高级语言程序设计

实验报告

南开大学 智能科学与技术
周晨
2011812
智能科学与技术班
2024 年 4 月 13 日

目录

高级语言程序设计大作业实验报告

一、 作业题目

带有图形界面的增强版计算器（Ultra Calculator）：不仅能够实现基础的加、减、乘、除运算，还增添了诸如指数、对数、三角函数（包括 sin、cos、tan）等复杂计算。此外，它还支持多层括号嵌套和不同运算的组合使用。

二、开发软件

Visual Studio 2022：用于整体的程序开发和编译。

Qt Creator 13.0.0：用于增强型计算器的核心开发。

Qt Designer 6.7.0：用于设计计算器的用户界面（UI）。

GitHub：版本控制系统，用于管理项目的多个版本和迭代。

三、 课题要求

1) **基础运算**：支持加法（+）、减法（-）、乘法（×）、除法（÷）
2) **高级数学运算**：包括指数（^）、对数（log）、三角函数（sin, cos, tan）及多层括号嵌套和不同运算的组合运算。
3) **界面与使用方式友好性**:界面需符合用户日常使用习惯，具有美观性和操作便捷性。

4) **错误处理**：能够处理用户输入错误和数学错误

5) **稳定性**：大量测试以验证功能，程序在常规使用下不出现崩溃或未响应

6) **数据安全**：程序不收集或存储用户输入的敏感数据

四、 主要流程

1. 创建 Qt Widget 应用

设计并实现一个用户友好的计算器应用，从创建一个基础的 Qt Widget 应用开始，逐步集成计算逻辑和用户界面。

2. 整体功能构建

原理示意图：

其中，用户界面（User Interface）：定义了应用程序的布局和外观，是用户与程序交互的界面。CalWidget 类（CalWidget Class）：是程序的核心，处理用户的输入，并控制应用程序的响应。数学逻辑（Mathematical Logic）：包括所有数学运算的逻辑，例如基础运算、指数、对数以及三角函数等。

1）用户界面设计：使用 Qt Designer，设计了一个直观的用户界面，用户可以通过点击按钮输入表达式，并实时看到他们的输入和计算结果。UI 设计文件为 calwidget.ui，该文件定义了界面的布局和风格。

2）计算功能实现：计算逻辑被编写在 calwidget.cpp 中，涵盖从基本到高级的所有数学运算。对于三角函数和对数等特殊运算，通过 QJSEngine 实现。

3）代码结构：calwidget.h 定义了 CalWidget 类的接口，包含了所有槽函数

（slots）和私有成员。所有的用户界面交互都通过这些槽函数来管理，确保了界面的响应性和良好的用户体验。

4）错误处理：程序在设计时考虑了多种错误输入情况，并在evaluate-Expression 函数中通过检查匹配的括号数量和输入格式来处理。如果发现错误，UI 会显示相应的错误消息。

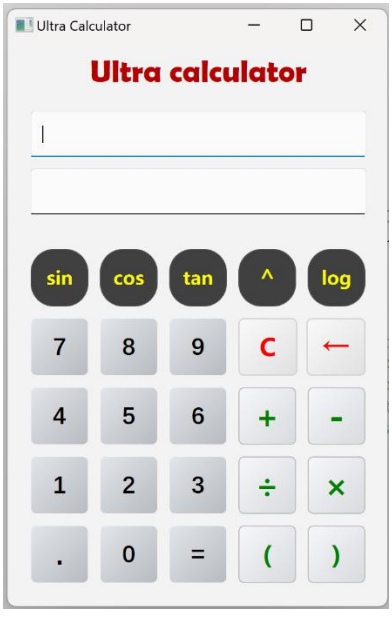
3. 核心功能实现（详细代码见文末链接）

1）CalWidget 类负责整个计算器的运行逻辑和用户交互。

```
7  CalWidget::CalWidget(QWidget *parent) : QWidget(parent), ui(new Ui::CalWidget)
8  {
9      ui->setupUi(this);
10     openParensCount = 0;
11 }
```

2）使用 QJSEngine 来评估用户输入的表达式并执行计算。

3）UI 界面（calwidget.ui）通过 Qt Designer 设计，并具有响应用户输入的按钮。



4）对数学函数如正弦（sin）、余弦（cos）、正切（tan）及对数（log）运算进行了特别处理，在符合用户日常使用习惯的前提下适配 QJSEngine 的语法。例如：使用符号“^”（x^y）而非“pow”；使用 log(x)(y)而不是 log(x,y)。

```cpp
QString CalWidget::evaluateExpression(const QString &expr)
{
    QJSEngine engine;

    // 识别数学函数
    engine.globalObject().setProperty("sin", engine.evaluate("(x) => Math.sin(x)"));
    engine.globalObject().setProperty("cos", engine.evaluate("(x) => Math.cos(x)"));
    engine.globalObject().setProperty("tan", engine.evaluate("(x) => Math.tan(x)"));
    engine.globalObject().setProperty("log", engine.evaluate("(base, antilog) => Math.log(antilog) / Math.log(base)"));
    engine.globalObject().setProperty("pow", engine.evaluate("(x, y) => Math.pow(x, y)"));

    QString jsExpr = expr;

    // 处理对数表达式
    QRegularExpression logRegEx("\\blog\\(([^)]+)\\)\\(([^)]+)\\)");
    jsExpr.replace(logRegEx, "log(\\1, \\2)");    ⚠Don't create temporary QRegularExpression objects. Use a static QRe

    // 处理指数表达式
    QRegularExpression expRegEx("\\b(\\d+|\\))\\s*\\^\\s*\\((\\d+)\\)");
    QRegularExpressionMatchIterator it = expRegEx.globalMatch(jsExpr);    ⚠Don't create temporary QRegularExpression o
    while (it.hasNext()) {
        QRegularExpressionMatch match = it.next();
        QString base = match.captured(1);
        QString exponent = match.captured(2);
        QString replacement = QString("Math.pow(%1, %2)").arg(base, exponent);
        jsExpr.replace(match.capturedStart(0), match.capturedLength(0), replacement);
    }

    qDebug() << "Final JavaScript expression: " << jsExpr;

    QJSValue result = engine.evaluate(jsExpr);
    if (result.isError()) {
        return "Error: " + result.toString();
    }
    return result.toString();
}
```

