Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Пермский национальный исследовательский**

**политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**О Т Ч Ё Т**

**по Творческой работе №1**

Дисциплина: основы алгоритмизации и программирования

Тема: Разработка калькулятора

Вариант 7

Выполнил работу

студент группы РИС-20-1б

Коваленко Н. А.

Проверил

Доцент кафедры ИТАС

Полякова О.А.

Пермь, 2021

Постановка задачи

Разработать алгоритм калькулятора в соответствии с заданием.

Реализовать алгоритм в виде программы на алгоритмическом языке С++.

Разработать интерфейс средствами QT.

Углы, измеряемые в градусной системе мер (1 градус=60 минут, 1 минута=60 секунд)

Сложение, вычитание, умножение, деление, нахождение тригонометрических и обратных тригонометрических функций, изменение знака угла.

Анализ задачи

Калькулятор основан на функциях и классах.

При вызове определенной функции, советующей выбранной пользователем операцией, создается объект класса и запускается определенный конструктор для определенных вводимых данных. Связано это с математической стороной градусных единиц измерения. Например, вызов операции сложения подразумевает ввод пары значений градусных измерений (градусы, минуты, секунды), вызов операции поиска синуса подразумевает ввод одного значения градусных измерений (градусы, минуты, секунды)

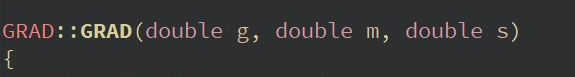
Далее вызывается соответствующий метод для получения результата той или иной операции.

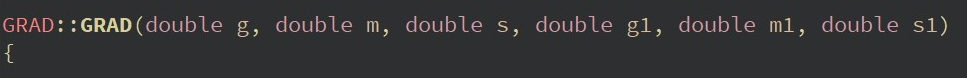
Определить какие операции должны быть выполнены по заданию:

Создание класса GRAD для работы с градусными мерами:

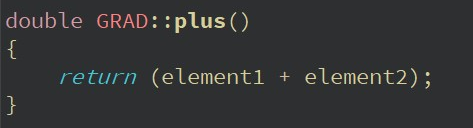


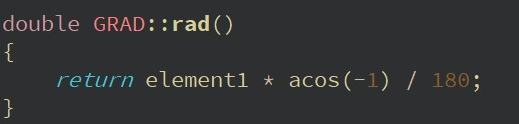
Создание двух типов конструкторов. В зависимости от операции осуществляется вызов одного из конструкторов





Создание методов класса GRAD для вычисления осуществления соответствующих операций, например

 - операция сложения

 - перевод из градусов в радианы

Создание полей класса GRAD для хранения вводимых значений и работы методов класса



Для решения задачи используются глобальные переменные:

Несколько переменных с плавающей точкой для хранения градусных единиц измерения:



Несколько целочисленных переменных для хранения коэффициентов и управления меню:



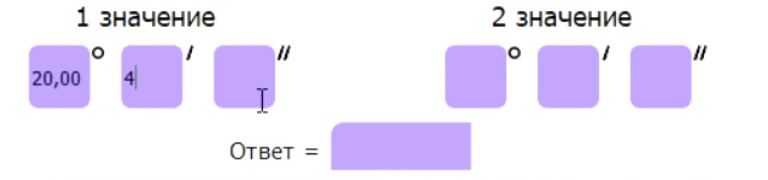
Анализ задачи со стороны дизайна:

Изначально пользователю доступны все операции на выбор:

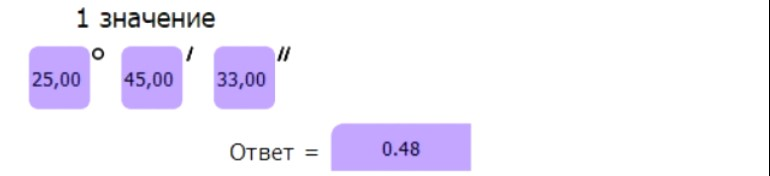


После выбора операции появляются окна для ввода значений. Тип окон зависит от выбранной операции:

Например, при вызове операции сложения появляются окна для ввода двух градусных мер:

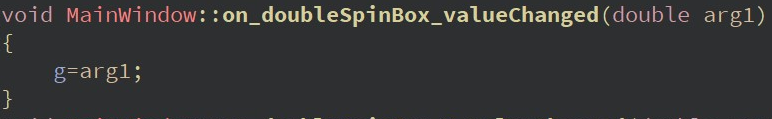


При вызове операции поиска синуса появляется окно для ввода одной градусной меры:

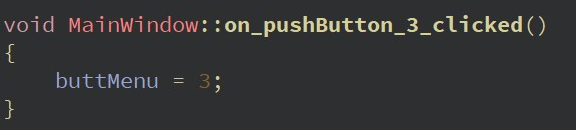


После ввода всех значений, пользователь нажимает на кнопку “answer” и появляется окно для вывода ответа:  

