МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

«Национальный исследовательский  
Нижегородский государственный университет им. Н.И. Лобачевского»

(ННГУ)

Институт информационных технологий, математики и механики

ЛАБОРАТОРНАЯ РАБОТА

«Паровозик из Ромашкова»

Выполнил: студент группы 381603-4

Леонтьев Вадим Евгеньевич

Подпись \_\_\_\_\_\_\_\_\_\_\_\_

Руководитель:

Барышева Ирина Викторовна

Подпись \_\_\_\_\_\_\_\_\_\_\_\_

Нижний Новгород

2017

Оглавление

[Введение. 3](#_Toc501401894)

[1. Постановка задачи 4](#_Toc501401895)

[2. Структура проекта 5](#_Toc501401896)

[3. Инструкция для пользователя 6](#_Toc501401897)

[4. Контрольный пример 8](#_Toc501401898)

[Заключение 9](#_Toc501401899)

[Список литературы 10](#_Toc501401900)

[Приложение 11](#_Toc501401901)

[Программа с maim 11](#_Toc501401902)

[Класс myRecttangle 11](#_Toc501401903)

[Класс windo 12](#_Toc501401904)

[Класс door 12](#_Toc501401905)

[Класс corpus 12](#_Toc501401906)

[Класс wheel 12](#_Toc501401907)

[Класс carriage 13](#_Toc501401908)

[Класс truba 14](#_Toc501401909)

[Класс myline 14](#_Toc501401910)

[Класс scepka 14](#_Toc501401911)

[Класс engene 15](#_Toc501401912)

[Класс parovoz 15](#_Toc501401913)

[Класс train 17](#_Toc501401914)

[Описание Wondows Form 18](#_Toc501401915)

Введение.

Визуальное программирование.

Объектно - ориентированное программирование (ООП) — методология программирования, основанная на представлении программы в виде совокупности объектов, каждый из которых является экземпляром определенного класса, а классы образуют иерархию наследования.

Выделяют четыре основных аспекта ООП.

Абстракция данных. Абстрагирование означает выделение значимой информации и исключение из рассмотрения незначимой.

Инкапсуляция. Инкапсуляция – свойство системы, позволяющее объединить данные и методы, работающие с ними, в классе.

Наследование – свойство системы, позволяющее описать новый класс на основе уже существующего с частично или полностью заимствующейся функциональностью. Класс, от которого производится наследование, называется базовым, родительским или суперклассом. Новый класс — потомком, наследником, дочерним или производным классом.

Полиморфизм подтипов. Полиморфизм подтипов – свойство системы, позволяющее использовать объекты с одинаковым интерфейсом без информации о типе и внутренней структуре объекта.

Понятия.

Класс — универсальный, комплексный тип данных, состоящий из тематически единого набора «полей» и «методов», то есть он является моделью информационной сущности с внутренним и внешним интерфейсами для оперирования своим содержимым. В частности, в классах широко используются специальные блоки из одного или чаще двух спаренных методов, отвечающих за элементарные операции с определенным полем. Другим проявлением интерфейсной природы класса является то, что при копировании соответствующей переменной через присваивание, копируется только интерфейс, но не сами данные, то есть класс – ссылочный тип данных.

Объект – сущность в адресном пространстве вычислительной системы, появляющаяся при создании экземпляра класса.

В рамках выполнения лабораторной работы наравне с ООП рассматриваются инструменты автоматизации создания графического интерфейса. Визуальное программирование – способ создания программы для ЭВМ путём использования графических объектов вместо написания её текста. Визуальное программирование часто представляют как следующий этап развития текстовых языков программирования. Наглядным примером может служить утилита Microsoft Visual Studio, где редактируются графические объекты и одновременно отображается соответствующий текст программы. В последнее время визуальному программированию стали уделять больше.

Визуальное программирование в основном используется для создания программ с графическим интерфейсом для операционных систем с графическим интерфейсом пользователя. Среда визуального программирования позволяет написать Веб-приложение для браузеров. Среда визуального программирования позволяет создать консольное.

Одним из примеров визуального программирования является язык CLR компании Micosoft, входящий в пакет Visual Studio.

Система разработки программного обеспечения Microsoft Visual Studio 2010 является продуктом номер один на рынке программного обеспечения. Используя эту систему, можно «малой кровью» и очень быстро написать, почти сконструировать, как в детском конструкторе, довольно-таки функционально сложные как настольные приложения, так и приложения, исполняемые в браузере. В центре системы Visual Studio 2010 находится среда программирования или платформа .NET Framework Платформа .NET Framework предоставляет среду управляемого выполнения, возможности упрощения разработки и развертывания.

1. Постановка задачи

Целью данной работы является демонстрация основных принципов ООП на небольшом примере, а также демонстрация возможностей визуального программирования с использованием инструмента CLR в Microsoft Visual Studio.

