|  |
| --- |
| **Министерство образования и науки Российской Федерации**  Федеральное государственное автономное образовательное  учреждение высшего образования  **«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ**  **ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»** |

Подразделение: Инженерная школа энергетики

Направление подготовки: 09.04.03 – Прикладная информатика

Отделение: Электроэнергетики и электротехники

**Проектная документация**

**Отчёт по лабораторной работе №5**

по дисциплине: «Основы объектно-ориентированного программирования»

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Выполнил студент гр. О-5КМ01 | |  | \_\_\_\_\_\_\_\_\_\_ | |  | Чугуевский А.С. |
|  | |  |  | |  |  |
|  | |  |  | \_\_\_ \_\_\_\_\_\_ 2022 г. | | |
|  | |  |  | |  |  |
| Отчёт принял | доцент, к.т.н. |  | \_\_\_\_\_\_\_\_\_\_ | |  | Калентьев А. А. |
|  |  |  |  | |  |  |
|  |  |  |  | \_\_\_ \_\_\_\_\_\_ 2022 г. | | |

Томск 2022

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc74956671)

[1 Основная часть 4](#_Toc74956672)

[1.1 UML диаграмма вариантов использования 4](#_Toc74956673)

[1.2 UML диаграмма классов 6](#_Toc74956674)

[1.3 Описание классов, образующих связь типа «общее-частное» 8](#_Toc74956675)

[1.4 Дерево ветвлений Git 9](#_Toc74956676)

[1.5 Тестирование программы 10](#_Toc74956677)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 22](#_Toc74956678)

[ПРИЛОЖЕНИЕ А 23](#_Toc74956679)

**ВВЕДЕНИЕ**

Корректная и полная документация сопровождает разработку программного обеспечения (далее – ПО) от появления идеи до выпуска конечного продукта. Написание документации является обязательным критерием разработки и последующей поддержки проекта [1].

Целью данной лабораторной работы является разработка проектной документации на созданный программный продукт.

Для достижения поставленной цели должны быть выполнены следующие задачи:

* Составление технического задания (далее – ТЗ) на разработанную программу (Приложение А);
* Составление UML диаграммы вариантов использования для разработанной программы;
* Составление UML диаграммы классов;
* Описание классов, образующих связь типа «общее-частное»;
* Привести дерево ветвлений Git;
* Провести тестирование программы.

**1 Основная часть**

**1.1 UML диаграмма вариантов использования**

Вариант использования (use case) — это описание множества последовательных действий (включая вариации), которые выполняются некоторым субъектом с целью получения результата, значимого для некоторого действующего лица [1]. ВИ предполагает взаимодействие действующих лиц и системы или другого объекта. Действующее лицо представляет собой логически связанное множество ролей, которые играют пользователи системы во время взаимодействия с ней.

Диаграмма вариантов использования для разработанного ПО приведена на рисунке 1.

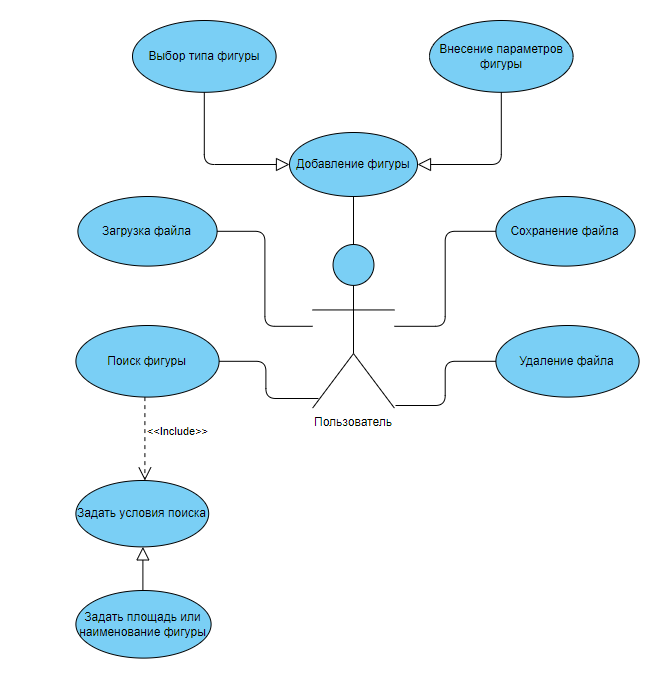


Рисунок 1 – Диаграмма вариантов использования

**1.2 UML диаграмма классов**

Диаграмма классов — это центральная методика моделирования, которая используется практически во всех объектно-ориентированных методах. Эта диаграмма описывает типы объектов в системе и различные виды статических отношений, которые существуют между ними. Широко применяется не только для документирования и визуализации, но также для конструирования посредством прямого или обратного проектирования.

Диаграмма классов приведена на рисунке 2.

