МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Российский государственный университет им. А.Н. Косыгина   
(Технологии. Дизайн. Искусство)»**

Институт ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ И ЦИФРОВЫЕ ТРАНСФОРМАЦИИ

**Отчет по лабораторной работе № 5**

**по дисциплине «Объектно-ориентированное программирование»**

**Тема: «Операторные функции. Реализация класса MyInt для перегрузки операций»**

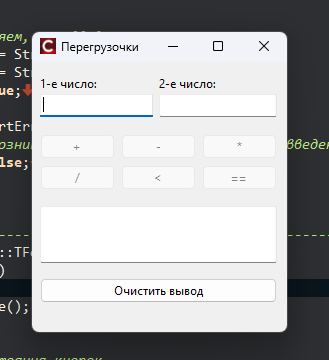
Выполнил: Сидоров Д. С., группа ИТС-123

Проверил: к.т.н., доц. Семёнов А. А.

Москва, 2025г

**Операторные функции. Реализация класса MyInt для перегрузки операций**

Задание: изучить и повторить проиллюстрированный ниже пример, демонстрирующий реализацию класса MyInt, демонстрирующего перегрузку арифметических операций и операций сравнения для объектов этого класса. В классе нужно реализовать нулевой конструктор, инициализирующий поля нулевыми значениями, и ненулевой, инициализирующий поля класса значениями. Необходимо создать семь методов класса, которые будут реализовывать операции +, -, \*, /, <, == с объектами класса и выводить результат в Memo. Повторив рассмотренный пример, необходимо самостоятельно доделать реализацию кнопок -, \*, / (обработать ситуацию деления на ноль), ==. Добавить фильтр на ввод ошибочных данных, переместить класс в unit1.h.

  
Рисунок 1. Пример работающего приложения.

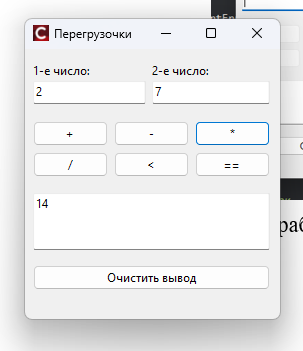


Рисунок 2. Приложение с активным интерфейсом

**Листинг кода**

**Unit1.cpp**

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit1.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm1 \*Form1;

// Проверка валидности введенных значений

bool \_\_fastcall TForm1::IsInputValid()

{

// Проверяем, что поля не пустые

if (LabeledEdit1->Text.Trim() == "" || LabeledEdit2->Text.Trim() == "") {

return false;

}

try {

// Проверяем, что введены числа

int val1 = StrToInt(LabeledEdit1->Text);

int val2 = StrToInt(LabeledEdit2->Text);

return true;

}

catch (EConvertError &e) {

// Если возникла ошибка конвертации - значит введены не числа

return false;

}

}

//---------------------------------------------------------------------------

\_\_fastcall TForm1::TForm1(TComponent\* Owner)

: TForm(Owner)

{

UpdateButtonsState();

}

// Обновление состояния кнопок

void \_\_fastcall TForm1::UpdateButtonsState()

{

bool isValid = IsInputValid();

// Обновляем состояние всех кнопок операций

Button1->Enabled = isValid;

Button2->Enabled = isValid;

Button3->Enabled = isValid;

Button4->Enabled = isValid;

Button5->Enabled = isValid;

Button6->Enabled = isValid;

}

// Обработчик изменения текста в первом поле ввода

void \_\_fastcall TForm1::LabeledEdit1Change(TObject \*Sender)

{

UpdateButtonsState();

}

//---------------------------------------------------------------------------

// Обработчик изменения текста во втором поле ввода

void \_\_fastcall TForm1::LabeledEdit2Change(TObject \*Sender)

{

UpdateButtonsState();

}

class MyInt {

private:

int I;

public:

MyInt() { I=0; }

MyInt(int i) { I=i; }

void show(){

Form1->Memo1->Lines->Add(I);

}

// Оператор сложения (уже есть)

MyInt operator+(MyInt a2) {

return (I + a2.I);

}

// Оператор вычитания

MyInt operator-(MyInt a2) {

return (I - a2.I);

}

// Оператор умножения

MyInt operator\*(MyInt a2) {

return (I \* a2.I);

}

// Оператор деления

MyInt operator/(MyInt a2) {

if (a2.I == 0) {

Form1->Memo1->Lines->Add("Ошибка: деление на ноль!");

return 0;

}

return (I / a2.I);

}

// Оператор сравнения < (уже есть)

bool operator<(MyInt a2) {

if(I < a2.I) return true;

else return false;

