Scheme解释器报告

黄俊翔 5140309571

项目内容

该大作业为一个在C++ 11下编写，实现R5RS标准下的Scheme解释器。

举例：

输入：(display (+ 1 2))

输出：3

输入：

(define y1 (list 1 2 3 4))

(display (map (lambda (x) (\* x x)) y1))

输出：(1 4 9 16)

输入：

(define (fib n)

(cond ((= n 0) 0)

((= n 1) 1)

(else (+ (fib (- n 1))

(fib (- n 2))))))

(display (fib 20))

输出：6765

编译环境

在Dev-C++ 5.3.04下编写及编译。

编译时加入以下命令：-std=c++11 -Wl,--stack,100000000

在连接器命令行加入以下命令：-static-libgcc

已实现功能

在历史no side effect中，能正确实现：

utilities

basic operations

try some big numbers

fast-exp

Fibonacci（注：若使用递归斐波那契则32需要一分钟运行时间）

(define unused\_var (fib 50000)) ：implemented tail call optimization

the compute

predicates

pairs and lists（小部分特殊函数未实现，hanoi已实现）

let-bindings

以及不能被作为复杂抽象时调用对象的、仅仅基础的map、apply、length等操作

未实现功能

pairs and lists中的cadr, list-ref, memq, table, assq

infinite stream

N-queens

closure

Test with side effect

实现介绍

用C++去实现Scheme听说比用Java啥的实现Scheme来得复杂……

实际上，由于我个人的奇怪原因，这个C++代码写得看起来更加丑陋……

举个例子，这个解释器中，没有class，没有stl容器如vector、map等（但用了string），实际上指针几乎都没有一个（只有用于访问string的迭代器）。

所有变量全部由一个num数组存放。num代表所有变量类型。（或list的一个单元）

函数本来是单独的，以单独的string数组存放。后来也有作为变量（作为参数）的函数出现。

变量的作用域表示为数组的一段连续区域。

（当然，这么写的后果就是架构不够完善，变量作用域可能会出现问题。存在list这样不能以一个变量保存的情况下尤甚。抢救之后换硬币、一个运用lambda的fib和汉诺塔终于是都过了，但是到了八皇后这里，由于调用实在复杂，调试难度高，成功运行一皇后之后，三皇后又出事，难度进一步提升，抢救无果，继续强行拯救可能会导致之前的内容出现问题，故放弃治疗）

部分内置函数的功能并没有在解释器中直接实现，而是模仿书上的写法，将一段Scheme代码交给解释器去运行来定义内置函数。如apply的实现是一段Scheme代码:

(define (apply tue jupiter)

(if (null? (cdr jupiter))

(car jupiter)

(tue (car jupiter) (apply tue (cdr jupiter)))))

我觉得这样写更能体现Scheme这种语言的精妙之处，也是验证解释器是否足够强大的象征。（不过最后我这样写的map和apply在执行八皇后的时候似乎力不从心了）

感想

解释器，确实是个从未写过的东西。由于知道自己学习新语言的能力弱小，也知道自己在C++程序设计基础方面有很多漏洞，也没写过什么大项目，故选择这个项目来寻求治疗。

写了之后感觉整个人都萌萌哒了。在这个项目中，我感觉学到了很多东西。

首先感谢魔法书，它让我对程序设计有了一些新的认识。连循环都没有的语言，运用层层抽象，能够以令人惊讶的简洁描述一个过程。看了此书真感觉作为魔法师的学徒还是要加油努力，对于程序设计本身的理解还有很长的路要走。

其次是再次学习C++程序设计基础，算是弥补一下第一学期没好好看书……虽然到最后还是设计了一个通用类型，确实想不出怎么才能让C++支持一个函数可以返回不同类型的返回值。不过也算是再长了些经验。

还有就是写一些……大程序的经验吧。我确实没有写过一个拥有一千八百八十行代码的程序，代码仓库也是头一次用……在出问题时候的调试也让我感觉到了一股浓烈的GG气息，确实感觉到了不小的挑战。多亏这个项目的福，感觉也长了不少姿势。

个人感想的话，写这个解释器一开始挺无助的……我真的连解释器是什么都不知道，不知道要干什么……后来还是慢慢找到了一些感觉。写解释器的感觉真的很神奇，和以前写代码是为了实现一个目标，为了算出一个结果不一样。当我写完一段的时候，我心里知道，这个代码本身可以说什么问题也不解决，但是我却知道不论给它输入什么样的解决问题的Scheme代码（实现范围内的），也不论这个程序是个斐波那契还是换硬币，它都能计算出正确结果。不过后期周末憋图书馆里一debug就一天过去的感觉也是久违了……这个事情也告诉我实现基层架构还是要尽量实现好一点。

感谢助教对我的无私帮助，这个暑假辛苦您了。如果没有您，也许我又一个暑假一事无成？（笑）

不论怎样，不论最后这个解释器是否完成，

我都觉得当初我选择解释器这个项目，能够遇见您，真是太好了。

最后，

敬活在计算机中的神灵，祝福那些将他们的思想镶嵌在重重括号之间的程序猿们。

P.S. 输入Requiescat in pace，程序将会自行终止。