МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Забайкальский государственный университет»

(ФГБОУ ВО «ЗабГУ»)

Факультет Энергетический

Кафедра Информатики, вычислительной техники и прикладной математики

**КОНТРОЛЬНАЯ РАБОТА**

по дисциплине

«Объектно-ориентированное программирование»

Выполнил ст. гр. ИВТз-18

(группа)

Токпанов Т. К.

(фамилия, инициалы)

Проверил

Старший преподаватель

Ветров С.В.

Чита 2021 г.

Задание 1. Простой класс

Задание 1. Абстрактный тип данных: класс.

А) Описать абстрактный тип данных – дата:

ADT Date.

Данные:

year – год;

month – месяц;

day – день;

Операции:

Конструктор:

Процесс - задать значения по умолчанию:

year = 1;

month = 1;

day = 1;

Операция – задать год;

Вход: значение year;

Предусловия: от 1 до 2100;

Процесс: задаем year;

Выход: нет;

Постусловие: нет;

Операция – задать месяц;

Вход: значение month;

Предусловия: от 1 до 12;

Процесс: задаем month;

Выход: нет;

Постусловие: нет;

Операция – задать день;

Вход: значение day;

Предусловия: от 1 до 30;

Процесс: задаем day;

Выход: нет;

Постусловие: нет;

Операция – прочитать значение год;

Вход: нет;

Предусловия: нет;

Процесс: читаем значение year;

Выход: значение year;

Постусловие: нет;

Операция – прочитать значение месяц;

Вход: нет;

Предусловия: нет;

Процесс: читаем значение month;

Выход: значение month;

Постусловие: нет;

Операция – прочитать значение день;

Вход: нет;

Предусловия: нет;

Процесс: читаем значение day;

Выход: значение day;

Постусловие: нет;

Операция – добавить к текущей дате n-дней;

Вход: значение n;

Предусловия: при n >=30 -пересчитывать месяцы, при >=360 -годы;

Процесс: задаем n;

Выход: нет;

Постусловие: нет;

Конец ADT Date.

Б) Представить в нотации UML.

|  |
| --- |
| Class |
| Date  year: integer  month: integer  day: integer |
| setYear ()  setMonth ()  setDay ()  addDay ()  getYear ()  getMonth ()  getDay ()  Date\_to\_string () |

3) Описать класс на С++

date.h

#ifndef DATE\_H

#define DATE\_H

#include <iostream>

#include <string>

using namespace std;

class Date {

private:

// polya classa

int year;

int month;

int day;

public:

// constructor

Date();

//operations

void setYear(int yy);

void setMonth(int mm);

void setDay(int dd);

void AddDay(int add);

int getYear();

int getMonth();

int getDay();

string Date\_to\_string();

};

#endif // DATE\_H

main.cpp

#include<iostream>

#include "date.h"

using namespace std;

int main() {

Date dmy;

int dd, mm, yy, n;

cout << "Year?" << endl;

cin >> yy;

dmy.setYear(yy);

cout << "Month?" << endl;

cin >> mm;

dmy.setMonth(mm);

cout << "Day?" << endl;

cin >> dd;

dmy.setDay(dd);

cout << dmy.Date\_to\_string() << endl;

cout << "add day?" << endl << "n = ";

cin >> n;

dmy.AddDay(n);

cout << dmy.Date\_to\_string();

return 0;

}

date.cpp

#include <iostream>

#include "date.h"

using namespace std;

// constructor

Date::Date() {

year = 1;

month = 1;

day = 1;

}

//operation

void Date::setYear(int yy) { // set year

if (1 <= yy)

year = yy;

}

void Date::setMonth(int mm) { // set month

if (mm <= 12 && mm >= 1)

month = mm;

}

void Date::setDay(int dd) { // set day

if (dd <= 31 && dd >= 1)

if (month != 2 && month != 4 && month != 6 && month != 9 && month != 11)

day = dd;

else

{

if (month != 1 && month != 2 && month != 3 && month != 5 && month != 7 &&

month != 8 && month != 10 && month != 12)

day = dd;

else

{

if (month == 2 && ((year % 4) == 0 && dd <= 29))

day = dd;

else if (dd <= 28)

day = dd;

