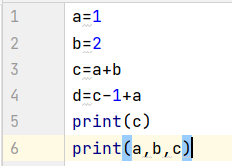
**输入输出说明：**

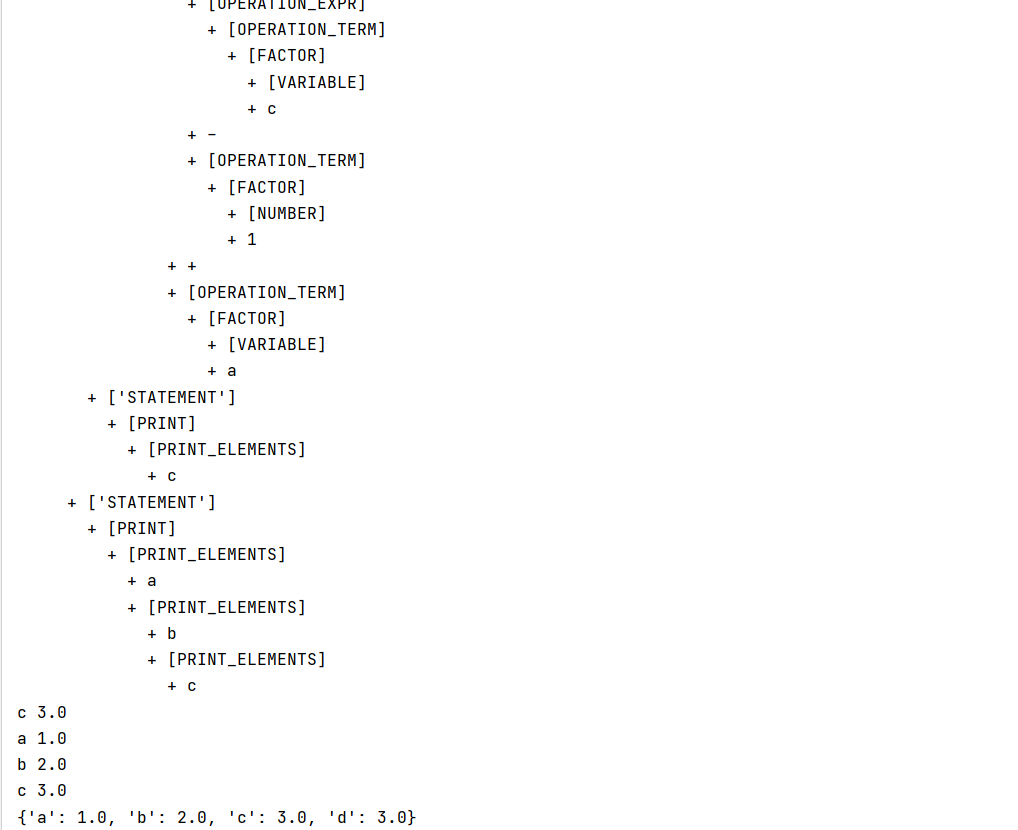
本程序的运行需要在**example.py文件**中输入需要解析的表达式，（example.py文件需放在和所有程序放在一个目录）执行本程序则只需要运行**main.py文件即可**。

**解析样例如下：**



**程序输出结果如下：**



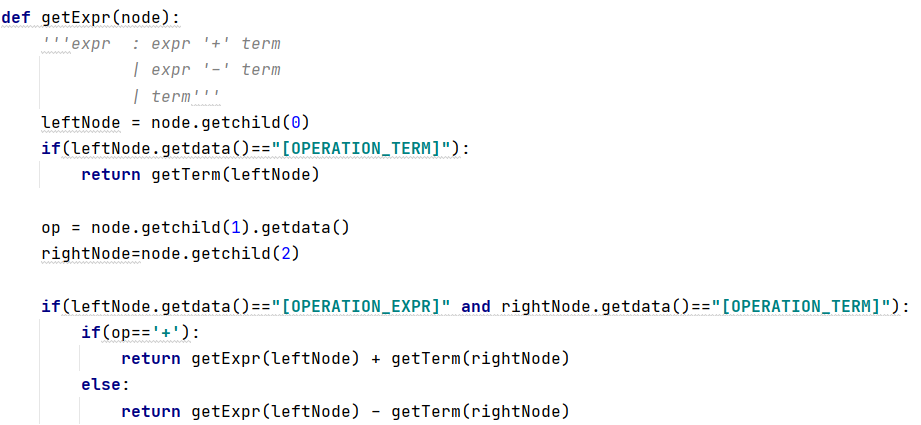


**设计思路：**

本次实验的难点主要在于遍历生成语法树，所以着重分析translation.py中的方法：

getExpr() getTerm()分别负责加减部分和乘除部分，使用递归

首先是getExpr()部分

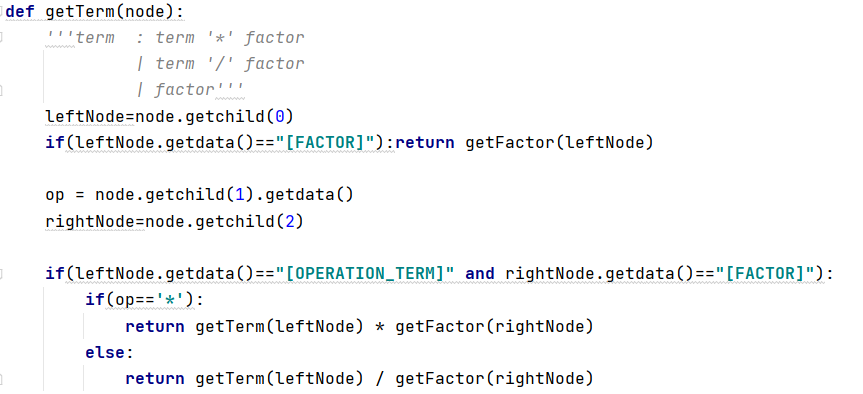


首先判断是否为文法的第三种情况 是的话直接返回值

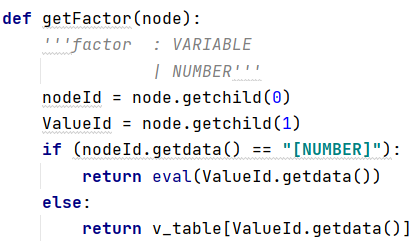
若不是则获取二三节点的值

根据运算符不同 进行递归调用

getTerm部分也同理 根据乘除号进行递归调用 或者直接返回值

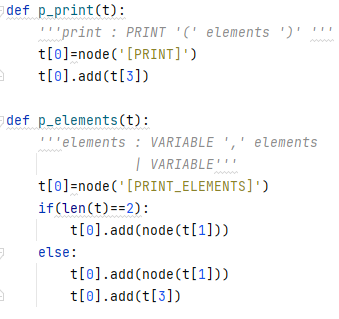


getFactor部分

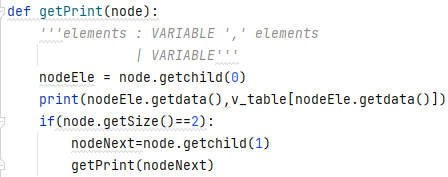


根据标签判断是常数还是变量 是变量则去变量表中获取

多变量PRINT函数部分



更新了文法 可以在print中递归的容纳多个元素



翻译部分也是一个递归调用函数。

如果判断当前节点有2个子节点，则打印节点并进行一次递归调用

否则直接打印数据。此种方法可以按照命令顺序自上而下正确打印。而非倒序调用。