Приложение должно осуществлять графическое построение сложного объекта (поезд) на основании введенных данных:

1. Положение на экране.
2. Количество вагонов.
3. Масштаб.
4. Скорость.

Приложение должно быть снабжено оконной оболочкой с возможностью ввода данных. При нажатии на кнопку приложение должно производить забор исходных данных и построение по ним объекта. Вторая кнопка запускает простейшую анимацию – движение поезда по горизонтальной оси с заданной скоростью.

В реализации проекта каждый логическая сущность соответствует определенному классу. Классы связаны иерархическими (класс – наследник класса) и композиционными (класс – поле класса) связями. Действия, осуществляемые определенным объектом, должны быть скрыты от вызывающего объекта (инкапсуляция). Классы, имеющие общие поля и методы, должны быть связаны иерархией наследования. Вызов методов с одинаковыми смысловыми функциями должен происходить одноименным образом (полиморфизм).

2. Структура проекта

Для решения поставленной задачи разработан проект со структурой построения классов UML (Рисунок 1).

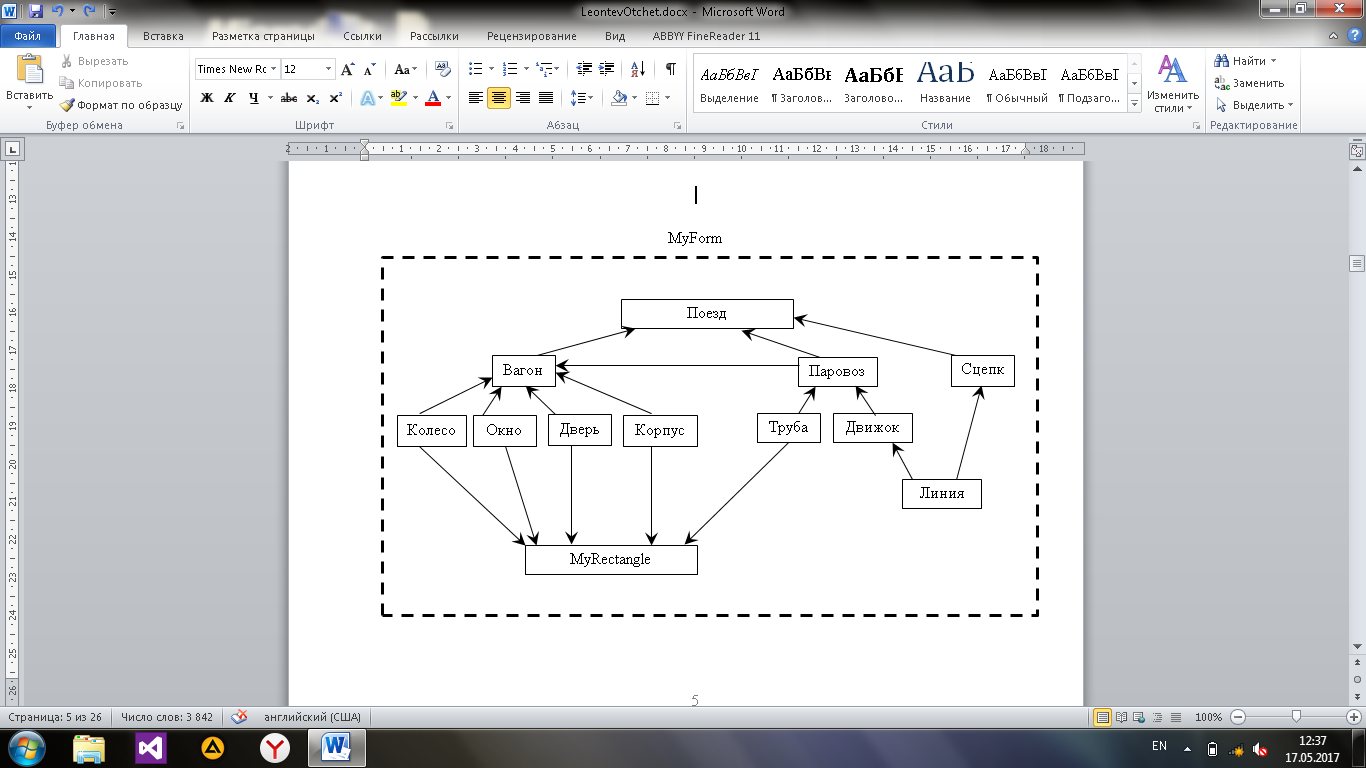


Рисунок 1. Схема проекта.

