编译原理实验二报告

181860109 吴润泽 181860109@smail.nju.edu.cn

实现功能

在本次实验中，我的任务为**要求2.3**，在实现符号表存储检索和类型检查的基础上，修改结构体的等价机制，实现结构等价。

在词法分析和语法分析均正确之后，采用自顶向下的方式进行语义分析，进行符号信息的添加维护和类型检查。

符号表

对于符号表，我采用了哈希表的实现方式，并采用开散列的方式处理冲突。

符号类型

在手册的基础上我定义了五种符号类型:

```
struct Type_ {
    enum { BASIC, ARRAY, STRUCTURE, STRUCTTAG, FUNCTION } kind;
    bool need_free;
    bool struct_def_done;
    union {
        enum { NUM_INT, NUM_FLOAT } basic;  // 基本类型
        struct {
            Type elem;
            int size;
        } array;                // 数组类型信息包括元素类型与数组大小构成
        FieldList structure;  // point to a struct definition
        FieldList member;     // struct definition
        struct {
            int argc;  // number of arguments
```

```
            FieldList argv;
            Type ret;  // return type
        } function;
    } u;
};
```

- `BASIC` 为基本类型定义，包含`INT,FLOAT`两种类型；
- `ARRAY` 为数组类型，以链表的方式将数组每一层的串起来；
- `STRUCTAG` 为结构体的定义，记录结构体中所有域的类型；

  在定义结构体时，可能会出现递归定义以及在结构体内层定义相同名字的结构体这两种情况。我的处理是在结构定义出现时就将其加入符号表，同时`struct_def_done`标记置为`0`，当内层出现该结构体定义的变量时，则得以判断其未定义，内层出现重名结构体，也可以判断其重复定义。
- `STRUCTURE` 为结构体变量的定义，其类型为结构体的定义，将结构体与结构体变量区分开，便于在语义分析中进行符号类型的判断；
- `FUNCTION` 为函数头的定义，记录参数个数、参数列表、返回值类型，统一存储，便于之后进行参数匹配和返回值类型匹配。

符号域

```
struct FieldList_ {
    char* name;      // 域的名字
    Type type;       // 域的类型
    FieldList tail;  // 下一个域
};
```

符号域包含符号的类型和符号的名字，结构体域以及参数列表都采用这种链表形式串起来。

哈希结点

```
struct HashNode_ {
    FieldList data;
    HashNode link;
};
```

出现哈希冲突时，将新加入的结点作为该散列项的链表头结点。

类型检查

按照手册所给出的错误类型，进行一一对应的类型检查。

结构等价

```
case STRUCTTAG:
    while (a_member != NULL || b_member != NULL) {
        if (a_member == NULL || b_member == NULL) return false;
        if (type_matched(a_member->type, b_member->type) == 0) return
false;
        a_member = a_member->tail;
        b_member = b_member->tail;
    }
```

对于结构体的类型匹配，采用结构等价的机制，即递归地匹配结构体各个域是否类型相同。

编译&测试方式

进入 `Code` 文件夹所在路径，执行 `make` 命令，即可获得 **parser** 可执行文件。

执行 `./parser filename` 命令，即可对一个待测试的源文件进行词法、语法、语义的分析。

实验感悟

本次实验是实现编译器的语义分析部分，涉及符号表的存储检索以及类型检查，实验难度并不大，实现的思路清晰明了，但符号表的设计、符号插入的时机、类型检查的具体逻辑顺序，均需要仔细地考量。同时，实验涉及较多的指针赋值操作，在实验调试时非法访问的情况时有发生，因此有效的断言和gdb调试工具的使用尤为重要。