目录

这是我的课程设计报告但是隐藏第一页个人信息。

一、 课程设计目的

面向对象(Object Oriented Programming)是一种重要的程序设计思想。当整个项目具有一定规模和复杂度时，面向对象可以通过封装、继承、多态使程序设计模块化降低开发难度，缩短开发周期。本文主要讲述基于 QT 设计，实用科学计算器的构建思路和关键代码，并进行测试评估，目前完整项目已上传 GitHub 方便维护和升级。该项目具有较大的工程量和复杂度，可以帮助我们很好的感悟 C++的面向对象程序设计和实际设计开发流程。

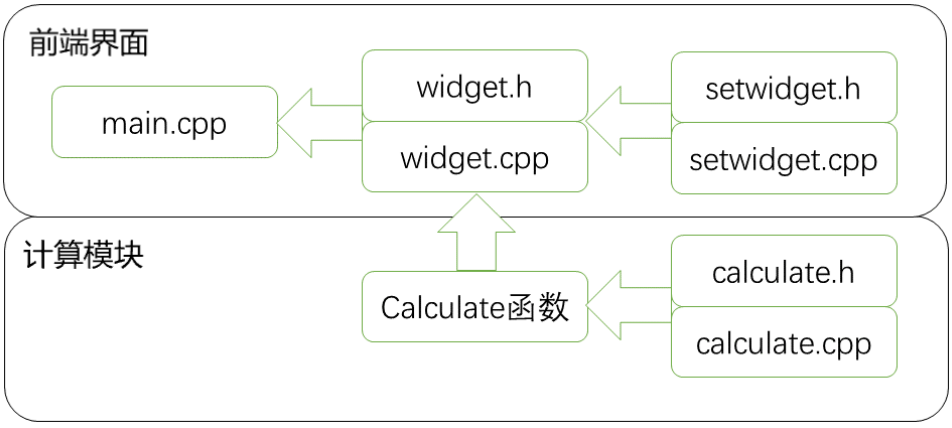
二、 课设的内容与思路

既然是实用科学计算器，本人设计时主要考虑实用和科学。实用首先体现在基于 **Qt** 设计的图形化前端，其次在于方便日常实际使用的细节优化；科学体现在提供了部分科学常数和函数，以及一些和科学计算器相同的特性。

2.1 面向对象的前端设计

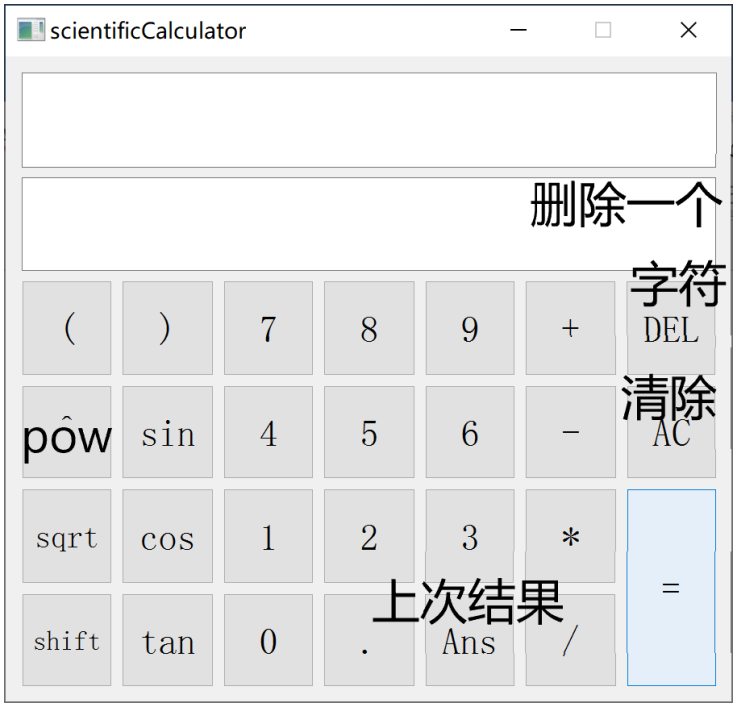
2.1.1 前后端分离的思想

完整逻辑如图文件调用结构。从图中可以看出，整个前端交互逻辑是完整的，前后端之间的接口是 **calculate** 函数，传入**算式字符串**，**上一次结果字符串**和**格式输出参数**，函数返回**结果字符串**。这使得前后端可以分别独立开发和测试。前端只需负责与用户交互逻辑的正确，并能正确获取三个输入参数即可，后端只需通过传入参数计算正确并返回即可。

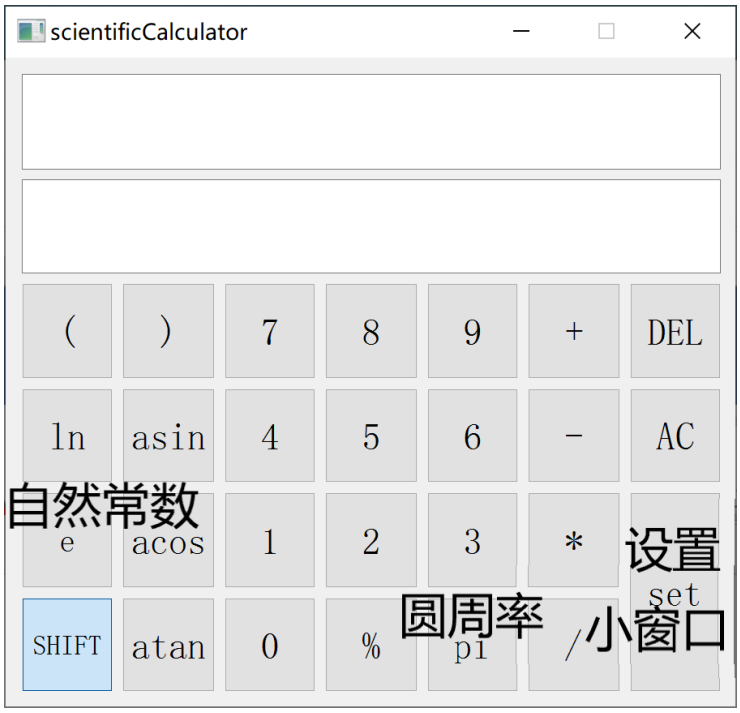


2.1.2 面向对象的前端逻辑

Qt 是一个面向对象的跨平台 C++图形化界面开发框架，使用特殊的代码生成扩展和宏。这使得 Qt 很容易扩展，并且允许真正的组件编程，开发门槛低。通过绘画 UI 界面，设置相应组件函数，调用函数即可实现基本开发。通过信号与槽的无缝对象通信，可以实现更复杂的前端响应。本计算器主窗口如图：