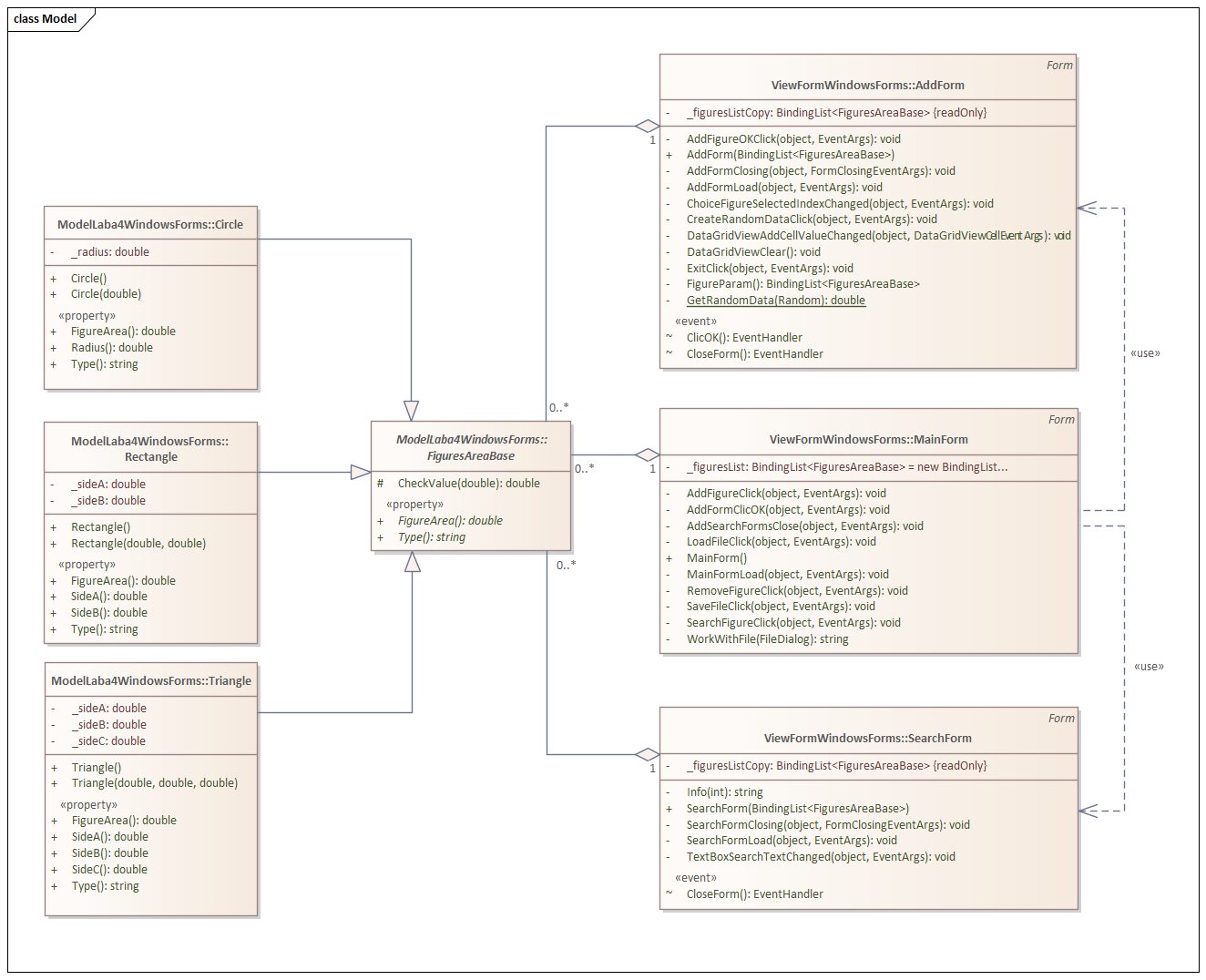


Рисунок 2 – UML диаграмма классов

**1.3 Описание классов, образующих связь типа «общее-частное»**

В таблице 1 приведено описание абстрактного класса *FiguresAreaBase* с его полями, свойствами и методами.

Таблица 1 – Описание класса *FiguresAreaBase*

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс *FiguresAreaBase* – абстрактный базовый класс для фигур | | |
| Свойства | | |
| + FigureArea | double | Площадь фигуры. |
| + Type | string | Тип фигуры. |
| Методы | | |
| # CheckValue(lineSegment) | double | Проверка знака (плюс или минус) отрезка фигуры.  lineSegment –параметр фигуры. |
|  |  |  |

В таблицах 2–4 приведены описания классов Circle, Rectangle и Triangle, которые наследуются от *FiguresAreaBase*.

