

再帰関数の定義

$$\text{fact}(n) = \begin{cases} 1 & (n=0\text{の時}) \\ n \times \text{fact}(n-1) & (\text{それ以外}) \end{cases}$$

```
(defun fact (n)
  (cond ((zerop n) 1)
        (t (* n (fact (- n 1))))))
```

```
(defun fact2 (n)
  (if (= n 0) 1 (* n (fact2 (- n 1)))))
```

39

$$\text{fib}(n) = \begin{cases} 1 & (n = 0\text{の時}) \\ 1 & (n = 1\text{の時}) \\ \text{fib}(n-1) + \text{fib}(n-2) & (\text{それ以外}) \end{cases}$$

$$\text{comb}(n,m) = \begin{cases} 1 & (m = 0 \text{ または } n = m \text{ の時}) \\ \text{comb}(n-1, m) + \text{comb}(n-1, m-1) & (\text{それ以外}) \end{cases}$$

40

$$\text{ack}(x, y) = \begin{cases} y + 1 & (x = 0 \text{ の時}) \\ \text{ack}(x-1, 1) & (x > 0, y=0 \text{ の時}) \\ \text{ack}(x-1, \text{ack}(x, y-1)) & (x > 0, y > 0 \text{ の時}) \end{cases}$$

41

リスト操作の関数

- リストの長さを求める関数 (LENGTH関数)

$$\text{len}(x) = \begin{cases} 0 & (x \text{ がNILの時}) \\ 1 + \text{len}(x \text{ から1つ要素を取ったもの}) & (\text{それ以外の時}) \end{cases}$$

42

- リストのn番目の要素(0オリジン)を返す関数

(my-nth 0 '(a b c)) ⇒ a

(my-nth 1 '(a b c)) ⇒ b

(my-nth 4 '(a b c)) ⇒ nil

43

my-nth (x,y)

= {
yの先頭要素 (x=0の時)
my-nth (x-1, yの先頭要素を取り除いたリスト)
(それ以外の時)

- リストをくっつける関数

(my-append '(a b) '(c d)) ⇒ (a b c d)

(my-append '(a b) nil) ⇒ (a b)

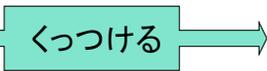
(my-append nil '(a b)) ⇒ (a b)

(my-append nil nil) ⇒ nil

(my-append '((a b) c) '(d (e) f))
⇒ ((a b) c d (e) f)

45

my-append (x, y)

= {
 y (xが空の時)
 xの先頭要素 ←  →
 my-append (xの先頭要素を除いたリスト, y)
 (xが空でない時)

プリント3