按下 shift 键后主窗口界面：



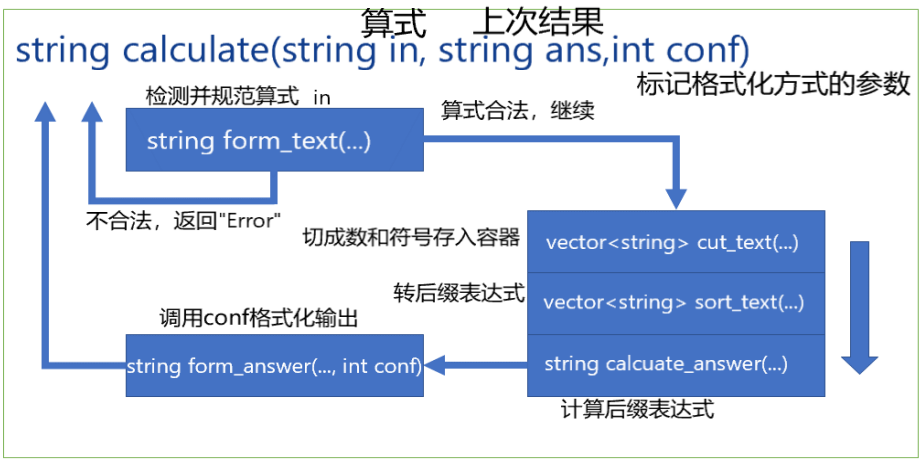
2.1.3 前端实用设计细节

细节处理主要在于对现实中计算器的特性模拟和对计算机设备的协调。借用真实计算机中 shift 的特性减少键位，利用如图的小窗口调整输出格式，使计算器更简洁。计算器窗口默认在桌面置顶方便操作，输入输出框字体选用 Consolas 防止认错一些相近字符。允许同时进行模拟按键输入和键盘输入，利用 connect 函数（跳转查看）实现敲回车出结果，带来流畅的使用体验。

2.2 面向对象的算法设计

2.2.1 计算过程的算法思路

计算过程最重要的就是对输入字符串的解读。要对字符串各成分进行划分，再进行计算。利用二叉树进行算法设计，思路简单，程序模块化，但是模块设计较难，众多判断递归操作难以调试，故我选用堆栈设计逆波兰算法[1]。先对字符串进行**合法检测与规范化**，再**成分划分**，然后使用堆栈逆波兰算法**转化为后缀表达式**，**计算后缀表达式**，接着**格式化结果**。将这 5 步写成 5 个**过程函数**，各自通过独立验证后将五部分整合在 calculate 函数综合验证，如图：



2.2.2 面向对象的数据处理

整个算法在运转过程中，数字和符号大部分时间都是以字符串形式参与，仅在计算时将数字 string 转化为成 long double 类型，计算完成再转回 string，这样在成分划分时，可以把小数如“1.0”，百分数如“5.8%”都划分为数字对象。同时，把数字和符号分别装入容器，对容器进行出入栈和算法操作，可以使算法过程抽象化，易于封装使用。

2.2.3 计算功能的设计考虑

这一部分的细节主要有规范字符串过程的强大错误检测，提供上次计算结果和一些科学常数参于运算，以及符号库的可拓展性。

三、 程序的内容与实现

3.1 基于 QT 的界面实现

3.1.1 主窗口

main.cpp 只负责调用 widget 类产生计算器窗口。成员变量 shift 是 bool 类型，记录 shift 键状态；config 是 int 类型，初始为 5（即保留 5 位小数）；ans_global，text，answer 都是 QString 类型，分别记录**上次结果**，**算式**和**结果**。普通按键调用函数如图：

```
void Widget::on_Button_0_clicked()
{   按下“0”后，读取文本框内容再追加“0”
    text = ui->lineEdit_text->text() + "0";
    ui->lineEdit_text->setText(text);
}    在文本框显示text内容
```

shift 翻转键函数：

```
//shiftOpt
void Widget::on_Button_shift_toggled(bool checked)
{                    两状态按键，根据shift键状
    shift = checked;
    if(shift == true) 态刷新一些按键的显示值
    {
        ui->Button_shift->setText("SHIFT");
        ui->Button_sin->setText("asin");
```

shift 键使其他一些按键能拥有两个按键效果，实现原理如下：

```
void Widget::on_Button_pow_ln_clicked()
{
    if(shift == true)
    {
        text = ui->lineEdit_text->text() + "ln(";
        ui->lineEdit_text->setText(text);
    }
    else  shift值决定追加在text字符串后的内容
    {
        text = ui->lineEdit_text->text() + "^(";
        ui->lineEdit_text->setText(text);
    }
}
```

“=”键在 shift 键按下后变成 set 键，用于生成小窗口界面设置输出格式：

```
void Widget::on_Button_enter_clicked()
{
    if(shift == true) 此时是set键，用于产生小窗口对象
    {
        setwidget *configwidget = new setwidget;
        configwidget->show();
        Widget::connect(configwidget, &setwidget::sendToWidget, this, &Widget::confSlot);
    }
    else
    {   此时是“=”键，进行类型转换和调用calculate函数计算，之后
        text = ui->lineEdit_text->text();                                显示结果
        string temp = text.toStdString();
        ans_global = calculate(temp, ans_global, config);
        answer = QString(QString::fromLocal8Bit(ans_global.c_str()));
        ui->lineEdit_answer->setText(answer);
    }
}
```

主窗口参数设置：

```
Widget::Widget(QWidget *parent)
    : QWidget(parent)
    , ui(new Ui::Widget)
{
    ui->setupUi(this);

    this->setWindowTitle("scientificCalculator"); 窗口标题设置

    this->setWindowFlags(Qt::WindowStaysOnTopHint); 停留桌面顶层设置

    QFont textFont("Consolas",14); 字体设置
    ui->lineEdit_text->setFont(textFont);
    ui->lineEdit_answer->setFont(textFont);

    ui->Button_shift->setCheckable(true); 设置检测shift键状态

    connect(ui->lineEdit_text, SIGNAL(returnPressed()), this, SLOT(on_Button_enter_clicked()));
}       连接“=”按键和键盘ENTER键
```

3.1.2 小窗口

小窗口是自定义 setwidget 类，从 set 键调用生成，通过赋值相加的一种编码方式来保存参数：

按下 commit 键后将信号传出，在 widget 类中定义 SLOT 槽函数接收该信号，并赋值给 config，之后会随着“=”触发传入 calculate 函数。

```
void setwidget::on_commitButton_clicked()
{
    conf = conf_1 + conf_2 + conf_3;
    emit sendToWidget(conf);
    this->close();
}
```

将conf以信号传出

关闭小窗口

3.2 基于 C++的计算实现

3.2.1 工具函数

```
int priority(string opt)
{
    if (opt == "(" ||
        opt == "j" || opt == "x" || opt == "y" || opt == "z" ||
        opt == "k" || opt == "u" || opt == "v" || opt == "w")
        return MAX_PRIORITY;
    else if (opt == "^" || opt == "%") return 3;
    else if (opt == "*" || opt == "/") return 2;
    else if (opt == "+" || opt == "-") return 1;
    else                               return 0;//num
}
```