根据输入类型要求，使用相同的基类指针构造不同的子类对象。使用基类指针虚函数引用调用子类实现，达到多态效果。

4.功能测试

在程序每一个版本实现后，所有功能都经过了严格的单元测试，以确保它们能在各种条件下正常工作。测试范围从简单的加法到复杂的表达式（包括嵌套括号和混合运算）以及对错误处理的测试。

五、 功能测试

1. 测试方法

对每个按钮和计算功能进行测试，确保其按预期工作。此外，还有对错误处理的测试，如输入不匹配的括号。

2. 测试过程

通过模拟用户交互，对每个按钮进行单独测试，确保它们的功能与预期相符。此外，通过构造复杂表达式来测试计算器的计算能力和准确性。

3. 错误修正

在测试过程中发现的任何问题都被记录并修复。例如，在开发日志中提到的除法崩溃问题，通过增加检查和改进算法来解决。

4. 测试结果

最终，经过多次迭代和改进，计算器应用程序在功能、稳定性和用户界面方面都达到了预定的要求。所有基本和高级运算都能正确执行，用户界面清晰、反应灵敏。

5. 测试样例

输入	输出	目的
20+20	40	测试加法运算是否正确
34-87	-53	测试减法运算是否正确
45×374	16830	测试乘法运算是否正确
95÷18	5.27777	测试除法运算是否正确
5^3	125	测试指数运算是否正确
log(3)(67)	3.82727	测试对数运算是否正确
log(6)(6^2)	2	测试嵌套运算是否正确
(log(6)(6^3)-2)×5	5	测试嵌套复合运算是否正确
65×(9	Error: Unmatched parentheses	测试错误处理是否正常

六、 收获与总结

1.收获

1）技术提升：在处理指数、对数和三角函数等复杂数学运算时，深化了对

Qt 框架的理解。

2）问题解决：通过解决各个版本中遇到的问题，如功能实现的复杂性、界面美观性的改善等，提高了解决实际开发问题的能力。

3）用户体验：开发者开发应用软件时，要特别注重用户界面设计以及使用逻辑。应用软件设计出来是给用户使用的，必须符合用户日常使用习惯，简化操作，使得最终产品既美观又易用。例如，大多数用户习惯于使用符号“^”（x^y）而非“pow”；习惯于使用 log(x)(y)而不是 log(x,y)。此时，就要对相关函数进行特别处理，在符合用户日常使用习惯的前提下适配 QJSEngine 的语法。

```
122 QString CalWidget::evaluateExpression(const QString &expr)
123 {
124     QJSEngine engine;
125 
126     // 识别数学函数
127     engine.globalObject().setProperty("sin", engine.evaluate("(x) => Math.sin(x)"));
128     engine.globalObject().setProperty("cos", engine.evaluate("(x) => Math.cos(x)"));
129     engine.globalObject().setProperty("tan", engine.evaluate("(x) => Math.tan(x)"));
130     engine.globalObject().setProperty("log", engine.evaluate("(base, antilog) => Math.log(antilog) / Math.log(base)"));
131     engine.globalObject().setProperty("pow", engine.evaluate("(x, y) => Math.pow(x, y)"));
132 
133     QString jsExpr = expr;
134 
135     // 处理对数表达式
136     QRegularExpression logRegEx("\\blog\\(([^)]+)\\)\\(([^)]+)\\)");
137     jsExpr.replace(logRegEx, "log(\\1, \\2)");    ⚠Don't create temporary QRegularExpression objects. Use a static QRe
138 
139     // 处理指数表达式
140     QRegularExpression expRegEx("\\b(\\d+|\\))\\s*\\^\\s*\\((\\d+)\\)");
141     QRegularExpressionMatchIterator it = expRegEx.globalMatch(jsExpr);    ⚠Don't create temporary QRegularExpression o
142     while (it.hasNext()) {
143         QRegularExpressionMatch match = it.next();
144         QString base = match.captured(1);
145         QString exponent = match.captured(2);
146         QString replacement = QString("Math.pow(%1, %2)").arg(base, exponent);
147         jsExpr.replace(match.capturedStart(0), match.capturedLength(0), replacement);
148     }
149 
150     qDebug() << "Final JavaScript expression: " << jsExpr;
151 
152     QJSValue result = engine.evaluate(jsExpr);
153     if (result.isError()) {
154         return "Error: " + result.toString();
155     }
156     return result.toString();
157 }
```

2.总结

通过本项目的开发，我不仅增强了对 Qt 框架和 C++编程的深入理解，也提高了面向对象设计和用户界面设计的技能。虽然遇到了一些挑战，但通过逐步分析和解决问题，最终实现了一个功能强大且用户友好的计算器应用程序。我将继续完善这款计算器，增加更多功能，并优化用户体验。感谢所有在此项目中给予我帮助和指导的老师和同学们，你们的支持是我不断前进和改进的动力。

附录：

本项目在 GitHub 的网址：https://github.com/Chandler-Recardo-Zhou/Dawn