Таблица 2 – Описание класса Circle

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс Circle – круг | | |
| Поля | | |
| – \_radius | double | Радиус круга |
| Свойства | | |
| + Radius | double | Радиус круга |
| + FigureArea | double | Площадь фигуры. |
| + Type | string | Тип фигуры. |

Таблица 3 – Описание класса Rectangle

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс Rectangle – прямоугольник | | |
| Поля | | |
| – \_sideA | double | Сторона А прямоугольника |
| – \_sideB | double | Сторона B прямоугольника |
| Свойства | | |
| + SideA | double | Сторона А прямоугольника |
| + SideB | double | Сторона B прямоугольника |
| + FigureArea | double | Площадь фигуры. |
| + Type | string | Тип фигуры. |

Таблица 4 – Описание класса Triangle

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс Triangle – треугольник | | |
| Поля | | |
| – \_sideA | double | Сторона А треугольника |
| – \_sideB | double | Сторона B треугольника |
| – \_sideC | double | Сторона C треугольника |
| Свойства | | |
| + SideA | double | Сторона А треугольника |
| + SideB | double | Сторона B треугольника |
| + SideC | double | Сторона C треугольника |
| + FigureArea | double | Площадь фигуры. |
| + Type | string | Тип фигуры. |

**1.4 Дерево ветвлений Git**

На рисунке 3 представлено дерево ветвлений Git, полученное по окончании работы с проектом.

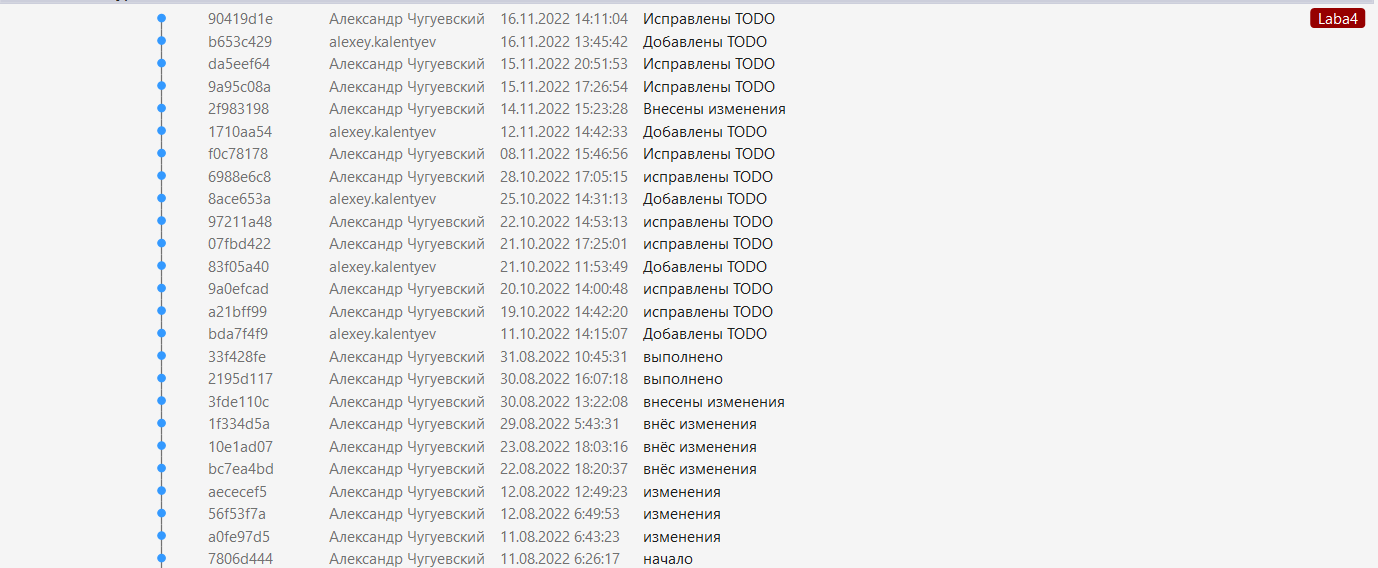


Рисунок 3 – Дерево ветвлений Git

**1.5 Тестирование программы**

Далее приводится процесс функционального тестирования программы.

Графический интерфейс пользователя представлен на рисунке 4.

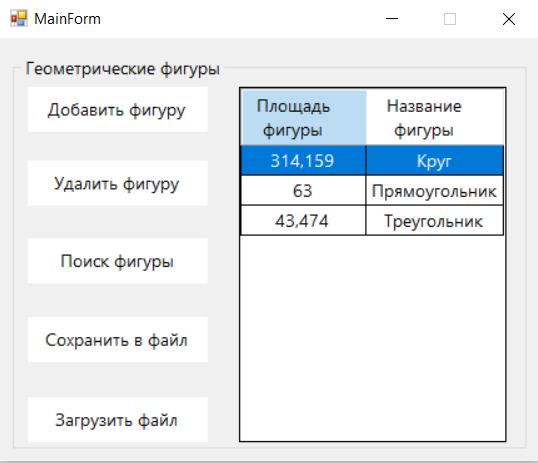


Рисунок 4 – Графический интерфейс пользователя

**1.5.1 Тестовый случай «Добавить элемент»**

Для добавления элемента необходимо вызвать соответствующую форму путём нажатия кнопки «Добавить фигуру» (рисунок 5).



Рисунок 5 – Форма для добавления элемента

После ввода данных необходимо нажать кнопку «ОК», элемент появится в таблице главной формы (рисунки 6 и 7).

