МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Кафедра «Математическое обеспечение и применение ЭВМ»

Курсовой проект

по дисциплине «Программирование»

на тему: «Разработка объектно-ориентированного приложения. Классы прямоугольник и треугольник»

ПГУ 09.03.04 – 02КП201. 20 ПЗ

Направление подготовки – 09.03.04 «Программная инженерия»

Выполнил студент: _____ Макаричева Е.М.

Группа: 20ВП1

Руководитель:

к.т.н., доцент _____ Гурьянов Л.В.

Проект защищен с оценкой _____

Преподаватели _____

_____

_____

Дата защиты _____

Пенза 2021

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра «Математическое обеспечение и применение ЭВМ»

«УТВЕРЖДАЮ»

заведующий кафедрой

_____П.П. Макарычев

«___»_____ 2021г.

ЗАДАНИЕ

на курсовой проект

по дисциплине «Программирование»

на тему: «Разработка объектно-ориентированного приложения. Классы прямоугольник и треугольник»

1. **Студент гр.** *20ВП1* **факультета** ВТ **направления** 09.03.04

Макаричева Елизавета Михайловна

2. **Руководитель работы** *Гурьянов Лев Вячеславович*
3. **Время проектирования** с «15» февраля 2021 по «30» мая 2021
4. **Тема проекта**: *Разработка объектно-ориентированного приложения. Классы прямоугольник и треугольник*
5. **Техническое задание на курсовую работу (назначение, технические требования)**

Назначение: *создание и визуализация следующих типов фигур:*

треугольник, прямоугольник, сложная фигура (прямоугольник, вписанный в треугольник)

Основные функции приложения:

а) показать фигуру;

б) скрыть фигуру;

в) *переместить фигуру;*

Структура данных: *стек как статический массив*

***Технология разработки**: ООП*

***Язык программирования**:C++*

***Среда исполнения**: Windows*

6. Содержание работы

6.1. **Пояснительная записка (перечень вопросов, подлежащих разработке, расчетов, обоснований, описаний)**

1. *Анализ предметной области*
2. *Анализ функциональных требований*
3. *Проектирование*
4. *Реализация*
5. *Тестирование*
6. *Оформление пояснительной записки*

7. Календарный график по выполнению работы

Наименование этапов работы	Объем работы (%)	Срок выполнения	Подпись руководителя
1) Анализ предметной области и требований	10	25.02.2021	
2) Проектирование	20	17.03.2021	
3) Реализация	30	20.04.2021	
4) Тестирование	20	10.05.2021	
5) Оформление пояснительной записки	20	30.05.2021	

Дата выдачи задания «15»февраля 2021г.

Руководитель курсового проекта _____ Гурьянов Л.В.

Задание к исполнению принял «15 »февраля 2021г.

Студент _____ Макаричева Е.М.

Содержание

ВВЕДЕНИЕ

В курсовом проекте требуется разработать объектно - ориентированное приложение создания и визуализации геометрических фигур: треугольника, прямоугольника и сложной фигуры (прямоугольник, вписанный в треугольник).

Для моделирования программных средств используется язык UML. Разработка осуществляется на языке C++ на платформе Microsoft Visual Studio 19.

Процесс создания программных средств включает следующие этапы: анализ предметной области и требований, проектирование, реализация и тестирование.

Графическая часть проекта включает диаграмму классов и диаграмму компонентов, выполненных в нотации UML.

1. РАЗРАБОТКА ОБЪЕКТНО-ОРИЕНТИРОВАННОГО ПРИЛОЖЕНИЯ. КЛАССЫ ПРЯМОУГОЛЬНИК И ТРЕУГОЛЬНИК

1.1. Анализ предметной области

Для предметной области[1], приведенной на рисунке 1, сложная фигура представляет собой прямоугольник, вписанный в треугольник. Сложная фигура определяется точкой O, описывающей координаты вершины (x, y) и длиной стороны (a).

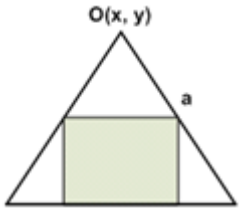

Рисунок 1 – Заданная геометрическая фигура

Модель предметной области для приведенной выше фигуры представлена на рисунке 2.

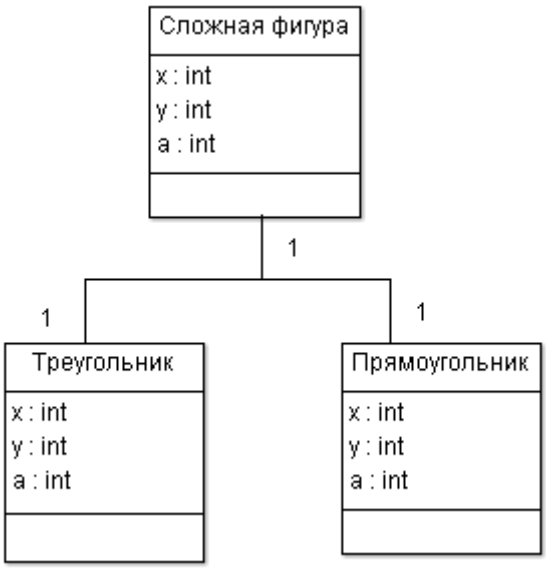

Рисунок 2 – Модель предметной области

Сложная фигура состоит из одного треугольника и одного прямоугольника.