Проект располагается в форме (Windows Form) описание которой находится в Приложении «Windows Form». Форма строиться на классах:

1. MyRectangle (описание класса находится в Приложении «Класс MyRectangle»), который служит для создания и заливки цветом различных прямоугольников.
2. Линия (описание класса находится в Приложении «Класс myline»), служащая для создание сцепок и движка.
3. Колесо (описание класса находится в Приложении «Класс whill»), описание метода создание и заливки цветом круга.
4. Окно (описание класса находится в Приложении «Класс windo»), для создании прямоугольника, служащим окном.
5. Дверь (описание класса находится в Приложении «Класс door»), служащий для создания двери.
6. Корпус (описание класса находится в Приложении «Класс corpus»), служащий корпусом
7. Труба (описание класса находится в Приложении «Класс tube»), создаёт и заливает прямоугольник для трубы.
8. Движок (описание класса находится в Приложении «Класс engin»), состаящая из 3х линий.
9. Вагон (описание класса находится в Приложении «Класс carriage»)
10. Паровоз (описание класса находится в Приложении «Класс parov»)
11. Сцепка (описание класса находится в Приложении «Класс scepka»)
12. Поезд (описание класса находится в Приложении «Класс Train»)

3. Инструкция для пользователя

После запуска программы появиться окно программы. В котором нужно выбрать начальные координаты (х и у) прорисовки паровозика, выбрать число вагонов в поезде, масштаб прорисовки и скорость движения паровозика (рис. 2).

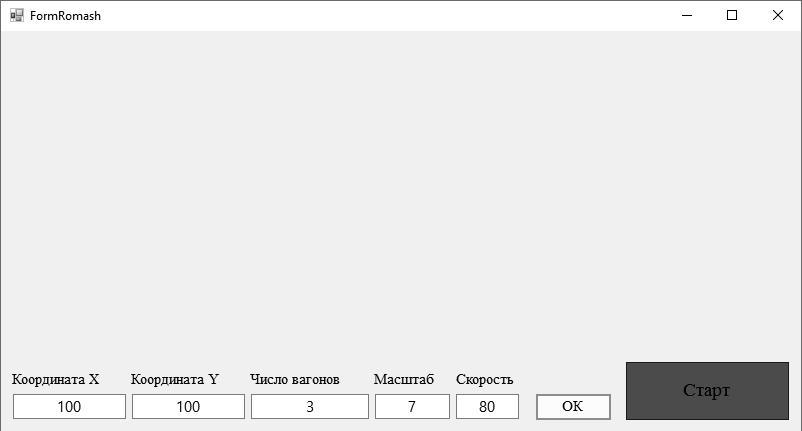


Рисунок 2. Главное окно.

Нажать на кнопку ОК. После чего нарисуется паровозик (рис. 3).

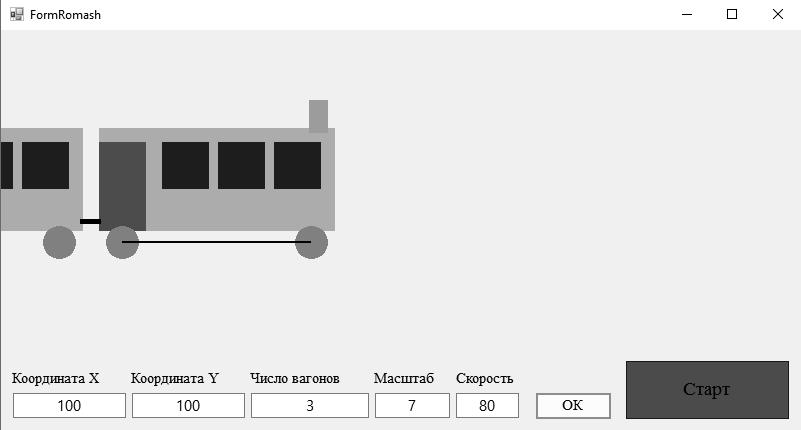
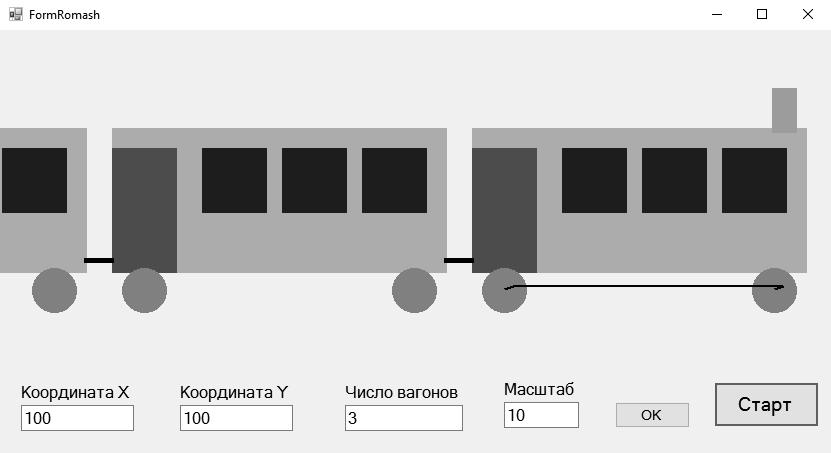


Рисунок 3. Прорисовка паравозика

Затем нажать на кнопку старт – паровозик поехал (рис. 4).