}

// Оператор равенства

bool operator==(MyInt a2) {

return (I == a2.I);

}

};

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button1Click(TObject \*Sender)

{

// Сложение

MyInt i1, i2, i3;

i1 = StrToInt(LabeledEdit1->Text);

i2 = StrToInt(LabeledEdit2->Text);

i3 = i1 + i2;

i3.show();

}

void \_\_fastcall TForm1::Button2Click(TObject \*Sender)

{

// Вычитание

MyInt i1, i2, i3;

i1 = StrToInt(LabeledEdit1->Text);

i2 = StrToInt(LabeledEdit2->Text);

i3 = i1 - i2;

i3.show();

}

void \_\_fastcall TForm1::Button3Click(TObject \*Sender)

{

// Умножение

MyInt i1, i2, i3;

i1 = StrToInt(LabeledEdit1->Text);

i2 = StrToInt(LabeledEdit2->Text);

i3 = i1 \* i2;

i3.show();

}

void \_\_fastcall TForm1::Button4Click(TObject \*Sender)

{

// Деление

MyInt i1, i2, i3;

i1 = StrToInt(LabeledEdit1->Text);

i2 = StrToInt(LabeledEdit2->Text);

i3 = i1 / i2;

i3.show();

}

void \_\_fastcall TForm1::Button5Click(TObject \*Sender)

{

// Сравнение <

MyInt i1, i2, i3;

i1 = StrToInt(LabeledEdit1->Text);

i2 = StrToInt(LabeledEdit2->Text);

if(i1 < i2) Form1->Memo1->Lines->Add("Истина");

else Form1->Memo1->Lines->Add("ЛОЖЬ");

}

void \_\_fastcall TForm1::Button6Click(TObject \*Sender)

{

// Сравнение ==

MyInt i1, i2, i3;

i1 = StrToInt(LabeledEdit1->Text);

i2 = StrToInt(LabeledEdit2->Text);

if(i1 == i2) Form1->Memo1->Lines->Add("Истина");

else Form1->Memo1->Lines->Add("ЛОЖЬ");

}

void \_\_fastcall TForm1::Button7Click(TObject \*Sender)

{

Form1->Memo1->Clear();

}

**Unit1.h**

//---------------------------------------------------------------------------

#ifndef Unit1H

#define Unit1H

//---------------------------------------------------------------------------

#include <System.Classes.hpp>

#include <Vcl.Controls.hpp>

#include <Vcl.StdCtrls.hpp>

#include <Vcl.Forms.hpp>

#include <Vcl.ExtCtrls.hpp>

#include <Vcl.Mask.hpp>

//---------------------------------------------------------------------------

class TForm1 : public TForm

{

\_\_published: // IDE-managed Components

TLabeledEdit \*LabeledEdit1;

TLabeledEdit \*LabeledEdit2;

TButton \*Button1;

TButton \*Button2;

TButton \*Button3;

TButton \*Button4;

TButton \*Button5;

TButton \*Button6;

TMemo \*Memo1;

TButton \*Button7;

void \_\_fastcall Button1Click(TObject \*Sender);

void \_\_fastcall Button2Click(TObject \*Sender);

void \_\_fastcall Button3Click(TObject \*Sender);

void \_\_fastcall Button4Click(TObject \*Sender);

void \_\_fastcall Button5Click(TObject \*Sender);

void \_\_fastcall Button6Click(TObject \*Sender);

void \_\_fastcall Button7Click(TObject \*Sender);

void \_\_fastcall LabeledEdit1Change(TObject \*Sender);

void \_\_fastcall LabeledEdit2Change(TObject \*Sender);

private: // User declarations

bool \_\_fastcall IsInputValid();

void \_\_fastcall UpdateButtonsState();

public: // User declarations

\_\_fastcall TForm1(TComponent\* Owner);

};

//---------------------------------------------------------------------------

extern PACKAGE TForm1 \*Form1;

//---------------------------------------------------------------------------

#endif

**Вывод:** В ходе выполнения лабораторной работы был разработан класс MyInt, позволяющий перегружать арифметические операции для работы с целыми числами. Были успешно реализованы операторные функции для основных арифметических операций, операций сравнения и унарных операций. Изучены принципы перегрузки операторов как методов класса и как внешних функций. Работа с классом MyInt подтвердила возможность создания пользовательских типов данных с интуитивно понятным синтаксисом операций. Данная реализация продемонстрировала важность и удобство механизма перегрузки операторов в C++ для повышения читаемости кода и расширения функциональности пользовательских типов данных.