# 第二次实验报告

好像也没说要写，但是想写就写了！本人是ddl狂战士，开始写lab2是ddl前24h的那个凌晨。从开始到写完大概花了4h。来讲讲在这个深夜里发生的神奇的事情。

## 主要内容

就我个人而言，我认为本次实验的难点主要在以下几点：

- g4文件的编写。SysYParser 对着手册写，看上去简单，但稍不留神就可能写错/认错符号等。本人在一开始就注意到了 *EQ* 和 *ASSIGN* 两个会让我一不小心混淆。于是很得意地写完了。写完之后测了一下错误处理，发现诶不对啊，还是错的。一检查发现，我还有个 *EQ* 写错了，感觉自己像个小丑.....
- 另外耗时比较久的是去理解visitor的行为。尝试犯了一下visitor父类、父类的父类...的代码，发现翻到后面 *visit(Parser tree)这个方法啥都没写。于是先打印了一下visit* 和 *visitChildren* 的相关信息，观察对某个 *Parser tree* 调用后的表现。分析后就知道了我应该去逐层遍历一下tree，调用 *visit*。那后面也很简单了，去找 *Node* 给相关信息的接口，重新组织一下信息，再注意一些小细节，就完成了。

## 一点反思

- lab1到现在的ide相关的历史遗留问题还没解决。正常编译，我们其实是不需要 *package src* 的，因为这些文件本就放在同一个目录下。但是如果不这么做，vscode它无法从同一目录去读其他文件，这就导致我的自动补全功能、或者找一些接口的时候会变得比较麻烦。应该就是vscode的问题，其他的ide没试过？好在现在作业还比较简单，不知道以后会咋样，后边得想想办法。
- 感觉自己的 *Visitor* 实现得并不是很好。在 *visitTerminal* 里面，我直接获取类型编号，然后看了一下我生成的 *SysYParser.java*,知道了哪种类型的范围是多少，值是多少，再去分类讨论。这样子做总感觉有点打破抽象层了，可维护性也不强。假如说我突然想稍微改一下我的g4，再生成一份，这里马上就不适用了。这是感觉写的不太好的地方。