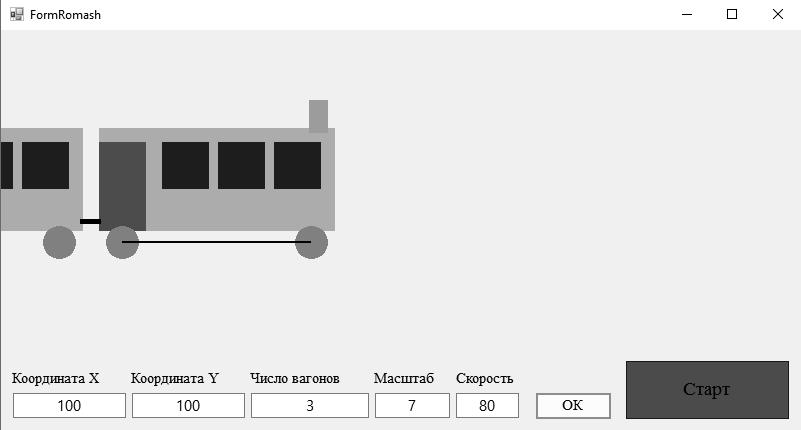


Рисунок 4. Паровозик в движении.

4. Контрольный пример

Контрольный пример привожу на параметрах:

1. Положение на экране Х = 100, Y = 100
2. Количество вагонов = 3 и 7
3. Масштаб = 10 и 15
4. Скорость = 80

В ходе проверки должны выполняться следующие этапы:

– Все детали поезда должны оставаться на своём месте

– Вагоны со сцепками не должны наезжать друг на друга

Поставленный эксперимент (рис. 4) показывает верное построение проекта.

Заключение

В приведенном проекте, несмотря на его масштаб, иллюстрируется использование четырех основных принципов ООП:

Создание отдельного класса для каждой значимой сущности модели отвечает принципу абстракции данных.

Изоляция введенных пользователем данных в класс UserData и использование его другими объектами со скрытием реализации является одним из примеров использования инкапсуляции.

Использование наследования позволяет избежать дублирования кода (например, описание Locomotive сокращается за счет использования родительского класса CoachItself).

Применение однообразной формы move() и show() для разных классов делает код более читаемым и скрывает детали алгоритма для каждого отдельного объекта, что дает полиморфизм.

При решении задачи были использованы средства Visual Studio Windows Forms и продемонстрировано применение инструментов визуального программирования для упрощения программирования пользовательского интерфейса.

Список литературы

1. Павловская Т. А. C/C++. Программирование на языке высокого уровня. — СПб.:Питер, 2003. – 461
2. Буч Г. Объектно-ориентированный анализ и проектирование с применением приложений. – М : «Вильямс», 2008 – 720 стр.
3. Шилдт Г.С. С++. Базовый курс. – М.: «Вильямс», 2010. – 624
4. Сабуров С.В. Языки программирования C и C++. \_ М.: Бук\_пресс, 2006. – 647 с.
5. Шень А. Программирование: теоремы и задачи. – М.: МЦНМО, 2004. – 296
6. Окулов С. М. Программирование в алгоритмах – М.: БИНОМ, 2002. – 341

Приложение

Программа с maim

#include "FormRomash.h"

using namespace System;

using namespace System::Windows::Forms;

[STAThreadAttribute]

void main(array<String^>^ args) {

Application::EnableVisualStyles();

Application::SetCompatibleTextRenderingDefault(false);

ProbRomashkovo::FormRomash Form;

Application::Run(%Form);

}

Класс myRecttangle

#pragma once

#include <vcclr.h>

#include <math.h>

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

class myRecttangle

{

protected:

int xl;

int yl;

int width,height;

gcroot <Color> cline, cfon;

public:

myRecttangle(int \_x = 0, int \_y = 0, int \_width = 0, int \_height = 0)

{

xl = \_x;

yl = \_y;

width = \_width;

height = \_height;

cfon = Color::Black;

cline = Color::Black;

}

~myRecttangle(){};

void Show (Graphics^ g)

{

Pen^ b = gcnew Pen(cline, 5);

Brush^ a = gcnew SolidBrush(cfon);

g->DrawRectangle(b, xl, yl, width, height);

g->FillRectangle(a, xl, yl, width, height);

}

void move (int dx)

{

xl += dx;

}

void setcfon(gcroot <Color> \_cfon)

{

cfon = \_cfon;

}

void setcline(gcroot <Color> \_cline)

{

cline = \_cline;

}

};

Класс windo

class windo: public myRecttangle{

public:

windo(int x = 0, int y = 0, int h = 0): myRecttangle(x, y, 6 \* h, 6 \* h)

{

cfon = Color::Blue;

cline = Color::Blue;

}

};

Класс door

class door: public myRecttangle{

public:

door(int x = 0, int y = 0, int h = 0): myRecttangle(x, y, 6 \* h, 12 \* h)

{

cfon = Color::Red;

cline = Color::Red;

}

};