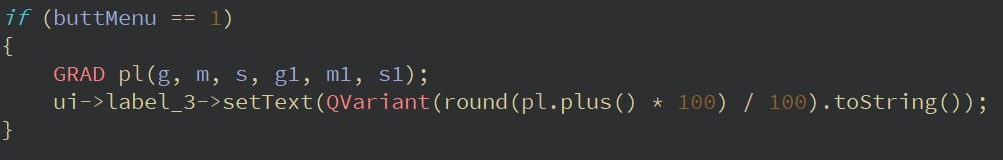

Ввод данных осуществляется посредством виджетов QDoubleSpinBox



Также ввод данных в переменную buttMenu, которая отвечает за выбранную операцию, происходит посредством фиксирования нажатия на кнопку



Вывод данных осуществляется посредством виджетов Label



В калькуляторе используются ограничители вводимых данных. Это связано с областью определения функций арксинуса и арккосинуса. Ограничители настроены свойствами соответствующих QDouble Spin Box-ов.

Код основных методов:

#include "mainwindow.h"

#include "ui\_mainwindow.h"

#include <math.h>

#include "CLASS.h"

#include <iomanip>

#include <cmath>

double g, m, s, g1, m1, s1, ch, zn;

int k, buttMenu;

MainWindow::**MainWindow**(QWidget \*parent)

: QMainWindow(*parent*)

, ui(*new* Ui::MainWindow)

{

ui->setupUi(*this*);

ui->label\_4->hide();

ui->label\_5->hide();

ui->label\_6->hide();

ui->label\_7->hide();

ui->doubleSpinBox->hide();

ui->doubleSpinBox\_2->hide();

ui->doubleSpinBox\_3->hide();

ui->doubleSpinBox\_4->hide();

ui->doubleSpinBox\_5->hide();

ui->doubleSpinBox\_6->hide();

ui->doubleSpinBox\_7->hide();

ui->doubleSpinBox\_8->hide();

ui->doubleSpinBox\_9->hide();

ui->doubleSpinBox->*clear*();

ui->doubleSpinBox\_2->*clear*();

ui->doubleSpinBox\_3->*clear*();

ui->doubleSpinBox\_4->*clear*();

ui->doubleSpinBox\_5->*clear*();

ui->doubleSpinBox\_6->*clear*();

ui->doubleSpinBox\_7->*clear*();

ui->doubleSpinBox\_8->*clear*();

ui->doubleSpinBox\_9->*clear*();

ui->label\_3->hide();

}

MainWindow::~***MainWindow***()

{

*delete* ui;

}

void MainWindow::**on\_doubleSpinBox\_valueChanged**(double arg1)

{

g=arg1;

}

void MainWindow::**on\_doubleSpinBox\_2\_valueChanged**(double arg1)

{

m=arg1;

}

void MainWindow::**on\_doubleSpinBox\_3\_valueChanged**(double arg1)

{

s=arg1;

}

void MainWindow::**on\_doubleSpinBox\_6\_valueChanged**(double arg1)

{

g1=arg1;

}

void MainWindow::**on\_doubleSpinBox\_5\_valueChanged**(double arg1)

{

m1=arg1;

}

void MainWindow::**on\_doubleSpinBox\_4\_valueChanged**(double arg1)

{

s1=arg1;

}

void MainWindow::**on\_doubleSpinBox\_7\_valueChanged**(double arg1)

{

ch = arg1;

}

void MainWindow::**on\_pushButton\_clicked**()

{

buttMenu = 1;

}

void MainWindow::**on\_pushButton\_2\_clicked**()

{

buttMenu = 2;

}

void MainWindow::**on\_pushButton\_3\_clicked**()

{

buttMenu = 3;

}

void MainWindow::**on\_pushButton\_4\_clicked**()

{

buttMenu = 4;

}

void MainWindow::**on\_pushButton\_14\_clicked**()

{

*if* (buttMenu == 1)

{

GRAD pl(g, m, s, g1, m1, s1);

ui->label\_3->setText(QVariant(round(pl.plus() \* 100) / 100).toString());

}

*if* (buttMenu == 2)

{

GRAD min(g, m, s, g1, m1, s1);

ui->label\_3->setText(QVariant(round(min.minus() \* 100) / 100).toString());

}

*if* (buttMenu == 3)

{

GRAD umno(g, m, s);

ui->label\_3->setText(QVariant(round(umno.mylti(ch) \* 100) / 100).toString());

}

*if* (buttMenu == 4)

{

GRAD di(g, m, s);

ui->label\_3->setText(QVariant(round(di.div(ch) \* 100) / 100).toString());

