# lab2 实验报告

本实验已完成所有的必做和选做内容

## 一、编译与测试命令

编译可以使用：

> make parser（或直接make）

通过makefile提供的方式可以对本实验项目进行编译，生成程序名为parser，此外我们写了一个简单的脚本用于对 `../Test/` 中的所有单元测试进行一键测试，使用

> ./test.sh

## 二、实现过程

相比于lab1，本次实验的内容主要体现在 `semantic.c` , `semantic.h` 两个文件中。当正确完成词法分析和语法分析之后，我们需要对语法树进行下一步的语义分析。语义分析的主体内容可以分为两部分：**构建符号表数据结构**和**遍历语法树进行语义分析**。

### 1、构建符号表数据结构

> 符号表用于记录程序中出现的所有的变量名，函数名，自定义类型名等符号名称。符号表上的操作主要包括：填表、查表和删表三种，当遇到程序中的说明或定义语句时就需要进行填表；当遇到符号使用（或在填表前）就需要先查表；而当结束子作用域时就需要删表。

- **符号类型**
  根据需要，我们把符号类型分为 `enum {BASIC,ARRAY,STRUCTURE,STRUCTVAR,FUNC} kind;` 这五种，分别表示基本变量类型，数组变量类型，结构体名类型，结构体变量类型，函数类型。
  前四种主要按讲义描述的方式实现，而对于函数类型，我们需要记录：参数个数，参数类型，返回类型，函数状态（声明/定义）。
  为了类型等价判断，我们实现了 `int equvilence(Type* t1,Type* t2);` 函数，可以对数组等价、结构体等价、函数等价进行综合判断，等价返回1；否则返回0。
- **数据结构**
  在我的实验中选取的数据结构是哈希表和链表（主要与作用域有关）相结合。
  - 哈希表中需要记录：

```
struct Element_{
    char* name;       //符号名
    Type* type;       //符号类型
    Element* next;
    int dep;          //符号的深度（作用域）
    int line;         //符号所在行
};
```

- 与哈希表相关的操作主要有：

```
Element* Search(char* ); 查找
void Insert(Element* );  插入
void Delete(Element* );  删除
```

为了实现作用域相关的内容，我们在哈希表之外还交叉使用了链表来记录属于同一作用域所有符号。由于每次插入总是将新符号插入到哈希表槽中的头部，所以查找总是返回离当前使用处最近的定义，这与作用域的影响效果正好匹配，而当离开子作用域时，只需遍历记录链表将想应符号删除。

```
#define CLEARSCOPE(f) \
  if(f!=NULL){      \
      FieldList* tmp=f; \
      while(tmp!=NULL){  \
          Element* e = Search(tmp->name);\
          Delete(e);    \
          tmp=tmp->next; \
      }\
  }


CompSt{
  ...
  FieldList* f = deflist(...);
  ...
  ...
  CLEARSCOPE(f) //清除作用域
}
```

### 2、遍历语法树进行语义分析

> 核心思想：使用继承属性和综合属性，自顶向下进行语义分析。对于每个递归语法单元分析函数，参数接受继承属性，返回值包含了综合属性。

具体而言，我们从根节点开始对整棵树进行向下遍历，通过识别其子结点的语法单元名字(为了尽量提高有效代码的复用性，采用逐步判断的方式，对于完全相同的产生式前缀，可以先进行分析，然后来到不同产生式的第一个不同的语法单元时再分类判断)来判断由哪个产生式归纳来的，从而分类进行分析。

- 对 `extdeflist` ， `extdef` 以及 `deflist` , `def` 这种，使用 `Fieldlist*` 作为返回值，从而将所有的定义语句定义的符号，通过一个链表记录下来。
- 在 `extdef` , `def` 内部，我们需要先分析 `Specifier` 得到定义的类型，然后将得到的 `Type*` ，作为继承属性传入后面的子结点分析。

## 三、几个问题

- 结构体内域名的作用域问题，在讲义上这一点并没有讲的很清楚，为了与其他类型的作用域统一，我门实现的时不同结构体内的域名（或与变量之间）可以重复。
- 由于不知道后续实验是否使用属性，所以在遍历过程中，我们暂时没有将继承属性和综合属性记录在每个节点结构体内。