Класс corpus

class corpus: public myRecttangle{

public:

corpus(int x = 0, int y = 0, int h = 0): myRecttangle(x, y, 33 \* h, 14 \* h)

{

cfon = Color::YellowGreen;

cline = Color::YellowGreen;

}

};

Класс wheel

class wheel: public myRecttangle{

public:

wheel(int x = 0, int y = 0, int h = 0): myRecttangle(x, y, 4 \* h, 4 \* h)

{

cfon = Color::Gray;

cline = Color::Gray;

}

void Show(Graphics ^g)

{

Pen^ b = gcnew Pen(cline, 5);

Brush^ a = gcnew SolidBrush(cfon);

g->DrawEllipse(b, xl, yl, width, height);

g->FillEllipse(a, xl, yl, width, height);

}

};

Класс carriage

class carriage{

private:

corpus corp;

door dver;

windo \*okno;

wheel leftcoleso, rightcoleso;

public:

carriage(int x = 0, int y = 0, int h = 0)

{

corp = corpus(x, y, h);

dver = door(x, y + 2 \* h, h);

okno = new windo[3];

for (int i = 0; i < 3; i++)

{

okno[i] = windo(x + 9 \* h + 8 \* i \* h, y + 2 \* h, h);

}

leftcoleso = wheel(x + 1 \* h, y + 14 \* h, h);

rightcoleso = wheel(x + 28 \* h, y + 14 \* h, h);

}

carriage (const carriage &tmp)

{

corp = tmp.corp;

dver = tmp.dver;

okno = new windo[3];

for(int i=0; i < 3; i++)

{

okno[i] = tmp.okno[i];

}

leftcoleso = tmp.leftcoleso;

rightcoleso = tmp.rightcoleso;

}

carriage &operator= (carriage tmp)

{

corp = tmp.corp;

dver = tmp.dver;

for(int i = 0; i < 3; i++)

{

okno[i] = tmp.okno[i];

}

leftcoleso = tmp.leftcoleso;

rightcoleso = tmp.rightcoleso;

return \*this;

}

void Show(Graphics ^g)

{

corp.Show(g);

dver.Show(g);

for (int i = 0; i < 3; i++)

{

okno[i].Show(g);

}

leftcoleso.Show(g);

rightcoleso.Show(g);

}

void move(int dx)

{

corp.move(dx);

dver.move(dx);

for (int i = 0; i < 3; i++)

{

okno[i].move(dx);

}

leftcoleso.move(dx);

rightcoleso.move(dx);

}

~carriage()

{

delete[] okno;

}

};

Класс truba

class truba:public myRecttangle{

public:

truba(int x = 0, int y = 0, int h = 0): myRecttangle(x, y, 2 \* h, 4 \* h)

{

cfon = Color::RosyBrown;

cline = Color::RosyBrown;

}

};

Класс myline

class myline

{

protected:

int xll, yll, xrl, yrl;

gcroot <Color> cfonli;

public:

myline(int \_xll = 0, int \_yll = 0, int \_xrl = 0, int \_yrl = 0)

{

xll = \_xll;

yll = \_yll;

xrl = \_xrl;

yrl = \_yrl;

cfonli = Color::Aquamarine;

}

~myline(){};

void Show (Graphics^ g)

{

Pen^ b = gcnew Pen(cfonli, 5);

g->DrawLine(b, xll, yll, xrl, yrl);

}

void move (int dx)

{

xll += dx;

xrl += dx;

}

void setcfonli(gcroot <Color> \_cfonli)

{

cfonli = \_cfonli;

}

};

Класс scepka

class scepka: public myline{

public:

scepka(int x = 0, int y = 0, int h = 0): myline(x, y, x + 3 \* h, y)

{

cfonli = Color::Black;

}

void move(int dx)

{

xll += dx;

xrl += dx;

}

};

Класс engene

class engine: public myline

{

private:

int \_xl, \_yl, \_xr, \_yr, \_h;

double r, angle;

public:

engine(int x = 0, int y = 0, int h = 0): myline(x + 3 \* h, y, x + 27 \* h + 3 \* h, y)

{

angle = 0;

\_yr = yll;

\_xl = xll;

\_xr = xll + 27 \* h;

\_yl = yll;

\_h = h;

r = 1 \* \_h;

cfonli = Color::Black;

}

void move(int dx)

{

\_xl += dx;

\_xr += dx;

angle = angle + dx / (r);

if (angle > 6.28)

{

angle = angle - 6.28;

}

xll = (double)\_xl + r \* cos(angle);

yll = (double)\_yl + r \* sin(angle);

xrl = (double)\_xr + r \* cos(angle);

yrl = (double)\_yr + r \* sin(angle);

}

void Show(Graphics^ g)

{

Pen^ b = gcnew Pen(cfonli, 2);

g->DrawLine(b, \_xl, \_yl, xll, yll);

g->DrawLine(b, xll, yll, xrl, yrl);

g->DrawLine(b, \_xr, \_yr, xrl, yrl);

