**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

федеральное государственное автономное образовательное учреждение высшего образования

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ**

**ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

Инженерная школа энергетики

Отделение электроэнергетики и электротехники

Направление: 09.04.03 Прикладная информатика

Отчет по лабораторной работе №5

**«Проектная документация»**

По дисциплине:

Основы объектно-ориентированного программирования



|  |  |  |
| --- | --- | --- |
| Выполнил: студент гр. О-5КМ21 | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) | Кичигин Е. Е. |
| Проверил: доцент, к.т.н. | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (дата) | Калентьев А. А. |
|  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (балл) |  |

Томск - 2024

Оглавление

[Введение 3](#_Toc169018558)

[1 UML диаграмма вариантов использования 4](#_Toc169018559)

[2 UML диаграмма классов 4](#_Toc169018560)

[3 Описание классов, образующих связь типа «общее-частное» 6](#_Toc169018561)

[4 Дерево ветвлений Git 8](#_Toc169018562)

[5 Тестирование программы 9](#_Toc169018563)

[5.1 Тестовый случай «Добавить фигуру» 9](#_Toc169018564)

[5.2 Тестовый случай «Удалить» 11](#_Toc169018565)

[5.3 Тестовый случай «Отфильтровать список фигур» 13](#_Toc169018566)

[5.4 Тестовый случай «Сброс фильтра» 19](#_Toc169018567)

[5.5 Тестовый случай «Сохранить файл» 20](#_Toc169018568)

[5.6 Тестовый случай «Загрузить файл» 22](#_Toc169018569)

[Список источников 24](#_Toc169018570)

[ПРИЛОЖЕНИЕ А 25](#_Toc169018571)

**Введение**

Назначение программной документации заключается в детальном описании всех аспектов разрабатываемого программного обеспечения. В контексте данной лабораторной работы, документация играет ключевую роль, предоставляя структурированную информацию о разработке программы, предназначенной для вычисления площади различных геометрических фигур: круга, прямоугольника и треугольника. Эта документация предназначена для облегчения понимания, сопровождения и дальнейшего развития программы.

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

Для достижения поставленной цели должны быть выполнены следующие задачи:

* Составление UML диаграммы вариантов использования для разработанной программы;
* Составление UML диаграммы классов;
* Описание классов, образующих связь типа «общее-частное»;
* Дерево ветвлений Git;
* Тестирование программы;
* Техническое задание.

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].

Диаграмма вариантов использования для разработанного ПО приведена на рисунке 1.

