Haskell Refresher Pt 2: ADTs and Pattern Matching
Creating new types

• Example: `data Maybe a = Nothing | Just a`

• Nothing and Just are constructors – functions that construct new instances of the type
Destructing/pattern matching

• Constructors can be used to disassemble or destructure with pattern matching

• Example:
  ```haskell```
  ```plaintext```
  ```
  unwrapWithDefault :: a -> Maybe a -> a
  unwrapWithDefault _ (Just x) = x
  unwrapWithDefault x Nothing = x
  ```
  ```haskell```

• (You used pattern matching on lists in your first lab)
A more complex (recursive) example

```haskell
data Expr = Add Expr Expr  
           | Mul Expr Expr  
           | Val Int

removeZero :: Expr -> Expr
removeZero (Add x (Val 0)) = removeZero x
removeZero x = x
```
Other pattern matching syntax

• An case-expression example:
  removeZero x = case x of
    (Add x (Val 0)) -> removeZero x
    x -> x

• A pattern can have Boolean guards:
  cmp :: Int -> Int -> Int
  cmp a b | a > b = 1
           | a == b = 0
           | otherwise = -1 -- a < b