编译原理实验四

——金鑫 121220307

一、 实现功能：

1) 完成必做要求，在中间代码的基础上生成目标代码

2) 可以出现结构体：
   可以出现结构体变量，但结构体变量不可以作为函数的参数。

3) 可以出现高维数组：
   可以出现高位数组，不过数组不能过作为函数的参数。
   高维数组计算地址采用递归计算的方式。

4) 数据结构
   因为考虑到实验的完整性，仍然保留了实验二的作用域的分析功能，所以原有符号表结构保持不变，此外将用到实验三的结构：
   实验三的数据结构：
   采用聊表实现中间代码的存储与优化。

```
typedef struct Operand_* Operand;
typedef struct InterCode_* InterCode;
```

   实验四的数据结构：

```
typedef struct Reg_* Reg;
typedef struct Var_* Var;
```

5) 其中 semantic.h 和 semantic.c 主要负责语法树的遍历以及语义分析。
   Table.h 和 table.c 主要负责符号表相关的操作，包括查询插入和删除。
   Intercode.h 和 intercode.c 主要负责中间代码的操作，包括插入删除和打印
   Objectcode.h 和 objectcode.c 主要负责目标代码的操作，包括翻译和打印

二、 编译命令

1) 1. 在 Code 文件目录下直接 make parser
2) 2. ./parser ../Test/1.cmm ../Test/1.s(1 可替换为文件名)
3) 3. 也可在 Lab 主目录下输入./parser ./Test/1.cmm ../Test/1.s