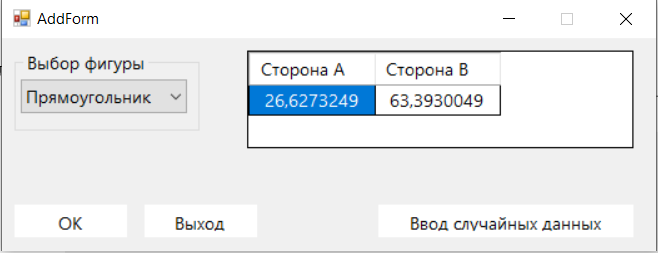


Рисунок 6 – Заполнение полей

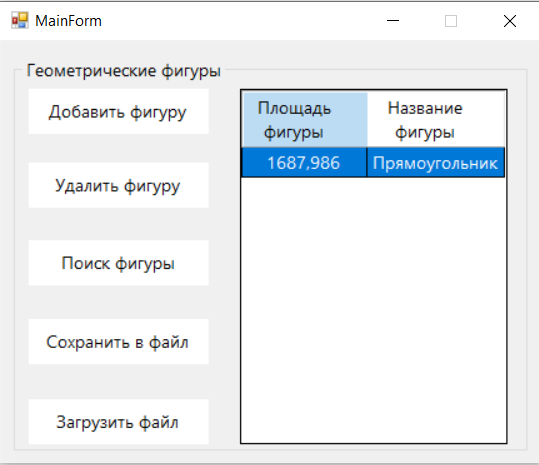


Рисунок 7 – Успешное добавление нового элемента

В программе предусмотрена система обработки некорректного ввода данных пользователем. Например, при попытке ввести текстовые данные вместо числовых появится соответствующее сообщение об ошибке (рисунки 8-9).

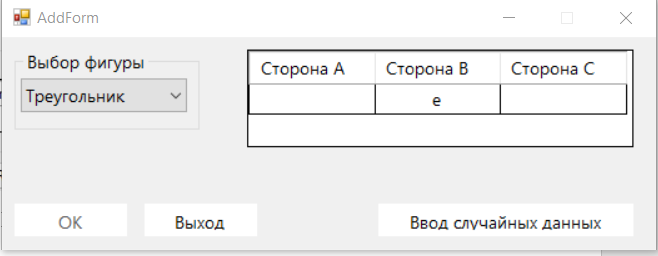


Рисунок 8 – Некорректный ввод (текстовые данные)

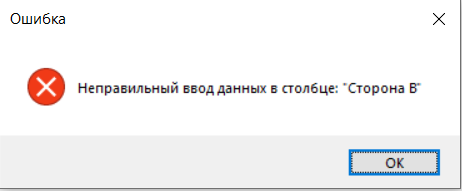


Рисунок 9 – Сообщение об ошибке

Подобная обработка предусмотрена для всех параметров фигуры.

**1.5.2 Тестовый случай «Удалить элемент»**

Для удаления одного или нескольких элементов необходимо выбрать их в таблице и нажать на кнопку «Удалить фигуру» (рисунки 10 и 11).

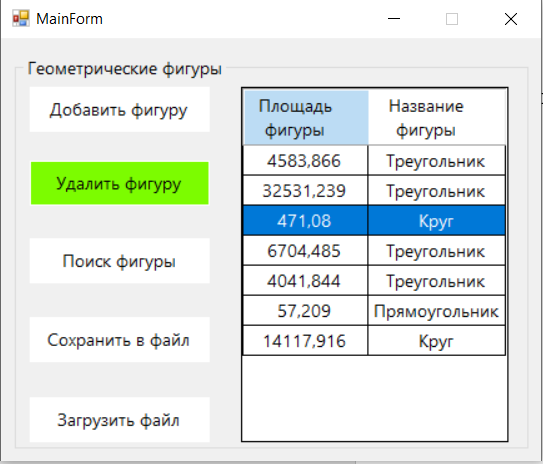


Рисунок 10 – Выбор элемента в таблице

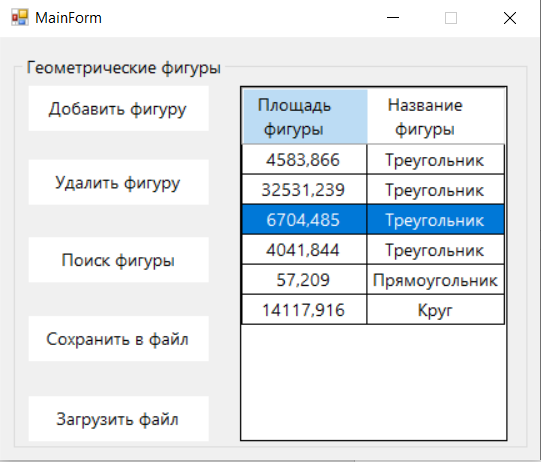


Рисунок 11 – Результат нажатия кнопки «Удалить фигуру»

