CS 360
Programming Languages
Day 10 - Foldr

Swift

JavaScript

Dart

```
(define (length lst)
  (if (null? lst) 0
    (+ 1 (length (cdr lst)))))

(define (sum-list lst)
  (if (null? lst) 0
    (+ (car lst) (sum-list (cdr lst)))))

(define (map func lst)
  (if (null? lst) '()
    (cons (func (car lst)) (map func (cdr lst)))))
```

All of these have:

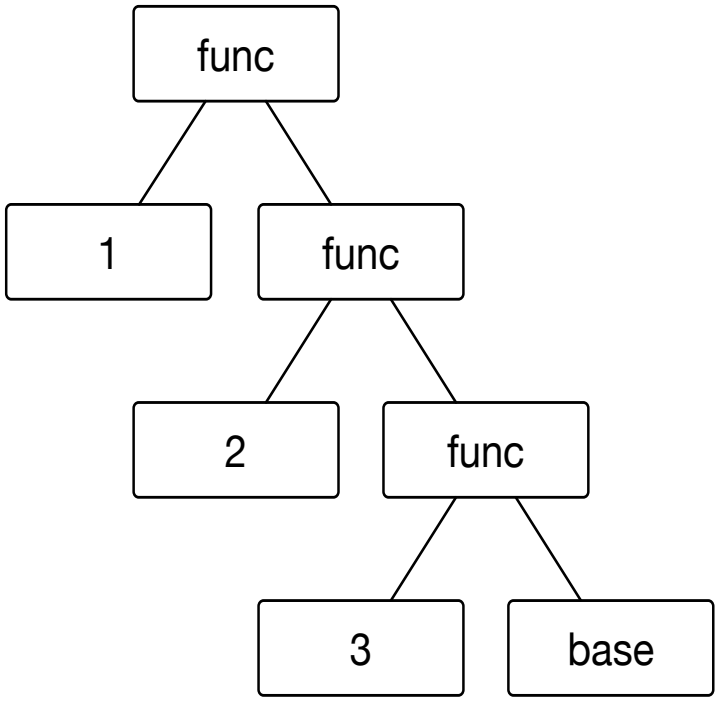
- A base case when the list is null (orange)
- A return value for the base case (green)
- A recursive case where we combine (red) something with the car of the list (purple) with a recursive call on the cdr (blue)

One function to rule them all

```
(define (foldr func base lst)
  (if (null? lst) base
    (func (car lst)
          (foldr func base (cdr lst)))))
```

(foldr func base lst)

Say `lst = '(1 2 3)`



- Foldr applies **`func`** repeatedly to pairs of items, starting from the right end of the list.
- The first two items are the last item in the list and the base element.
- The function must be a function of two items.

  `(f 1 (f 2 (f 3 base)))`
- In general, for `lst = (x1 x2 … xn)`
- **`(f x1 (f x2 (f x3 (f … (f xn base)))…)`**

Examples

- `(foldr + 0 lst)`
- `(foldr (lambda (item acc) (+ 1 acc)) 0 lst)`

Examples with foldr

These are useful and do not use “private data”

```
(define (f1 lst) (foldr + 0 lst))
(define (f2 lst)
  (foldr (lambda (x acc) (and (>= x 0) acc)) #t lst))
```

These are useful and do use “private data”

```
(define (f3 lo hi lst)
  (foldr
    (lambda (x acc)
      (+ (if (and (>= x lo) (<= x hi)) 1 0) acc)) 0 lst))

(define (f4 g lst)
  (foldr (lambda (x acc) (and (g x) acc)) #t lst))
```

You try:

- Write reverse using foldr.
- Write max using foldr.
  - Try to make it so the "base" argument to foldr is not a huge negative number. (write it this way first if it's easier, then change it)
- Write map using foldr.
- Write filter using foldr.