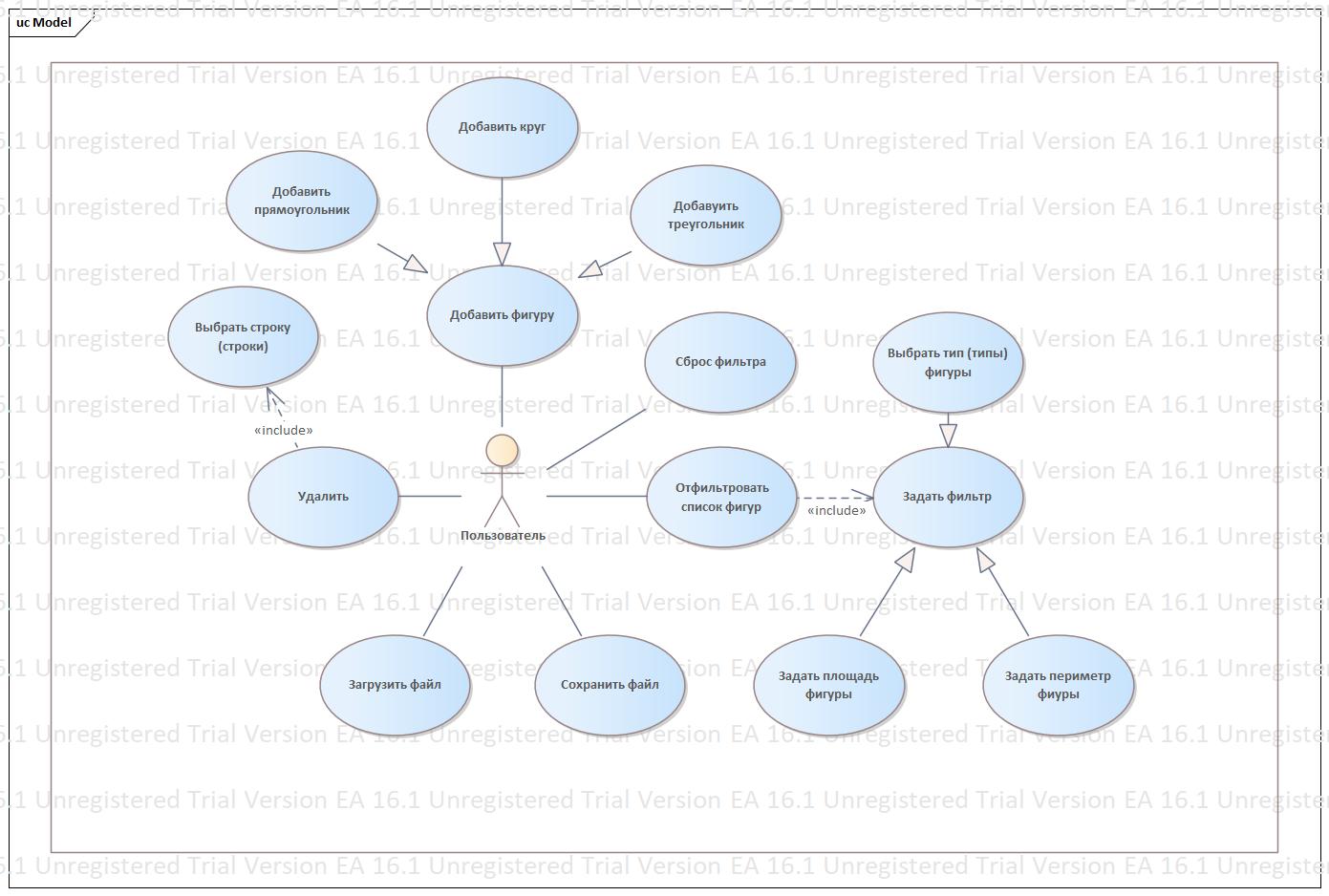


Рисунок 1 – Диаграмма вариантов использования

1. **UML диаграмма классов**

Диаграмма классов — один из видов UML-диаграмм, позволяющий описать статический аспект программной системы за счёт описания классов и их взаимосвязей в системе [1].

Диаграмма классов приведена на рисунке 2.

