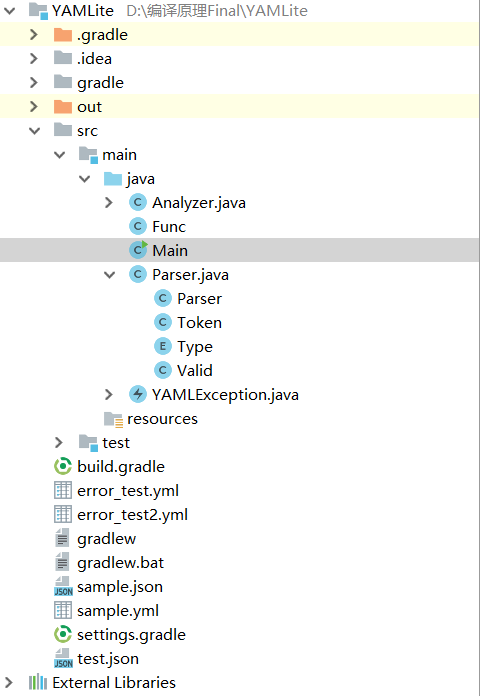
# 编译原理实验报告

### 综述：

本次实验对我而言难度比较大，无论是对于基本的分词和语法分析器的构造，还是对于YAML这种缩进文法的处理，对我都是一个很大的挑战。最终完成的项目在检测语法语义上面表现比较稳定，但对于查询函数多层嵌套的处理，yaml转json等的实现比较勉强，效果也不尽如人意。

代码包含Parser.java,Analyser.java,Func.java,YAMLException.java四个功能类和程序的入口Main.java。使用的是gradle

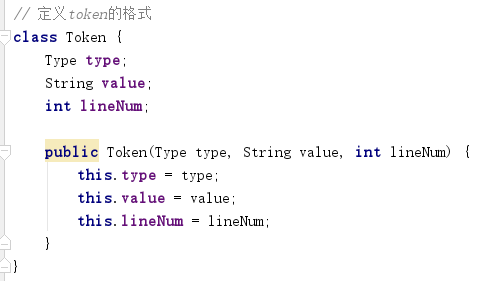
程序目录结构：



1. **分词器的构造实现**

该部分代码集中在Parser.java类中，实现思路如下。

定义Token类储存token信息，token类中包含类型，值和所在的行这三类信息，其中类型分为9种，分别为Identiier, Operator, ChangeLine, Space, Int, Double, ScientificNotation, Str, Bool，因为是缩进问法，所以空格和换行信息也要保存。



分词过程中，将文件内容一个字符一个字符读入，以空格和换行作为分割的标志，然后判断读入且未处理的字符是标识符，操作符还是数字。例如，如果以”开头，则可以进入判断是否为字符串的函数处理，如果以数字开头，则去检测这个数字表达是否合法，并判断是整数，浮点数还是科学计数法。在分词的过程中，可以检测到token表达不合法的错误。

将所有token保存到tokenList中，便于下一步进行语法分析。

在进行语法分析前将多余的空格（比如注释前的空格），注释和连续换行（只保留一个）等对于分析无用的信息清理掉。

1. **语法和语义分析器的构造实现**

按照比较规范的做法，应当构造分析表，并按照分析表对应的分析方法进行代码的书写，但是这是一个缩进文法（1型文法），也没有相应的产生式，实现起来困难重重，于是便采用了比较原始的按行分析的方法。

首先，定义一个ValueStore类进行存值，在这个类中还要保留其缩进的层数。读取一个token便将其处理放入队列中储存，在进行enqueue操作时，要检查这个元素的进入是否合法，我定义了如下规则：

1. 上一个元素缩进为0且为键值对结构，则下一个元素缩进必定为0。
2. 下一个元素的缩进不得比上一个元素多两层以上或少两层以上，上一个部分定义完毕返回缩进为0的情况除外。
3. 同一级的数组元素缩进相同。
4. 在数组结束之前，不得有元素的缩进小于等于数据起始的缩进。
5. 缩进层数必须为整数。



1. **功能的实现**

**YAML转json**

对json语法了解有限，可能很多地方不是很完善。根据储存的值，做如下解析：

1. 所有identifier位于双引号之间。
2. 键与值之间以冒号隔开。
3. 字符串位于双引号之间。
4. 数字直接将值输出即可。
5. 检测到数组则将数组以对应的格式输出。
6. 数组结束的标志是返回起始的缩进。
7. 在数组内部的键值对要用{}括起来。



**查询功能**

这个的实现是最勉强的，只能支持一层嵌套查询，而且不能返回数组内容。（主要原因：没有找到合适的存储结构存值）

实现方法：

首先判断是否有访问数组元素的情况，如果有，记录这个索引，并记录要访问的键的名称。

在存储的值中寻找同名的identifier，如果不是数组结构，则直接返回这个对应的值，如果是数组结构，则根据counter遍历数组元素，找到对应的索引后，如果是元素，则直接返回，如果是键值对，则再检查键名是否相同。

存在的问题在于对于多重键值对的处理，仅能判断语法问题但不能对于其值进行访问，也无法生成json语句。

报错机制：

定义了YAMLiteException这个继承自Exception的类来处理报错。一共定义了十种错误，对应着在词法分析和语法分析中的错误。对于出现在词法分析的错误可以给出具体在哪一行哪一个位置，在语法分析中，因为相关的字符都转化成了token，因此在报错时只报哪一行存在问题。关于报错方面存在的问题是，最开始考虑报错机制的时候，想到的是一旦发生错误就马上退出终止错误，但后期在改进的时候考虑到catch exception之后可以继续运行，直到全部完成，这样可以把所有错误报完。但是在实际操作中，由于在前期代码中缺乏对报错后继续运行的支持，所以如果文件中存在多处错误，很难全部报错。