1.2. Анализ функциональных требований

В техническом задании на курсовой проект определены следующие функциональные требования:

– показать фигуру;

– скрыть фигуру;

– переместить фигуру;

Диаграмма вариантов использования[2], соответствующая данным требованиям приведена на рисунке 3. На диаграмме варианты «Показать прямоугольник» и «Скрыть прямоугольник» расширяют функциональность варианта «Переместить прямоугольник». Аналогичное отношение расширения используется для вариантов треугольника и сложной фигуры. Это расширение обосновано алгоритмом функции «Переместить фигуру», который сначала удаляет геометрическую фигуру, а потом показывает ее на новом месте (относительно новой точки «привязки»).

Рисунок 3 – Диаграмма вариантов использования

Сценарии[1] некоторых вариантов использования представлены в таблицах 1-3.

Таблица 1 – Сценарий варианта использования «Скрыть прямоугольник»

Наименование: Скрыть прямоугольник
ID: 2
Краткое описание: система удаляет изображение прямоугольника
Действующие лица: пользователь
Предусловие: прямоугольник создан и нарисован цветом фона
Основной поток: 1. Пользователь инициирует удаление изображения прямоугольника 2. Система рисует изображение прямоугольника цветом фона
Постусловие: изображение прямоугольника скрыто

Таблица 2 – Сценарий варианта использования «Переместить прямоугольник»

Наименование: Переместить прямоугольник
ID: 3
Краткое описание: система удаляет изображение квадрата и рисует его в новых координатах
Действующие лица: пользователь
Предусловие: прямоугольник создан и нарисован в новых координатах
Основной поток: 1. Пользователь задает новые координаты вершины треугольника (x, y) Точка расширения «Скрыть» 2. Система переопределяет координаты квадрата Точка расширения «Показать»
Постусловие: прямоугольник перемещен

Таблица 3 – Сценарий варианта использования «Показать треугольник»

Наименование: Показать треугольник
ID: 1
Краткое описание: система рисует изображение прямоугольника с координатами вершины (x, y) и заданной стороной (a)
Действующие лица: пользователь
Предусловие: треугольник показан
Основной поток: 1. Пользователь задает координаты вершины и длину стороны; 2. Система рисует изображение треугольника по пользовательским данным
Постусловие: треугольник создан

1.3. Проектирование

Структура ПС отображена на диаграмме классов[2] (Рисунок 4).

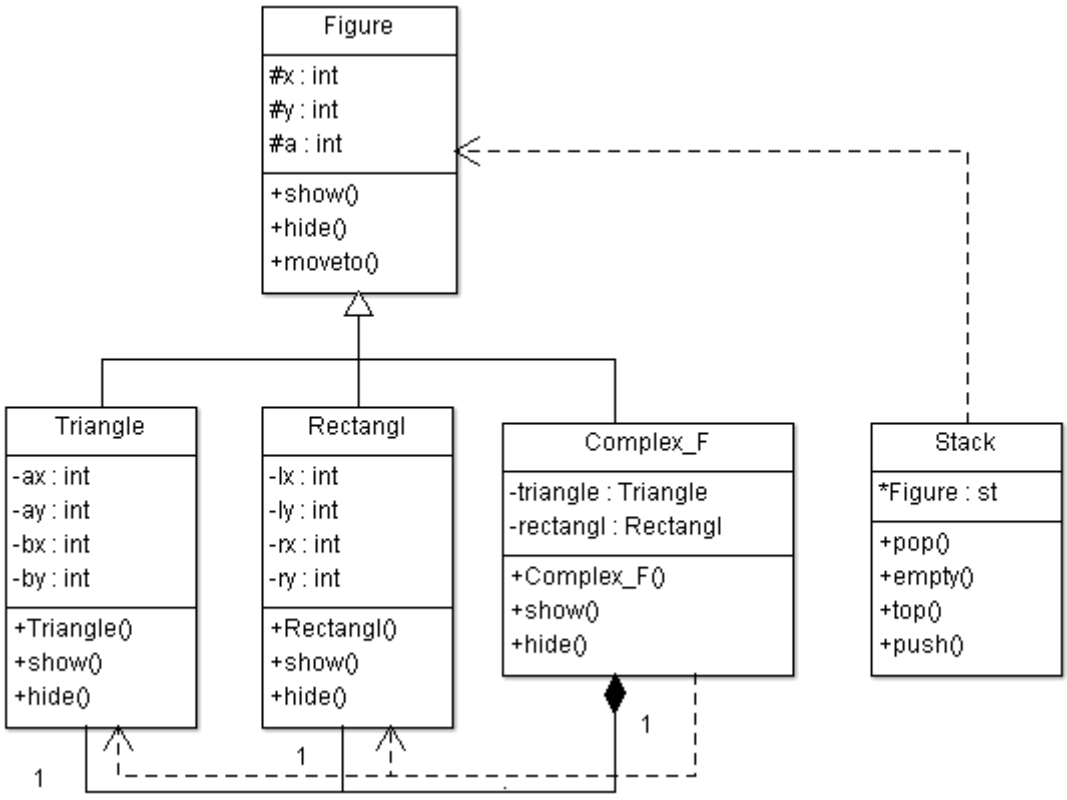

Рисунок 4 – Диаграмма классов

На приведенной диаграмме классы треугольник (Triangle) и прямоугольник (Rectangl) наследуют из базового класса[3] Figure координаты вершины фигуры (x, y) и длину стороны (a); класс сложной фигуры (Complex_F) содержит композицию из объектов классов Triangle и Rectangl. Stack содержит описание структуры данных для хранения, тип которой совпадает с типом базового класса фигур (*Figure).

Спецификация классов

Таблица 4 – Спецификация класса Figure

Название класса:	Figure
Назначение класса:	Базовый родительский класс, объединяющий поля и методы, свойственные всем типам фигур
Члены класса:	x: int – абсцисса вершины треугольника; y: int – ордината вершины треугольника; a: int – длина стороны треугольника; rt : RECT – прямоугольное окно консольного приложения.
Функции класса:	show() – показать фигуру; hide() – скрыть фигуру; moveto(int, int) – переместить фигуру, получив новые координаты вершины.

