hofs-theory

Part A

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

Base Case

```
;; Base Case: For an empty list '(),

;; Taking the LHS and applying it to '()
((o ((curry map) f) ((curry map) g)) '())                ; LHS
(((curry map) f) (((curry map) g) '()))                  ; apply-compose
(((curry map) f) (map g '()))                            ; apply-curry
((map f) (map g '())                                     ; apply-curry
(map f '())                                              ; map-nil
'()                                                      ; map-nil


;;Taking the RHS and applying it to '(),
(((curry map) (o f g)) '())                              ; RHS
(map (o f g) '())                                        ; apply-curry
'()                                                      ; map-nil

;; => LHS = RHS when applied on an empty list
```

Inductive Case

```
;; Inductive Case: For a non-empty list (cons y ys),

;; Taking the LHS and applying it to (cons y ys),
((o ((curry map) f) ((curry map) g)) (cons y ys))       ; LHS
(((curry map) f) (((curry map) g) (cons y ys)))         ; apply-compose
(((curry map) f) (map g (cons y ys))                    ; apply-curry
(map f (map g (cons y ys))                              ; apply-curry
(map f (cons (g y) (map g ys)))                         ; map-cons
(cons (f (g y)) (((curry map) f) (map g ys)))           ; map-cons
(cons (f (g y)) (((curry map) f) (((curry f) g) ys)))   ; reverse apply-curry on inner map
```

```
(cons (f (g y)) ((o ((curry map) f) ((curry f) g)) ys)) ; reverse apply-compose
(cons (f (g y)) ((curry map) (o f g) ys))              ; reverse apply-compose
(cons (f (g y)) (map (o f g) ys))                      ; apply-curry
(cons ((o f g) y) (map (o f g) ys))                    ; reverse apply-compose
(map (o f g) (cons y ys))                              ; reverse map-cons
(((curry map) (o f g)) (cons y ys))                    ; reverse apply-curry gives us RHS

;; here, we obtain the RHS ((curry map) (o f g))

;; => LHS = RHS when applied on a non-empty list
```