**C++程序设计（精品课程）大型实验**

**实验题目：浮点及大整数加减乘除模计算器**

**姓名：叶璇**

**学号：201806060825**

**班级：健行1802班**

**时间：2019年6月10日**

第一章 计算器描述

1. **计算器语法**
2. 计算器所做的运算是处理一些long double 类型的数据表示的表达式。包括加、减、乘、除、等号和括号操作。遵守一般的优先级顺序，括号优先于乘除、乘除优先于加减。
3. 由于具有优先级，我们可以使用递归来实现。递归文法描述如下：

Program: --------------------- 程序

END

exprList END

exprList: --------------------- 表达式清单

expression END

expression END exprList

expression: --------------------- 表达式

expression + term

expression – term

term

term: --------------------- 因式项

term / primary

term \* primary

primary: --------------------- 初等项

NUMBER --------------------- 浮点数

NAME --------------------- 名字

NAME = expression

* Primary

(expression)

1. **计算器制作要求**
2. 计算器能够进行语法识别，NUMBER按照long double 的语法接受输入，NAME是变量名，受到系统的限制。END表示当前表达式到此结束，以回车表示。
3. 当计算器接收输入的数据时，能够按照事先写好的算法将结果计算到底，同时输出一个具有6位小数精度的浮点数。同时，能够做出错误反应。错误类型在具体程序中给出。
4. **测试数据制作及代码**
5. 实验测试输入数据在各个实验清单in.txt文件中。
6. 实验测试输出数据在各个实验清单out.txt文件中。
7. 实验程序全部的源代码在各个文件夹程序清单中。

第二章 过程化方法

1. **过程化框架结构设计**
2. 总控制设计：主函数main 不断从in.txt文件中读取数据，然后经过判断，计算，将计算结果写入out.txt文件中，直到数据全部处理完毕。为了精确到6位小数，有预先处理“cout<<fixed;”语句。实验详细代码在实验清单中。
3. 词法处理设计：计算器应该会处理读取的每一个单词，然后判断下一步过程。Token.h和Token.cpp实现了此过程，源代码在清单中。
4. 语法分析处理：计算器在词法分析处理的基础上，进一步对语法进行分析，程序中将expression、term、primary三个语法单位分别实现。
5. 异常处理设计：计算器需要对输入的数据进行判断，判断是否符合语法要求，当不符合语法要求时，能够抛出异常进行下一步的处理。详细代码在清单中。
6. **代码运行测试过程**
7. 过程化方法比较简单，但是实验过程中同样遇到一些问题。在编译完成后，输出的结果总是不对，后来经过调试，发现程序在if(token.tok != END) throw UnexpectedToken();处会直接抛出异常，调试发现，运行到这里时，token.tok的值并不是END，而是一个默认初始值，导致异常抛出。本人最终解决不了，于是将这行代码屏蔽掉，结果正常。
8. 过程化方法实验总体还算成功，测试数据是书本上的，结果和书本上的一样，应该没有什么大问题。实验详细代码在《浮点数计算器过程版》清单中。

第三章 对象化方法

1. **系统模型和类设计**
2. 程序类设计：对象化方法，此时需要设计多个类。首先的是程序类。类中有一个表达式成员，同时有一个不断运行处理数据的run()函数。Run()函数的实现与过程化方法类似。
3. 表达式类设计：将过程化的三个语法单元函数封装在一个表达式类中。表达式类中有单词成员数据，包含三个词法单元函数。同时增加一个getToken()函数，来读取tok,进一步判断处理数据。
4. 单词类设计：在单词类设计中，封装了单词的各类信息，包括数据，标志，等。同时设计了一个初始化函数，有一个readToken()函数，判断单词的类型。
5. 异常类设计：同过程化方法的异常类一样，在此不再赘述，详情请参考上文。
6. **代码运行测试过程**
7. 对象化方法同过程化方法在运行结果方面没有差异，相同的输入数据，输出的数据一样，就是实现的过程不一样，一个是一步一步按过程进行，另一个是将过程，数据封装在类里面，让类与类打交道。编译过程中也没用遇到什么问题。

第四章 对象化系统扩充

1. **系统修改方案**
2. 主程序框架：大整数计算器的设计，主要是将long double换成StrInt 大整数类型进行计算，其余代码不用做太大的修改。
3. 大整数类设计：此部分主要是进行StrInt 大整数类的设计。类中包含大整数类的符号，数值，位数限制，以及一些运算符的重载。详细代码见本部分实验的文件清单。
4. **代码运行测试过程**
5. 由于是前面2个实验代码的整理与小幅度修改，编译过程并么有遇到问题，而且程序运行的结果也与预期的一样，整体来说，实验还算成功。

第五章 总结

1. **实现过程总结**
2. 本次实验是做一个简易计算器，一个是过程版，另一个是对象版。
3. 个人感觉，过程版看似代码量更少，而对象版相比来说，多一点。不过个人觉得过程版适合简单的，代码量少的程序，而对象版则适合复杂的程序。对象版程序比较容易修改，像第3个实验是对大整数的计算，如果用过程版，也可以，但是使用对象版，则更加方便，只需要重新设计一个大整数类型即可，而其余代码几乎不需要改动。对象化编程对大型，复杂的程序来说，可以相对来说比较简单，比较有条理。
4. **实验收获**
5. 技术上：通过这3个大型实验，本人自己看书，不懂的百度，问同学，技术有了一定的提高，尤其是调试方面，通过调试，初步学会如何去解决程序遇到的BUG，总体来说，这次实验很值得。
6. 学习方法：这次的实验，通过自己的努力，基本完成了实验要求。学习上，让我知道，学习上不懂的可以百度，问同学，这样解决问题的速度比自己苦苦思所不得快很多，但是没有自己独自想清楚来的深刻，不过，我更喜欢那句话，众人拾柴火焰高，这个社会需要团结。所以，我以后遇到不会的问题，自己会先去思考，然后实在是想不明白的就去向他人寻求帮忙。
7. **对本次大型实验的看法和建议**
8. 看法：本次实验安排的挺合理，实验内容也很紧凑，先是过程版，然后是对象版，最后是一个大整数类型版。从过程到对象，让我们深刻认识到对象化的简便，安全，实用，最后到大整数类型，进一步升华对象化编程的优势，让我们体会到对象和过程不同的魅力。
9. 建议：希望以后的实验不要给出太多的源代码，应该给我们一些必要的提示，然后让我们自己去编写。这样我们学会的东西应该会更多。

附录

注：实验全部的源代码在每个文件夹下。输入输出数据也在各个.txt文件中。