# 第一次作业：

第一次作业是单层括号的展开，总体来说难度重点在于如何进行表达式的解析。对于表达式我的处理是首先将单层的（）\*\*形式进行展开，然后再分析表达式、建树、求值。对于建模部分我利用两个数确定每个“基本”的因子，一个是系数一个是指数（现在看来这样写并不利于后续扩展），同时普通的因子也可以看作是一个表达式。我忽略了项那一步，认为最后的表达式是若干因子相加构成的。表达式是一个ArrayList，它里面存的就是我的因子类，然后相邻两项利用加法连接。所以总体而言在建立完成表达式树之后一切都变得相对简单。对于精度问题我直接选择BigInteger。

在bug方面由于作业要求较为清晰，情况也没那么多，所以我没有被找到bug。但是找到了一个bug，那位同学的连+-号存在着问题。

# 第二次作业

第二次作业加入了三角因子与自定义函数，我对自己的因子进行了扩展，新增了SinCos类归属于普通因子内，对于一个普通的因子我定义成了a\*x\*（若干sin、cos相乘），同时在建树设立节点是增加了对sin、cos、函数的分析。对于函数我只需要再将他的替换结果再次送入分析类中即可获得他的表达式类，在这点上我的架构为我提供了不错的便利，但是在因子的扩展以及化简方面我的架构就十分的不友好。

在bug方面主要是如果调用的自定义函数内部含有三角函数的话，我的sin里会有两曾括号从而导致我的正则表达式提取不出有效信息。而这个点直接让我的强测一片惨淡。这也告诉了我面对一下规则较为复杂而且很有可能存在我想不到的情况是尽量慎用正则表达式。其次在互测内大家好像都是因为自定义函数处出现的错误，所以互测方面彼此并没有发现bug。

# 第三次作业

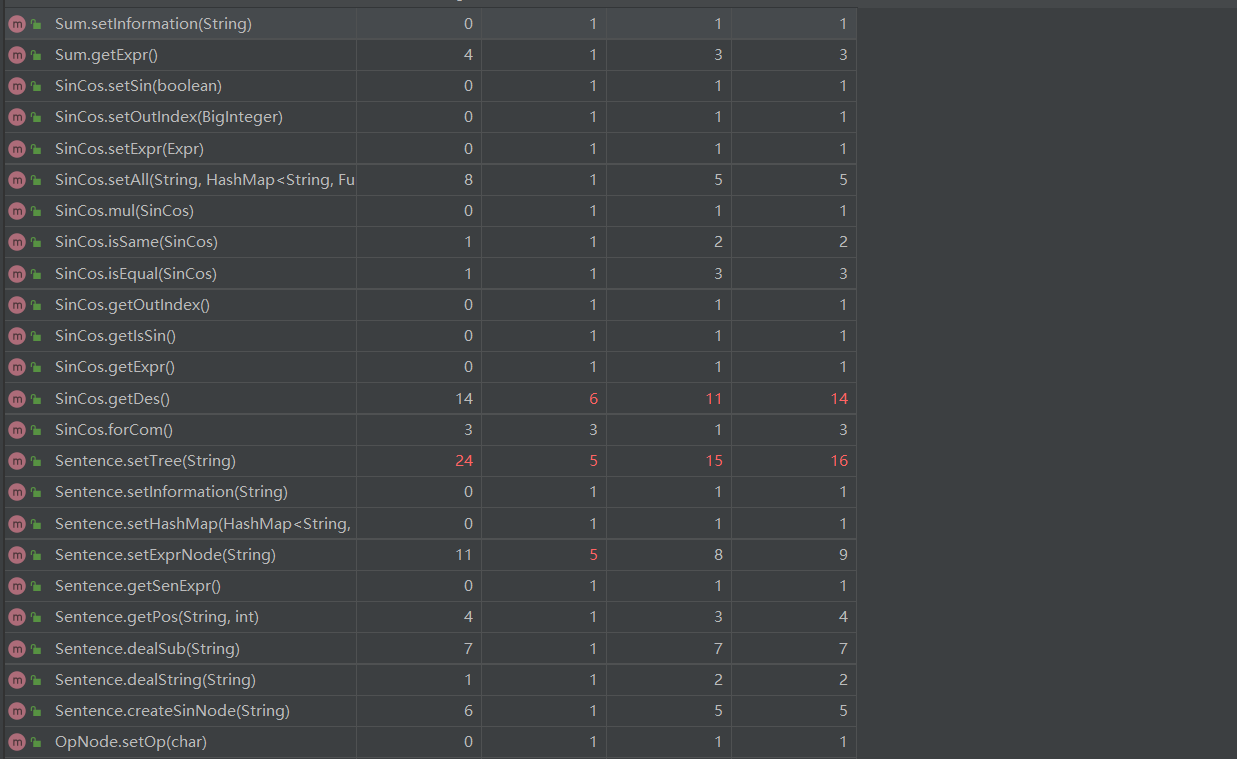
第三次作业无非就是在自定义函数和三角函数内部可以实现多层嵌套。我原先的架构只需要将我的（）\*\*放到最后设立节点时解析、在SinCos内部换为一个表达式类、重写判断化简条件就完成了要求。只花了3个小时就结束了战斗。