“j”，“k”等符号就是sqrt等函数的替换编码，后面form_text细说

define一个最高优先级，当前值是5

经form_text规范化后不是符号就是数字

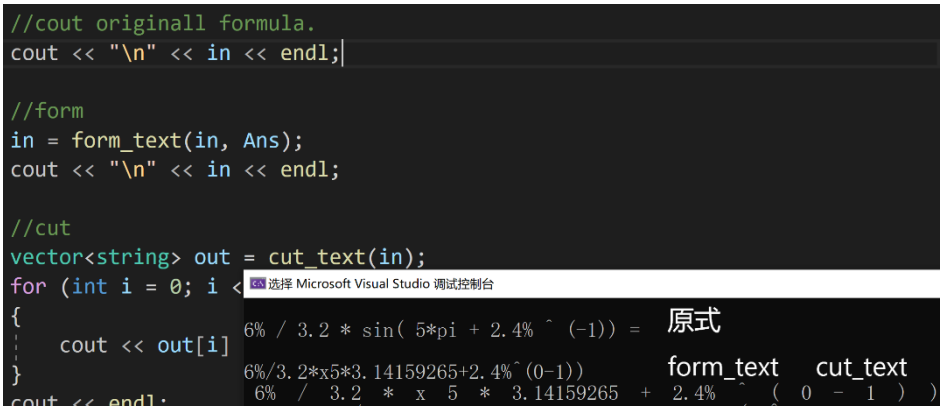
此函数用于判断字符串是符号还是数字，同时也判断各个符号的优先级。可以发现三角函数开方等运算自带括号。同时可以认为括号对其里面的内容做乘 1 操作。故可以认为括号和函数都是优先级最高的单目运算符。扩展运算符需在此函数中添加该符号优先级。

digitize 函数把数字如“11.4%”转化成 long double 类型，s_add，s_sin 等运算函数调用 digitize 实现 string 输入转 long double 计算再转 string 输出。

3.2.2 计算的过程函数

string form_text(string in, string ans)函数首先清除字符串中的空格和等号，再对字符串进行一系列操作：（并非按如下顺序，而是穿插进行）

1) 如果仅剩单个字符，若是数字继续操作，若是其他符号返回异常字符串“@”，返回“@”后不进行后续的**过程函数**。
2) 只有负号为第一个字符，或负号前是“(”时出现负数或负式，在此种情况前面补零就可以把所有负式转化为“**0-式子**”。
3) 替换编码：sin 等一系列函数有多字符，将函数替换为单字符方便操作，如“sin(”替换为“x”，这样就和“(”一样为单字符，方便统一操作。算式中的“Ans”、“e”、“pi”转化为具体数值的字符串。
4) 统计了**有效字符集** 1（计算器上能按出的所有字符，在代码中是 check_1），每个字符都有效才进行上述替换编码。开辟新的**有效字符集** 2 再检测（在代码中是 check_2，字符集 1 中的字母和字符集 2 中的字母**没有交集**）。如“ssin(”是可以通过**字符集** 1 的，但替换后为“sx”，这时“s”不在**字符集** 2 中，返回“@”。经此筛选后的算式字符串所有字符均有意义，但仍不一定合法，如出现“++”。**纠错集** 3（在代码中是 check_3）检测此类问题。三次筛选检测依次筛出**陌生字符**，**重复字母**，**重复符号**，实测检测效果很好。



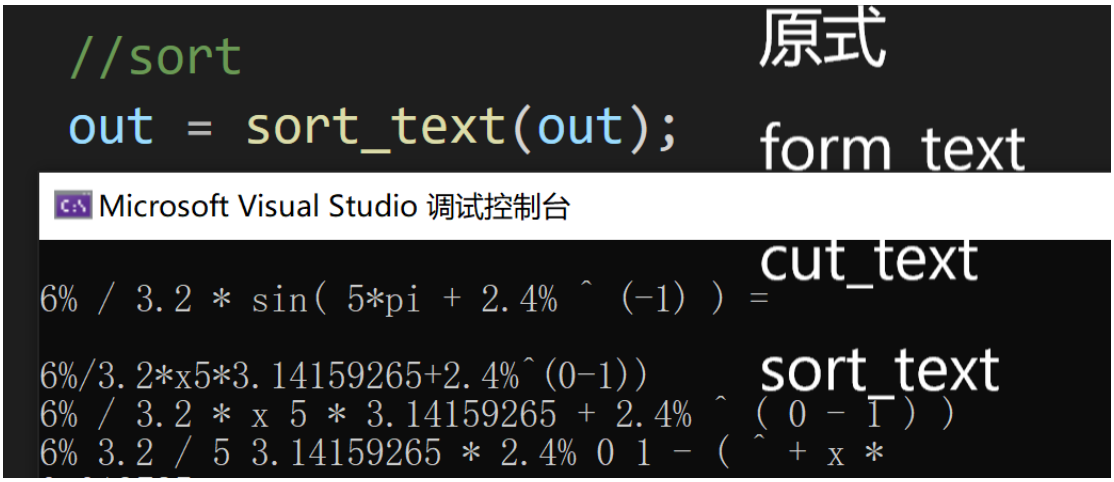
代码较长不便展示，跳转查看。

vector<string> cut_text(string in)函数接收算式字符串进行成分划分，即划分成数字和操作符。内容较为简单，展示效果如上图。

vector<string> sort_text(vector<string> in)函数接收算式转化为**后缀表达式**，此过程最重要。循环遍历遵循如下过程：

1) 遍历到数字就直接输出。
2) 遇到最高优先级符号压入栈。
3) 遇到“)”就依次输出栈中操作符直到栈空或遇到最高优先级符号，把该符号也输出。（2，3 是我的算法创新点，用于处理“sin()”等函数）
4) 遇到其他操作符时先看栈顶运算符：若栈顶是最高优先级就压入栈；若当前操作符的优先级高于栈顶的，压入栈；否则就依次输出栈中运算符直到栈空，或者栈顶的优先级不大于当前操作符。
5) 遍历完后栈中若有操作符就依次输出。

数字，操作符优先级统一用工具函数 priority 判断，直接按该逻辑写即可，但要注意在多处增加对栈空的判断，防止 pop 查询空栈顶报错。如图效果：



string calcuate_answer(vector<string> in)函数接收**后缀表达式**进行计算，遍历逻辑很简单：

1) 遇到数字入栈。
2) 遇到单目运算符取栈顶数字运算（若是“(”不运算）再压入栈。
3) 遇到双目运算符取两数字运算，要注意运算顺序是“后数 操作符 先数”。
4) 遍历完后栈中数就是结果。

数字，操作符也用 priority 函数判断，同样要注意在多处增加对栈空的判断，防止 pop 查询空栈报错。此处还要注意的问题是遇到双目运算符，但是栈中只有一个数的情况，这显然是算式不合法，同时也是我的 string form_text(string in, string ans)函数唯一检测不出的异常，这里我处理方法是先取一个数，若栈空就直接放弃该操作符，数字回栈。可以防止程序闪退。

string form_answer(string in, int conf)函数接收结果和格式输出参数 conf，对 conf 相当于一个解码过程，该过程对应 Qt 小窗口类中的编码过程。得到目标格式后对结果字符串格式化后输出。

```
string form_answer(string in, int conf)
{
    int len = 0;
    in = to_string(digitize(in));
    if (conf >= 100)in = s_mul(in, "100");
    while (in.length() - in.find(".") - conf % 10 - 2)
    {
        in.pop_back();
    }
    if(conf/10%10)in.pop_back();
    else
    {
        if (in[in.length() - 1] > '5')
        {
            len = in.length() - in.find(".") - 1;
            in = to_string(long long int(digitize(in) * pow(10, len)) + 1).insert(in.length() - len - 1, ".");
        }
        in.pop_back();
    }
    if(in[in.length() - 1]=='.')in.pop_back();
    if (conf >= 100)in.append("%");
    return in;
}
```

