编译原理 实验四 实验报告

181250117 秦锐鑫

设计与实现

实现功能

1. 完成 基本表达式、语句、函数调用的中间代码翻译
2. 部分完成 数组的中间代码翻译

实现方式

新建 intercode.h 文件，定义 操作数Operand 与 中间代码InterCode 结构体，同实验手册

将关于中间代码生成的代码放到 ir.h 与 ir.cpp 中，等待语义检查全部完成并通过之后再生成中间代码。

使用 cpp stl 中的 list （双向链表）来存储中间代码

```
void Translate(tree_node* ptr,std::map<std::string, struct Sysmtable_item>&
Sysmtable);
```

借助 Translate() 函数，将根节点作为参数传入，该函数根据不同节点类型进行不同的处理

```
    case ENUM_Stmt:
        TranslateStmt(ptr, Sysmtable);
        break;
    case ENUM_Dec:
        TranslateDec(ptr, Sysmtable);
        break;
    case ENUM_FunDec:
        TranslateFunDec(ptr, Sysmtable);
        break;
```

void TranslateStmt(tree_node* ptr,std::map<std::string, struct Sysmtable_item>& Sysmtable)
翻译 Stmt 产生式相关内容

void TranslateDec(tree_node* ptr,std::map<std::string, struct Sysmtable_item>& Sysmtable)
主要翻译数组声明

TranslateFunDec() 翻译函数声明

void TranslateExp(tree_node* ptr,std::map<std::string, struct Sysmtable_item>&
Sysmtable,Operand* place)
翻译基本表达式

void TranslateCond(tree_node* ptr, Operand* label_true,Operand*
label_false,std::map<std::string, struct Sysmtable_item>& Sysmtable) 翻译条件表达式

印象深刻的 bug

关于数组的取地址、解引用需要小心处理