}

};

Класс parovoz

class paravoz

{

private:

corpus corppar;

door dverpar;

windo \*oknopar;

wheel leftcolesopar, rightcolesopar;

truba tube;

engine dvig;

public:

paravoz(int x = 0, int y = 0, int h = 0)

{

corppar = corpus(x, y, h);

dverpar = door(x, y + 2 \* h, h);

oknopar = new windo[3];

for(int i = 0; i < 3; i++)

{

oknopar[i] = windo(x + 9 \* h + 8 \* h \* i, y + 2 \* h, h);

}

leftcolesopar = wheel(x + 1 \* h, y + 14 \* h, h);

rightcolesopar = wheel(x + 28 \* h, y + 14 \* h, h);

tube = truba(x + h \* 30, y - 4 \* h, h);

dvig = engine(x, y + 16 \* h, h);

}

~paravoz()

{

delete[] oknopar;

}

paravoz (const paravoz &tmp)

{

corppar = tmp.corppar;

dverpar = tmp.dverpar;

oknopar = new windo[3];

for(int i = 0; i < 3; i++)

{

oknopar[i] = tmp.oknopar[i];

}

leftcolesopar = tmp.leftcolesopar;

rightcolesopar = tmp.rightcolesopar;

dvig = tmp.dvig;

tube = tmp.tube;

}

paravoz &operator= (paravoz tmp)

{

corppar = tmp.corppar;

dverpar = tmp.dverpar;

for(int i = 0; i < 3; i++)

{

oknopar[i] = tmp.oknopar[i];

}

leftcolesopar = tmp.leftcolesopar;

rightcolesopar = tmp.rightcolesopar;

dvig = tmp.dvig;

tube = tmp.tube;

return \*this;

}

void Show(Graphics ^g)

{

corppar.Show(g);

dverpar.Show(g);

for (int i = 0; i < 3; i++)

{

oknopar[i].Show(g);

}

leftcolesopar.Show(g);

rightcolesopar.Show(g);

dvig.Show(g);

tube.Show(g);

}

void move(int dx)

{

corppar.move(dx);

dverpar.move(dx);

for (int i = 0; i < 3; i++)

{

oknopar[i].move(dx);

}

leftcolesopar.move(dx);

rightcolesopar.move(dx);

dvig.move(dx);

tube.move(dx);

}

};

Класс train

class train

{

private:

paravoz first;

carriage \*vagons;

scepka \*sceps;

int kolvo;

public:

train(int x = 0, int y = 0, int h = 0, int n = 0)

{

kolvo = n;

first = paravoz(x, y, h);

vagons = new carriage[kolvo];

for(int i = 0; i < kolvo; i++)

{

vagons[i] = carriage(x - (33 + 3) \* h \* (i + 1), y, h);

}

sceps = new scepka[kolvo];

for(int i = 0; i < kolvo; i++)

{

sceps[i] = scepka(x - (33 + 3) \* h \* i - 3 \* h, y + 13 \* h, h);

}

}

~train()

{

delete[] vagons;

delete[] sceps;

}

train (const train &tmp)

{

kolvo = tmp.kolvo;

first = tmp.first;

vagons = new carriage[kolvo];

for(int i = 0; i < kolvo; i++)

{

vagons[i] = tmp.vagons[i];

}

sceps = new scepka[kolvo];

for(int i = 0; i < kolvo; i++)

{

sceps[i] = tmp.sceps[i];

}

}

train &operator= (train tmp)

{

kolvo = tmp.kolvo;

first = tmp.first;

for(int i = 0 ; i < kolvo; i++)

{

vagons[i] = tmp.vagons[i];

}

for(int i = 0; i < kolvo; i++)

{

sceps[i] = tmp.sceps[i];

}

return \*this;

}

void Show(Graphics ^g)

{

first.Show(g);

for(int i = 0; i < kolvo; i++)

{

vagons[i].Show(g);

}

for(int i = 0; i < kolvo; i++)

{

sceps[i].Show(g);

}

}

void move(int dx)

{

first.move(dx);

for(int i = 0; i < kolvo; i++)

{

vagons[i].move(dx);

}

for(int i = 0; i < kolvo; i++)

{

sceps[i].move(dx);

}

}

};

Описание Wondows Form

#pragma once

#include "AllClassTrain.h"

int xs , ys, vagcount, mashtab, speed;

namespace ProbRomashkovo

{

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

/// <summary>

/// Сводка для FormRomash

/// </summary>

public ref class FormRomash : public System::Windows::Forms::Form

{

public:

FormRomash(void)

{

InitializeComponent();

//

//TODO: добавьте код конструктора

//

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~FormRomash()

{

if (components)

{

delete components;

}

}

private: System::Windows::Forms::Button^ button1;

private: System::Windows::Forms::Button^ button2;

private: System::Windows::Forms::TextBox^ textBox1;

private: System::Windows::Forms::TextBox^ textBox2;

private: System::Windows::Forms::TextBox^ textBox3;

private: System::Windows::Forms::TextBox^ textBox4;

private: System::Windows::Forms::Label^ label1;

private: System::Windows::Forms::Label^ label2;

private: System::Windows::Forms::Label^ label3;

private: System::Windows::Forms::Label^ label4;

private: System::Windows::Forms::Timer^ timer1;

private: System::Windows::Forms::Label^ label5;

private: System::Windows::Forms::TextBox^ textBox5;

private: System::ComponentModel::IContainer^ components;

private:

/// <summary>

/// Требуется переменная конструктора.