记录answer长度
例如计算器只输入“1”，经过此变成“1.000000”，方便格式化
保留多一位，方便后面截止此位或四舍五入
截止此位
四舍五入
整数去掉小数点
百分数补上百分号

string calculate(string in, string ans, int conf)整和以上 5 个函数，流程图跳转，代码如下：

```
string calculate(string in, string ans, int conf)
{
    string out;
    out = form_text(in, ans);
    if (out == "@")
    {
        out = "Error";
        return out;
    }
    else
    {
        ans = form_answer(calcuate_answer(sort_text(cut_text(out))), conf);
        return ans;
    }
}
```

四、程序的测试与结论

4.1 测试基本要求

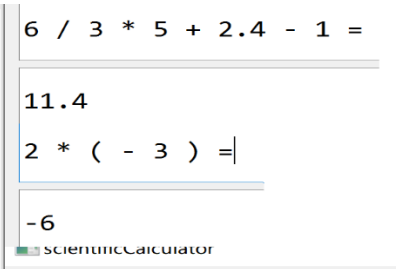
各要求都是达标的：

二、基本要求

1. 能够支持四则运算功能，如输入；

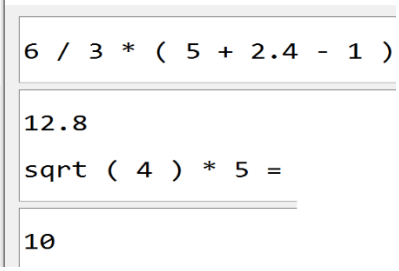
6 / 3 * 5 + 2.4 - 1 =　(输出结果为11.4)

2 * (- 3) =　(输出结果为-6)



2. 能够支持带括号的优先级运算，如；

6 / 3 * (5 + 2.4 - 1) =　(输出结果为 12.8)



3. 能够支持常用数学函数表达式（开方、幂、三角函数、对数等）；

sqrt (4) * 5 =　(输出结果为10)

甚至可以直接复制例题敲回车。

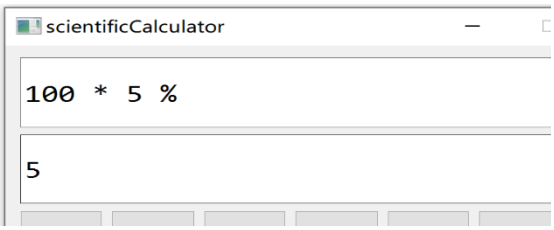
4. 支持百分号运算，如：

100 * 5 % =　(输出结果为5)

100 + (100 * 5 %) =　(输出结果为105)

10 - (10 * 20 %) =　(输出结果为8)

(30 / 60) % =　(输出结果为50)



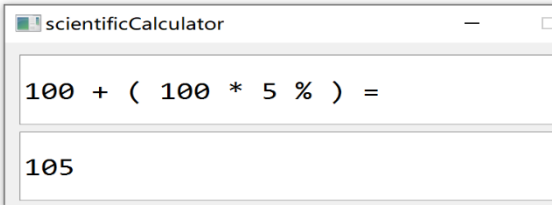
4. 支持百分号运算，如：

100 * 5 % =　(输出结果为5)

100 + (100 * 5 %) =　(输出结果为105)

10 - (10 * 20 %) =　(输出结果为8)

(30 / 60) % =　(输出结果为50)



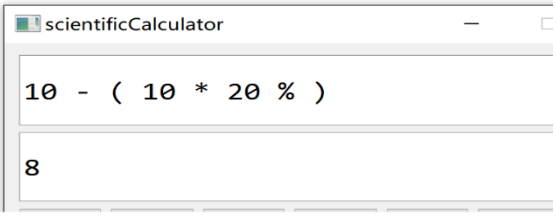
4. 支持百分号运算，如：

100 * 5 % =　(输出结果为5)

100 + (100 * 5 %) =　(输出结果为105)

10 - (10 * 20 %) =　(输出结果为8)

(30 / 60) % =　(输出结果为50)



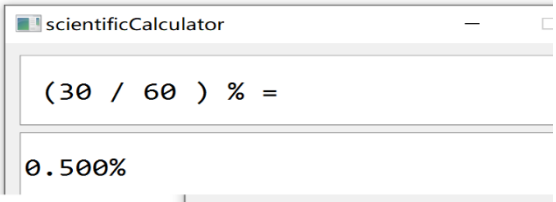
4. 支持百分号运算，如：

100 * 5 % =　(输出结果为5)

100 + (100 * 5 %) =　(输出结果为105)

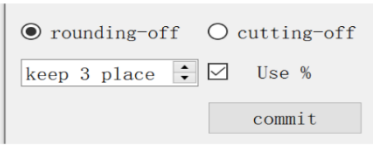
10 - (10 * 20 %) =　(输出结果为8)

(30 / 60) % =　(输出结果为50)



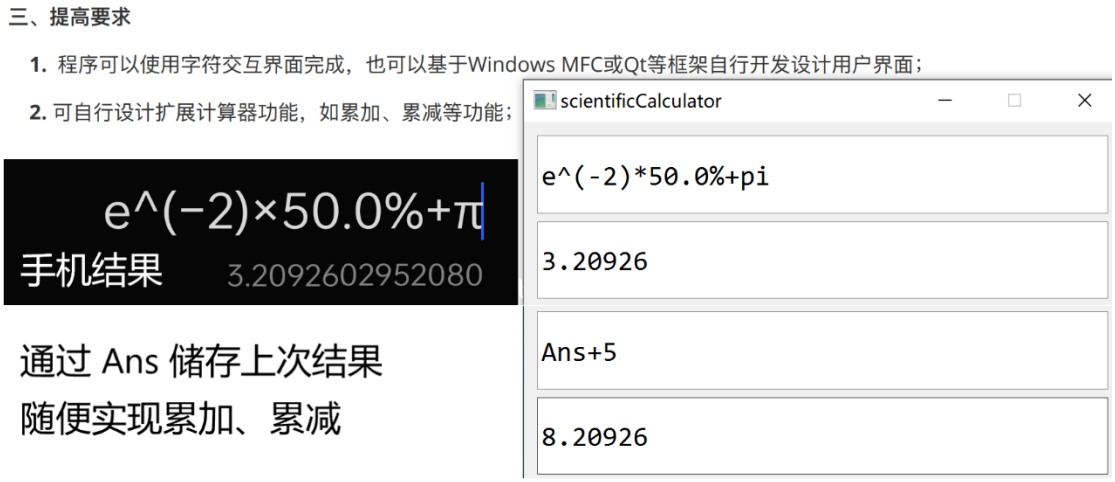
5. 对运算结果按指定格式显示，如：

能够根据设定参数对运算结果的小数部分进行位数截取或四舍五入

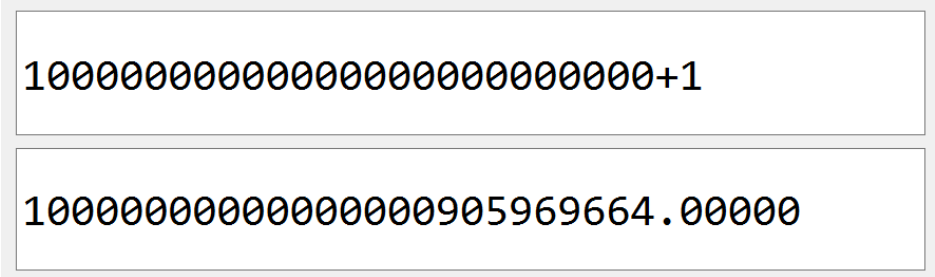


4.2 测试提高要求

Qt 图形化界面，“Ans”键实现的储存结果用于累加等累次运算，以及很多其他功能、细节都进行了设计。



目前存在的不足是大数运算，因为 long double 的低位精度不够：



五、课程设计总结

本项目主要基于面向对象思想设计。现有基于 Qt 开发的图形化界面方便使用，又考虑了实际使用体验和实体计算器特性优化了一些细节；计算部分，程序实现的模块化使内容易于拓展，算法的抽象化使计算器不易出 bug。不足之处在于计算过程使用了 long double，大数计算时低位精度不够高。当然这个问题可以通过高精度算法解决，现有很多开源 BigInt 类提供高精度运算，而我的算法过程一直使用的 string 类可以很容易地套用这些高精度算法，这也会是我的下一个改进目标。