}

}

}

void Date::AddDay(int add) { // add day

if (0 < add)

{

month = month + add / 30;

day = day + add % 30;

if (day > 30) {

day = day % 30;

month = month + 1;

}

year = year + month / 12;

if (month > 12) {

month = month % 12;

}

}

}

int Date::getYear() {

return year;

}

int Date::getMonth() {

return month;

}

int Date::getDay() {

return day;

}

string Date::Date\_to\_string() {

return to\_string(day) + "/" + to\_string(month) + "/" + to\_string(year);

}

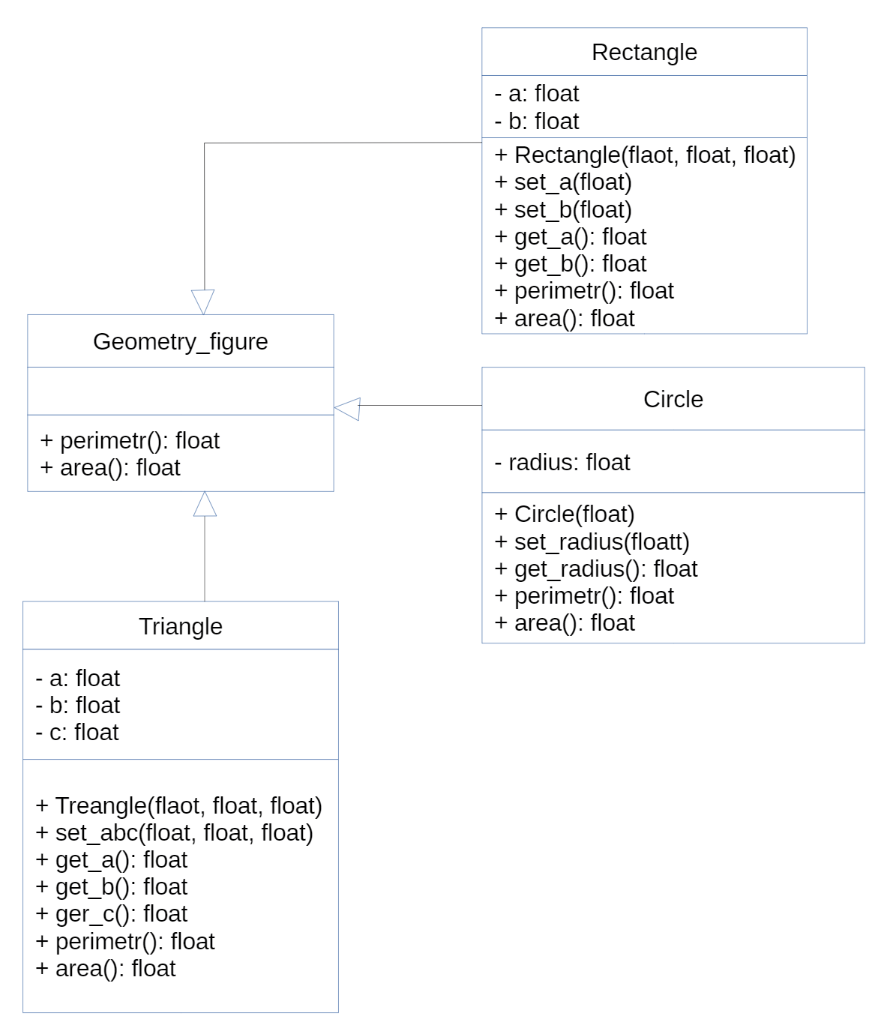
Задание 2. Наследование

Опишем UML диаграмму классов геометрических фигур.

Классы Rectangle, Circle и Triangle являются наследниками родительского класса Geometry\_figure.

Родительский класс Geometry\_figure имеет 2 виртуальных метода для расчета периметра и площади.

