PA1-B 实验报告

2018011289 包涵

实验简述

任务简述

- 抽象类和抽象函数，只要根据ast节点新加的属性，按照文档说明，在type和symbol遍历中相应处理就好。类是否定义了所有抽象方法的判断有点麻烦，通过一个hashset来记录所有未overwrite的抽象方法，然后看有没有都overwrite掉。
- 局部类型推断在不管Lambda的情况下还是很简单的，只要把expr返回的类型设置到var上就好了。
- Lambda表达式太难了，由于细节太多，好多都忘了，下面只能大致描述一下
    - 类型在pa1里面已经实现了。但是要找到所有的lambda表达式，所有可以有expr的ast节点，在type_pass里都要过一遍。
    - 返回类型推断的话，先递归找所有的return语句，然后找最大上界，上界和下界的函数互相递归调用。
    - 函数变量的话，把VarSel改好，能正确返回对应的类型，并且用decaf-rs文档中的方法，判断“当前”变量就可以了。length函数新加了一个TyKind。
    - 调用的话，用VarSel返回的类型去做判断就好了。

挑战和解决方法

- Lambda表达式过于难了。由于类型推断相关文档没有理解对，花了大量的时间在overfit样例上，以后一定要认真看有符号的地方。
- 对Rust的语言特性太不熟悉了，什么ownership，lifetime根本搞不清楚，花了大量时间在通过编译上。当然，Rust好的一点是过了编译，运行基本不会出问题。

问题回答

1. 实验框架中是如何实现根据符号名在作用域中查找该符号的？在符号定义和符号引用时的查找有何不同？ Generally speaking，是通过遍历ScopeStack，在打开的作用域中查找符号名有没有在hashset中。符号定义时，查找打开的作用域中已经定义过的id即可。符号引用时，需要判断符号是否已经完成了定义，就需要额外通过finish_loc来过滤。而且符号引用时，可以查找到类里在该方法位置之后定义的方法，因为type_pass后已经加到scope里了。

2. 对 AST 的两趟遍历分别做了什么事？分别确定了哪些节点的类型？ 第一趟遍历，声明了所有的类、函数、变量和lambda表达式，加入确定了显式定义的类型。确定了ClassDef, FuncDef,部分VarDef，Stmt中的部分LocalVarDef，声明了所有expr中的lambda表达式。所有的var类型和部分lambda表达式的返回值类型都没有确定。对于声明过程中的错误进行了处理。 第二趟遍历，确定所有未确定的节点类型，包括赋值、函数调用、符号引用等，并且判断各种类型是否正确。确定了所有表达式节点的类型，确定了lambda表达式的类型和返回值，推断了var类型变量的类型。

3. 在遍历 AST 时，是如何实现对不同类型的 AST 节点分发相应的处理函数的？请简要分析。
   根据AST节点的内容，调用相应的函数。比如遇到VarSel节点，那么需要知道owner的类型，就要调用expr(owner)。遇到FuncDef节点，就要调用相应的FuncDef函数，并且对param调用VarDef，对body调用Block。这个对于非面向对象的语言来说，似乎比较straightforward.