参考文献

[1] 王昊,陈雅.两种逆波兰转换算法分析、程序设计及比较【J】.情报科学.2005(06):955-960
[2] 郑致力.算术表达式解析引擎的设计及实现【D】.北京邮电大学.2012(03)

附件

源码已上传 GitHub: https://github.com/Radioactive-jkl/cppSciCal

也可以下载打包好的可执行文件直接使用。

小窗口 Setwidget 类

Setwidget.h

```
#ifndef SETWIDGET_H
#define SETWIDGET_H

#include <QWidget>


namespace Ui {
class setwidget;
}

class setwidget : public QWidget
{
    Q_OBJECT

public:

    bool use_per = false;

    int conf = 5,
        conf_1 = 0,
        conf_2 = 5,
        conf_3 = 0;

    explicit setwidget(QWidget *parent = nullptr);
    ~setwidget();

private slots:
    void on_roundButon_clicked();

    void on_cutButton_clicked();

```

```
    void on_spinBox_valueChanged(int arg1);

    void on_commitButton_clicked();

    void on_checkBox_stateChanged(int arg1);

private:
    Ui::setwidget *ui;

signals:
    void sendToWidget(int co);
};

#endif // SETWIDGET_H
```

Setwidget.cpp

```
#include "setwidget.h"
#include "ui_setwidget.h"

setwidget::setwidget(QWidget *parent) :
    QWidget(parent),
    ui(new Ui::setwidget)
{
    ui->setupUi(this);

    this->setWindowTitle("set");

    ui->spinBox->setRange(0, 5);
    ui->spinBox->setSingleStep(1);
    ui->spinBox->setPrefix("keep ");
    ui->spinBox->setSuffix(" place");
}
setwidget::~setwidget()
{
    delete ui;
}

void setwidget::on_roundButon_clicked()
{
    conf_1 = 0;
}

void setwidget::on_cutButton_clicked()
```

```
{
    conf_1 = 10;
}

void setwidget::on_spinBox_valueChanged(int arg1)
{
    conf_2 = arg1;
}


void setwidget::on_commitButton_clicked()
{
    conf = conf_1 + conf_2 + conf_3;
    emit sendToWidget(conf);
    this->close();
}


void setwidget::on_checkBox_stateChanged(int arg1)
{
    switch (arg1)
    {
    case Qt::Checked:
        conf_3 = 100;
        break;
    case Qt::Unchecked:
        conf_3 = 0;
    }
}
```

主窗口 widget 类

Widget.h

```
#ifndef WIDGET_H
#define WIDGET_H

#include <QWidget>
#include "setwidget.h"
#include "calculate.h"


QT_BEGIN_NAMESPACE
namespace Ui { class Widget; }
QT_END_NAMESPACE
```

```
12.
13.class Widget : public QWidget
14.{
15.    Q_OBJECT
16.
17.public:
18.    Widget(QWidget *parent = nullptr);
19.    ~Widget();
20.
21.    bool shift = false;
22.    int config = 5;
23.    string ans_global = "0"; （再点击跳回）
24.
25.private slots:
26.    void on_Button_0_clicked();
27.
28.    void on_Button_1_clicked();
29.
30.    void on_Button_2_clicked();
31.
32.    void on_Button_3_clicked();
33.
34.    void on_Button_4_clicked();
35.
36.    void on_Button_5_clicked();
37.
38.    void on_Button_6_clicked();
39.
40.    void on_Button_7_clicked();
41.
42.    void on_Button_8_clicked();
43.
44.    void on_Button_9_clicked();
45.
46.    void on_Button_left_clicked();
47.
48.    void on_Button_right_clicked();
49.
50.    void on_Button_shift_toggled(bool checked);
51.
52.    void on_Button_add_clicked();
53.
54.    void on_Button_sub_clicked();
55.
```

```
    void on_Button_mul_clicked();

    void on_Button_div_clicked();

    void on_Button_pow_ln_clicked();

    void on_Button_sqrt_e_clicked();

    void on_Button_sin_clicked();

    void on_Button_cos_clicked();

    void on_Button_tan_clicked();

    void on_Button_point__clicked();

    void on_Button_ans_pai_clicked();

    void on_Button_del_clicked();

    void on_Button_ac_clicked();

    void on_Button_enter_clicked();

private:
    Ui::Widget *ui;
    QString text = "";
    QString answer = "";

public slots:
    void confSlot(int co);
};
#endif // WIDGET_H
```