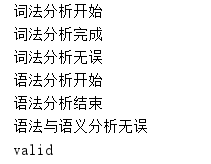


结果演示：

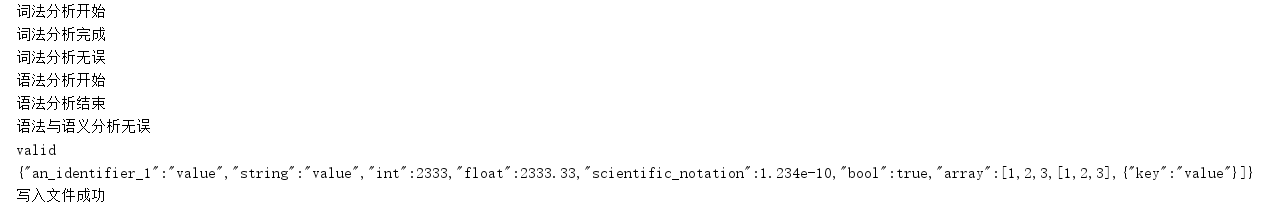
sample.yml



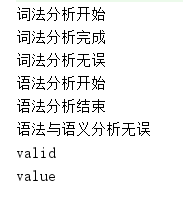
java -jar YAMLite.jar sample.yml



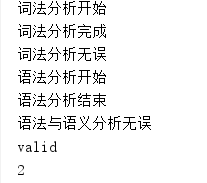
java -jar YAMLite.jar -json sample.yml



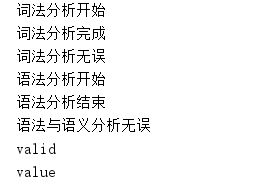
java -jar YMLite.jar -find string sample.yml



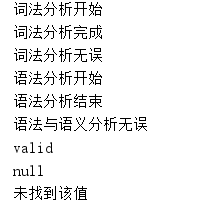
java -jar YAMLite.jar -find array[1] sample.yml



java -jar YAMLite.jar -find array[5].key sample.yml

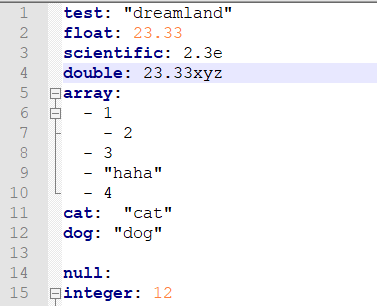


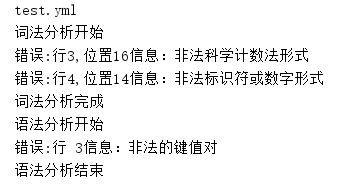
java -jar YAMLite.jar -find notexist sample.yml

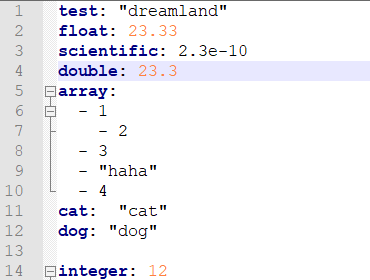


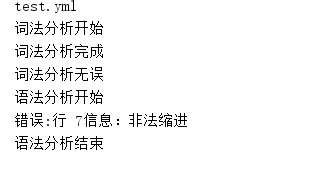
错误处理：

test.yml



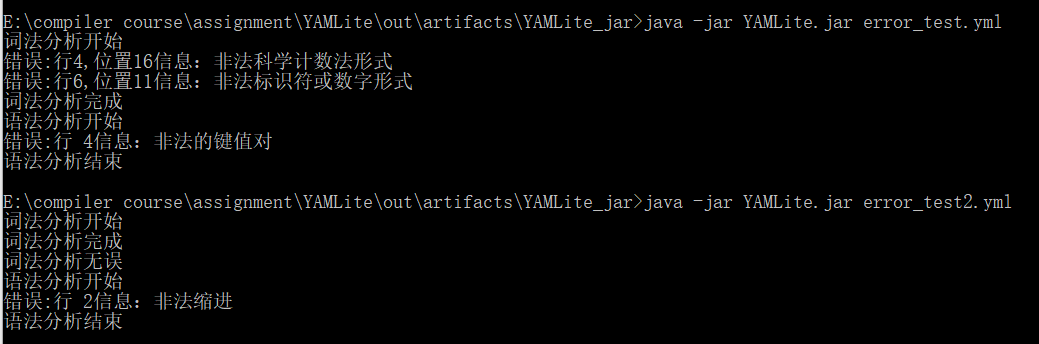






error\_test.yml

error\_test.yml



### 总结

这次作业前前后后大致花了26个小时的时间做，总结起来还是自己太菜，代码写的太少。首先在整个工程上缺乏相应的规划，导致代码缺乏统一的规范，使得我在做的过程中出现了大量的反复。而且在后期我想改进的时候，往往这里代码改一点会导致其他地方出现更多的bugs，最后我的各个方面的实现都非常勉强。

在今后的学习过程中，我要更加重视代码的书写，提升自己的代码能力，并提高数据结构等方面的能力，写出更高质量的代码。