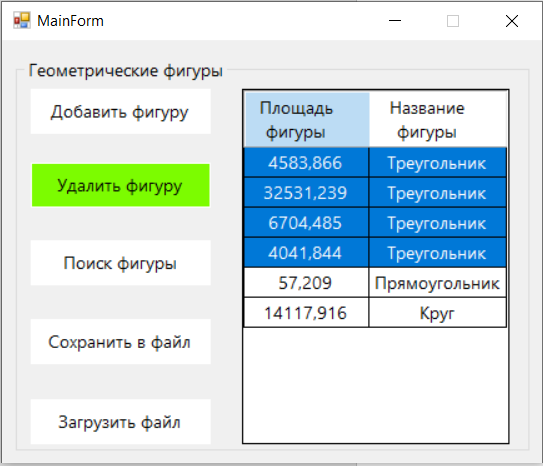


Рисунок 12 – Выбор нескольких элементов для удаления

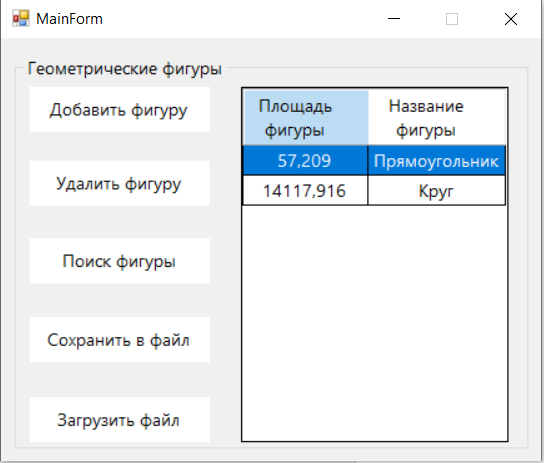


Рисунок 13 – Результат удаления выбранных элементов

**1.5.3 Тестовый случай «Найти элемент»**

Для поиска элементов в списке необходимо нажать кнопку «Поиск фигуры» (рисунок 14). Откроется соответствующая форма для поиска (рисунок 15).

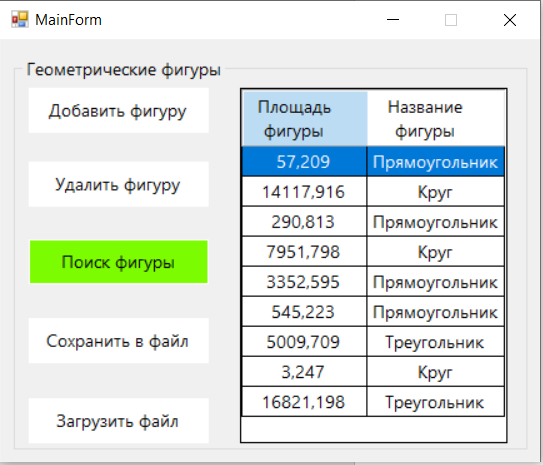


Рисунок 14 – Вызов формы для поиска элементов

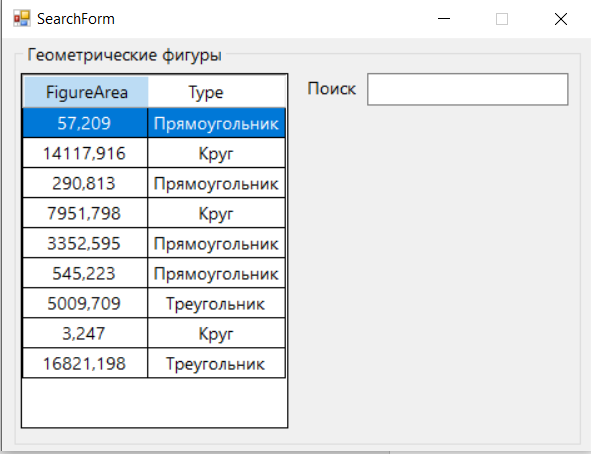


Рисунок 15 – Форма для поиска

Далее пользователь вводит в поле «Поиск» наименование фигуры или её площадь (рисунки 16, 17, 18).