Widget.cpp

```
#include "widget.h"
#include "ui_widget.h"


Widget::Widget(QWidget *parent)
    : QWidget(parent)
    , ui(new Ui::Widget)
{
```

```
9.      ui->setupUi(this);
10.
11.     this->setWindowTitle("scientificCalculator");
12.
13.     this->setWindowFlags(Qt::WindowStaysOnTopHint);
14.
15.     QFont textFont("Consolas",14);
16.     ui->lineEdit_text->setFont(textFont);
17.     ui->lineEdit_answer->setFont(textFont);
18.
19.     ui->Button_shift->setCheckable(true);
20.
21.     connect(ui->lineEdit_text, SIGNAL(returnPressed()), thi
    s, SLOT(on_Button_enter_clicked()));（再点击跳回）
22.
23.}
24.
25.Widget::~Widget()
26.{
27.     delete ui;
28.}
29.
30.
31./***********
32. * numbers *
33. ***********/（再点击跳回）
34.
35.void Widget::on_Button_0_clicked()
36.{
37.     text = ui->lineEdit_text->text() + "0";
38.     ui->lineEdit_text->setText(text);
39.}
40.
41.void Widget::on_Button_1_clicked()
42.{
43.     text = ui->lineEdit_text->text() + "1";
44.     ui->lineEdit_text->setText(text);
45.}
46.
47.void Widget::on_Button_2_clicked()
48.{
49.     text = ui->lineEdit_text->text() + "2";
50.     ui->lineEdit_text->setText(text);
51.}
```

```
52.
53.void Widget::on_Button_3_clicked()
54.{
55.    text = ui->lineEdit_text->text() + "3";
56.    ui->lineEdit_text->setText(text);
57.}
58.
59.void Widget::on_Button_4_clicked()
60.{
61.    text = ui->lineEdit_text->text() + "4";
62.    ui->lineEdit_text->setText(text);
63.}
64.
65.void Widget::on_Button_5_clicked()
66.{
67.    text = ui->lineEdit_text->text() + "5";
68.    ui->lineEdit_text->setText(text);
69.}
70.
71.void Widget::on_Button_6_clicked()
72.{
73.    text = ui->lineEdit_text->text() + "6";
74.    ui->lineEdit_text->setText(text);
75.}
76.
77.void Widget::on_Button_7_clicked()
78.{
79.    text = ui->lineEdit_text->text() + "7";
80.    ui->lineEdit_text->setText(text);
81.}
82.
83.void Widget::on_Button_8_clicked()
84.{
85.    text = ui->lineEdit_text->text() + "8";
86.    ui->lineEdit_text->setText(text);
87.}
88.
89.void Widget::on_Button_9_clicked()
90.{
91.    text = ui->lineEdit_text->text() + "9";
92.    ui->lineEdit_text->setText(text);
93.}
94.
95.
```

```
/***********
 * options *
 ***********/

 //commonOpt
 void Widget::on_Button_left_clicked()
 {
     text = ui->lineEdit_text->text() + "(";
     ui->lineEdit_text->setText(text);
 }

 void Widget::on_Button_right_clicked()
 {
     text = ui->lineEdit_text->text() + ")";
     ui->lineEdit_text->setText(text);
 }

 void Widget::on_Button_add_clicked()
 {
     text = ui->lineEdit_text->text() + "+";
     ui->lineEdit_text->setText(text);
 }

 void Widget::on_Button_sub_clicked()
 {
     text = ui->lineEdit_text->text() + "-";
     ui->lineEdit_text->setText(text);
 }

 void Widget::on_Button_mul_clicked()
 {
     text = ui->lineEdit_text->text() + "*";
     ui->lineEdit_text->setText(text);
 }

 void Widget::on_Button_div_clicked()
 {
     text = ui->lineEdit_text->text() + "/";
     ui->lineEdit_text->setText(text);
 }

 //shiftOpt（再点击跳回）
 void Widget::on_Button_shift_toggled(bool checked)
 {
```

```
140.      shift = checked;
141.      if(shift == true)
142.      {
143.          ui->Button_shift->setText("SHIFT");
144.          ui->Button_sin->setText("asin");
145.          ui->Button_cos->setText("acos");
146.          ui->Button_tan->setText("atan");
147.          ui->Button_point_->setText("%");
148.          ui->Button_pow_ln->setText("ln");
149.          ui->Button_sqrt_e->setText("e");
150.          ui->Button_ans_pai->setText("pi");
151.          ui->Button_enter->setText("set");
152.      }
153.      else
154.      {
155.          ui->Button_shift->setText("shift");
156.          ui->Button_sin->setText("sin");
157.          ui->Button_cos->setText("cos");
158.          ui->Button_tan->setText("tan");
159.          ui->Button_point_->setText(".");
160.          ui->Button_pow_ln->setText("^");
161.          ui->Button_sqrt_e->setText("sqrt");
162.          ui->Button_ans_pai->setText("Ans");
163.          ui->Button_enter->setText("=");
164.      }
165.  }
166.
167.  void Widget::on_Button_pow_ln_clicked()
168.  {
169.      if(shift == true)
170.      {
171.          text = ui->lineEdit_text->text() + "ln(";
172.          ui->lineEdit_text->setText(text);
173.      }
174.      else
175.      {
176.          text = ui->lineEdit_text->text() + "^(";
177.          ui->lineEdit_text->setText(text);
178.      }
179.  }
180.
181.  void Widget::on_Button_sqrt_e_clicked()
182.  {
183.      if(shift == true)
```

```
184.     {
185.         text = ui->lineEdit_text->text() + "e";
186.         ui->lineEdit_text->setText(text);
187.     }
188.     else
189.     {
190.         text = ui->lineEdit_text->text() + "sqrt(";
191.         ui->lineEdit_text->setText(text);
192.     }
193. }
194.
195. void Widget::on_Button_sin_clicked()
196. {
197.     if(shift == true)
198.     {
199.         text = ui->lineEdit_text->text() + "asin(";
200.         ui->lineEdit_text->setText(text);
201.     }
202.     else
203.     {
204.         text = ui->lineEdit_text->text() + "sin(";
205.         ui->lineEdit_text->setText(text);
206.     }
207. }
208.
209. void Widget::on_Button_cos_clicked()
210. {
211.     if(shift == true)
212.     {
213.         text = ui->lineEdit_text->text() + "acos(";
214.         ui->lineEdit_text->setText(text);
215.     }
216.     else
217.     {
218.         text = ui->lineEdit_text->text() + "cos(";
219.         ui->lineEdit_text->setText(text);
220.     }
221. }
222.
223. void Widget::on_Button_tan_clicked()
224. {
225.     if(shift == true)
226.     {
227.         text = ui->lineEdit_text->text() + "atan(";
```

```
228.          ui->lineEdit_text->setText(text);
229.      }
230.      else
231.      {
232.          text = ui->lineEdit_text->text() + "tan(";
233.          ui->lineEdit_text->setText(text);
234.      }
235.  }
236.
237.  void Widget::on_Button_point__clicked()
238.  {
239.      if(shift == true)
240.      {
241.          text = ui->lineEdit_text->text() + "%";
242.          ui->lineEdit_text->setText(text);
243.      }
244.      else
245.      {
246.          text = ui->lineEdit_text->text() + ".";
247.          ui->lineEdit_text->setText(text);
248.      }
249.  }
250.
251.  void Widget::on_Button_ans_pai_clicked()
252.  {
253.      if(shift == true)
254.      {
255.          text = ui->lineEdit_text->text() + "pi";
256.          ui->lineEdit_text->setText(text);
257.      }
258.      else
259.      {
260.          text = ui->lineEdit_text->text() + "Ans";
261.          ui->lineEdit_text->setText(text);
262.      }
263.  }
264.
265.  //controlOpt
266.  void Widget::on_Button_del_clicked()
267.  {
268.      text = ui->lineEdit_text->text();
269.      text.chop(1);
270.      ui->lineEdit_text->setText(text);
271.  }
```

```
272.
273.  void Widget::on_Button_ac_clicked()
274.  {
275.      text = ui->lineEdit_text->text();
276.      answer = ui->lineEdit_answer->text();
277.      text.clear();
278.      answer.clear();
279.      ui->lineEdit_text->setText(text);
280.      ui->lineEdit_answer->setText(answer);
281.  }
282.
283.  void Widget::on_Button_enter_clicked()（再点击跳回）
284.  {
285.      if(shift == true)
286.      {
287.          setwidget *configwidget = new setwidget;
288.          configwidget->show();
289.          Widget::connect(configwidget, &setwidget::sendToW
    idget, this, &Widget::confSlot);
290.      }
291.      else
292.      {
293.          text = ui->lineEdit_text->text();
294.          string temp = text.toStdString();
295.          ans_global = calculate(temp, ans_global, config);
296.          answer = QString(QString::fromLocal8Bit(ans_globa
    l.c_str()));
297.          ui->lineEdit_answer->setText(answer);
298.      }
299.  }
300.
301.  void Widget::confSlot(int co) （再点击跳回）
302.  {
303.      config = co;
304.      //qDebug("%d", config);
305.  }
```

