CSE 114A Lecture 10

agenda:
- Type classes!

(+) :: Num a ⇒ a → a → a

3 + 4 ✓

3.4 + 5.7 ✓

"pikachu" + "venusaur" ✗

True + True ✗

(\x → x) + (\x → x) ✗

Int, Double, etc. all implement the Num type class.

One way to think of a type class:
A set of operations that you can do on values of of any type that implements that type class.

- For types that implement Num (e.g., Int, Double, etc.), you can do (+), (-), (*), (abs), etc.
- For types that implement Eq, you can do (==) (/=).

Another example:

(==) :: Eq a ⇒ a → a → Bool

5 == 6 ✓

"pikachu" == "venusaur" ✓

True = True ✓

(\x → x) == (\x → y) ✗

"How to make ad hoc polymorphism less ad hoc"
— Wadler and Blott, 1988

↑ Introduced the idea of typeclasses

- Haskell - type classes (since the 90s)
- Rust - traits (since 2012)
- Java - interfaces (since ~2010-2015)
- C++ - concepts ?