实验报告—step7

2018011340 计 83 郭峥岩

一、 实验内容

1. 整体框架

本次实验中 step7 的整体架构与之前的框架相比增加了一个极为重要的概念-作用域。

作用域概念的出现导致框架发生变化：在遍历 AST 生成 IR 码之前首先进行一次**名称解析**，以便 AST 生成 IR 时即可确定每个变量的不同作用域。舍弃之前的在产生 IR 码的时候使用一个 offsetManager 来管理的方法。采用名称解析时产生一符号表的方法。

2. Step7 的功能实现

（1） 实现程序的语法规范的改动：

没有发生变化

（2） IR 类的更改和产生：

IR 类没有发生变化。

名称解析的过程维护了一个符号表，采用栈结构，每个元素为一个字典。Key 代表该作用域中新声明的变量，value 是该变量对应的 frameaddr。每进入一个作用域，就压入一个空字典。离开作用域即弹栈。frameaddr 代表着“此变量刚声明之前，所有开作用域中的变量总数”

（3） 汇编代码的产生：

汇编代码的产生即按照实验指导书的内容将其从 IR 指令逐步翻译成汇编指令即可。

3. 实现细节

声明变量时：首先需要查看其是否在栈顶列表之中，如果存在，则产生重复定义的错误，否则即分配空间，并将变量名与内存地址进行相应的关联。

使用变量时：从栈顶向下查找，如果找不到就报错。

二、 思考题

1. 请将下述 MiniDecaf 代码中的 `???` 替换为一个 32 位整数，使得程序运行结束后会返回 0。

```
int main() {
 int x = ???;
 if (x) {
     return x;
 } else {
     int x = 2;
 }
 return x;
}
```

解答：int x = 0

2. 在实验指导中，我们提到“就 MiniDecaf 而言，名称解析的代码也可以嵌入 IR 生成里”，但不是对于所有语言都可以把名称解析嵌入代码生成。试问被编译的语言有什么特征时，名称解析作为单独的一个阶段在 IR 生成之前执行会更好？

解答：特征：语言中的变量在声明或者定义之前就能使用，则名称解析作为单独阶段在 IR 之前执行会更好。
举例：如果x = 4; int x;是合法的代码段。此时只有将名称解析作为一个单独的阶段才能正确分析出 x 的类型并且进行正确的解析和操作。

三、 参考资料

助教所写的代码：md-dzy branch

四、 总结

本次总体而言进行的比较顺利，主要的问题在于框架的大致修改，在参考了助教的大致框架之后自己进行填写代码就比较简单。