计算 calculate 文件

Calculate.h

```
1.  #ifndef CALCULATE_H
2.  #define CALCULATE_H
3.
4.
```

```cpp
#include <iostream>
#include <string>
#include <stack>
#include <vector>
#include <cmath>


using namespace std;

/*****************
 * toolFunctions *
 *****************/

//Define the highest priority.
#define MAX_PRIORITY 5

//Rank opts and select nums.
int priority(string opt);

//change numbers' string into double.
long double digitize(string num);

//Operations for string
string s_add(string num1, string num2);

string s_sub(string num1, string num2);

string s_mul(string num1, string num2);

string s_div(string num1, string num2);

string s_pow(string num1, string num2);

string s_sqrt(string num);

string s_ln(string num);

string s_sin(string num);

string s_cos(string num);

string s_tan(string num);

string s_asin(string num);
```

```
49.
50.string s_acos(string num);
51.
52.string s_atan(string num);
53.
54.string s_per(string num);
55.
56.
57./**********************
58. * calculateFunctions *
59. **********************/
60.
61.string form_text(string in, string ans);
62.
63.vector<string> cut_text(string in);
64.
65.vector<string> sort_text(vector<string> in);
66.
67.string calcuate_answer(vector<string> in);
68.
69.string form_answer(string in, int conf);
70.
71.//Integrate all calculateFunctions
72.string calculate(string in, string ans, int conf);
73.
74.#endif // CALCULATE_H
```