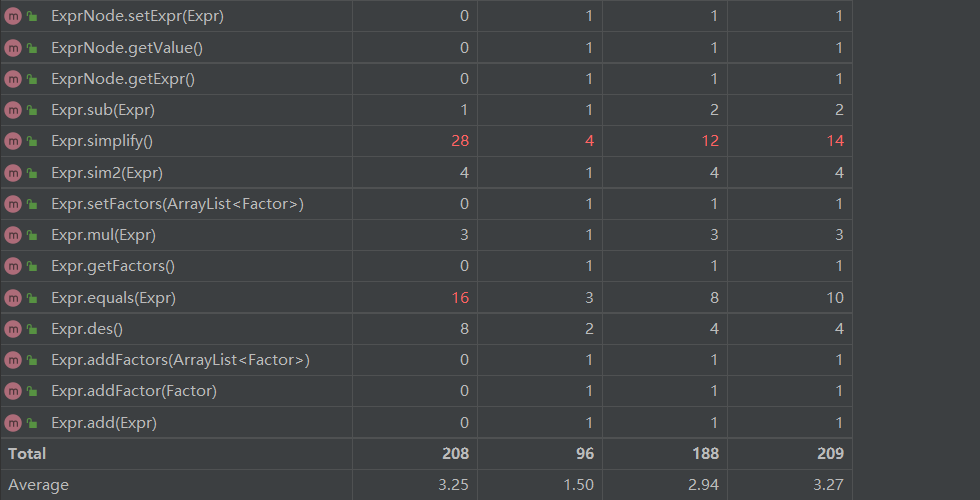
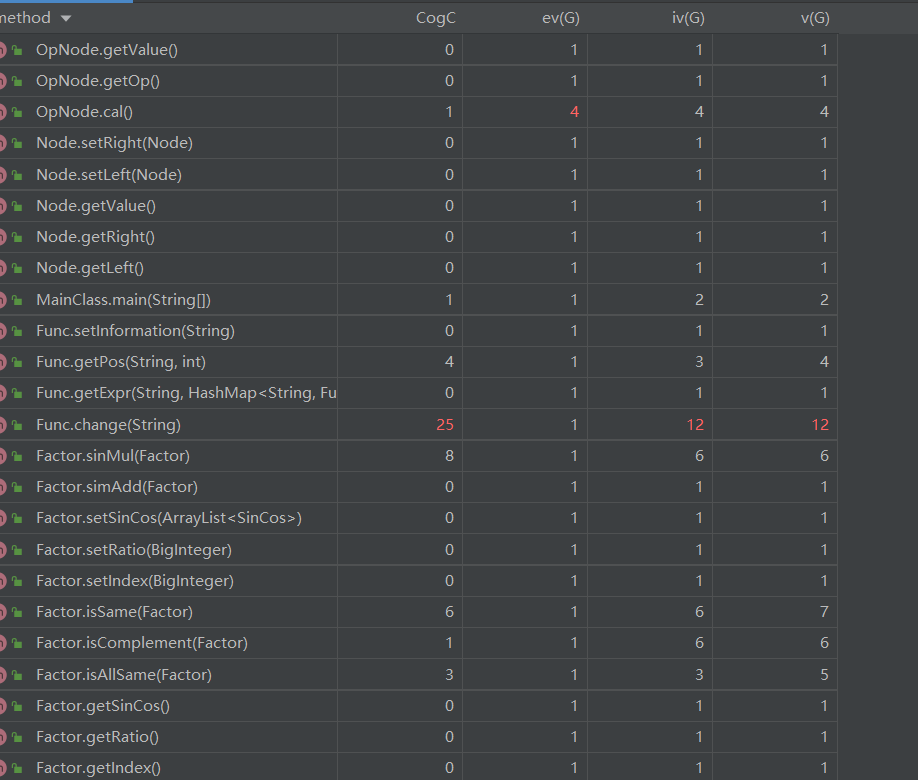
值得注意的是，我在对函数进行取值的时候采用的是利用逗号将三个自定义函数引用的变元提取出来进行运算，但是在这次作业中，我需要在遇到括号时直接跳到对应的括号处。其次，还有在对输出的化简上，sin\*\*2+cos\*\*2需要比较内部的表达式是否相等，在这里我需要重写expr的equals方法。

在bug方面，由于第二次作业时保证不会出现sum(i\*\*2)的情况而第三次却又可能出现，但是我只注意了新增功能是否正确没有管原先要求的变化，导致了这个新bug的产生。在互测时，我输入了sin(0)\*\*0成功抓到了一个人，但是我sum的这个bug被许多人抓到了。值得我说的是，我这一个点被几个人反复hack，着实是搞人心态，最后我甚至被别人hack了10次。不过最好的方式是自己没有bug，最好还是自己要多做测试啊。。。。

