Today

- cons
- reverse

---

**cons**
- Takes two arguments
- Makes the first arg. the car of the new list
- Makes the second arg. the cdr of the new list
- The second argument MUST be a list

```
(cons (reverse '(b c d)) a)
(cons (reverse (cdr lst)) (car lst))
```

---

**reverse**

Wrong recursion

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

---

Lectures in the next few days

- Recursive lambdas
- Generalized lists (and higher order procedures)
- Polish expression evaluation
- Luck function
- Boxed lists
- Tree descent
- Graphics
- Untyped lambda calculus
- Untyped lambda calculus and numbers
- Introduction to Haskell
- Introduction to Scheme
- Introduction to Python
- Introduction to C++
- Introduction to C
- Introduction to Python game development libraries
- Writing an executable program
- Tree recursion
- Graphics
- Untyped lambda calculus
- Untyped lambda calculus and numbers
- Introduction to Haskell
- Introduction to Scheme
- Introduction to Python
- Introduction to C++
- Introduction to C
- Introduction to Python game development libraries
- Writing an executable program