Today

- lambda
- every
- keep
- Accumulate
- Global variables
- Tick-tack-toe (ttt)

Today

- Lambda
- Every
- Keep
- Accumulate
- Global variables
- Tick-tack-toe (ttt)

Every example

(define (change-1st-char char sent)
  (every
   (lambda (wd) (word char (bf wd)))
   sent))

> (change-l"st"-char 'x' (eat get let))

(lambda ('eat) (word 'x 'eat))
(lambda ('get) (word 'x 'get))
(lambda ('let) (word 'x 'let))

‘xat
‘xet
‘xat
‘xet
‘xet

Announcements

- Midterm two Tuesday July 29th
- This one will probably be harder than the first
- Colleen will be out of town Thurs-Monday
- Homework
  - Miniproject 3 starts Today! Due Friday July 25th

Today

- Lambda
- Every
- Keep
- Accumulate
- Global variables
- Tick-tack-toe (ttt)

Today

- Lambda
- Every
- Keep
- Accumulate
- Global variables
- Tick-tack-toe (ttt)

Every example

(define (change-1st-char char sent)
  (every
   (lambda (wd) (word char (bf wd)))
   sent))

> (change-l"st"-char 'x' (eat get let))

(lambda ('eat) (word 'x 'eat))
(lambda ('get) (word 'x 'get))
(lambda ('let) (word 'x 'let))

‘xat
‘xet
‘xat
‘xet
‘xet

Announcements

- Midterm two Tuesday July 29th
- This one will probably be harder than the first
- Colleen will be out of town Thurs-Monday
- Homework
  - Miniproject 3 starts Today! Due Friday July 25th

Today

- Lambda
- Every
- Keep
- Accumulate
- Global variables
- Tick-tack-toe (ttt)

Today

- Lambda
- Every
- Keep
- Accumulate
- Global variables
- Tick-tack-toe (ttt)

Every example

(define (change-1st-char char sent)
  (every
   (lambda (wd) (word char (bf wd)))
   sent))

> (change-l"st"-char 'x' (eat get let))

(lambda ('eat) (word 'x 'eat))
(lambda ('get) (word 'x 'get))
(lambda ('let) (word 'x 'let))

‘xat
‘xet
‘xat
‘xet
‘xet

Announcements

- Midterm two Tuesday July 29th
- This one will probably be harder than the first
- Colleen will be out of town Thurs-Monday
- Homework
  - Miniproject 3 starts Today! Due Friday July 25th

Today

- Lambda
- Every
- Keep
- Accumulate
- Global variables
- Tick-tack-toe (ttt)

Today

- Lambda
- Every
- Keep
- Accumulate
- Global variables
- Tick-tack-toe (ttt)

Every example

(define (change-1st-char char sent)
  (every
   (lambda (wd) (word char (bf wd)))
   sent))

> (change-l"st"-char 'x' (eat get let))

(lambda ('eat) (word 'x 'eat))
(lambda ('get) (word 'x 'get))
(lambda ('let) (word 'x 'let))

‘xat
‘xet
‘xat
‘xet
‘xet

Announcements

- Midterm two Tuesday July 29th
- This one will probably be harder than the first
- Colleen will be out of town Thurs-Monday
- Homework
  - Miniproject 3 starts Today! Due Friday July 25th
keep

(keep procedure sent)
- procedure
  - a procedure that takes in one argument
  - a procedure that returns #t or #f
- sent
  - a sentence with 0 or more words
  - a word with 0 or more letters

keep example
(define (keep-equal? wd sent)
  (keep
    (lambda (one-wd)
      (equal? wd one-wd))
    sent))

> (keep-equal? 'cat '(car cat con))

(accumulate procedure sent)
- procedure
  - a procedure that takes in two arguments
  - a procedure that combines things together
- sent
  - a sentence with 1 or more words
  - a word with 1 or more letters

accumulate
(accumulate (lambda () …) '(cat dog hat tag sag)

How to use global variables
(define (pi) 3.1415)
(define circumference
  (lambda (diameter)
    (* diameter (pi))))

(define pi 3.1415)
(define circumference
  (lambda (diameter)
    (* diameter pi)))
Triples (another representation of a board)

```
X _ _ O O X _ _ _
```

( x23 oo7 xo7 2o8 3x9 xo9 3o7 )