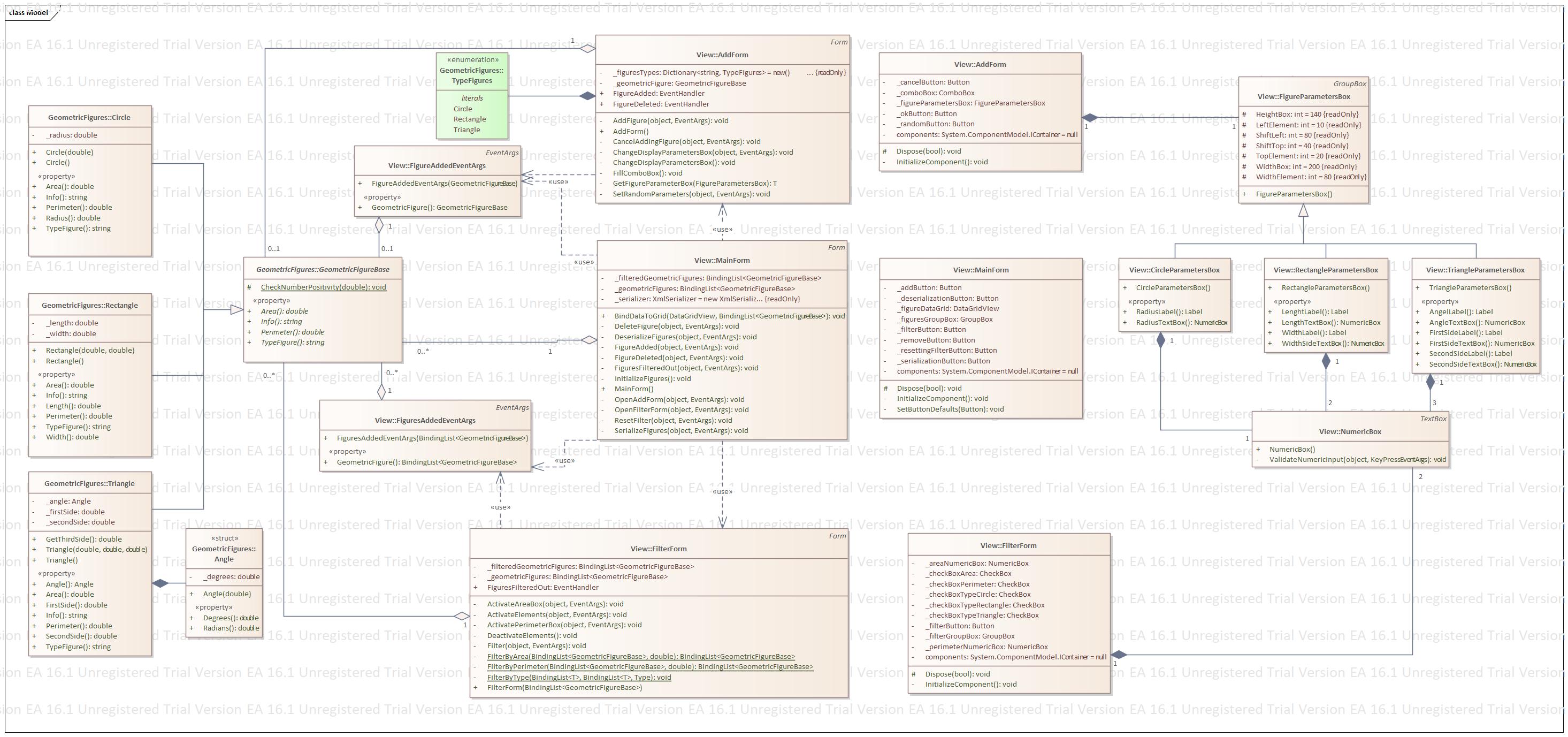


Рисунок 2 – UML диаграмма классов

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

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

Таблица 1 – Описание класса GeometricFigureBase

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс *GeometricFigureBase* – абстрактный базовый класс, описывает общую природу всех геометрических фигур | | |
| Свойства | | |
| + Area | string | Площадь фигуры.  Абстрактное свойство, переопределяется в производных классах. |
| + Info | string | Информация о фигуре.  Абстрактное свойство, переопределяется в производных классах. |
| + Perimeter | double | Периметр фигуры.  Абстрактное свойство, переопределяется в производных классах. |
| + TypeFigure | string | Тип фигуры  Абстрактное свойство, переопределяется в производных классах. |
| Методы | | |
| # CheckNumberPositivity | double | Проверяет корректность ввода параметров фигур  double – любой параметр фигуры |

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

Таблица 2 – Описание класса Circle

| Название | Тип | Описание |
| --- | --- | --- |
| Описание класса | | |
| Класс Circle – круг | | |
| Поля | | |
| – \_radius | double | Радиус круга |
| Свойства | | |
| + Area | double | Площадь круга |
| + Info | string | Информация о круге |
| + Perimeter | double | Периметр круга |
| + Radius | double | Радиус круга |
| + TypeFigure | string | Тип фигуры |

Таблица 3 – Описание класса Rectangle

| Название | Тип | Описание |
| --- | --- | --- |
| Описание класса | | |
| Класс Rectangle – прямоугольник | | |
| Поля | | |
| – \_length | double | Длинна прямоугольника |
| – \_width | double | Ширина прямоугольника |
| Свойства | | |
| + Area | double | Площадь прямоугольника |
| + Info | string | Информация о прямоугольнике |
| + Length | double | Длинна прямоугольника |
| + Perimeter | double | Периметр прямоугольника |
| + TypeFigure | string | Тип фигуры |
| + Width | double | Ширина прямоугольника |