Все методы наследуются классами Rectangle, Circle и Triangle. Классы-наследники в свою очередь имеют свои методы для расчета площади и периметра фигур.



Реализуем эти классы на C++

main.h

#include <iostream>

using namespace std;

const float pi = 3.14159265358979323846;

class Geometry\_figure

{

public:

virtual float perimetr() = 0;

virtual float area() = 0;

};

class Triangle : public Geometry\_figure

{

private:

float its\_a, its\_b, its\_c;

public:

Triangle(float a, float b, float c);

void set\_abc(float a, float b, float c);

float get\_a();

float get\_b();

float get\_c();

float perimetr();

float area();

};

class Rectangle : public Geometry\_figure

{

private:

float its\_a, its\_b;

public:

Rectangle(float a, float b);

void set\_a(float a);

void set\_b(float b);

float get\_a();

float get\_b();

float perimetr();

float area();

};

class Circle : public Geometry\_figure

{

private:

float its\_radius;

public:

Circle(float radius);

void set\_radius(float radius);

float get\_radius();

float perimetr();

float area();

};

main.cpp

#include "main.h"

#include <iostream>

using namespace std;

int main()

{

setlocale(LC\_ALL, "Russian");

Triangle T(2, 2, 2);

Rectangle R(5, 4);

Circle C(3);

cout << "Сторона a треугольника равна: " << T.get\_a() << endl;

cout << "Сторона b треугольника равна: " << T.get\_b() << endl;

cout << "Сторона c треугольника равна: " << T.get\_c() << endl;

cout << "Сторона a прямоугольника равна: " << R.get\_a() << endl;

cout << "Сторона b прямоугольника равна: " << R.get\_b() << endl;

cout << "Радиус окружности: " << C.get\_radius() << endl;

cout<<"Площадь теругольника: "<<T.area()<<" Периметр: "<<T.perimetr()<<endl;

cout<<"Площадь прямоугольника: "<<R.area()<<" Периметр: "<<R.perimetr()<<endl;

cout<<"Площадь круга: "<<C.area()<<endl;

T.set\_abc(5,4,6);

R.set\_b(10);

C.set\_radius(20);

cout << "Сторона a треугольника равна: " << T.get\_a() << endl;

cout << "Сторона b прямоугольника равна: " << R.get\_b() << endl;

cout << "Радиус окружности: " << C.get\_radius() << endl;

return 0;

}

methods.cpp

#include "main.h"

#include <cmath>

Triangle::Triangle(float a, float b, float c)

{

set\_abc(a, b, c);

}

void Triangle::set\_abc(float a, float b, float c)

{

if (a > 0 and b > 0 and c > 0)

{

if (a + b > c and b + c > a and a + c > b)

{

its\_a = a;

its\_b = b;

its\_c = c;

}

}

}

float Triangle::get\_a()

{

return its\_a;

}

float Triangle::get\_b()

{

return its\_b;

}

float Triangle::get\_c()

{

return its\_c;

}

float Triangle::perimetr()

{

return its\_a + its\_b + its\_c;

}

float Triangle::area()

{

float p = perimetr() / 2;

return sqrt(p \* (p - its\_a) \* (p - its\_b) \* (p - its\_c));

}

Rectangle::Rectangle(float a, float b)

{

if (a > 0 and b > 0)

{

its\_a = a;

its\_b = b;

}

}

void Rectangle::set\_a(float a)

{

if (a > 0)

{

its\_a = a;

}

}

void Rectangle::set\_b(float b)

{

if (b > 0)

{

its\_b = b;

}

}

float Rectangle::get\_a()

{

return its\_a;

}

float Rectangle::get\_b()

{

return its\_b;

}

float Rectangle::perimetr()

{

return (its\_a + its\_b) \* 2;

}

float Rectangle::area()

{

return its\_a \* its\_b;

}

Circle::Circle(float radius)

{

if (radius > 0)

{

its\_radius = radius;

}

}

void Circle::set\_radius(float radius)

{

if (radius > 0)

{

its\_radius = radius;

}

}

float Circle::get\_radius()

{

return its\_radius;

}

float Circle::perimetr()

{

return pi \* its\_radius \* its\_radius;

}

float Circle::area()

{

return 2 \* pi \* its\_radius;

}

Задание 3. Калькулятор.