Таблица 5 – Спецификация класса Triangle

Название класса:	Triangle
Назначение класса:	Класс сущности фигуры треугольник, наследник класса Figure
Члены класса:	ax: int – абсцисса нижней левой вершины треугольника; ay: int – ордината нижней левой вершины треугольника; bx: int – абсцисса нижней правой вершины треугольника; by: int – ордината нижней правой вершины треугольник.
Функции класса:	Triangle(int, int, int) – конструктор с параметрами (длиной стороны и координатами вершины треугольника); Triangle() – конструктор по умолчанию; show() – показать треугольник; hide() – скрыть треугольник.

Таблица 6 – Спецификация класса Rectangl

Название класса:	Rectangl
Назначение класса:	Класс сущности фигуры прямоугольник, наследник класса Figure
Члены класса:	lx: int - абсцисса нижней левой вершины прямоугольника; ly: int – ордината нижней левой вершины прямоугольника; rx: int – абсцисса верхней правой вершины прямоугольника; ry: int – ордината верхней правой вершины прямоугольника.
Функции класса:	Rectangl(int, int, int) – конструктор с параметрами (длиной стороны и координатами вершины треугольника); Rectangl() – конструктор по умолчанию; show() – показать треугольник; hide() – скрыть треугольник.

Таблица 7 – Спецификация класса Complex_F

Название класса:	Complex_F
Назначение класса:	Класс сущности сложной фигуры, наследник класса Figure
Члены класса:	Rectangl:rectangl – объект класса Rectangl, составляющая сложной фигуры; Triangle:triangle – объект класса Triangle, составляющая сложной фигуры
Функции класса:	Complex_F(int, int, int) - конструктор с параметрами (длиной стороны и координатами вершины треугольника); show() – показать сложную фигуру; hide() – скрыть сложную фигуру.

1.4. Реализация

Диаграмма компонентов[2,3] приведена на рисунке 5.

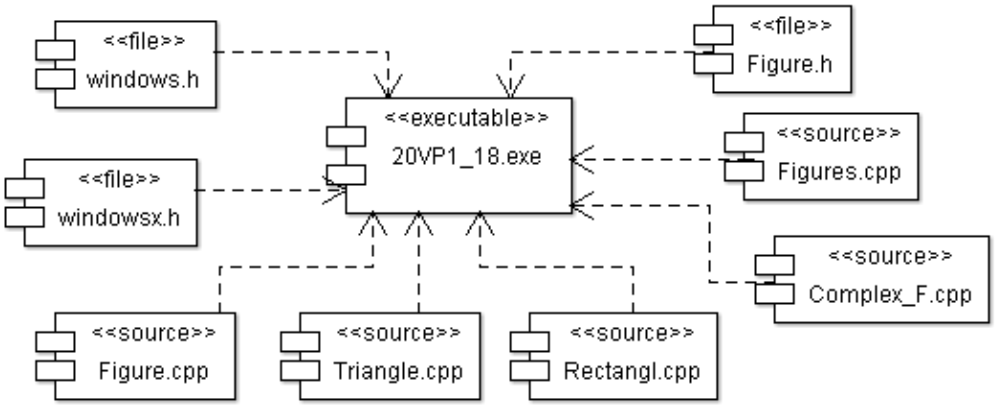

Рисунок 5 – Диаграмма компонентов

Описание компонентов приведено в таблице 8.

Таблица 8 – Компоненты

Компоненты	Назначение
20VP1_18.exe	Исполняемый файл приложения
Figure.h	Заголовочный файл проекта
Figures.cpp	Исходный файл главной программы (main)
Complex_F.cpp	Исходный файл класса Complex_F
Rectangl.cpp	Исходный файл класса Rectangl
Triangle.cpp	Исходный файл класса Triangle
Figure.cpp	Исходный файл класса Figure
windows.h	Системный заголовочный файл
windowsx.h	Системный заголовочный файл

На рисунке 6 показана структура компонентов исполняемого приложения.

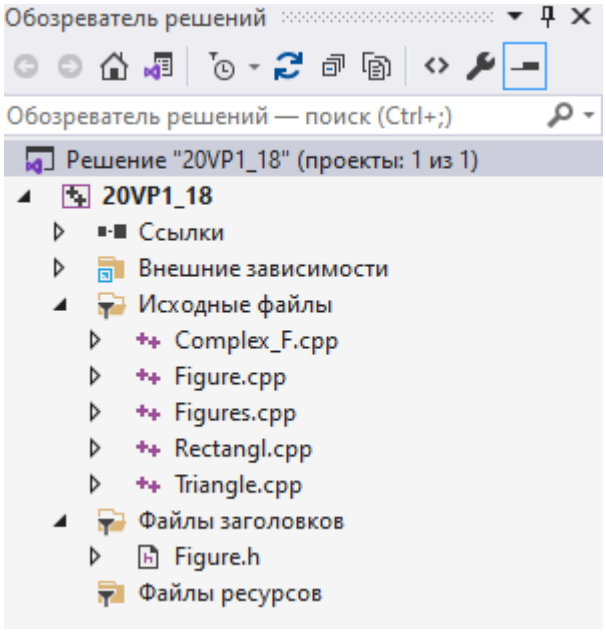

Рисунок 6 – Структура компонентов

1.5. Тестирование

Результаты функционального тестирования представлены в таблице 9.

Таблица 9 – План тестирования

Вариант использования	**Тест**	**Результат**
Показать треугольник	**Тест 1** Координаты вершины: 200, 200 Длина стороны: 150	Тест пройден. Треугольник нарисован правильно (Рисунок 7).
	Тест 2 Координаты вершины: 100, 300 Длина стороны: 50	Тест пройден. Треугольник нарисован правильно.
	Тест 3 Координаты вершины: 400, 300 Длина стороны:50	Тест пройден. Успешно обработано исключение выхода треугольника за границы окна (Рисунок 9).
Скрыть треугольник	**Тест 1** Координаты вершины: 200, 200 Длина стороны: 150	Тест пройден. Треугольник скрыт (Рисунок 8).
	Тест 2 Координаты вершины: 100, 300 Длина стороны: 50	Тест пройден. Треугольник скрыт.

	Тест 3 Координаты вершины: 345, 100 Длина стороны: 75	Тест пройден. Треугольник скрыт.
Переместить треугольник	**Тест 1** Новые координаты вершины: 100, 100	Тест пройден. Треугольник перемещен в новые координаты (Рисунок 10).
	Тест 2 Новые координаты вершины: 100, 346	Тест пройден. Треугольник перемещен в новые координаты.
	Тест 3 Новые координаты вершины: 256, 400	Тест пройден. Успешно обработано исключение выхода треугольника за границы окна.
Показать прямоугольник	**Тест 1** Координаты вершины: 200, 100 Длина стороны: 100	Тест пройден. Прямоугольник нарисован правильно (Рисунок 7).
	Тест 2 Координаты вершины: 380, 250 Длина стороны: 200	Тест пройден. Прямоугольник нарисован правильно.
	Тест 3 Координаты вершины: 920, 250 Длина стороны: 200	Тест пройден. Успешно обработано исключение выхода прямоугольника за границы окна (Рисунок 9).
Скрыть прямоугольник	**Тест 1** Координаты вершины: 200, 100 Длина стороны: 100	Тест пройден. Прямоугольник скрыт (Рисунок 8).
	Тест 2 Координаты вершины: 380, 250 Длина стороны: 200	Тест пройден. Прямоугольник скрыт.
	Тест 3 Координаты вершины: 120, 230 Длина стороны: 78	Тест пройден. Прямоугольник скрыт.
Переместить прямоугольник	**Тест 1** Новые координаты вершины: 300, 350	Тест пройден. Прямоугольник перемещен в новые координаты (Рисунок 10).
	Тест 2 Новые координаты вершины: 440, 135	Тест пройден. Прямоугольник перемещен в новые координаты.

	Тест 3 Новые координаты вершины: 680, 351	Тест пройден. Успешно обработано исключение выхода прямоугольника за границы окна.
Показать сложную фигуру	**Тест 1** Координаты вершины: 350, 200 Длина стороны: 100	Тест пройден. Сложная фигура нарисована правильно (Рисунок 7).
	Тест 2 Координаты вершины: 195, 310 Длина стороны: 75	Тест пройден. Сложная фигура нарисована правильно.
	Тест 3 Координаты вершины: 645, 380 Длина стороны: 50	Тест пройден. Успешно обработано исключение выхода сложной фигуры за границы окна(Рисунок 9).
Скрыть сложную фигуру	**Тест 1** Координаты вершины: 350, 200 Длина стороны: 100	Тест пройден. Сложная фигура скрыта (Рисунок 8).
	Тест 2 Координаты вершины: 195, 310 Длина стороны: 75	Тест пройден. Сложная фигура скрыта.
	Тест 3 Координаты вершины: 670, 210 Длина стороны: 80	Тест пройден. Сложная фигура скрыта.
Переместить сложную фигуру	**Тест 1** Новые координаты вершины: 450, 300	Тест пройден. Сложная фигура перемещена в новые координаты (Рисунок 10).
	Тест 2 Новые координаты вершины: 120, 180	Тест пройден. Сложная фигура перемещена в новые координаты.
	Тест 3 Новые координаты вершины: 594, 355	Тест пройден. Успешно обработано исключение выхода сложной фигуры за границы окна.

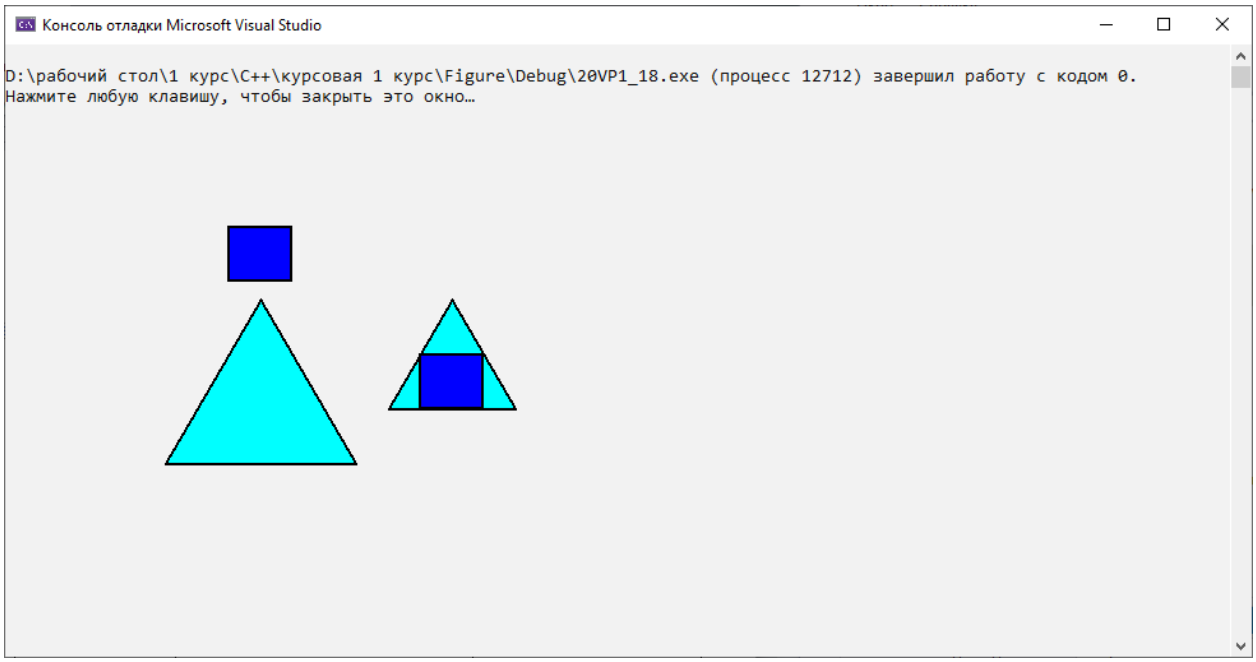

Рисунок 7 – Тестирование функции show()

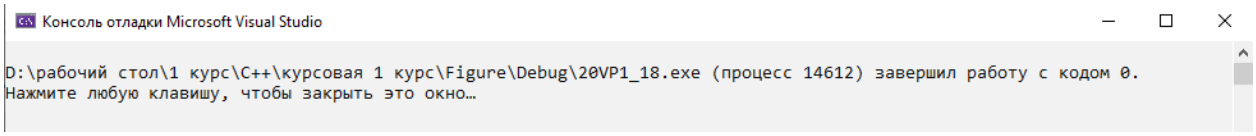

Рисунок 8 – Тестирование функции hide()

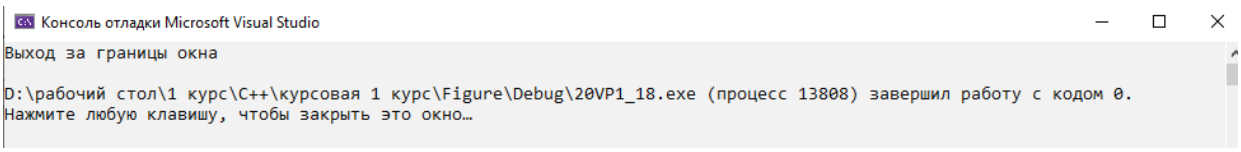

Рисунок 9 – Тестирование функции show(). Обработка исключения

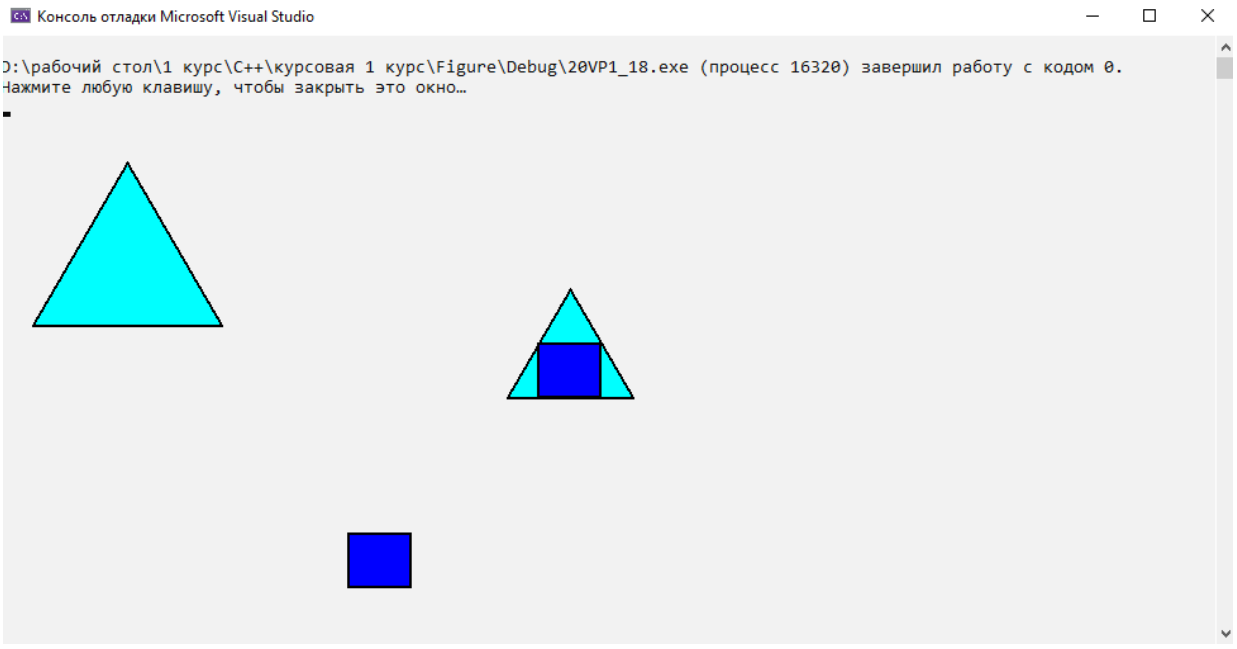

Рисунок 10 – Тестирование функции moveto(int, int)

ЗАКЛЮЧЕНИЕ

В процессе разработки курсового проекта произведён анализ предметной области (составлена модель предметной области) и функциональных требований к проекту (разработана диаграмма вариантов использования, описаны сценарии некоторых вариантов использования). В процессе проектирования построена диаграмма классов, спроектированы и реализованы классы: Figure, Triangle, Rectangl, Complex_F; описаны их спецификации. Результатом разработки стало приложение «20VP1_18.exe» для работы с геометрическими фигурами: треугольник, прямоугольник, сложная фигура (прямоугольник, вписанный в треугольник). Структура приложения отражена на диаграмме компонентов. Заключительным этапом разработки приложения стало его тестирование, которое было пройдено успешно.

Список использованных источников

1. Л.В.Гурьянов. Введение в программирование на языке С++/Л.В.Гурьянов, Л.С. Гурьянова, Е.А.Дзюба, Д.В.Такташкин. – Лабораторный практикум: Издательство ПГУ, 2010. – 91с.

2. Джим Арлоу. UML2 и Унифицированный процесс. Практический объектно-ориентированный анализ и проектирование/Джим Арлоу, Айла Нейштадт. – Санкт-Петербург, Издательство Символ-Плюс, 2007. – 624с.

3. Т. А. Павловская. С/С++. Программирование на языке высокого уровня. – СПб.: Питер, 2003. – 461 с.

Приложение А. Код приложения

(обязательное)

Главная программа Figures.cpp

```
#include"Figure.h"

int main() {
	SetConsoleTitle((LPCWSTR)L"20VP1_18");// заголовок консоли
	setlocale(LC_ALL, "Russian");
	stack<Figure*> st; // создание стека для хранения фигур
	// заполнение стека фигурами
	st.push(reinterpret_cast<Figure*>(new Triangle(200, 200, 150)));
	st.push(reinterpret_cast<Figure*>(new Rectangl(200, 100, 100)));
	st.push(reinterpret_cast<Figure*>(new Complex_F(350, 200, 100)));
	try {
		while (!st.empty()) {
			st.top()->show();
			st.pop();
		}
	}
	catch (Figure::Border) {
		cout << "Выход за границы окна" << endl;
		// обработка исключения выхода за границы окна
	}
	return 0;
}
```

Figure.h

```
#pragma once
#include <iostream>
#include <windows.h>
#include<windowsx.h>
#include<stack>
using namespace std;

const int NotUsed = system("color F0");// цвет фона окна

class Figure {// базовый класс
protected:
	int x;//абсцисса вершины треугольника
	int y;//ордината вершины треугольника
	int a;// сторона равностороннего треугольника
	RECT rt;// прямоугольник
public:
	virtual void show();//показать
	virtual void hide();//скрыть
	void moveto(int, int);// переместить фигуру

	class Border {};//для обработки исключений
};


class Rectangl :public Figure {//прямоугольник
```

```
private:
	int lx, ly, rx, ry;
//l - левый нижний угол, r - правый верхний угол
public:
	Rectangl() {
		x = 0;
		y = 0;
		a = 0;
	}
	Rectangl(int new_x, int new_y, int new_a);
//конструктор с параметрами, задающими прямоугольник
	void show() override;
//override - для переопределения виртуальной функции базового класса
	void hide() override;
};

class Triangle:public Figure {//треугольник
private:
	int ax, ay, bx, by;
//a - нижний левый угол треугольника, b - нижний правый угол
треугольника
public:
	Triangle() {
		x = 0;
		y = 0;
		a = 0;
	}
	Triangle(int new_x, int new_y, int new_a);
//конструктор с параметрами, задающими треугольник
	void show()override;
	void hide()override;
};
class Complex_F :public Figure
{
private:
	Rectangl rectangl;
	Triangle triangle;
public:
	Complex_F(int new_x, int new_y, int new_a);
//конструктор с параметрами, задающими сложную фигуру
	void show()override;
	void hide()override;
};
```

Figure.cpp

```
#include"Figure.h"

void Figure::show() {}
void Figure::hide() {}
```

```
void Figure::moveto(int new_x, int new_y) {
	hide();
	x = new_x;
	y = new_y;
	show();
}
```

Triangle.cpp

```
#include"Figure.h"

Triangle::Triangle(int new_x, int new_y, int new_a) {//конструктор с
параметрами
	x = new_x;// объявление начальных координат
	y = new_y;
	a = new_a;
//расчет координат для построения
	ax = x - a / 2;
	ay = y + ((1.73 * a) / 2);
	bx = x + a / 2;
	by = y + ((1.73 * a) / 2);
}
void Triangle::show() {
	HWND hwnd = GetConsoleWindow();//Получаем идентификатор окна
	HDC hdc = GetDC(hwnd);//получаем контекст изображения
	HPEN pen = CreatePen(PS_SOLID, 2, RGB(0, 0, 0));//создаем перо
	HBRUSH brush = CreateSolidBrush(RGB(0, 255, 255));//создаем кисть
	GetClientRect(hwnd, &rt);//получаем размер окна
	SelectObject(hdc, pen);//назначаем перо для рисования
	SelectObject(hdc, brush);//назначаем кисть для рисования
	//объявляем координаты для построения
	ax = x - a / 2;
	ay = y + ((1.73 * a) / 2);
	bx = x + a / 2;
	by = y + ((1.73 * a) / 2);
	if ((ax <= rt.left) ||
		(y <= rt.top) || (bx >= rt.right)
		|| (by >= rt.bottom))throw Border();
//отлавливание выхода за границы окна
	POINT points[] = { {bx, by}, {x, y}, {ax, ay} };
	Polygon(hdc, points, 3);// рисуем треугольник
	//освобождаем ресурсы
	DeleteObject(pen);
	DeleteObject(brush);
	DeletePen(pen);//удаление пера
	DeleteBrush(brush);//удаление кисти
	ReleaseDC(hwnd, hdc);//освобождение контекста изображения
}

void Triangle::hide() {
	HWND hwnd = GetConsoleWindow();//Получаем идентификатор окна
	HDC hdc = GetDC(hwnd);//получаем контекст изображения
```

```
	HPEN pen = CreatePen(PS_SOLID, 2, RGB(242, 242, 242));//создаем
перо
	HBRUSH brush = CreateSolidBrush(RGB(242, 242, 242));//создаем
кисть
	GetClientRect(hwnd, &rt);//получаем размер окна
	SelectObject(hdc, pen);//назначаем перо для рисования
	SelectObject(hdc, brush);//назначаем кисть для рисования
	POINT points[] = { {bx, by}, {x, y}, {ax, ay} };
	Polygon(hdc, points, 3);// рисуем треугольник
	//освобождаем ресурсы
	DeleteObject(pen);
	DeleteObject(brush);
	DeletePen(pen);//удаление пера
	DeleteBrush(brush);//удаление кисти
	ReleaseDC(hwnd, hdc);//освобождение контекста изображения
}
```

Rectangle.cpp

```
#include"Figure.h"

Rectangl::Rectangl(int new_x, int new_y, int new_a) {
	x = new_x;
	y = new_y;
	a = new_a;
	rx = x + a / 4;//правый верхний угол
	ry = y + ((1.73 * a) / 4);
	lx = x - a / 4;//левый нижний угол
	ly = y + ((1.73 * a) / 2);
}
void Rectangl::show() {
	HWND hwnd = GetConsoleWindow();//Получаем идентификатор окна
	HDC hdc = GetDC(hwnd);//получаем контекст изображения
	HPEN Bpen = CreatePen(PS_SOLID, 2, RGB(0, 0, 0));//создаем перо
	HBRUSH hbrush = CreateSolidBrush(RGB(0, 0, 255));//создаем кисть
	GetClientRect(hwnd, &rt);//получаем размер окна
	SelectObject(hdc, Bpen);//назначаем перо для рисования
	SelectObject(hdc, hbrush);//назначаем кисть для рисования
	//объявляем координаты для построения
	rx = x + a / 4;
	ry = y + ((1.73 * a) / 4);
	lx = x - a / 4;
	ly = y + ((1.73 * a) / 2);
	if ((lx <= rt.left) ||
		(ry <= rt.top) || (rx >= rt.right)
		|| (ly >= rt.bottom))throw Border();
//отлавливание выхода за границы окна
	Rectangle(hdc, lx, ly, rx, ry);// рисуем прямоугольник
	//освобождаем ресурсы
	DeleteObject(Bpen);
	DeleteObject(hbrush);
	DeletePen(Bpen);//удаление пера
```

```
	DeleteBrush(hbrush);//удаление кисти
	ReleaseDC(hwnd, hdc);//освобождение контекста изображения
}
void Rectangl::hide() {
	HWND hwnd = GetConsoleWindow();//Получаем идентификатор окна
	HDC hdc = GetDC(hwnd);//получаем контекст изображения
	HPEN Bpen = CreatePen(PS_SOLID, 2, RGB(242, 242, 242));//создаем
перо
	HBRUSH hbrush = CreateSolidBrush(RGB(242, 242, 242));//создаем
кисть
	GetClientRect(hwnd, &rt);//получаем размер окна
	SelectObject(hdc, Bpen);//назначаем перо для рисования
	SelectObject(hdc, hbrush);//назначаем кисть для рисования
	Rectangle(hdc, lx, ly, rx, ry);// рисуем прямоугольник
	//освобождаем ресурсы
	DeleteObject(Bpen);
	DeleteObject(hbrush);
	DeletePen(Bpen);//удаление пера
	DeleteBrush(hbrush);//удаление кисти
	ReleaseDC(hwnd, hdc);//освобождение контекста изображения
}
```

Complex_F.cpp

```
#include"Figure.h"

Complex_F::Complex_F(int new_x, int new_y, int new_a) {
	x = new_x;
	y = new_y;
	a = new_a;
	rectangl = Rectangl(x, y, a);
	triangle = Triangle(x, y, a);
}
void Complex_F::show() {
	rectangl = Rectangl(x, y, a);
	triangle = Triangle(x, y, a);
	triangle.show();
	rectangl.show();
}
void Complex_F::hide() {
	triangle.hide();
	rectangl.hide();
}
```

Приложение Б. Графическая часть

(обязательное)

