编译原理lab6

1. 项目结构

本次实验项目结构较上次实验基本没有变化。主要还是在LLVMvisitor类上进行函数的重载。

1. 实验思路

1、对全局变量进行处理：

首先在visitVarDecl中，先判断当前作用域是否是global，之后再通过对constExpContext是否为null的判断，来判断是否为数组。之后创建数组操作和局部变量差别不大，只不过使用LLVMAddGlobal来建立LLVMValueRef。全局数组就算没有初始化也要将所有值都置为0，由于没有builder，故不能使用GEP指令，只能建立一个LLVMConstArray存放数组初始化的值，之后再通过LLVMSetInitializer将值传入数组进行初始化。

全局变量则更为简单，直接使用LLVMAddGlobal来建立LLVMValueRef，之后判断是否要初始化，需要则使用LLVMSetInitializer进行初始化。

上述全局变量和数组也都要再currentScope中进行define。

1. 对if else语句处理：

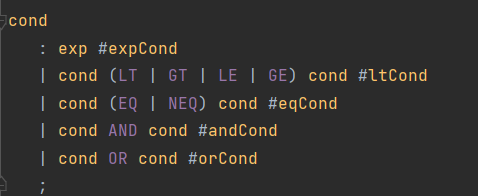
首先新创建三个block，分别为true，false，entry，分别代表if{}, else{]以及之后的函数的其他部分。

首先要对if(cond)中的cond部分进行判断，即visit(cond)，若结果为i1则LLVMBuildCondBr

进行跳转，否则(为i32，即cond部分是exp)则通过LLVMBuildICmp (builder, LLVMIntNE, condRe, zero,"");判断是否为0，将结果传入LLVMBuildCondBr进行跳转。

之后按顺序使用LLVMPositionBuilderAtEnd将visit(stmt(0))和visit(stmt(1))得到的IR语句分别加到true和false block中，并使得二者最后都通过LLVMBuildCondBr跳转到entry block中。

之后则是对cond语句的判断的函数的重载，主要重载的函数都是下面这些语句相关



ExpCond不需要重载

LtCond和EqCond需要使用LLVMBuildICmp，且为了防止出现a<b<c这种，在通过visit得到两边的操作数后要先对其进行判断，类型是否是i1.若是，则通过LLVMBuildZExt转为i32再参与处理。

And和or本次实验并未考察，但仍进行了相应处理，不再赘述。

1. 遇见的问题
2. 关于大小比较，我的第一反应是从LLVMValue中得到int值再进行比较，但关于全局变量和数组使用LLVMConstIntGetSExtValue得到的int值是乱码，最后在仔细阅读LLVM文档后以及在lli编译帮助下使用LLVM相应api解决。
3. LLVMBuildIcmp按理来说返回的应该是个i1类的LLVMConstInt，故我定义了两个分别为0和1的i1类的LLVMConstInt，来和LLVMBuildIcmp返回值通过.equal进行比较，但结果都是false。最后则使用其他方法，规避了对LLVMBuildIcmp返回值究竟是true还是false的判断。