Step 8 Report

刘轩奇 2018011025

2020年10月6日

1 工作内容

本人选择不使用辅助工具 ANTLR 因此自己实现了 lexer 和 parser。

1.1 文件说明

`montLexer.h/cpp` 词法分析器；

`montParser.h/cpp` 语法分析器；

`montConceiver.h/cpp` 产生中间代码；

`montAssembler.h/cpp` 从中间代码产生汇编代码；

`montLog.h` 记录编译错误信息；

`montCompiler.cpp` MiniDecaf 编译器，包含主函数，编译成功返回 0 否则返回 -1，并将错误信息输出到 `std::cerr`。

1.2 本步骤完成的工作

1 词法分析 添加了新的 Token：

* `FOR` 关键字 for；
* `WHILE` 关键字 while；
* `DO` 关键字 do；
* `BREAK` 关键字 break；
* `CONTINUE` 关键字 continue。

2 句法分析 添加或修改了以下的 AST 节点：

* `statement` 现在可以生成 `for`, `while`, `do-while` 循环对应的节点，以及 `continue`, `break` 语句对应的节点。
* `while` 此类型节点包括 `while` 和 `do-while` 两种循环类型。
* `for` 此类型节点表示 `for` 循环。
* `empty` 空节点。这种节点是为了提供一个占位，因为 `for` 循环的三个表达式可能为空。

3 产生中间代码 根据指导书产生相应的中间代码即可。其中需要注意的是 `for` 循环中三个表达式若非空则还应当生成对应的 `POP` 代码。

4 产生汇编代码 没有改变。

2 思考题

1 while 循环的两种翻译方式哪一种更好？

答 第二种更好。假设循环体恰好执行且仅执行一次，则第一种翻译方式执行了两次跳转，第二种翻译方式没有执行跳转。