/// </summary>

#pragma region Windows Form Designer generated code

/// <summary>

/// Обязательный метод для поддержки конструктора - не изменяйте

/// содержимое данного метода при помощи редактора кода.

/// </summary>

void InitializeComponent(void)

{

this->components = (gcnew System::ComponentModel::Container());

this->button1 = (gcnew System::Windows::Forms::Button());

this->button2 = (gcnew System::Windows::Forms::Button());

this->textBox1 = (gcnew System::Windows::Forms::TextBox());

this->textBox2 = (gcnew System::Windows::Forms::TextBox());

this->textBox3 = (gcnew System::Windows::Forms::TextBox());

this->textBox4 = (gcnew System::Windows::Forms::TextBox());

this->label1 = (gcnew System::Windows::Forms::Label());

this->label2 = (gcnew System::Windows::Forms::Label());

this->label3 = (gcnew System::Windows::Forms::Label());

this->label4 = (gcnew System::Windows::Forms::Label());

this->timer1 = (gcnew System::Windows::Forms::Timer(this->components));

this->label5 = (gcnew System::Windows::Forms::Label());

this->textBox5 = (gcnew System::Windows::Forms::TextBox());

this->SuspendLayout();

//

// button1

//

this->button1->BackColor = System::Drawing::SystemColors::Info;

this->button1->BackgroundImageLayout = System::Windows::Forms::ImageLayout::None;

this->button1->Cursor = System::Windows::Forms::Cursors::AppStarting;

this->button1->FlatStyle = System::Windows::Forms::FlatStyle::Popup;

this->button1->Font = (gcnew System::Drawing::Font(L"Times New Roman", 11.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->button1->Location = System::Drawing::Point(535, 363);

this->button1->Name = L"button1";

this->button1->Size = System::Drawing::Size(75, 26);

this->button1->TabIndex = 0;

this->button1->Text = L"ОК";

this->button1->UseVisualStyleBackColor = false;

this->button1->Click += gcnew System::EventHandler(this, &FormRomash::button1\_Click);

//

// button2

//

this->button2->BackColor = System::Drawing::Color::Green;

this->button2->FlatStyle = System::Windows::Forms::FlatStyle::Popup;

this->button2->Font = (gcnew System::Drawing::Font(L"Times New Roman", 14, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->button2->Location = System::Drawing::Point(625, 331);

this->button2->Name = L"button2";

this->button2->Size = System::Drawing::Size(163, 58);

this->button2->TabIndex = 1;

this->button2->Text = L"Старт";

this->button2->UseVisualStyleBackColor = false;

this->button2->Click += gcnew System::EventHandler(this, &FormRomash::button2\_Click);

//

// textBox1

//

this->textBox1->Font = (gcnew System::Drawing::Font(L"Times New Roman", 11.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204), true));

this->textBox1->Location = System::Drawing::Point(250, 363);

this->textBox1->Name = L"textBox1";

this->textBox1->Size = System::Drawing::Size(118, 25);

this->textBox1->TabIndex = 2;

this->textBox1->Text = L"3";

this->textBox1->TextAlign = System::Windows::Forms::HorizontalAlignment::Center;

//

// textBox2

//

this->textBox2->Font = (gcnew System::Drawing::Font(L"Times New Roman", 11.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204), true));

this->textBox2->Location = System::Drawing::Point(374, 363);

this->textBox2->Name = L"textBox2";

this->textBox2->Size = System::Drawing::Size(75, 25);

this->textBox2->TabIndex = 3;

this->textBox2->Text = L"7";

this->textBox2->TextAlign = System::Windows::Forms::HorizontalAlignment::Center;

//

// textBox3

//

this->textBox3->Font = (gcnew System::Drawing::Font(L"Times New Roman", 11.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204), true));

this->textBox3->Location = System::Drawing::Point(12, 363);

this->textBox3->Name = L"textBox3";

this->textBox3->Size = System::Drawing::Size(113, 25);

this->textBox3->TabIndex = 4;

this->textBox3->Text = L"100";

this->textBox3->TextAlign = System::Windows::Forms::HorizontalAlignment::Center;

