编译原理 Lab 7 实验报告

学号：201250185

姓名：王福森

我实现的功能

本次实验我通过 antlr 工具和 llvm工具，在 lab 6 的基础上，扩充了 SysY 语言的 IR 生成程序 FunctionAndVarIRVisitor.java，实现了全局变量、常量，全局变量数组、常量数组的声明、定义和初始化，实现了 if 语句的翻译，实现了 while 语句的翻译。

我是这么实现的

本次实验直接沿用 Lab 1 到 Lab 6 所写的 SysYLexer.g4、 SysYParser.g4 、TypeCheckListener.java、FunctionAndVarIRVisitor.java 文件，并且将 TypeCheckListener.java 内容整合到 FunctionAndVarIRVisitor.java 中。

在 main 函数中，我遍历一遍语法分析树，在生成符号表的同时，生成符合要求的 IR。其中，构建基本的符号表我沿用的是 lab 3 的 TypeCheckListener，并将其整合入 FunctionAndVarIRVisitor 中；为了生成符合要求的 IR， 我设计了一个继承 SysYParserBaseVisitor<LLVMValueRef> 的 FunctionAndVarIRVisitor，在遍历语法分析树的过程中构建逐步向符号表中各个符号填充 LLVMValueRef 字段，同时向 module 中加入函数、基本块和指令，用以生成符合文档要求格式的输入文件的 IR。

本次处理的核心是全局常量的定义与初始化、全局变量的定义与初始化、if 语句、while 语句和条件表达式的翻译。对于各个语句，只需要按照 sysY 语言的语义生成对应的指令即可。

我的精巧设计

这次 Lab 暴露出之前设计的问题，导致我只能将符号表构建和 IR 生成同步进行，同时为了解决 “error: instruction expected to be numbered”，我把所有 build 指令对应的 name 都设置成 “*”。

过程中有趣现象和印象很深的 bug

这次 Lab 我遇到一个几乎无法发掘的 bug，就是 normaltest10 和 hardtest1 的 core dump 问题。

我试了很多用例，反复看实验要求，每个部分单独测试，结果都是正常运行，但是提交上去还是 core dump。不得已之下，我寻求助教gg 的帮助，结果发现是各个模块联动产生的 bug。

我前几次 lab 都是遍历两次，第一次生成符号表，第二次生成 IR，这就导致如果同名变量声明在父子作用域，并且在两次声明之间使用到了改变量，我就会解析到子作用域中的变量，而不是正确的父作用域的变量，但是这个时候我还没有遍历到子作用域中变量的定义，没有为其设置对应的 LLVMValueRef 指针，这个时候这个指针还是 null，在我使用的时候用 LLVMBuildLoad 自然就 core dump了。

到现在我也觉得这个 bug 十分出人意料，感谢助教 gg ♥♥♥！！！