哈尔滨工业大学计算学部

《编译原理-中间代码生成》

实验报告

计算机科学与技术学院

|  |  |
| --- | --- |
| 课程名称： | 编译原理 |
| 实验名称： | 语义分析 |
| 指导教师： | [陈鄞](https://www.icourse163.org/u/6989061495) |
| 学生姓名： | 张中界 |
| 组 号： |  |
| 实验日期： | 2022.4.24 |
| 实验地点： |  |
| 实验成绩： |  |

一、程序功能

在词法分析、语法分析和语义分析程序的基础上，将C--源代码翻译为中间

代码（三地址代码形式）。

在本次实验中，我们对输人的C--语言源代码文件做如下假设：

1）假设1：不会出现注释、八进制或十六进制整型常数、浮点型常数或者变

量。

2）假设2：不会出现类型为结构体或高维数组（高于1维的数组）的变量。

3）假设3：任何函数参数都只能为简单变量，也就是说，结构体和数组都不

会作为参数传入函数中。

4）假设4：没有全局变量的使用，并且所有变量均不重名。

5）假设5：函数不会返回结构体或数组类型的值。

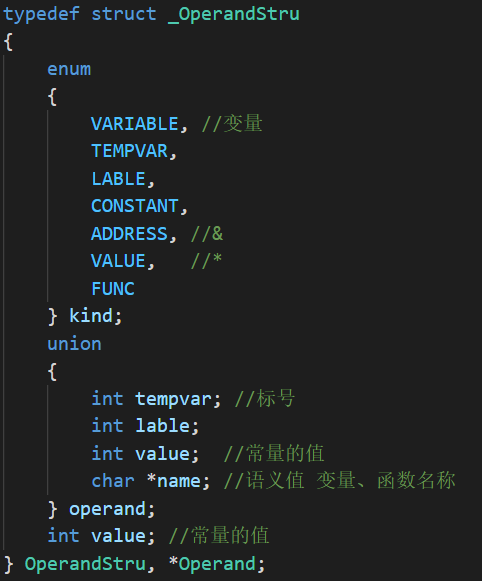
6）假设6：函数只会进行一次定义（没有函数声明）。

7）假设7，输人文件中不包含任何词法、语法或语义错误。

程序需要将符合以上假设的C-一源代码翻译为中间代码

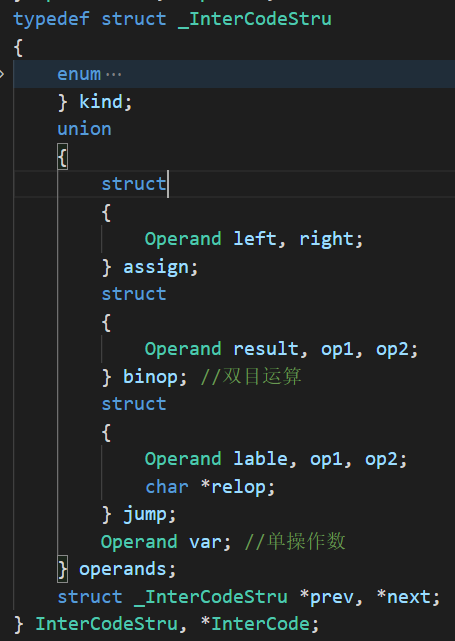
二、数据结构

1. 操作数



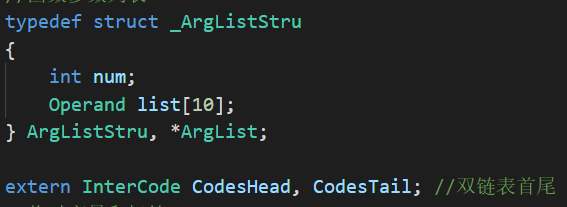
该结构体用于记录程序中出现的操作数，存储变量的语义值、函数名称、操作符等

2. 中间代码结构



kind记录中间代码的类型，operands记录操作数。

