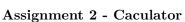
Student Name: 杨卓

Student ID: 2022141450295

高级语言程序设计-II





1 文件结构

由于要写一个计算器,将相关函数放在一个类 Expression 中,成员函数和成员变量的声明放在对应的 .h 头文件中,成员函数的定义再对应的 .cpp 中。同时为了规避命名的冲突,将这个类和定义的相关常量定义在 EXP 这个名字空间中。这个类表示了一行的表达式。

在主函数中对这个类进行测试。

2 常量设置

```
// accuracy
const double eps = 1e-6;

// Error message
const int NO_ERROR = 0;
const int ERROR_DIVIDE_BY_ZERO = 1;
const int ERROR_BRACKET = 1<<1;
const int ERROR_VARIABLE_NAME = 1<<2;
const int ERROR_CONST_VALUE = 1<<3;
const int ERROR_UNDEFINED_VARIABLE = 1<<4;</pre>
```

eps 是对运算过程中的精度控制,即使没怎么用到,但对于 double 类型的数据运算还是写一个。 下面的 6 个常量表示无错误和不同错误类型所代表的编号。这里采用二进制的思想记录,不会重 复也不容易弄错。

3 成员变量设置

```
private:
std::string expr;
int length, error_number;
static std::map<std::string,double> var_map;
std::stack<double> number_stack;
std::stack<int> option_stack;
```

使用 std::string 来存储每次输入的表达式。length 表示表达式的长度,即 expr.length()。error_number用来存储错误信息; var_map 用于存储变量名字和对应的值,用 std::map 实现。这是一个静态变量,因为对变量的定义是全局的,与对象的构建与否没有关系。

下面两个栈是用来处理操作的部分。number_stack 是数字栈, option_stack 为操作符栈,包括各种函数,用一个编号表示这些函数。

4 构造函数和主要的处理函数

```
public:
    Expression();
    Expression(std::string);
    void produce();// main produce function
```

这个构造函数起到清零和重置的作用。produce 就是整个主要处理过程。

5 其他成员函数

```
std::string getVariableName(const int &pos);// position of '='
std::pair<double, int> getNumberFrom(const int &pos);
bool basic_option(const char &option);

void function_produce(const int &option);

void value_calc();
bool isLegalName(const std::string &var_name);

std::pair<std::string, int> getName(const int &pos);

int judgeFunction(const std::string &fun_name);

bool judgeConstValue(const std::string &var_name);

std::pair<bool, double> calc();

void printErrorMessage();
```

从上到下功能的依次介绍:

getVariableName:传入等号的位置,得到等号前的变量名字,去除了前置和后置空格;getNumberFrom:从传入位置读取一串连续的浮点数字;basic_option:判断是否为基本操作符;function_produce对传入的函数的编号进行处理;value_calc:对当前栈顶的操作符利用操作数上的两个数值进行计算;isLegalName:对传入的名字进行合法性的判定;getName从传入位置开始,读取一段连续的字符串;judgeFunction:判断一个字符串是否是函数;judgeConstValue:判断给定字符串是否为常量PI和E;calc为计算值的主要函数;pringeErrorMessage:根据错误值输出错误信息。

6 使用方法

在主程序中调用 Expression.h, 在 EXP 这个名字空间中使用 Expression 类定义一个变量,利用 隐式转换,将一个表达式 string 赋值给它,调用 produce()函数处理。

这个字符串要合法,对一些常见错误可以发现并给出提示,但是对于完全不合法的表达式,如 3--2 这样的式子,会直接终止程序的运行。

为了方便程序的执行,可以在主程序中可以对 exit 进行判断,以此作为标志终止输入。