//

// textBox4

//

this->textBox4->Font = (gcnew System::Drawing::Font(L"Times New Roman", 11.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204), true));

this->textBox4->Location = System::Drawing::Point(131, 363);

this->textBox4->Name = L"textBox4";

this->textBox4->Size = System::Drawing::Size(113, 25);

this->textBox4->TabIndex = 5;

this->textBox4->Text = L"100";

this->textBox4->TextAlign = System::Windows::Forms::HorizontalAlignment::Center;

//

// label1

//

this->label1->AutoSize = true;

this->label1->Font = (gcnew System::Drawing::Font(L"Times New Roman", 11.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label1->Location = System::Drawing::Point(246, 340);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(98, 17);

this->label1->TabIndex = 6;

this->label1->Text = L"Число вагонов";

//

// label2

//

this->label2->AutoSize = true;

this->label2->Font = (gcnew System::Drawing::Font(L"Times New Roman", 11.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label2->Location = System::Drawing::Point(370, 340);

this->label2->Name = L"label2";

this->label2->Size = System::Drawing::Size(68, 17);

this->label2->TabIndex = 7;

this->label2->Text = L"Масштаб";

//

// label3

//

this->label3->AutoSize = true;

this->label3->Font = (gcnew System::Drawing::Font(L"Times New Roman", 11.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label3->Location = System::Drawing::Point(8, 340);

this->label3->Name = L"label3";

this->label3->Size = System::Drawing::Size(95, 17);

this->label3->TabIndex = 8;

this->label3->Text = L"Координата X";

//

// label4

//

this->label4->AutoSize = true;

this->label4->Font = (gcnew System::Drawing::Font(L"Times New Roman", 11.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label4->Location = System::Drawing::Point(127, 340);

this->label4->Name = L"label4";

this->label4->Size = System::Drawing::Size(96, 17);

this->label4->TabIndex = 9;

this->label4->Text = L"Координата Y";

//

// timer1

//

this->timer1->Interval = 20;

this->timer1->Tick += gcnew System::EventHandler(this, &FormRomash::timer1\_Tick);

//

// label5

//

this->label5->AutoSize = true;

this->label5->Font = (gcnew System::Drawing::Font(L"Times New Roman", 11.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label5->Location = System::Drawing::Point(452, 340);

this->label5->Name = L"label5";

this->label5->Size = System::Drawing::Size(66, 17);

this->label5->TabIndex = 10;

this->label5->Text = L"Скорость";

//

// textBox5

//

this->textBox5->Font = (gcnew System::Drawing::Font(L"Times New Roman", 11.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204), true));

this->textBox5->Location = System::Drawing::Point(455, 363);

this->textBox5->Name = L"textBox5";

this->textBox5->Size = System::Drawing::Size(63, 25);

this->textBox5->TabIndex = 11;

this->textBox5->Text = L"80";

this->textBox5->TextAlign = System::Windows::Forms::HorizontalAlignment::Center;

this->textBox5->TextChanged += gcnew System::EventHandler(this, &FormRomash::textBox5\_TextChanged);

//

// FormRomash

//

this->AutoScaleDimensions = System::Drawing::SizeF(6, 13);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->ClientSize = System::Drawing::Size(800, 400);

this->Controls->Add(this->textBox5);

this->Controls->Add(this->label5);

this->Controls->Add(this->label4);

this->Controls->Add(this->label3);

this->Controls->Add(this->label2);

this->Controls->Add(this->label1);

this->Controls->Add(this->textBox4);

this->Controls->Add(this->textBox3);

this->Controls->Add(this->textBox2);

this->Controls->Add(this->textBox1);

this->Controls->Add(this->button2);

this->Controls->Add(this->button1);

this->DoubleBuffered = true;

this->Name = L"FormRomash";

this->Text = L"FormRomash";

this->Load += gcnew System::EventHandler(this, &FormRomash::FormRomash\_Load);

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

private:

Graphics^ g;

train \*poezd;

private: System::Void button1\_Click(System::Object^ sender, System::EventArgs^ e)

{

g = this->CreateGraphics();

g->Clear(this->BackColor);

xs = Convert::ToInt32(textBox3->Text);

ys = Convert::ToInt32(textBox4->Text);

speed = abs(101 - Convert::ToInt32(textBox5->Text));

vagcount = Convert::ToInt32(textBox1->Text);

mashtab = Convert::ToInt32(textBox2->Text);

poezd = new train(xs,ys,mashtab,vagcount);

timer1->Enabled=false;

poezd->Show(g);

}

private: System::Void button2\_Click(System::Object^ sender, System::EventArgs^ e)

{

timer1->Enabled=true;

}

private: System::Void timer1\_Tick(System::Object^ sender, System::EventArgs^ e)

{

timer1->Interval = speed;

g = this->CreateGraphics();

g->Clear(this->BackColor);

poezd->move(2);

poezd->Show(g);

}

private: System::Void pictureBox1\_Click(System::Object^ sender, System::EventArgs^ e)

{

}

private: System::Void FormRomash\_Load(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void textBox5\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

}

};

};