3. 函数参数列表

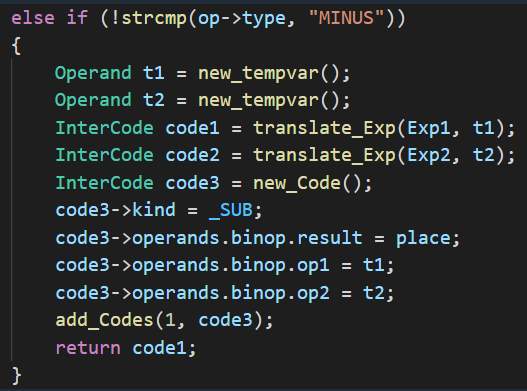


程序调用函数时用来记录传入的实参。

三、中间代码生成

**1. 基本表达式的翻译**

(1) 四则运算



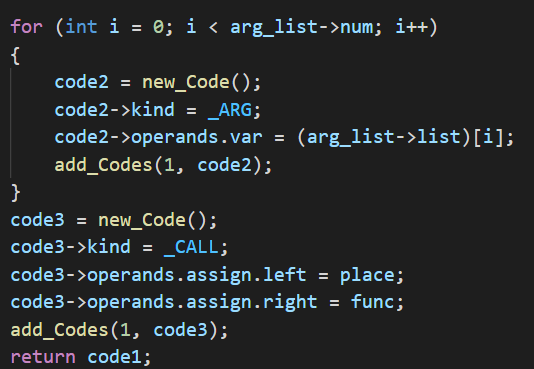
以减法为例，首先递归翻译减号两侧的两个表达式，结果存储在t1,t2两个临时变量中，然后生成一条中间代码，将实参place设置为运算结果，t1,t2分别为两个操作数；

对于取负操作，翻译为#0与Exp的减法；

(2) 函数调用

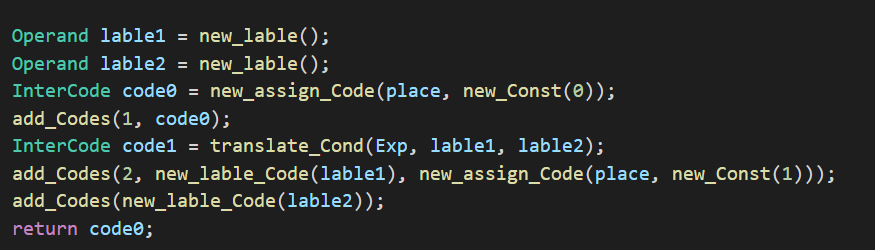


首先翻译实参，即构造ArgList结构体；



然后根据ArgList结构生成实参声明的代码，最后生成CALL代码；

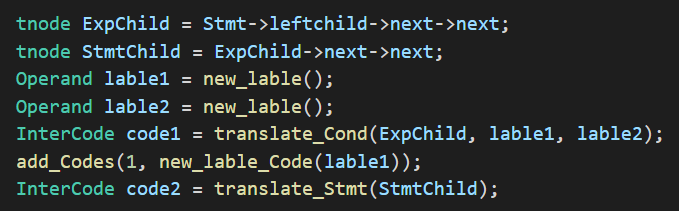
（3）条件表达式



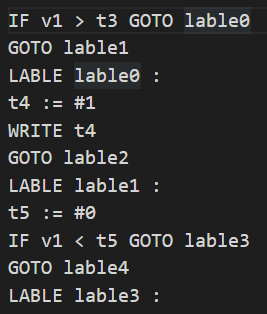
生成两个跳转label，一个用作条件成功和失败时的跳转，然后翻译条件判断后的代码块。

2. if-else语句的翻译

如果只有if没有else，那么遵从一般的翻译模式：

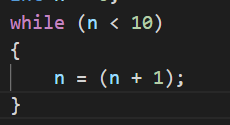


若有else,则需要再加一个label用作跳出if-else：

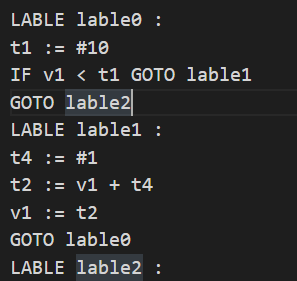


3. while语句的翻译

对于以下while语句：



对应的中间代码为：



需要三个label，label0用来记录条件判断的位置，label1用于执行循环体，label2跳出循环。

四、编译

make

./parser text