编译实验之LAB2：常量表达式

18375123 朱穆清

本次实验算是才真正的开始了编译器的编写。之前的pre和lab1搭建好了语法分析器和基础的程序框架，这次的lab2就要开始写汇编语句了。

这次实验的语法分析部分不难，只需要在antlr上做一些修改，增加addexp/mulexp/unaryexp等表达式的支持即可。我在g4语法文件中使用了 `op=(MUL|DIV|MOD)` 这样的语法表达式，方便在遍历AST时通过 `ctx.op` 对运算符进行判断。

接下来就是整体的设计思路。我的想法是在遍历AST的同时直接生成出IR代码，即遍历结束的同时生成也结束。因为树的遍历顺序是符合编译器语法解析顺序的，所以这种方法是可行的。我设置每个Visitor节点的函数返回值为String，以便于上下层次间的信息传递，例如子节点分析完成后返回数值或寄存器序号给父节点，父节点再生成对应的IR代码。

通过利用ANTLR生成的AST框架，我很方便的完成了整个AST的编写，在其中几个关键节点例如 `visitMulExp/visitAddExp` 等重点编写了如何输出IR代码：首先访问表达式的左子节点和右子节点，分别得到他们的返回值（即数值或寄存器序号，这个寄存器中保存了子节点的运算结果），然后根据操作符的不同（ADD/MUL/SUB等），将结果保存在一个新的寄存器中，并输出对应的IR代码即可，例如乘法的IR代码：`System.out.println("    " + reg + " = mul i32 " + a + ", " + b);`

其他的几个运算也类似。都实现之后就可通过本次实验的测试了！