- 学号: 221240093
- 姓名: 陈力峥
- 邮箱: 221240093@smail.nju.edu.cn

编译

编译采用 `Makefile` ，在 `Code` 目录下执行 `make` 即可，随后生成 `parser` 可执行文件。

`Code` 目录下运行 `parser` ： `./parser [input_file] [output_file]`

实验内容

完成实验三必做以及选做部分：

- `C--` 语言的翻译，包含：
  - 基本语句的翻译（表达式、控制流语句等）
  - 函数调用和参数传递
  - 数组和结构体的处理
  - 中间代码生成

实验细节

1. 主要实现文件
   - `translate.c/h` ：实现语法树到中间代码的转换
   - `command.c/h` ：定义中间代码的数据结构和操作
2. 部分细节解释
   - 中间代码设计
     - 采用四元组形式： `(op, arg1, arg2, result)`
     - 不同语句的参数规则：
       - 赋值语句： `(ASSIGN, src, NULL, dst)`
       - 二元运算： `(op, arg1, arg2, result)`
       - 条件跳转： `(COND_GOTO, arg1, arg2, label, relop)`
       - 函数调用： `(CALL, NULL, NULL, result)`
       - 参数传递： `(ARG, NULL, NULL, arg)`
   - 操作数设计
     - 类型区分：
       - `VAL` ：直接使用值
       - `ADDRESS` ：使用地址，用于STORE、DEREF、ADDR操作

    - 变量命名：
        - 程序变量：添加后缀 _ 避免与临时变量冲突
        - 临时变量：使用 tag 生成唯一标识
        - 标签：使用 tag 生成唯一标识

- 函数调用
    - 函数调用时，参数通过 ARG 操作传递，结果通过 CALL 操作返回
    - 对符号增加了 isParam 属性，用于区分参数和普通变量(因为数组和结构体作为参数时，传入的是地址，需要特殊标识)
- 语句优化
    - 例如在 Stmt 中，若当前为 RETURN 语句，则直接返回，后续不需要加入 GOTO/LABEL

```
if (stmt2->child != NULL && strcmp(stmt2->child->name, "RETURN") == 0) isElseRet
if (!isElseReturn)
    command gotoCmd = create_command(GOTO, NULL_OP, NULL_OP, label3, NULL_RELOP)
```

- 数组和结构体
    - 结构体访问
        - 通过 ADDR 获取结构体地址
        - 计算字段偏移量，使用 ADD 计算目标地址
        - 根据上下文决定是否需要 DEREF 获取值
    - 地址处理
        - ADDR ：获取变量地址
        - DEREF ：通过地址获取值
        - STORE ：将值存储到地址
        - 通过 operand_type 区分VAL和ADDRESS类型

```
// 结构体字段访问示例
// 如果structOp是VAL类型，需要先取地址
if (structOp->type == VAL) ...

int offset = calculateFieldOffset(members, fieldName);
// ADDRESS + CONSTANT = ADDRESS

command addCmd = create_command(ADD, structOp, offsetOp, temp, NULL_RELOP);

// 如果是赋值语句的左边，直接返回地址
if (exp->next != NULL && strcmp(exp->next->name, "ASSIGNOP") == 0) return temp;
...
// 否则需要取操作数
command derefCmd = create_command(DEREF, temp, NULL_OP, derefOp, NULL_RELOP);
```

- 数组访问
  - 递归处理多维数组
  - 计算元素大小和偏移量
  - 根据是否为左值决定是否 DEREF

```
// 数组访问示例
int elemSize = calculateTypeSize(arrayType->u.array.elem);
...
// 计算偏移量
...
command mulCmd = create_command(MUL, indexOp, offsetOp, temp1, NULL_RELOP);
...
// 计算最终地址
command addCmd = create_command(ADD, temp1, baseOp, temp2, NULL_RELOP);

// 检查是否在赋值语句的左边
bool isLValue = (exp->next != NULL && strcmp(exp->next->name, "ASSIGNOP") == 0);

// 如果是多维数组，递归处理
if (arrayType->u.array.elem->kind == ARRAY) { return temp2; }
else // 最后一维
    if (isLValue) { /* 在赋值左边，返回地址用于STORE */ }
    else { /* 在赋值右边，需要取数值 */ }
```