Calculate.cpp

```
1. #include <calculate.h>
2.
3.
4. /*****************
5.  * toolFunctions *
6.  *****************/
7.
8. int priority(string opt)
9. {
10.     if (opt == "(" ||
11.         opt == "j" || opt == "x" || opt == "y" || opt == "z
   " ||
12.         opt == "k" || opt == "u" || opt == "v" || opt == "w
   ")
13.         return MAX_PRIORITY;
```

```
14.     else if (opt == "^" || opt == "%") return 3;
15.     else if (opt == "*" || opt == "/") return 2;
16.     else if (opt == "+" || opt == "-") return 1;
17.     else                               return 0;//num
18. }
19. 
20. 
21. long double digitize(string num) （再点击跳回）
22. {
23.     long double d_num = 0;
24.     if (num.find('%') != num.npos)
25.     {
26.         num.pop_back();
27.         d_num = (atof(num.c_str()) / 100.0);
28.     }//deal with percentage like "11.4%"
29.     else d_num = atof(num.c_str());
30.     return d_num;
31. }
32. 
33. 
34. string s_add(string num1, string num2)
35. {
36.     return to_string(digitize(num1) + digitize(num2));
37. }
38. 
39. string s_sub(string num1, string num2)
40. {
41.     return to_string(digitize(num1) - digitize(num2));
42. }
43. 
44. string s_mul(string num1, string num2)
45. {
46.     return to_string(digitize(num1) * digitize(num2));
47. }
48. 
49. string s_div(string num1, string num2)
50. {
51.     return to_string(digitize(num1) / digitize(num2));
52. }
53. 
54. 
55. string s_pow(string num1, string num2)
56. {
57.     return to_string(pow(digitize(num1), digitize(num2)));
```

```
58.}
59.
60.
61.string s_sin(string num)
62.{
63.    return to_string(sin(digitize(num)));
64.}
65.
66.string s_cos(string num)
67.{
68.    return to_string(cos(digitize(num)));
69.}
70.
71.string s_tan(string num)
72.{
73.    return to_string(tan(digitize(num)));
74.}
75.
76.string s_asin(string num)
77.{
78.    return to_string(asin(digitize(num)));
79.}
80.
81.string s_acos(string num)
82.{
83.    return to_string(acos(digitize(num)));
84.}
85.
86.string s_atan(string num)
87.{
88.    return to_string(atan(digitize(num)));
89.}
90.
91.string s_sqrt(string num)
92.{
93.    return to_string(sqrt(digitize(num)));
94.}
95.
96.string s_ln(string num)
97.{
98.    return to_string(log(digitize(num)));
99.}
100.
101.  string s_per(string num)
```

```
102.  {
103.      return to_string(digitize(num) / 100.0);
104.  }
105.
106.
107.  /**********************
108.   * calculateFunctions *
109.   **********************/
110.
111.  string form_text(string in, string ans) （再点击跳回）
112.  {
113.
114.      //clear " "
115.      for (int i = in.find(" "); i <= in.length(); i = in.f
   ind(" "))
116.      {
117.          in.erase(i, 1);
118.      }
119.      if (in.empty())return "@";
120.      for (int i = in.find("="); i <= in.length(); i = in.f
   ind("="))
121.      {
122.          in.erase(i, 1);
123.      }
124.      if (in.empty())return "@";
125.      if (in.length() == 1)
126.      {
127.          string check_0 = "0123456789";
128.          for (int i = 0; i < in.length(); i++)
129.          {
130.              if (check_0.find(in[i]) == check_0.npos)retur
   n "@";
131.          }
132.      }
133.      // check all valid strings
134.      /**************************
135.       * ( ) s q r t i n c o a l *
136.       * ^       *
137.       * + - * /       *
138.       * A . e p %      *
139.       * 0 1 2 3 4 5 6 7 8 9    *
140.       **************************/
141.      string check_1 = "+-*/()sqrtincoal^0123456789A.ep%";
142.      for (int i = 0; i < in.length(); i++)
```

```
143.     {
144.         if (check_1.find(in[i]) == check_1.npos)return "@
    ";
145.     }
146.
147.     // fill "0"
148.     if(in[0]=='-')in.insert(0, "0");
149.     for (int i = 1; i < in.length(); i++)
150.     {
151.         if (in[i] == '-' && in[i - 1] == '(')
152.         {
153.             in.insert(i, "0");
154.         }
155.     }
156.
157.     // encode to other unoccupied code
158.     /**********************************
159.      * "sqrt(" "sin(" "cos(" "tan("  *
160.      * "ln(" "asin(" "acos(" "atan(" *
161.      * "Ans" "e" "pi"     *
162.      *         *
163.      * j x y z k u v w               *
164.      **********************************/
165.     vector<string> s_code = {"sqrt(","sin(","cos(","tan("
    ,"ln(","asin(","acos(","atan(", "e", "pi"};
166.     vector<string> e_code = {"j","x","y","z","k","u","v",
    "w", "2.71828183", "3.14159265"};
167.     for (int j = 0; j < s_code.size(); j++)
168.     {
169.         for (int i = in.find(s_code[j]); i <= in.length()
    ; i = in.find(s_code[j]))
170.         {
171.             in.erase(i, s_code[j].length());
172.             in.insert(i, e_code[j]);
173.         }
174.     }
175.
176.     // assign "Ans"
177.     for (int i = in.find("Ans"); i <= in.length(); i = in
    .find("Ans"))
178.     {
179.         in.erase(i, 3);
180.         in.insert(i, ans);
181.     }
```

```
182.
183.     // check new string
184.     /**********************
185.      * ( ) j x y z k u v w *
186.      * ^       *
187.      * + - * /      *
188.      * . %            *
189.      * 0 1 2 3 4 5 6 7 8 9 *
190.      **********************/
191.     string check_2 = "+-*/()jxyzkuvw^0123456789.%";
192.     for (int i = 0; i < in.length(); i++)
193.     {
194.         if (check_2.find(in[i]) == check_2.npos)return "@
    ";
195.     }
196.
197.     // check legal text
198.     vector<string> check_3 = {
199.         "++", "+-", "+*", "+/", "+^", "+.", "+%",
200.         "--", "-+", "-*", "-/", "-^", "-.", "-%",
201.         "**", "*+", "*-", "*/", "*^", "*.", "*%",
202.         "//", "/+", "/-", "/*", "/^", "/.", "/%",
203.         "^^", "^+", "^-", "^*", "^/", "^.", "^%",
204.         "..", ".+", ".-", ".*", "./", ".^", ".%",
205.         "%.", "(+", "(-", "(*", "(/", "(^", "(.", "(%",
206.         "()", ")+)", "-)", "*)", "/)", "^)", ".)", "%%", "
    )(",
207.         "j)", "j+", "j-
    ", "j*", "j/", "j^", "j.", "j%", ")j",
208.         "x)", "x+", "x-
    ", "x*", "x/", "x^", "x.", "x%", ")x",
209.         "y)", "y+", "y-
    ", "y*", "y/", "y^", "y.", "y%", ")y",
210.         "z)", "z+", "z-
    ", "z*", "z/", "z^", "z.", "z%", ")z",
211.         "k)", "k+", "k-
    ", "k*", "k/", "k^", "k.", "k%", ")k",
212.         "u)", "u+", "u-
    ", "u*", "u/", "u^", "u.", "u%", ")u",
213.         "v)", "v+", "v-
    ", "v*", "v/", "v^", "v.", "v%", ")v",
214.         "w)", "w+", "w-
    ", "w*", "w/", "w^", "w.", "w%", ")w"};
215.     for (int j = 0; j < check_3.size(); j++)
```

```
216.     {
217.         if (in.find(check_3[j]) != in.npos)return "@";
218.     }
219. 
220.     return in;
221. }
222. 
223. vector<string> cut_text(string in) （再点击跳回）
224. {
225.     vector<string> out;
226.     string temp;
227.     for (int i = 0; i < in.length(); i++)
228.     {
229.         if ((in[i] >= '0' && in[i] <= '9') || in[i] == '.
    ' || in[i] == '%')
230.         {
231.             temp += in[i];
232.         }
233.         else
234.         {
235.             if (!temp.empty())
236.             {
237.                 temp += '\0';
238.                 out.push_back(temp);
239.                 temp.clear();
240.             }
241. 
242.             temp = in[i] + '\0';
243.             out.push_back(temp);
244.             temp.clear();
245.         }
246.     }
247.     if (!temp.empty())
248.     {
249.         temp += "\0";
250.         out.push_back(temp);
251.         temp.clear();
252.     }
253.     return out;
254. }
255. 
256. vector<string> sort_text(vector<string> in) （再点击跳
    回）
257. {
```

```
258.      vector<string> out;
259.      stack <string> opt;
260.      for (int i = 0; i < in.size(); i++)
261.      {
262.          if (in[i] != ")" && priority(in[i]) == 0)out.push
   _back(in[i]);
263.          else if (in[i] == ")")
264.          {
265.              while (priority(opt.top()) != MAX_PRIORITY)
266.              {
267.                  out.push_back(opt.top());
268.                  opt.pop();
269.              }
270.              out.push_back(opt.top());
271.              opt.pop();
272.          }
273.          else if (priority(in[i]) == MAX_PRIORITY)opt.push
   (in[i]);
274.          else if ((!opt.empty()) && ((priority(opt.top())
   == MAX_PRIORITY) || (priority(in[i]) > priority(opt.top()))
   ))opt.push(in[i]);
275.          else
276.          {
277.              while (!opt.empty())
278.              {
279.                  if (priority(opt.top()) == MAX_PRIORITY)b
   reak;
280.                  else if (priority(in[i]) <= priority(opt.
   top()))
281.                  {
282.                      out.push_back(opt.top());
283.                      opt.pop();
284.                  }
285.              }
286.              opt.push(in[i]);
287.          }
288.      }
289.      while (!opt.empty())
290.      {
291.          out.push_back(opt.top());
292.          opt.pop();
293.      }
294.      return out;
295.  }
```

```
296.
297.  string calcuate_answer(vector<string> in) （再点击跳回）
298.  {
299.      stack <string> out;
300.      for (int i = 0; i < in.size(); i++)
301.      {
302.          if (priority(in[i]) == 0)out.push(in[i]);
303.          else if(!out.empty())
304.          {
305.              if (priority(in[i]) == MAX_PRIORITY)
306.              {
307.                  string num = out.top();
308.                  out.pop();
309.                  /* "(" equals to no-operation.
310.                  if (in[i] == "(");          */
311.                  if  (in[i] == "j")out.push(s_sqrt(num));
312.                  else if (in[i] == "x")out.push(s_sin (num
    ));
313.                  else if (in[i] == "y")out.push(s_cos (num
    ));
314.                  else if (in[i] == "z")out.push(s_tan (num
    ));
315.                  else if (in[i] == "k")out.push(s_ln  (num
    ));
316.                  else if (in[i] == "u")out.push(s_asin(num
    ));
317.                  else if (in[i] == "v")out.push(s_acos(num
    ));
318.                  else if (in[i] == "w")out.push(s_atan(num
    ));
319.                  else out.push(num);
320.              }
321.              else
322.              {
323.                  string num2 = out.top();
324.                  out.pop();
325.                  if (!out.empty())
326.                  {
327.                      string num1 = out.top();
328.                      out.pop();
329.                           if (in[i] == "+")out.push(s_add(
    num1, num2));
330.                      else if (in[i] == "-
    ")out.push(s_sub(num1, num2));
```

```
331.                     else if (in[i] == "*")out.push(s_mul(num1, num2));
332.                     else if (in[i] == "/")out.push(s_div(num1, num2));
333.                     else if (in[i] == "^")out.push(s_pow(num1, num2));
334.                 }
335.                 else if (in[i] == "%")out.push(s_per(num2));
336.                 else out.push(num2);
337.             }
338.         }
339.     }
340.     return out.top();
341. }
342.
343.
344. string form_answer(string in, int conf) （再点击跳回）
345. {
346.     int len = 0;
347.     in = to_string(digitize(in));
348.     if (conf >= 100)in = s_mul(in, "100");
349.     while (in.length() - in.find(".") - conf % 10 - 2)
350.     {
351.         in.pop_back();
352.     }
353.     if(conf/10%10)in.pop_back();
354.     else
355.     {
356.         if (in[in.length() - 1] > '5')
357.         {
358.             len = in.length() - in.find(".") - 1;
359.             in = to_string(long(digitize(in) * pow(10, len)) + 1).insert(in.length() - len - 1, ".");
360.         }
361.         in.pop_back();
362.     }
363.     if(in[in.length() - 1]=='.')in.pop_back();
364.     if (conf >= 100)in.append("%");
365.     return in;
366. }
367.
368. string calculate(string in, string ans, int conf)
369. {
```

```
370.      string out;
371.      out = form_text(in, ans);
372.      if ( out== "@")
373.      {
374.          out = "Error";
375.          return out;
376.      }
377.      else
378.      {
379.          ans = form_answer(calcuate_answer(sort_text(cut_t
   ext(out))), conf);
380.          return ans;
381.      }
382.  }
```

Main.cpp

```
1. #include "widget.h"
2.
3. #include <QApplication>
4.
5. int main(int argc, char *argv[])
6. {
7.     QCoreApplication::setAttribute(Qt::AA_EnableHighDpiScal
   ing);
8.     QApplication a(argc, argv);
9.
10.    Widget w;
11.    w.show();
12.
13.    return a.exec();
14.}
```