Student: Cao Quoc Thang Hoang

Id: s4759487

Question 1:

Show Either satisfies the *second* functor law:

```
fmap (g . h) = fmap g . fmap h
```

Let P() = fmap (g . h) = fmap g . fmap h of Either data_type

```
instance Functor (Either a) where
    fmap f (Right x) = Right (f x)
    fmap f (Left x) = Left x
```

If Either is Left (P(Left)):

```
fmap (g . h) (Left x)
= Left x                          apply fmap (g.h)
= fmap g (Left x)                 unapply (fmap g)
= (fmap g)  $ (fmap h) (Left x)   unapply (fmap h)
= (fmap g) . (fmap h) (Left x)    unapply composition
```

Therefore, P(Left).

If Either is Right (P(Right)):

```
fmap (g . h) (Right x)
= Right $ (g.h) x                 apply fmap g.h
= Right $ g $ h x
= Right $ (fmap g) $ h x          unapply fmap g
= Right $ (fmap g) $ (fmap h)  x  unapply fmap h
= (fmap g) $ Right $ (fmap h) x   by definition of fmap
= (fmap g) $ (fmap h) $ Right x   by definition of fmap
= (fmap g . fmap h) $ Right x     unapply composition
= (fmap g . fmap h)  (Right x)    definition of composition
```

Thus, P(Right). Therefore, Either satisfies the *second* functor law.

Question 2:

Show Either satisfies the *third* applicative law:

x <*> pure y = pure (\g -> g y) <*> x

Knowing that (\g -> g y) = ($ y) and x has Either data-type, thereby,

(Left x) <*> pure y = pure ($ y) <*> (Left x) (1)

Or

(Right x) <*> pure y = pure ($ y) <*> (Right x) (2)

RHS (2): If x = (Right h)

pure ($ y) <*> (Right h)	
= ($ y) <$> (Right h)	apply pure
= Right $ (($ y) h)	apply <$>
= Right $ (h $ y)	apply ($ y)
= h <$> Right y	unapply <$>
= Right h <*> Right y	unapply <*>
= x <*> Right y	by assumption
= x <*> pure y	unapply pure
= LHS (2)	

RHS (1): if x = (Left h)

pure ($y) <*> (Left h)	
= Right ($y) <*> (Left h)	apply pure
= Left h	apply <*>
= Left h <*> Right y	unapply <*>
= Left h <*> pure y	unapply pure
= Left h <*> x	
= LHS (1)	

Thus, P(Right). Therefore, Either satisfies the *third* applicative law.

Question 3:

Show your Applicative satisfies the *second* applicative law:

P(xs) <=> take k $ pure (g x) = take k $ pure g <*> pure x (1)

Base case: k = 0 <=> take 0 (pure (g x)) = take 0 (pure g <*> pure x)

LHS:

take 0 (pure g x)	
= []	apply take 0
= take 0 (repeat $ g x)	unapply take 0
= take 0 (fmap g $ repeat x)	unapply fmap
= take 0 (repeat g <*> repeat x)	unapply <*>
= take 0 (pure g <*> pure x)	unapply pure
= RHS	

The result follows.

Induction: Assume P(k) <=> take k (pure (g x)) = take k (pure g <*> pure x) and consider P(k+1) as

RHS	= take (k+1) (pure g <*> pure x)	
	= take (k+1) (repeat g <*> repeat x)	apply pure
	= take (k+1) (fmap g $ repeat x)	unapply fmap
	= take (k+1) (fmap g $ x:repeat x)	
	= take (k+1) (g x : fmap g $ repeat x)	by line 3
	= g x : take k fmap g $ repeat x	
	= take 1 repeat (g x) : take k $ repeat g <*> repeat x	unapply repeat
	= take 1 pure (g x): take k $ pure g <*> pure x	unapply pure
	= take 1 pure (g x) : take k $ pure g x	inductive hypothesis
	= take 1 pure (g x): take k $ pure g x	unapply take
	= take (k+1) pure g x	
	= LHS	

Thereby, P(k). Thus equation (1) follows from the Principle of Mathematical Induction.

Question 4:

Show your Applicative satisfies the *forth* applicative law:

x <*> (y <*> z) = (pure (.) <*> x <*> y) <*> z

substitute x = xs, y = ys and z = zs, we have:

P() <=> (xs) <*> ((ys) <*> (zs)) = (pure (.) <*> (xs) <*> (ys)) <*> (zs)

Base case:

P([]) <=> [] <*> ((ys) <*> (zs)) = (pure (.) <*> [] <*> (ys)) <*> (zs) (1)

RHS	= (pure (.) <*> [] <*> (ys)) <*> (zs)	
	= (pure (.) <*> []) <*> (zs)	by line 7
	= [] <*> (zs)	by line 8
	= []	by line 7
	= [] <*> ((ys) <*> (zs))	unapply <*>
	= LHS	

Induction: Assume P(xs) <=> (xs) <*> ((ys) <*> (zs)) = (pure (.) <*> (xs) <*> (ys)) <*> (zs), consider P(x:xs) <=> (x:xs) <*> ((y:ys) <*> (z:zs)) = (pure (.) <*> (x:xs) <*> (y:ys)) <*> (z:zs)

LHS	= (x:xs) <*> ((y:ys) <*> (z:zs))	
	= (x:xs) <*> ((y $ z) : (ys <*> zs))	by line 9
	= (x $ y $ z) : (xs <*> (ys <*> zs))	by line 9
	= ((x . y) z) : (xs <*> (ys <*> zs))	
	= ((.) x y) z : (pure (.) <*> (xs) <*> (ys)) <*> (zs)	by inductive hypothesis
	= ((.) x y): (pure (.) <*> (xs) <*> (ys)) <*> (z:zs)	by line 9
	= (pure (.) <*> (x:xs) <*> (y:ys)) <*> (z:zs)	by line 9
	= RHS	

Thereby, P(x:xs). Thus equation (1) follows from the Principle of Mathematical Induction