Lab3 实验报告

201240069 曹语桐 201240060 林彦葶

一、如何编译运行程序

在Code 目录下执行：

```
make clean  //清理之前编译链接生成的文件
make
```

生成可执行文件parser。

对测试文件../Test/test.cmm，若输出文件问../Test/test.ir，则在命令行下运行./parser ../Test/test.cmm ../Test/test.ir即可执行。

二、实现的主要功能

我们实现了讲义上所要求的全部必做和选做内容：生成输入文件对应的中间代码，并写入输出文件

1. 中间代码的存储：

   每一条中间代码的数据结构如下所示：

```
struct InterCode_{

 enum{IR_GETVAL,IR_GETADDR,IR_LABEL,IR_ASSIGN,IR_ADD,IR_SUB,IR_MUL,IR_DIV,IR_GOTO,IR_IFGOTO,
    IR_FUNCTIONNAME,IR_RETURN,IR_DEC,IR_ARG,IR_CALL,IR_PARAM,IR_READ,IR_WRITE}kind;
    union{
        Operand one; //LABEL,FUNCTION,GOTO,RETURN,ARG,PARAM,READ,WRITE
        struct {Operand left,right;}two;  //ASSIGN,GETADDR,GETVALUE,PASSIGN,CALL
        struct{Operand result,op1,op2;}three;   //ADD,SUB,MUL,DIV
        struct{Operand var;int size;}dec;   //DEC
        struct{Operand op1,op2,label;   char relop[5];}ifgoto; //  IFGOTO:if op1 [relop] op2 goto label
    }u;
};
```

   中间代码间用双向链表连接，每条中间代码作为其中一个结点

2. 操作数的存储

```
struct Operand_{
```

```
 enum{OP_VARIABLE,OP_CONSTANT,OP_TEMP,OP_TEMP_OFFSET,OP_LABEL,OP_FUNCTIONNAME,OP_ARRAYNAME,
    OP_STRUCTURENAME,OP_ADDRESS,OP_ADDRESS_LEFT}kind;
    union{
        char *name;//OP_VARIABLE,OP_LABEL,OP_FUNCTIONNAME,OP_ARRAYNAME,OP_STRUCTURENAME
        int no;//OP_TEMP,OP_TEMP_OFFSET,OP_ADDRESS,OP_ADDRESS_LEFT
        int number;//OP_CONSTANT
    };
    struct{ //for array and structture
        Type type;
        int param;  //是不是函数参数
        int offset;
    }optype;
};

```

3. 代码结构

为每个主要的语法单元X设计响应的翻译函数translate_X。主要的难点翻译函数在translate_exp、translate_stmt和translate_Cond三个函数上。此三个函数主要构造模式见书本，在此只注明几个需要注意的点：

- 数组/结构体大小

```
int get_size(Type type){
    if(type==NULL)  return 0;
    int size=0;
    switch(type->kind){
        case STRUCTURE:
            遍历type下structurefield中每一个子成员structmb;
            size=size+get_size(structmb);
        case ARRAY:
            Type arrmb_type=type->u.array.elem;
            size=get_size(arrmb_type)*type->u.array.size;
        case BASIC:
            size=4;
    }
    return size;
}
```

- 取址和取值
  - translate_FunDec()函数中，函数参数若为结构体或数组，则应打上isPARAM标记。后面处理它们时应将PARAM语句的参数都设置为地址。
  - translate_Exp()函数中，若调用的translate_Exp返回值为OP_ARRESS，OP_ARRAYNAME，OP_STRUCTURENAME类型，要注意添加一条取值语句。特别注意Exp->Exp ASSIGNOP Exp：

```
//代码片段（对应产生式Exp->Exp ASSIGNOP Exp）
translate_Exp(exp1,t1);
translate_Exp(exp2,t2);
if(t2->kind==OP_ADDRESS||t2->kind==OP_ARRAYNAME||t2->kind==OP_STRUCTURENAME)
    t2=get_value(t2);
if(t1->kind==OP_ADDRESS||t1->kind==OP_ARRAYNAME||t1->kind==OP_STRUCTURENAME) //右值可以getvalue赋给一个新的temp变量，左值不可以
    get_value_left(t1);
insert_ir(gen_ir(IR_ASSIGN,t1,t2,NULL));
insert_ir(gen_ir(IR_ASSIGN,place,t2,NULL));
```

- 函数调用相关
  - 需注意READ、WRITE函数应与其他函数分开处理，它们的中间代码样式不同。另外在语义分析构造符号表时，还应注意最先要将read和write函数加入符号表
  - 函数调用实参传递ARG和函数形参定义PARAM中，参数传递顺序相反。处理translate_arg时应反转arglist列表