# 架构分析

由于我是一条路走到黑，我这里就直接说我最终版的架构吧。我的架构并没有太多的继承、泛化，





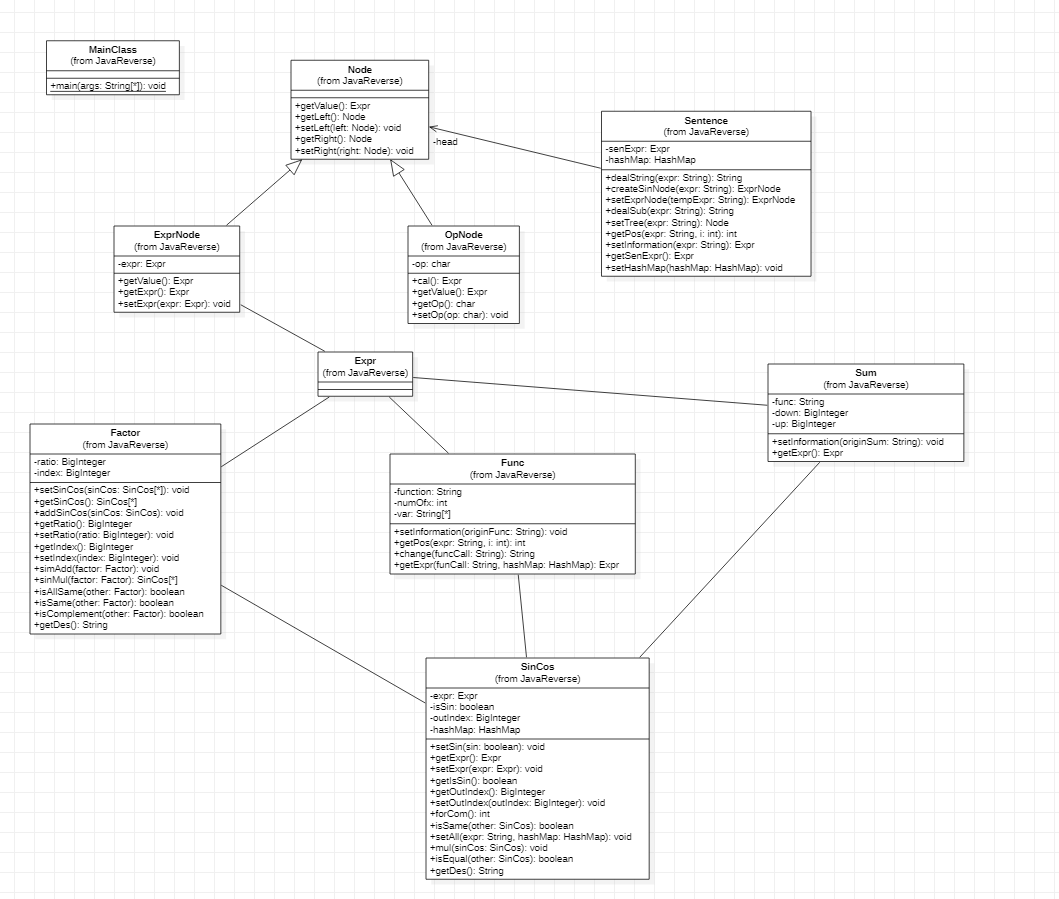
我的部分方法复杂度太高，比如simplify函数由于我用ArrayList不好化简又怕内存的克隆出错，只好运用循环。

下面是我的类的耦合情况：



、可见耦合度较高，这也是我之后要提升的。

下面是我的uml图，由于没有有效挖掘出他们之间的关系，于是在下列图例中我采用了Assoication；



我的主要架构一般就是一个类的某些属性就是另一个类，主要优点就是总体上看比较清晰易懂，但是代价就是在改动或者调试时十分困难，需要一层一层找到最下层。

其中Sentence类是我的字符串解析类，内设一个senExpr用来储存我的表达式解析结果。

而expr、factor、Sincos、func和sum则是我为了解析储存“单位因子”而建立的类。

至于opNode、Node和ExprNode是我为了建树而设立的类。

# 体会与反思

有关于这几次出现的bug，我认为主要有以下几点原因。

首先是不同作业的要求会发生改变，我却没有及时关注到(sum(i\*\*2)),之后我一定要经常注意这个问题。

其次是尽量少用一些特意化的正则表达式，一旦需求发生改变或者有某些情况发生改变，那么很容易出现bug。

这些bug都是一些小bug，都是我没有考虑到某些情况，这也是我不注重去特殊化的后果。在测试别人bug时，我主要是根据要求构造一些边界的BigInterger以及sin(0)\*\*0这类数据进行性测试。

总而言之，这单元作业让我感受最深的是架构的重要性，一个好的架构不仅可以容易扩展，更可以减少bug的产生。其次，就是要减少编程时对于问题的特殊要求。当然还要注意高内聚低耦合的特点。下一单元的作业，我一定要着重注意这些问题。