14. 向量和结构体

1. 简介

本章中，我将讲解向量和结构体。向量是一组通过整数索引的数据。同一个向量可以储存不同类型的数据，这有点与C语言的数组不同。与表相比，向量更加紧凑且存取时间更短。但从另外一方面来说，向量是通过副作用来操作的，这样会造成负担。Scheme中的结构体与C语言中的结构体类似。但Scheme中的结构体比C语言中的更容易使用，这是因为Scheme为结构体自动创建了读取函数和写入函数，这受益于Lisp/Scheme程序设计语言中的宏。

1. 向量
   1. 字面

向量通过闭合的‘#(’和‘)’表示，例如#(1 2 3)。作为字面值时，它们应该被引用，例如：

'#(1 2 3) ; 整数向量

'#(a 0 #\a) ; 由符号、整数和字符构成的向量

* 1. 定义在向量上的函数

下面的函数都是R5RS规定的函数：

(vector? obj)

如果obj是一个向量则返回#t。

(make-vector k)

(make-vector k fill)

放回一个有k个元素的向量。如果指定了第二个参数fill，那么所有的元素都会被初始化为fill。

(vector obj …)

返回由参数列表构成的向量。

(vector-length vector)

返回向量vector的长度。

(vector-ref vector k)

返回向量vector的索引为k的元素。（译注：和C语言类似，向量从0开始索引。）

(vector-set! vector k obj)

将向量vector的索引为k的元素修改为obj。

(vector->list vector)

将vector转换为表。

(list->vector list)

将list转换为向量。

(vector-fill! vector fill)

将向量vector的所有元素设置为fill。

例：一个对向量中元素求和的函数。

01: (define (vector-add v1 v2)

02: (let ((lenv1 (vector-length v1))

03: (lenv2 (vector-length v2)))

04: (if (= lenv1 lenv2)

05: (let ((v (make-vector lenv1)))

06: (let loop ((i 0))

07: (if (= i lenv1)

08: v

09: (begin

10: (vector-set! v i (+ (vector-ref v1 i) (vector-ref v2 i)))

11: (loop (1+ i))))))

12: (error "different dimensions."))))

练习1

编写一个用于计算两向量内积的函数。

1. 结构体

3．1 大体功能

虽然R5RS中没有定义结构体，但是在很多Scheme实现中，都实现有类似于Common Lisp中的结构体。这些结构体本质上来说都是向量。每一个槽都通过使用一个宏来命名，我将会在下一章（十五章）中讲解这个问题。Structures express data with different kinds of attributes clearly.定义结构体的宏自动为结构体创建读取函数和设置函数。你可以通过程序来写程序，这被认为是Lisp/Scheme最好之处之一。

3.2 MIT-Scheme中的结构体

在MIT-Scheme中，结构体通过函数define-structure来定义。为了使你更加容易理解，我会用一个实例来讲解。请考虑书籍。书籍都有下列属性：

* 标题
* 作者
* 出版商
* 出版年份
* ISBN号

因此结构体book就可以想下面这样定义：

(define-structure book title authors publisher year isbn)

下面演示了如何注册[《大教堂与市集》](http://www.oreilly.com/catalog/cathbazpaper/)。

(define bazaar

(make-book

"The Cathedral and the Bazaar"

"Eric S. Raymond"

"O'Reilly"

1999

0596001088))

然而，这样做多多少少有点不便，因为属性与值的关联并不清楚。参量keyword-constructor可以用于解决这个问题。下面的代码就是使用这个参量的重写版，这个版本中，属性与值的关联就非常清楚了。更进一步来说，有了这个参量参数的顺序就不重要了。

参量copier可用于为结构体创建一个拷贝拷贝函数。

(define-structure (book keyword-constructor copier)

title authors publisher year isbn)

(define bazaar

(make-book

'title "The Cathedral and the Bazaar"

'authors "Eric S. Raymond"

'publisher "O'Reilly"

'year 1999

'isbn 0596001088))

·一个名为“[结构体名称]?”的函数用于检查某对象是否为特定结构体。例如，可使用函数book?来检查bazaar是否为book结构体的一个实例。

* (book? bazaar)
* ;Value: #t

·一个名为“copy-[结构体名称]”的函数用于拷贝结构体。例如，下面的代码演示了将bazaar拷贝到cathedral。

* (define cathedral (copy-book bazaar))

·一个名为“[结构体名称]-[属性名称]”的函数用于读取结构体某属性的值。例如，下面的代码演示了如何读取bazaar的title属性。

* (book-title bazaar)
* ;Value 18: "The Cathedral and the Bazaar"

·一个名为“set-[结构体名称]-[属性名称]!”用于将某属性设定为特定值。下面的代码演示了如何将bazaar的year字段更新到2001（《大教堂与市集》2001年重版）

。

* (set-book-year! bazaar 2001)
* ;Unspecified return value
* (book-year bazaar)
* ;Value: 2001

请参阅[MIT/GNU Scheme Reference: 2.10 Structure Definitions](http://www.gnu.org/software/mit-scheme/documentation/scheme_3.html#SEC41)以获得关于结构体的跟多信息。