Таблица 4 – Описание класса Triangle

| Название | Тип | | Описание | |
| --- | --- | --- | --- | --- |
| Описание класса | | | | |
| Класс Triangle – треугольник | | | | |
| Поля | | | | |
| – \_angle | Angle | | Угол между сторонами треугольника | |
| – \_firstSide | double | | Первая сторона треугольника | |
| – \_secondSide | double | | Вторая сторона треугольника | |
| Свойства | | | | |
| + Angle | Angle | | Угол между сторонами треугольника | |
| + Area | double | | Площадь треугольника | |
| + FirstSide | double | | Первая сторона треугольника | |
| + Info | string | | Информация о треугольнике | |
| + Perimeter | double | | Периметр треугольника | |
| + SecondSide | double | | Вторая сторона треугольника | |
| + TypeFigure | string | | Тип фигуры | |
| Методы | | | | |
| + GetThirdSide | | double | | Рассчитывает значение длинны третьей стороны треугольника |

Таблица 5 – Описание структуры Angle

| Название | Тип | Описание |
| --- | --- | --- |
| Описание класса | | |
| Структура Angle– угол между двумя прямыми на плоскости | | |
| Поля | | |
| – \_degrees | double | Градусы |
| Свойства | | |
| + Degrees | double | Угол в градусах. |
| + Radians | double | Угол в радианах. |

1. **Дерево ветвлений Git**

Git [2] — распределённая система управления версиями файлов. Система спроектирована как набор утилит командной строки, специально разработанных с учётом их использования в скриптах. Это позволяет удобно создавать специализированные системы контроля версий на базе Git или пользовательские интерфейс.