}

*if* (buttMenu == 5)

{

GRAD si(g, m, s);

ui->label\_3->setText(QVariant(round(sin(si.rad()) \* 100) / 100).toString());

}

*if* (buttMenu == 6)

{

GRAD co(g, m, s);

ui->label\_3->setText(QVariant(round(cos(co.rad()) \* 100) / 100).toString());

}

*if* (buttMenu == 7)

{

GRAD ta(g, m, s);

ui->label\_3->setText(QVariant(round(tan(ta.rad()) \* 100) / 100).toString());

}

*if* (buttMenu == 8)

{

GRAD cta(g, m, s);

ui->label\_3->setText(QVariant(round(1/tan(cta.rad()) \* 100) / 100).toString());

}

*if* (buttMenu == 9)

{

double gradus = asin(zn)\*180/acos(-1);

ui->label\_3->setText(QVariant(round(gradus \* 100) / 100).toString());

}

*if* (buttMenu == 10)

{

double gradus = acos(zn)\*180/acos(-1);

ui->label\_3->setText(QVariant(round(gradus \* 100) / 100).toString());

}

*if* (buttMenu == 11)

{

double gradus = atan(zn)\*180/acos(-1);

ui->label\_3->setText(QVariant(round(gradus \* 100) / 100).toString());

}

*if* (buttMenu == 12)

{

double gradus = (1/atan(zn))\*180/acos(-1);

ui->label\_3->setText(QVariant(round(gradus \* 100) / 100).toString());

}

*if* (buttMenu == 13)

{

GRAD radi(g, m, s);

ui->label\_3->setText(QVariant(round(radi.rad() \* 100) / 100).toString());

}

g = m = s = g1 = m1 = s1 = ch = zn = 0;

}

void MainWindow::**on\_pushButton\_5\_clicked**()

{

buttMenu = 5;

}

void MainWindow::**on\_pushButton\_6\_clicked**()

{

buttMenu = 6;

}

void MainWindow::**on\_pushButton\_7\_clicked**()

{

buttMenu = 7;

}

void MainWindow::**on\_pushButton\_8\_clicked**()

{

buttMenu = 8;

}

void MainWindow::**on\_pushButton\_9\_clicked**()

{

buttMenu = 9;

}

void MainWindow::**on\_pushButton\_10\_clicked**()

{

buttMenu = 10;

}

void MainWindow::**on\_pushButton\_11\_clicked**()

{

buttMenu = 11;

}

void MainWindow::**on\_pushButton\_12\_clicked**()

{

buttMenu = 12;

}

void MainWindow::**on\_pushButton\_13\_clicked**()

{

buttMenu = 13;

}

void MainWindow::**on\_doubleSpinBox\_8\_valueChanged**(double arg1)

{

zn = arg1;

}

void MainWindow::**on\_doubleSpinBox\_9\_valueChanged**(double arg1)

{

zn = arg1;

}

Работа калькулятора:

Калькулятор в первичном состоянии:



Калькулятор при выполнении операции сложения (Ответом является значение в дробных градусах):



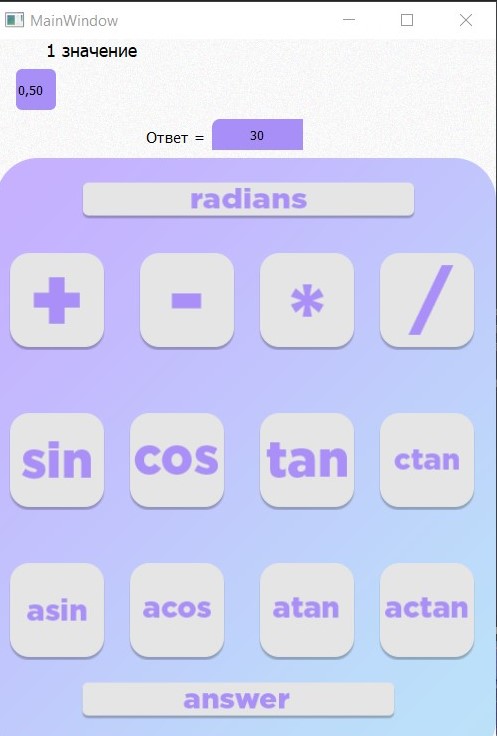
Калькулятор при выполнении операции умножения (Ответом является значение в дробных градусах):



Калькулятор при выполнении операции поиска синуса (Ответом является значение в радианах):



Калькулятор при выполнении операции поиска арксинуса (Ответом является значение в дробных градусах):



Калькулятор при выполнении операции поиска арктангенса (Ответом является значение в дробных градусах):



Калькулятор при выполнении операции перевода из градусов в радианы:



UML Диаграмма:

