特殊快（special form）

2022年12月19日 14:27

过程定义：

```
(define (<name> <formal parameters>) <body>)
```

参数解释：

<name>是一个符号，过程定义将在环境中关联于这个符号

<formal parameters>（形式参数）

<body>是一个表达式

条件表达式：

一般形式：

```
(cond (<p1> <e1>)
      (<p2> <e2>)
      .
      .
      .
      (<pn> <en>))
```

解释：先求**谓词**<p_1>，如果它的值是false，就去求<p_2>，如此继续，直到发现某个谓词时ture，此时返回相应的**序列表达式**<e>。

例如：写绝对值函数

```
(define (abs x)
    (cond ((> x 0) x)
          ((= x 0) 0)
          ((< x 0) (- x)) ))
```

还可以写成这样：

```
(define (abs x)
    (cond ((< x 0) (- x))
          (else x)))
```

特殊形式：

```
(if <predicate> <consequent> <alternative> )
```

用if写绝对值函数：

```
(define (abs x)
    (if (< x 0)
        (- x)
        x))
```

常用的三个复合运算符：

```
(and <e1> ... <e2>)
(or <e1> ... <e2>)
(not <e>)
```

符号：

```
and: /\
or : \/
not: !
```

用and和not表示or：a \/ b = !(!a /\ !b)

用or和not表示and： a/\b = !(!a \/ !b)

匿名过程：

1、**(lambda(<formal-parameters>) <body>)**

我们可以按如下方式来阅读lambda表达式：

```
(lambda    (x)        (+    x      4))
  ↑         ↑          ↑    ↑      ↑
该过程   以x为参数   它加起  x  和  4
```

应用：

格式：(匿名过程定义 参数)

即**((lambda (参数列表) (过程体)) 参数)**

应用这个匿名过程和给定参数求值

例如，((lambda (x) (* x x)) 10)的求值结果为100

定义过程的格式是(define (f arg) body)，

而定义表达式的格式是(define name body)

事实上，过程也可以以类似的方式定义（通过lambda语句)

过程的定义=使用lambda定义匿名过程+给予名字

格式：(define 过程名 匿名过程定义)

例子：(define square (lambda (x) (* x x)))

2、

定义过程时，有些时候一些小的计算过程会反复出现，

根据代换模型，这些反复出现的小的计算过程每出现一次我们就得计算一次。

那么：

```
((lambda (<var1> ...<varn>)
    <body>)
```

((lambda (<var_1> ...<var_n>)
    <$body$>)
 <exp_1>
 ⋮
 <exp_n>)

这种写法的意思是：参数列表中的每一个参数（var）都代表着一个表达式（exp），这些参数可以代替表达式在body中出现，使反复出现的参数只用计算一次

应用：

```
(define (square x) (* x x))
(define (cube x) (* x x x))
(define (quad x) (* x x x x))
(define (f x y)
  ((lambda (a b)
     (+ (* 1 (quad a))
        (* 2 (cube a) b)
        (* 3 (square a) (square b))
        (* 4 a (cube b))
        (* 5 (quad b))))
   (+ 2 y)
   (+ 1 (square x))))
```

```
欢迎使用 DrRacket, 版本 8.1 [cs].
语言: sicp，带调试; memory limit: 128 MB.
> (f 2 4)
12281
```

let语句：

将上述lambda的应用换一种写法：

将这些小表达式的位置从最后（即参数）提前到了过程体之前

格式：(let ((v_1 e_1) . . . (v_n e_n)) 过程体)

这里有一个匿名过程，参数为v1, . . . , vk

求值这个式子时，先求值e1，. . .，ek，再将过程体中的v1, . . . , vk分别替换为e1, . . . , ek的值，之后求值过程体。

注意格式，尤其是括号的数目

例如：

```
(define (square x) (* x x))
(define (cube x) (* x x x))
(define (quad x) (* x x x x))

(define (f x y)
  (let ((a (+ 2 y))
        (b (+ 1 (square x))))
    (+ (* 1 (quad a))
       (* 2 (cube a) b)
       (* 3 (square a) (square b))
       (* 4 a (cube b))
       (* 5 (quad b)))))
```

```
欢迎使用 DrRacket, 版本 8.1 [cs].
语言: sicp，带调试; memory limit: 128 MB.
> (f 2 4)
12281
>
```

内置语法

2022年12月30日 20:20

(newline)：打印一个换行
(display a)：打印a的值，例如(display "/")就是打印一个/
(random a)过程：随机地返回一个小于a的非负整数
(remainder a b)：求a除b的余数
(inc a)：将参数增加1
(dec a)：将参数减少1
(null? items)：用来判断items是否等于nil，即是否是表的“最右边”
(pair? n)：判断n是否是一个序对
(eq? a b)（a和b都是字符串）：判断a和b是否内容相同
(number? exp)：判断a的值是否是数字。
(=number? exp n)用来判断exp是一个数字且值为n
(symbol? x)：判断x是否是一个字符串