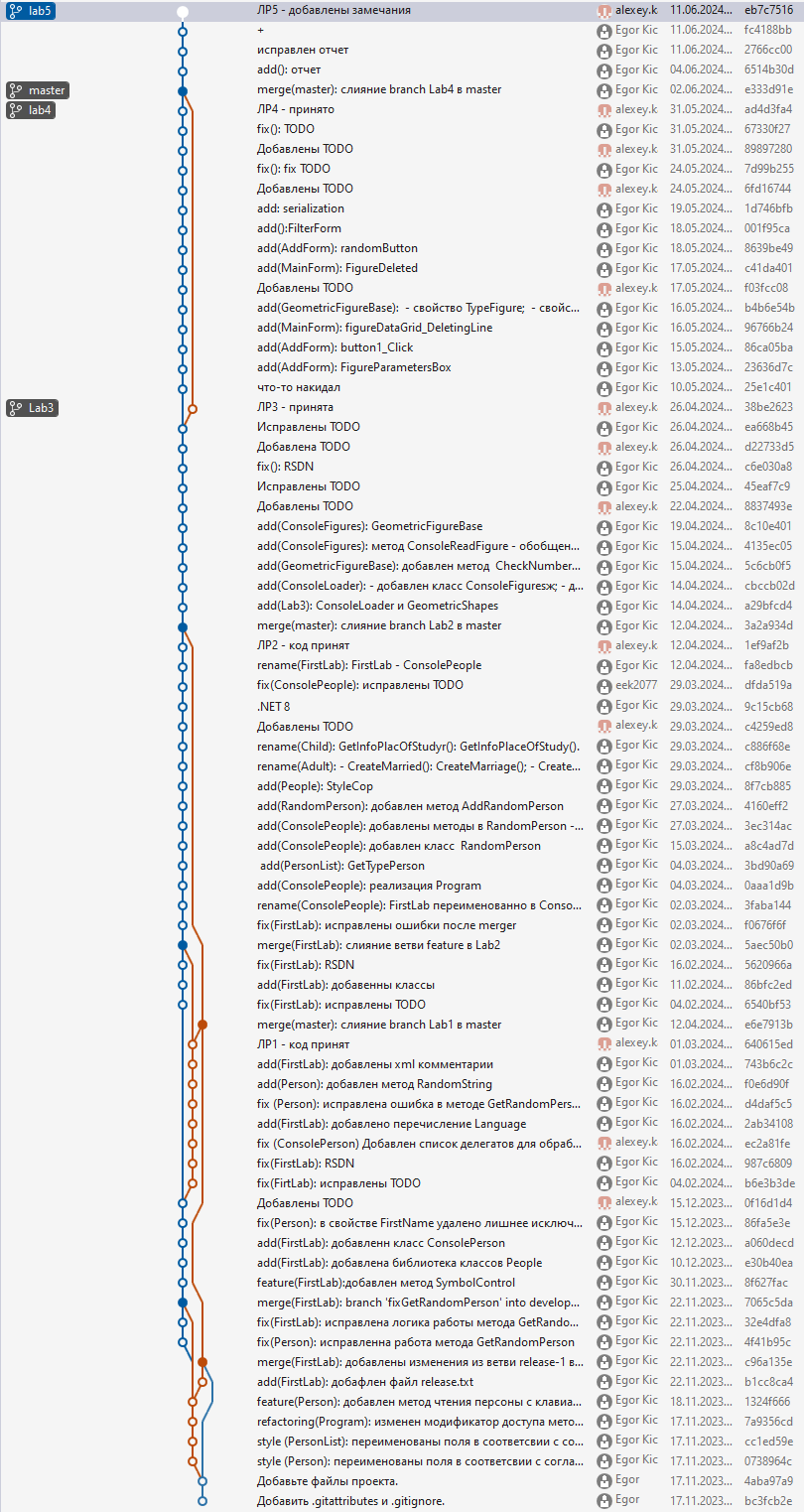


Рисунок 3 – Дерево Git

1. **Тестирование программы**

Далее приводится процесс функционального тестирования программы. Графический интерфейс пользователя представлен на рисунке 3

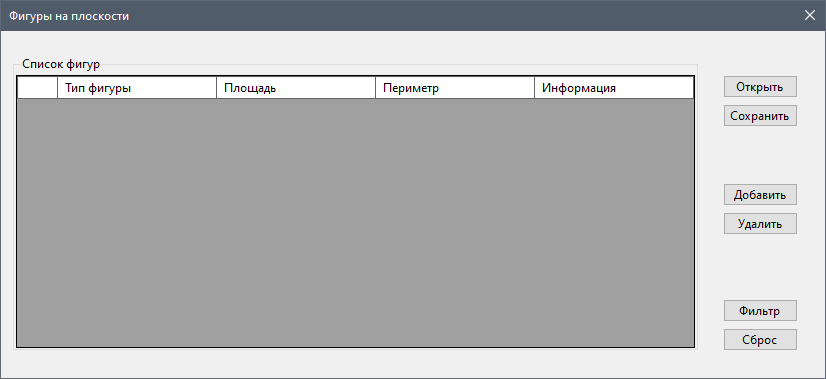


Рисунок 4 – Графический интерфейс пользователя

* 1. **Тестовый случай «Добавить фигуру»**

Для добавления фигуры необходимо вызвать соответствующую форму путём нажатия кнопки «Добавить», рисунок 4.

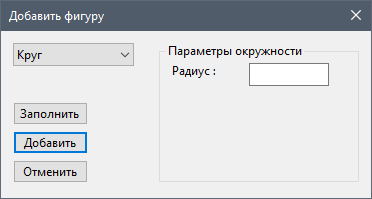


Рисунок 5 – Форма для добавления фигуры

Параметры любой из выбранных фигур (круга, прямоугольника, треугольника) можно ввести, выбрав соответствующую фигуру в выпадающем меню.

После ввода данных необходимо нажать кнопку «добавить», фигура появится в таблице главной формы, рисунки 5 – 7).