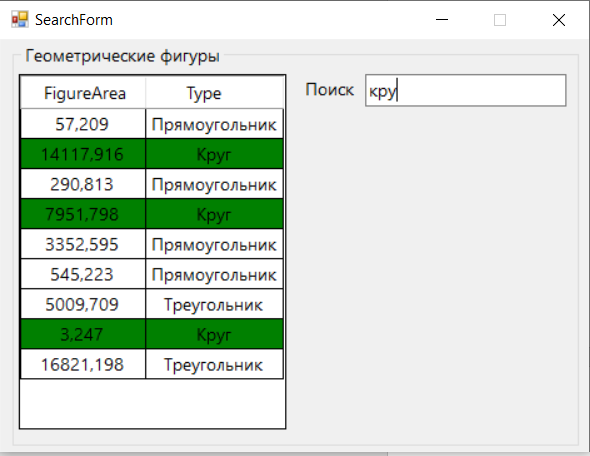


Рисунок 16 – Поиск элемента

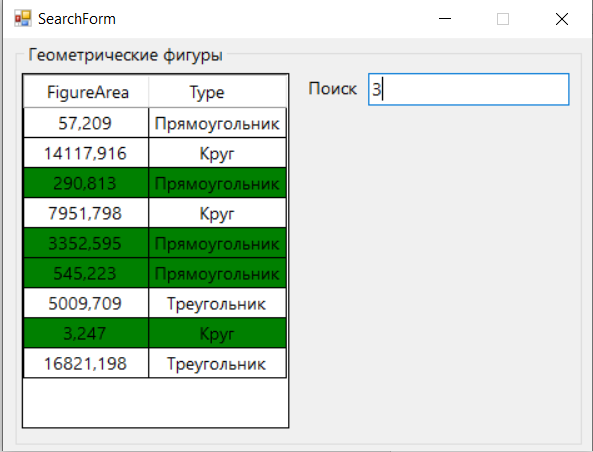


Рисунок 17 – Поиск элемента

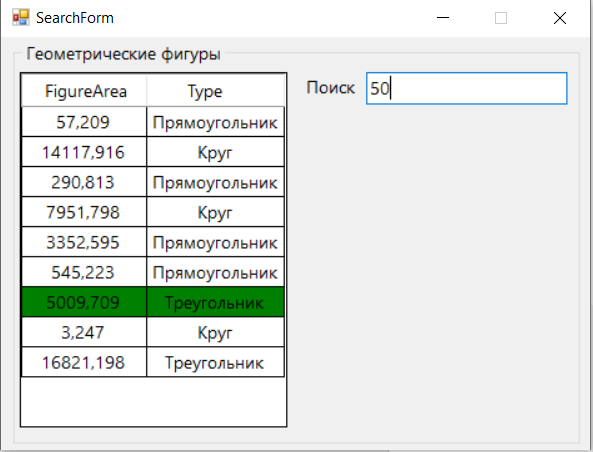


Рисунок 18 – Поиск элемента

**1.5.4 Тестовый случай «Сохранить данные»**

Для сохранения данных в таблице необходимо нажать на панели «Сохранить в файл» (рисунок 19). Откроется системный диалог сохранения файла, где пользователь выбирает директорию и указывает имя файла (рисунок 20).

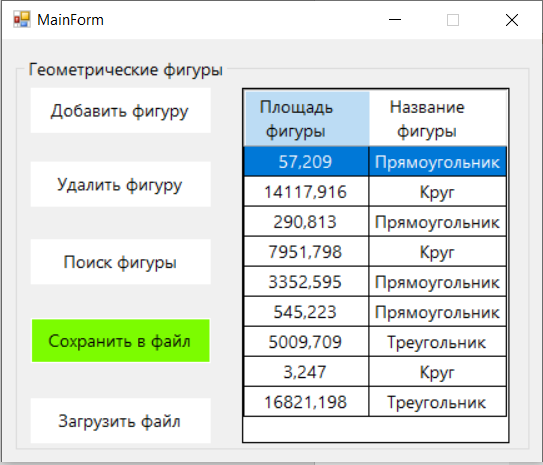


Рисунок 19 – Сохранение данных

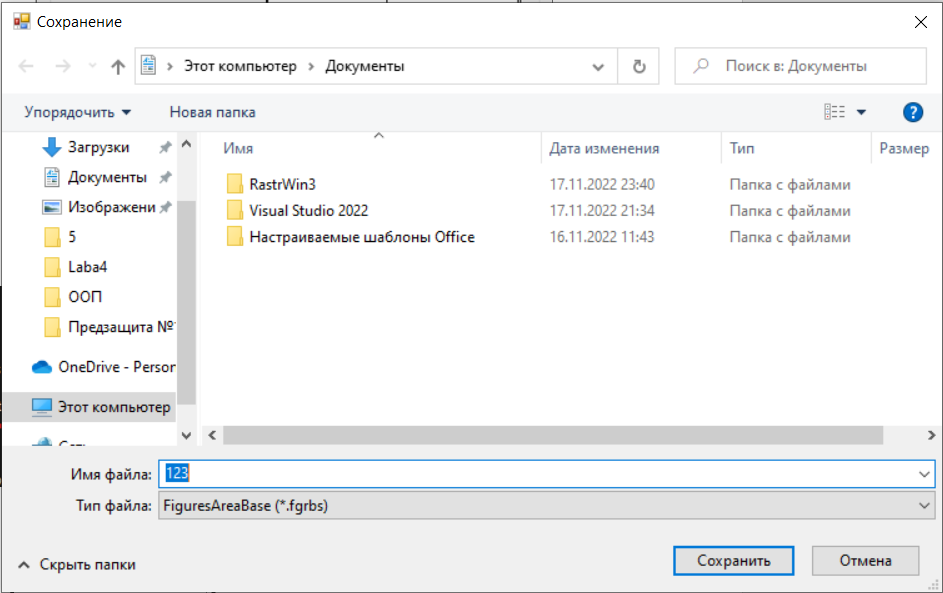


Рисунок 20 – Сохранение файла

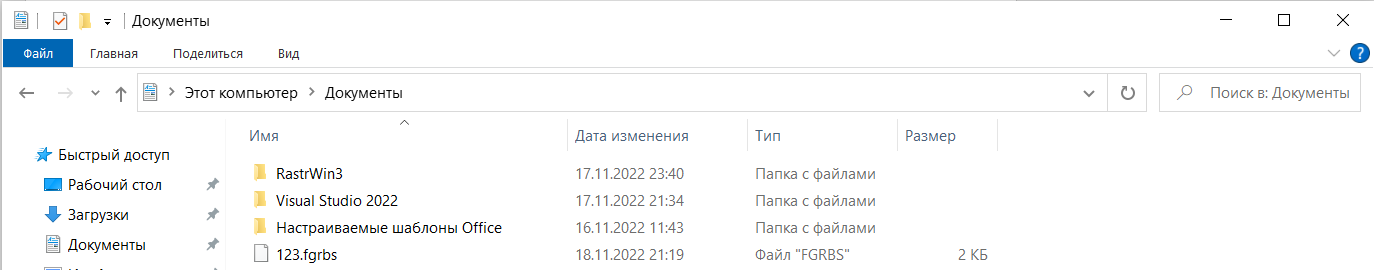


Рисунок 21 – Результат сохранения файла

**1.5.5 Тестовый случай «Загрузить данные»**

Для загрузки данных в таблицу необходимо нажать на соответствующую кнопку (рисунок 22).

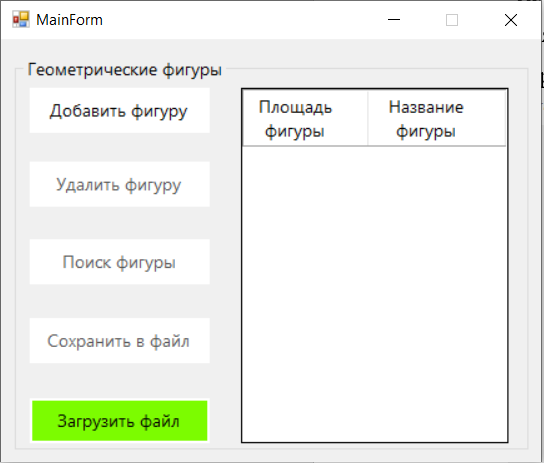


Рисунок 22 – Загрузить данные в таблицу

Далее откроется системный диалог загрузки файла (рисунок 23).

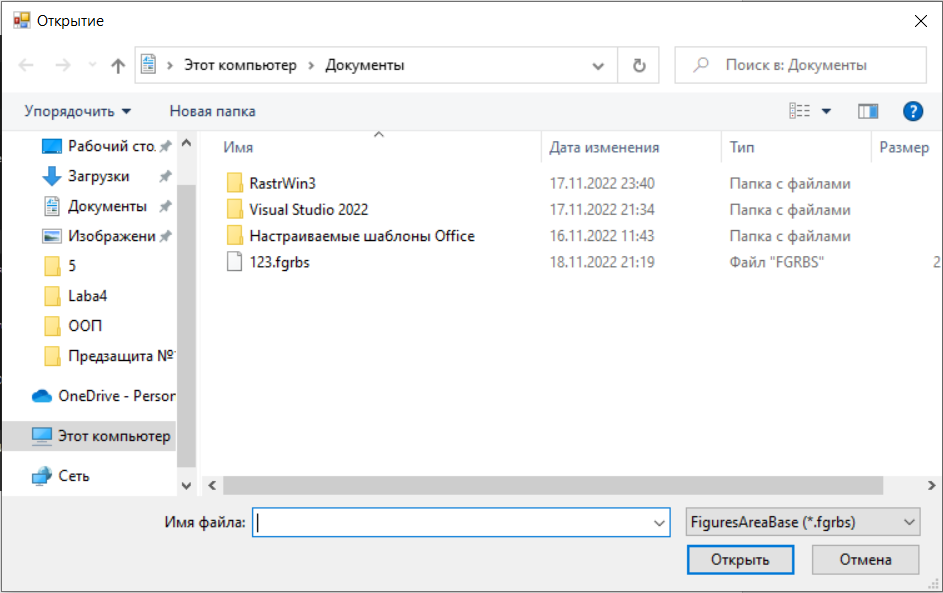


Рисунок 23 – Выбор файла для загрузки

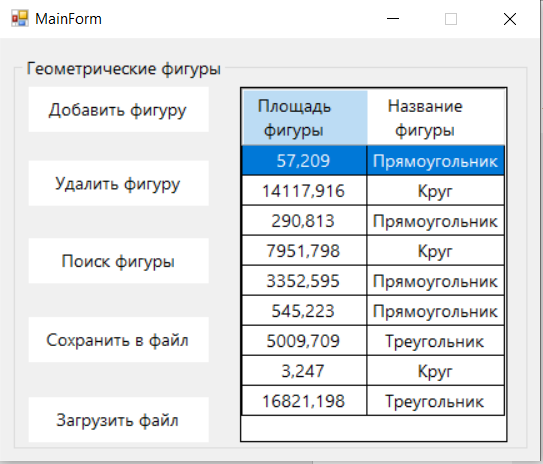


Рисунок 24 – Результат загрузки данных

В случае, если XSD схема файла не соответствует установленному формату, появится соответствующее сообщение (рисунок 25).