Создать калькулятор с графическим интерфейсом пользователя.

Калькулятор должен корректно обрабатывать любые входные данные.

Хранить историю вычислений. Помимо арифметических операций и

возведения в любою степень калькулятор должен вычислять

функции: sin, cos, tan, ln, exp. Использовать шаблон проектирования

Представление-Модель.

main.cpp

#include "mainwindow.h"

#include <QApplication>

int main(int argc, char \*argv[])

{

QApplication a(*argc*, argv);

MainWindow w;

w.show();

return a.exec();

}

mainwindow.cpp

#include <cmath>

#include "mainwindow.h"

#include "ui\_mainwindow.h"

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

: QMainWindow(parent)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

}

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

{

delete ui;

}

// цифра 1

void MainWindow::**on\_push\_button1\_clicked**()

{

QString s;

s=ui->line\_edit->text();

ui->line\_edit->setText(s+"1");

}

// цифра 2

void MainWindow::**on\_push\_button2\_clicked**()

{

QString s;

s=ui->line\_edit->text();

ui->line\_edit->setText(s+"2");

}

// цифра 3

void MainWindow::**on\_push\_button3\_clicked**()

{

QString s;

s=ui->line\_edit->text();

ui->line\_edit->setText(s+"3");

}

// цифра 4

void MainWindow::**on\_push\_button4\_clicked**()

{

QString s;

s=ui->line\_edit->text();

ui->line\_edit->setText(s+"4");

}

// цифра 5

void MainWindow::**on\_push\_button5\_clicked**()

{

QString s;

s=ui->line\_edit->text();

ui->line\_edit->setText(s+"5");

}

// цифра 6

void MainWindow::**on\_push\_button6\_clicked**()

{

QString s;

s=ui->line\_edit->text();

ui->line\_edit->setText(s+"6");

}

// цифра 7

void MainWindow::**on\_push\_button7\_clicked**()

{

QString s;

s=ui->line\_edit->text();

ui->line\_edit->setText(s+"7");

}

// цифра 8

void MainWindow::**on\_push\_button8\_clicked**()

{

QString s;

s=ui->line\_edit->text();

ui->line\_edit->setText(s+"8");

}

// цифра 9

void MainWindow::**on\_push\_button9\_clicked**()

{

QString s;

s=ui->line\_edit->text();

ui->line\_edit->setText(s+"9");

}

// цифра 0

void MainWindow::**on\_push\_button\_zero\_clicked**()

{

QString s;

s=ui->line\_edit->text();

ui->line\_edit->setText(s+"0");

}

// ","

void MainWindow::**on\_push\_button\_comma\_clicked**()

{

QString s;

if(!(ui->line\_edit->text().contains('.')))

ui->line\_edit->setText(ui->line\_edit->text()+'.');

}

// +

void MainWindow::**on\_push\_button\_plus\_clicked**()

{// при нажатии плюс значение поля запишется в первое значение

operation = "+";

first\_value=ui->line\_edit->text().toFloat();

ui->line\_edit->clear();

}

// -

void MainWindow::**on\_push\_button\_minus\_clicked**()

{

operation = "-";

first\_value=ui->line\_edit->text().toFloat();

ui->line\_edit->clear();

}

// \*

void MainWindow::**on\_push\_button\_multiply\_clicked**()

{

operation = "\*";

first\_value=ui->line\_edit->text().toFloat();

ui->line\_edit->clear();

}

// /

void MainWindow::**on\_push\_button\_division\_clicked**()

{

operation = "/";

first\_value=ui->line\_edit->text().toFloat();

ui->line\_edit->clear();