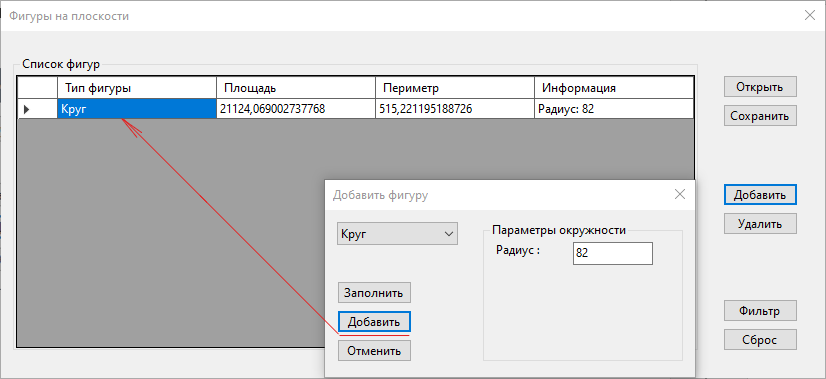


Рисунок 6 – Добавление круга

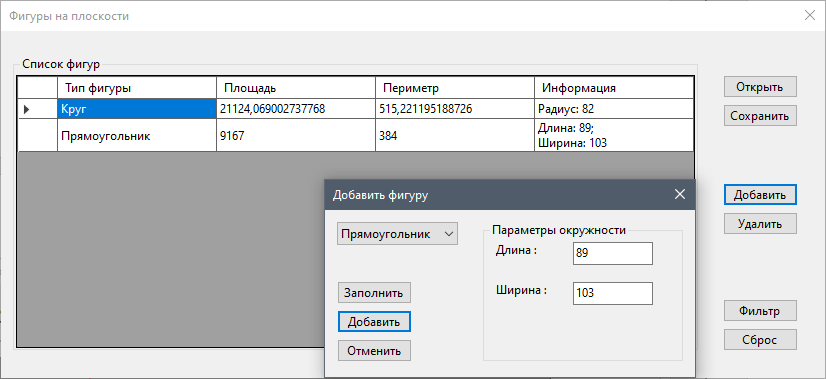


Рисунок 7 – Добавление прямоугольника

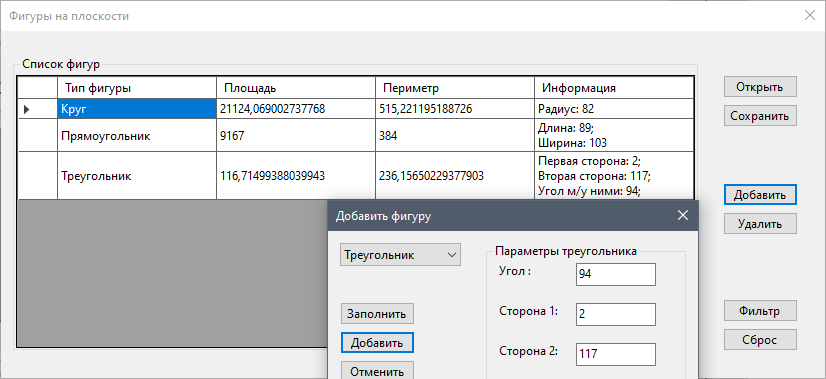


Рисунок 8 – Добавление треугольника

В программе предусмотрена система обработки некорректного ввода данных пользователем. Например, пользователь не сможет ввести несколько точек в числе или отрицательное число, также программа не позволит ввести любой символ кроме точки или цифры. Если пользователь оставляет поле незаполненным и нажимает «Добавить», то появится соответствующее сообщение об ошибке, рисунки 9–10.

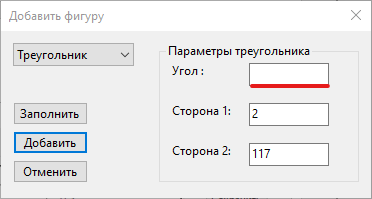


Рисунок 9 – Некорректный ввод

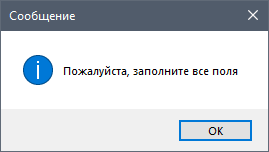


Рисунок 10 –Предупреждение пользователя о некорректном вводе

* 1. **Тестовый случай «Удалить»**

Для удаления одной или нескольких фигур необходимо выбрать их в таблице и нажать на кнопку «Удалить», рисунки 10 – 13.

