编译原理实验报告

目标代码生成

王梓轩 121220100

完成的功能点

在词法分析、语法分析、语义分析和中间代码程序的基础上，将 C--源代码翻译为目标代码。该程序将中间代码输出成线性结构，从而可以使用 SPIM 来测试目标代码的运行结果。

基本功能：读取 C—源文件，生成 MIPS 代码汇编文件。

测试样例 1、2 已通过。

编译及运行方法

依次输入：

```
make
./parser XXX1 XXX2
//在 SPIM 中使用该.s 文件
//make clean 可以删除编译过程中生成的一些文件
```

（XXX1 为测试用的 C--文件，XXX2 为生成的.m 文件，与 parser 在同一文件夹下）

实现方法

本实验的目的是实现目标代码，所以需要遍历一遍实验三生成的三地址代码链表。针对每个三地址结点，有相对应的处理方式，即考虑到该节点的语义，然后决定所生成目标代码的样式。

具体来说，将 MIPS 代码分成两部分：data 和 text 两部分，每个部分有一个链表。根据三地址代码的语义，判断其对应的 MIPS 代码类型，即属于 data 还是 text。遍历结束后，先输出 data 链表，然后输出".globl main"语句，接着输出 text 链表。

亮点

1. 寄存器分配策略：

采用的是朴素的寄存器分配方案，即：在需要操作数的时候将操作数加载到寄存器中进行运算，在指令运行完毕后将结果存回到栈或者内存（数组）中。

2. 利用数组模拟内存：

本实验中利用数组模拟内存的实现，即：将各个变量存放在数组中，在需要读取内存的时候从该数组中寻找，然后读出相应的数据。

3. 栈管理和函数调用：

栈用于保存临时变量，并且在函数调用过程中保存参数值、ra 寄存器旧值、fp 寄存器旧值及 sp 寄存器旧值，并据此确定 param 的起始位置。

具体的实现参考了 MIPS 代码的对栈的调用，还需注意栈的生长方向。

4. 遍历三地址代码：

通过三地址代码一一对应生成 MIPS 代码，避免了树状结构带来的冗杂，结构清晰。