Claim: `(o ((curry map) f) ((curry map) g)) == ((curry map) (o f g))`

We will prove these functions are equal by proving they always equate to the same thing for any input. We will do this by inducting over general input
`xs = (cons x ys)`

First, we will use laws to simplify both sides of the equation to make the rest of the proof more readable, without making any assumptions about xs:

LHS:

```
((o ((curry map) f) ((curry map) g)) xs)
   = { apply-compose law }
 (((curry map) f) (((curry map) g) xs))
   = { apply-curry law }
(((curry map) f) (map g xs))
   = { apply-curry law }
(map f (map g xs))
```

RHS:

```
(((curry map) (o f g)) xs)
   = { apply-curry law }
(map (o f g) xs)
```

Now, we can use induction to prove that these expressions are equivalent:

Base Case: `xs = '()`

```
(map f (map g xs))
   = { Definition of xs }
(map f (map g '()))
   = { map-nil law }
(map f '())
   = { map-nil law }
'()
   = { map-nil law, right to left }
(map f' '())
   = { define function f' as (o f g) }
(map (o f g) '())
```

Now that we have proven it for a base case, we can assume it to be true for a general xs' and prove it to be true for xs = (cons x xs'):

Proof:
Inductive Hypothesis: Assume `(map f (map g xs')) == (map (o f g) xs')`

```
(map f (map g xs))
   = { Definition of xs }
(map f (map g (cons x xs')))
   = { map-cons law }
(map f (cons (g x) (map g xs')))
   = { map-cons law }
(cons (f (g x)) (map f (map g xs')))
   = { apply-compose law right to left }
(cons ((o f g) x) (map f (map g xs')))
   = { Inductive Hypothesis }
(cons ((o f g) x) (map (o f g) xs'))
   = { map-cons law right to left }
(map (o f g) (cons x xs'))
   = { Definition of xs }
(map (o f g) xs)
```

Thus, we have proven that the claim holds for any general `xs`, So the claim is always true. QED