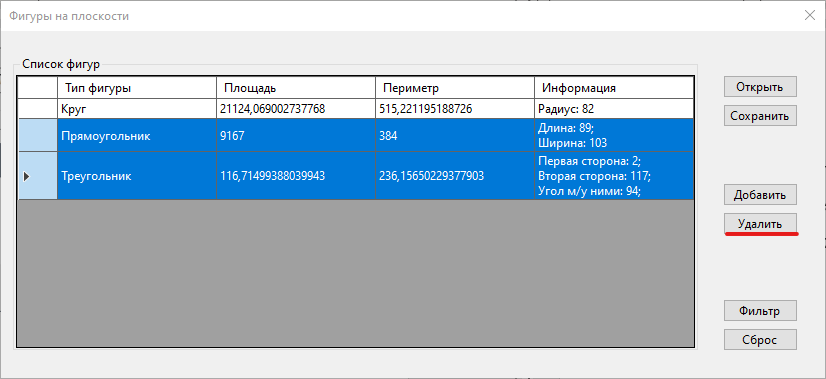


Рисунок 11 – Выбор фигур в таблице

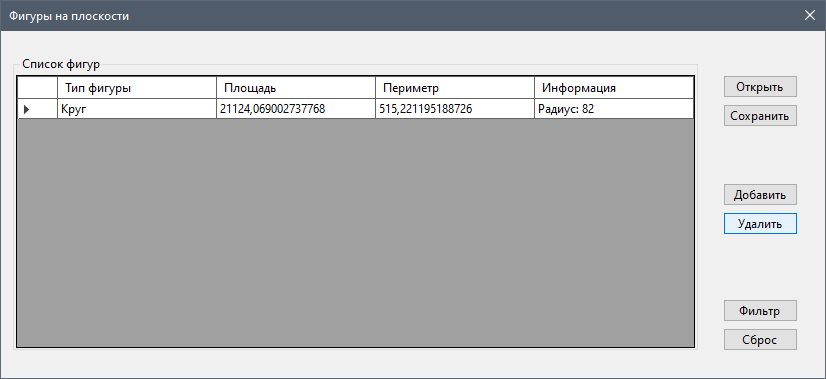


Рисунок 12 – Результат нажатия кнопки «Удалить»

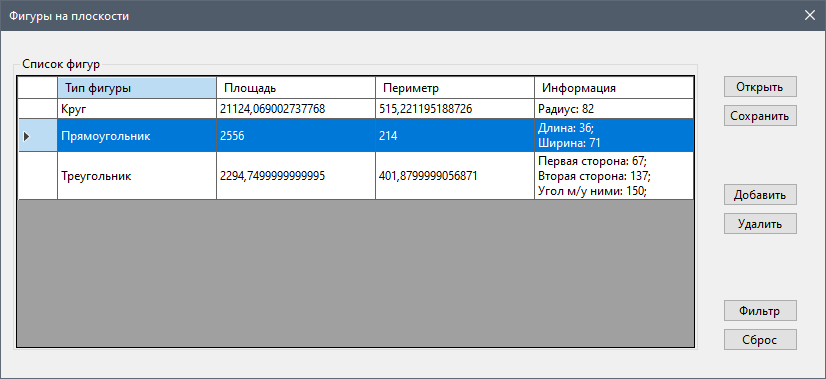


Рисунок 13 – Выбор фигуры в таблице

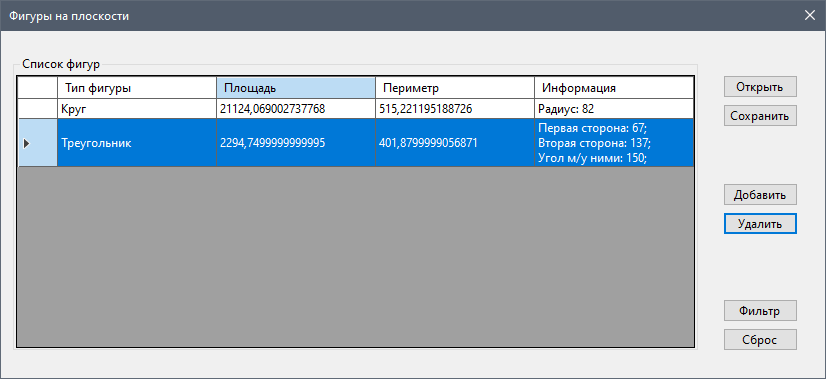


Рисунок 14 – Результат нажатия кнопки «Удалить»

* 1. **Тестовый случай «Отфильтровать список фигур»**

Для фильтрации фигур по определенным критериям необходимо нажать кнопку «Фильтр» (рисунок 14). Откроется соответствующая форма для поиска (рисунок 15).

