Haskell Programming

```
fmap :: (a -> b) -> Maybe a -> Maybe b
fmap f m = case m of
             Just a -> f a
             Nothing -> Nothing

thenDo :: Maybe a -> (a -> Maybe b) -> Maybe b
thenDo Nothing _ = Nothing
thenDo (Just x) f = f x

thenDo (Just x) (\y -> Just $ f y)
=> (\y -> Just $ f y) (Just x)
=> Just $ (\a -> b) x
=> Just y

thenDo Nothing (\b -> Just $ f b)
=> Nothing
```