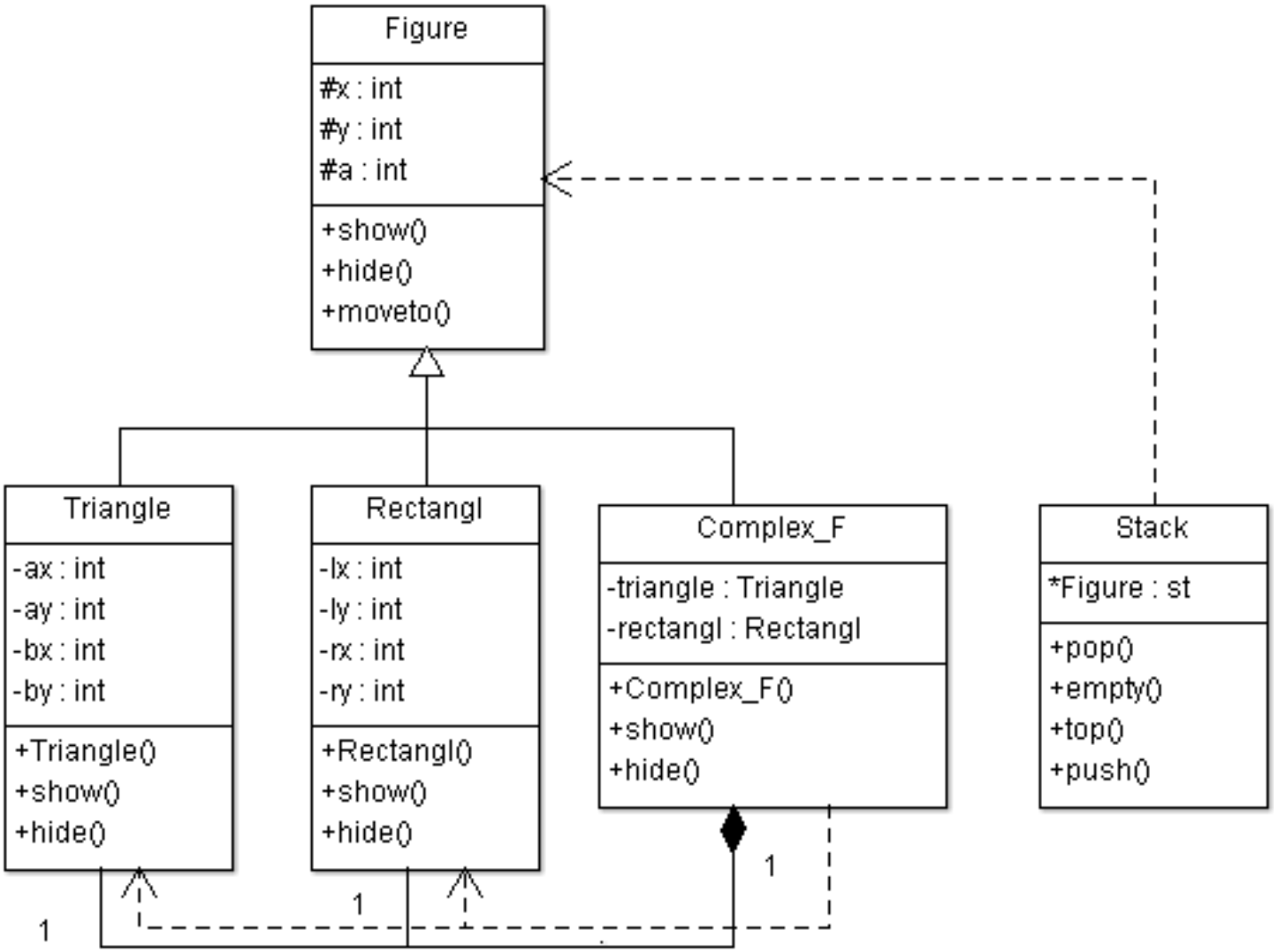

Рисунок Б1 – Диаграмма классов

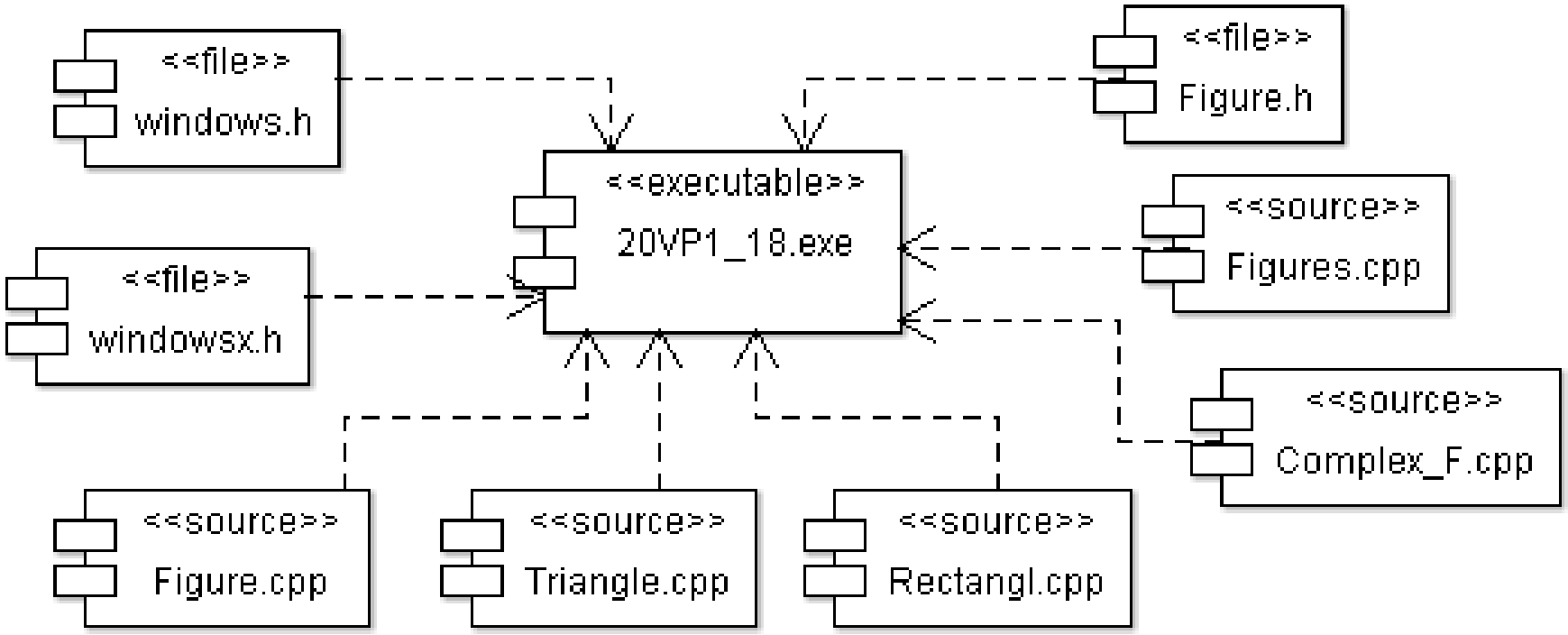

Рисунок Б2 – Диаграмма компонентов