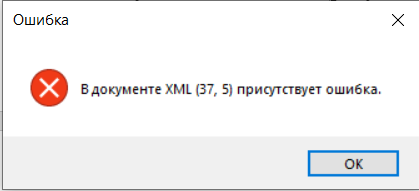


Рисунок 25 – Загрузка повреждённого файла

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Калентьев, А. А. Новые технологии в программировании : учебное пособие / А. А. Калентьев, Д. В. Гарайс, А. Е. Гориянов. – Томск : Эль Контент, 2014. – 176 с. – ISBN 978-5-4332-0185-9.

**ПРИЛОЖЕНИЕ А**

**Техническое задание на создание автоматизированной системы**

Программа для расчёта площадей геометрических фигур

Разработчик: студент гр. О-5КМ01 НИ ТПУ Чугуевский А.С.

Заказчик: Канд. техн. наук, доцент каф. КСУП ТУСУР Калентьев А. А.

Томск 2022

**1 Общие сведения**

**1.1 Полное наименование системы и её условное обозначение**

Полное наименование: «Программа для расчёта площадей геометрических фигур».

Условное обозначение: Система.

**1.2 Наименование предприятий разработчика и заказчика системы**

Заказчик: Канд. техн. наук, доцент каф. КСУП ТУСУР Калентьев А. А.

Разработчик: Студент гр. О-5КМ01 НИ ТПУ Чугуевский А.С.

**1.3 Плановые сроки начала и окончания работы по созданию системы**

Начало работ: 01 апреля 2022 г.

Окончание работ: 02 июня 2022 г.

**2 Назначение и цели создания системы**

**2.1 Назначение системы**

Система предназначена для расчёта площадей геометрических фигур: круга, прямоугольника, треугольника.

**2.2 Цели создания системы**

Система создаётся в целях сокращения времени при расчёте площади геометрических фигур.

**3 Характеристика объектов автоматизации**

Вычисление площади фигур важная задача. Систему можно использовать при расчетах в математических организациях.

**4 Требования к системе**

Таблица 4.1 – Префиксы мнемонических идентификаторов требований и их расшифровка

|  |  |
| --- | --- |
| Префикс | Тип требования |
| A | Архитектурное требование |
| С | Требование к программной или аппаратной совместимости |
| D | Требование к структуре данных |
| F | Функциональное требование |
| U | Требование к пользовательскому интерфейсу |

**4.1 Требования к архитектуре**

**А01.** Система должна быть реализована в виде настольного приложения.

**4.2 Требования к структуре данных**

**D01.** Данные о фигурах должны храниться в XML-файле с расширением \*.fgrbs.

**D01.01.** Формат XML-файла должен соответствовать следующей XSD-схеме:

<FiguresAreaBase xsi:type="Circle">

<Radius></Radius>

</FiguresAreaBase>

<FiguresAreaBase xsi:type="Rectangle">

<SideA></SideA>

<SideB></SideB>

</FiguresAreaBase>

<FiguresAreaBase xsi:type="Triangle">

<SideA></SideA>

<SideB></SideB>

<SideC></SideC>

</FiguresAreaBase>

**4.3 Функциональные требования**

**F01.** Система должна рассчитывать площадь для следующих геометрических фигур:

* круг;
* прямоугольник;
* треугольник.

**F01.01.** Площадь круга должна рассчитываться с учётом величины радиуса.

**F01.02.** Площадь прямоугольника должна рассчитываться с учётом величины двух сторон.

**F01.03.** Площадь треугольника должна рассчитываться с учётом величины трёх сторон.

**F02.** В системе должен быть реализован список фигур.

**F02.01.** Каждый элемент должен иметь следующие параметры:

* площадь фигуры;
* тип фигуры;

**F03.** В системе должна присутствовать функция добавления элементов в список.

**F04.** В системе должна присутствовать функция удаления элементов из списка.

**F05.** В системе должна присутствовать функция поиска элементов по параметрам, указанным в **F02.01**.

**F06.** В системе должна присутствовать функция сохранения списка элементов в файл (**D01**).

**F07.** В системе должна присутствовать функция загрузки списка элементов из файла (**D01**).

**4.4 Требования к пользовательскому интерфейсу**

**U01.** Система должна иметь графический интерфейс пользователя.

**U02.** Данные должны быть представлены в табличном виде.

**U03.** В системе должна быть реализована система обработки ошибок.

**4.5 Требования к программному обеспечению**

**C01.** Система должна работать на операционной системе Windows 10 Pro 20H2. Работоспособность на других выпусках и версиях не гарантируется.

**C02.** На рабочей станции должен быть установлен .NET Framework версии 4.7.2.

**4.6 Требования к аппаратному обеспечению**

**C03.** Для работы системы необходим ПК/ноутбук.

**C04.** 1 гигагерц (GHz) или быстрее.

**C05.** ОЗУ – не менее 1 ГБ для 32-разрядной системы и 2 ГБ для 64-разрядной.

**C06.** Накопитель – 20 ГБ свободного пространства для 64-разрядных и 16 ГБ свободного пространства для 32-разрядных систем.

**C07.** Разрешение экрана – минимум 800 х 600 пикселей.