}

// =

void MainWindow::**on\_push\_button\_calc\_clicked**()

{

second\_value=ui->line\_edit->text().toFloat();

if (operation=="+")

{result =first\_value+second\_value;}

if (operation=="-")

{result=first\_value-second\_value;}

if (operation=="\*")

{result=first\_value\*second\_value;}

if (operation=="/")

{result=first\_value/second\_value;}

if (operation=="sin")

{ PI=3.14159265;

value=first\_value\*PI/180;

result=sin(value);} //вычисляем синус

if (operation=="cos")

{ value=first\_value\*PI/180;

result=cos(value);} //вычисляем косинус

if (operation=="tng")

{ PI=3.14159265;

value=first\_value\*PI/180;

result=tan(value);} //вычисляем тангенс

if (operation=="pow")

{result=pow(first\_value,second\_value);} //возведение в степень

if(operation=="ln") //вычисляем натуральный логарифм

{result=log(first\_value);}

if(operation=="exp") //вычисляем экпоненту

{result=exp(first\_value);}

ui->line\_edit->setText(QString::number(result));

first\_value = result;

}

void MainWindow::**on\_push\_buttonC\_clicked**()

{

first\_value = 0;

ui->line\_edit->clear();

}

void MainWindow::**on\_push\_button\_sin\_clicked**()

{

operation="sin";

first\_value=ui->line\_edit->text().toFloat();

ui->line\_edit->clear();

}

void MainWindow::**on\_push\_button\_cos\_clicked**()

{

operation="cos";

first\_value=ui->line\_edit->text().toFloat();

ui->line\_edit->clear();

}

void MainWindow::**on\_push\_button\_tng\_clicked**()

{

operation="tng";

first\_value=ui->line\_edit->text().toFloat();

ui->line\_edit->clear();

}

void MainWindow::**on\_push\_button\_pow\_clicked**()

{

operation="pow";

first\_value=ui->line\_edit->text().toFloat();

ui->line\_edit->clear();

}

void MainWindow::**on\_push\_button\_ln\_clicked**()

{

operation="ln";

first\_value=ui->line\_edit->text().toFloat();

ui->line\_edit->clear();

}

void MainWindow::**on\_push\_button\_exp\_clicked**()

{

operation="exp";

first\_value=ui->line\_edit->text().toFloat();

ui->line\_edit->clear();

}

mainwindow.h

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QMainWindow>

QT\_BEGIN\_NAMESPACE

namespace **Ui** { class **MainWindow**; }

QT\_END\_NAMESPACE

class **MainWindow** : public QMainWindow

{

Q\_OBJECT

public:

**MainWindow**(QWidget \*parent = nullptr);

~***MainWindow***();

private slots:

void **on\_push\_button1\_clicked**();

void **on\_push\_button\_plus\_clicked**();

void **on\_push\_button\_calc\_clicked**();

void **on\_push\_button2\_clicked**();

void **on\_push\_button3\_clicked**();

void **on\_push\_button4\_clicked**();

void **on\_push\_button5\_clicked**();

void **on\_push\_button6\_clicked**();

void **on\_push\_button7\_clicked**();

void **on\_push\_button8\_clicked**();

void **on\_push\_button9\_clicked**();

void **on\_push\_buttonC\_clicked**();

void **on\_push\_button\_minus\_clicked**();

void **on\_push\_button\_multiply\_clicked**();

void **on\_push\_button\_division\_clicked**();

void **on\_push\_button\_comma\_clicked**();

void **on\_push\_button\_zero\_clicked**();

void **on\_push\_button\_sin\_clicked**();

void **on\_push\_button\_cos\_clicked**();

void **on\_push\_button\_tng\_clicked**();

void **on\_push\_button\_pow\_clicked**();

void **on\_push\_button\_ln\_clicked**();

void **on\_push\_button\_exp\_clicked**();

private:

Ui::MainWindow \*ui;

float first\_value;

float second\_value;

float result;

float PI;

float value;

QString operation;

};

#endif // MAINWINDOW\_H

