CS 421 --- Monads Activity

Manager	Keeps team on track	
Recorder	Records decisions	
Reporter	Reports to class	
Reflector	Assesses team performance	

The Join operation

Here is an alternative explanation to Monads than what was in the video. We'll start by talking about the join operation. Observe these four examples.

Input	**Output**
`join [[2,3],[4,5,6]]`	`[2,3,4,5,6]`
`join [[3,1],[],[6,4]]`	`[3,1,6,4]`
`join (Just Nothing)`	`Nothing`
`join (Just (Just 340))`	`Just 340`

Problem 1) What is the purpose of the above code? Not just what it does, but why we might want to do it?

Problem 2) Give an implementation of `join` for these two types.

Problem 3) Here is the implementation of `bind` for List and Maybe.

```
x >>= f = concatMap f x

Nothing >>= f = Nothing
Just x  >>= f = f x
```

How is this related to `join` above?

What will this do?

The point of these examples is to help you become more familiar with how monads behave.

```haskell
inc x = x >>= (\a -> return $ a + 1)

add x y = do
  a <- x
  b <- y
  return $ a + b

-- alternative notation

add' x y = x >>= (\a ->
           y >>= (\b -> return $ a + b))

t1 = Just 10        t4 = []
t2 = Nothing        t5 = [2]
t3 = Just 20        t6 = [5,3,8]
                    t7 = [9,3]
```

Problem 4) For each of these examples, decide as a team which one you think is the best answer.

Example	Candidate A	Candidate B
`add t1 t3`	`Just 30`	`Just (Just 30)`
`add t1 t2`	`Just 10`	`Nothing`
`inc t4`	`[]`	`[1]`
`inc t5`	`[3]`	`[2]`

Problem 5) Give the output to these examples. Be sure everyone on your team agrees with the answers.

Example	Output
`inc t6`	
`add t4 t5`	
`add t5 t7`	
`add t6 t7`	

The Either Monad

Here is the code for `Either`. The `Left` constructor is meant to contain an ``error message'' or failure, and the `Right` constructor is meant to contain the actual data.

```haskell
data Either a b = Left a
                | Right b

instance Functor (Either e) where
  fmap _ (Left x)  = Left x
  fmap f (Right x) = Right (f x)

instance Applicative (Either e) where
  pure = Right
  (Right f) <*> (Right x)  = Right (f x)
  (Left x)  <*> _          = Left x
  _         <*> (Left x)   = Left x
```

Problem 6)

Give the output to these examples. Be sure everyone on your team agrees with the answers.

Example	Output
`fmap (+1) (Left 10)`	
`fmap (+1) (Right 10)`	
`(+) <$> (Right 10) <*> (Right 32)`	
`(*) <$> (Left "FOO") <*> (Right 99)`	

Problem 7)

Give the implementations of `join` and `bind` for `Either`. The first line will be `instance Monad (Either e) where`

Monads Activity--- Reflector's Report

Manager	Keeps team on track	
Recorder	Records decisions	
Reporter	Reports to Class	
Reflector	Assesses team performance	

1. What was a strength of your team's performance for this activity?

2. What could you do next time to increase your team's performance?

3. What insights did you have about the activity or your team's interaction today?