Announcements

- HW due Friday - bring to class!
- Next HW - on Haskell
Last time

- Types in Haskell

- Type classes (not an O-O class!)

  Ex: Num, Ord, Eq, Show, Read
Strong pattern matching
Can use pattern matching in unexpected ways.
add Vectors example

Another:

> let my_list = [(1,3), (4,3), (5,6)]
> [(a+b) | (a,b) <- my_list]
Our own head function

\[
\text{head'} : \text{[a]} \rightarrow \text{a} \\
\text{head'} [\text{[]} ] = \text{error 'Can't call head on empty list'} \\
\text{head'} (\text{x : --}) = \text{x}
\]

See first two .hs
Exercise

Use pattern matching:

Write a function which computes the sum of the first 3 elements in a list.

Note: don't call it sum!

Now write a function which computes the sum of all elements in a list.
(Hint — use recursion!)
Guards

A way of testing if properties hold.
(a lot like if statements)

Example: where InAlpha :: String → String

where InAlpha word

head word = 'h' = "beginning"
head word = 'r' = "middle"
otherwise = "end"

guards
**Where Bindings**

We can set local names for expressions.

**Ex:**

\[
\text{initials} ::= \text{String} \rightarrow \text{String} \rightarrow \text{String}
\]

\[
\text{initials} \; \text{firstname} \; \text{lastname} = \[
\text{[f]} \; ++ \; "." \; ++ \; \text{[l]} \; ++ \; "." 
\]
\]

where
\[
\text{[f]} = \text{firstname} \]
\[
\text{[l]} = \text{lastname}
\]