_a 2 = \frac{\log 2}{\log a} \)

Exponent properties

- \( a^b \cdot a^c = a^{b+c} \)

- \( a^b \cdot a^c = a \cdot a \cdot \ldots \cdot a \cdot a \cdot \ldots \cdot a \)
  
    \(b \text{ times} \quad \text{c times} \)
Exponent properties

Which is bigger?
1) $x^2$
2) $x^{2.1}$

Exponent properties

Which is bigger (for $x > 1$)?
1) $2^{x+1}$
2) $2^x + 2$
Exponent properties

$(ab)^c = a^c b^c$

Which is bigger?

1) $12^3$
2) $4^6$

Which is bigger?

1) $12^3 = (4\times3)^3 = 4^3 3^3$
2) $4^6 = 4^2 4^4$
Exponent properties

\[(a^b)^c = \]

Which is bigger \(x > 1\)?

1) \(2^{2x}\)
2) \(4^x\)

Exponent properties

\[(a^b)^c = a^{bc}\]

\((a^b)^c = (a \cdot a \cdot \ldots \cdot a) \cdot (a \cdot a \cdot \ldots \cdot a) \cdot \ldots \cdot (a \cdot a \cdot \ldots \cdot a)\)

\[b \text{ times} \quad b \text{ times} \quad b \text{ times} \quad \ldots \]
\[c \text{ times}\]

Exponent properties

\[(a^b)^c = a^{bc}\]

Which is bigger \(x > 1\)?

1) \(2^{2x} = (2^2)^x = 4^x\)
2) \(4^x\)
Pseudocode

A way to discuss how an algorithm works that is language agnostic and without being encumbered with actual implementation details.

Should give enough detail for a person to understand, analyze and implement the algorithm.

Proofs

What is a proof?
A deductive argument showing a statement is true based on previous knowledge (axioms)

Why are they important/useful?
Allows us to be sure that something is true
In algo: allow us to prove properties of algorithms
Proof techniques?

- example
- counterexample
- enumeration
- by case
- by inference (aka direct proof)
- trivially
- contradiction
- contradiction
- induction (strong and weak)