实验三

141220145，张瑞琦，zhangruiqi2@qq.com

一、功能简述

1. 实现基本翻译功能
2. 实现要求3.1：可以出现结构体类型的变量，结构体变量可以作为函数的参数
3. 对中间代码进行了优化
    - 合并常量表达式
    - 优化控制流语句
    - 优化临时变量
    - 减少重复的Label
    - 优化代数式
    - 除去恒等式

二、编译方式

编译命令：make
测试运行：make test

三、功能实现

中间代码数据结构

采用双向链表的数据结构。链表结点类型为struct InterCodeNode，操作数的类型为Operand。数据结构定义在ir.h中。

中间代码翻译

遍历语法树，根据结点类型调用相应的transalte函数，向双向链表添加中间代码。这部分实现在translate2IR.c中。

结构体翻译

对结构体中域的赋值：得到域的地址x，*x = y

取结构体中域的值：得到域的地址x，y = *x
这部分关键要找到结构体中域的offset。这里我先使用函数get_id_list得到所有使用的变量名，比如student.friend.name 在id_list中表示为 “student” “friend” “name”。再调用函数getOffset遍历结构体的fieldlist，这样就可以快速找到域。

常量合并

遍历语法树（预处理），如果结点规则为add, sub, mul, div, and ,or ,not, assignop, relop，判断其左右结点是否为常数，是则根据结点运算类型进行计算，并删除子结点。

控制流语句的优化

如果GOTO跳转的语句为下一条，删除GOTO语句。对于IF语句，可以通过改变翻转条件，使得GOTO语句跳转到下一句，从而减少GOTO语句。

优化临时变量

在生成中间代码时，如果刚生成的一条语句为赋值语句，并且左操作数为临时变量，则可以删除临时变量，用其右操作数代替。比如：
t1 = v1
t2 = t1 + v2
可以优化为:
t2 = v1 + v2

减少重复的Label

对于这种类型的中间代码：
LABEL 2:　　和　　LABEL 2:
GOTO 3　　　　LABEL 3:

将所有跳转到LABEL2的语句改为跳转到LABEL3。这样处理后，对所有中间代码进行遍历，删除没有无条件和有条件跳转跳转到的LABEL语句。

优化代数式

加0变为赋值语句，乘1变为赋值语句等等。

去除恒等式

如果赋值号两边为同一个变量，则删除这条代码。这个可能是优化代数式遗留下来的语句。