

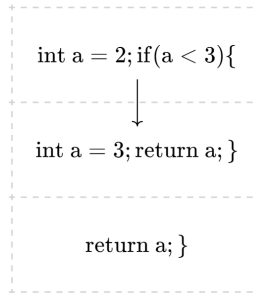
# STAGE3: 作用域和循环 实验报告

邢竞择 2020012890

## 1 Step 7

由于在前几个 stage 已经完成了对该功能的支持，所以不需添加任何代码即可通过。

### 1.1 思考题



## 2 Step8

### 2.1 实验内容

- 在 parser 阶段添加了 ForStmt, DoWhileStmt 的解析，在 ast 中定义了 ForStmt, DoWhileStmt, ContStmt 节点
- 在 SemPass1 中添加了 DoWhileStmt, ForStmt 的 visit 函数

```
1 void SemPass1::visit(ast::DoWhileStmt *s) {
2     s->loop_body->accept(this);
3     s->condition->accept(this);
4 }
5
6 void SemPass1::visit(ast::ForStmt *s) {
7     Scope *scope = new LocalScope();
8     s->ATTR(scope) = scope;
9     scopes->open(scope);
10    if (s->init != NULL)
11        s->init->accept(this);
12    if (s->condition != NULL)
13        s->condition->accept(this);
14    if (s->rear != NULL)
15        s->rear->accept(this);
16    s->loop_body->accept(this);
17
18    scopes->close();
19 }
```

- 在 SemPass2 中，添加了 ForStmt, DoWhileStmt 的 visit 函数，需对 cond 进行类型检查

- 在 translation 中，添加了 ForStmt, DoWhileStmt 的三地址码生成函数。For 与 DoWhile 不同于 While，需要三个标签才能完成跳转，以 for 为例

```

1 void Translation::visit(ast::ForStmt *s) {
2     // init...L1..cond..body..L3..rear..L2
3     Label L1 = tr->getNewLabel();
4     Label L2 = tr->getNewLabel();
5     Label L3 = tr->getNewLabel();
6
7     Label old_break = current_break_label;
8     current_break_label = L2;
9     Label old_continue = current_continue_label;
10    current_continue_label = L3;
11    loop_level++;
12
13    if (s->init != NULL)
14        s->init->accept(this);
15    tr->genMarkLabel(L1);
16    if (s->condition != NULL) {
17        s->condition->accept(this);
18        tr->genJumpOnZero(L2, s->condition->ATTR(val));
19    }
20    s->loop_body->accept(this);
21    tr->genMarkLabel(L3);
22    if (s->rear != NULL)
23        s->rear->accept(this);
24    tr->genJump(L1);
25    tr->genMarkLabel(L2);
26
27    current_break_label = old_break;
28    current_continue_label = old_continue;
29    loop_level--;
30 }

```

- 在 translation 中，添加了 current\_continue\_label，用来记录当前位置若出现和 continue 应当跳转到何处，添加了 loop\_level，用来检查 ContStmt 和 BreakStmt 是否出现在循环外

## 2.2 思考题

第二种更好。假设循环会进行多次，那么第二种每次循环只执行 body, cond, bnez，而第一种则需要执行 cond, beqz, body, br。第二种相当于把前一次的循环体和后一次的条件接在一起了，所以比较节省，但生成的程序会更长。