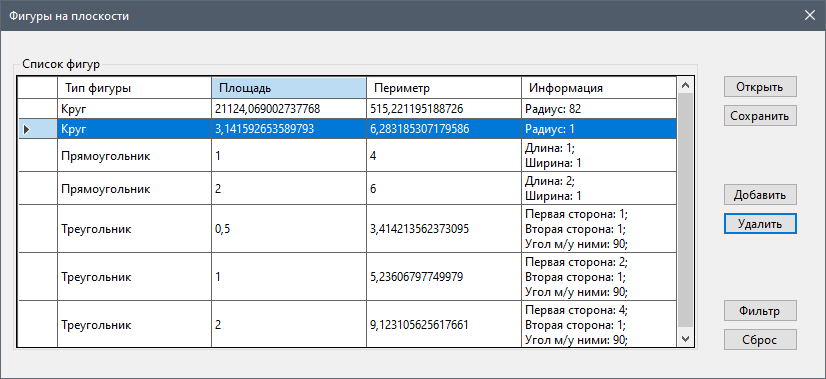


Рисунок 15 – Вызов формы для фильтрации списка фигур

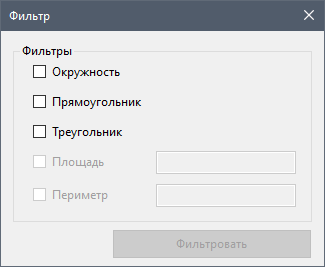


Рисунок 16 – Форма фильтрации списка фигур

Ниже приведены примеры фильтрации списка фигур по разным критериям.

* Тип «Окружность»

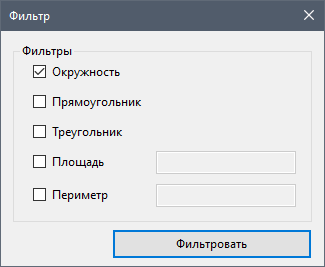


Рисунок 17 – Поиск фигуры по типу «Окружность»

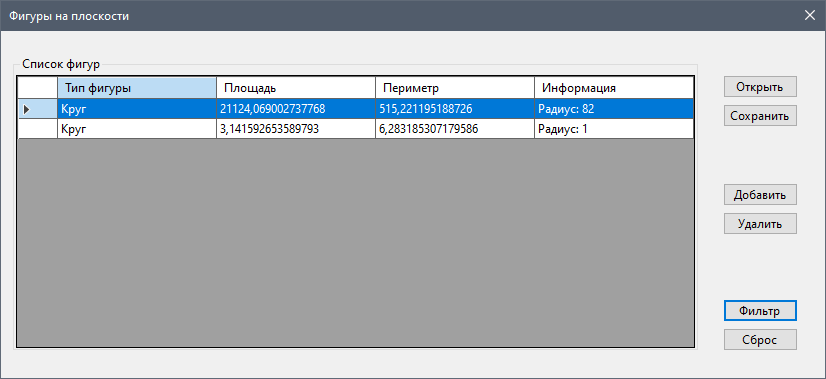


Рисунок 18 – Результат поиск фигуры по типу «Окружность»

* Тип «Прямоугольник»

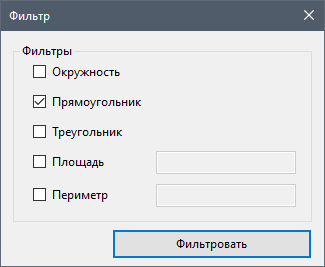


Рисунок 19 – Поиск фигуры по типу «Прямоугольник»

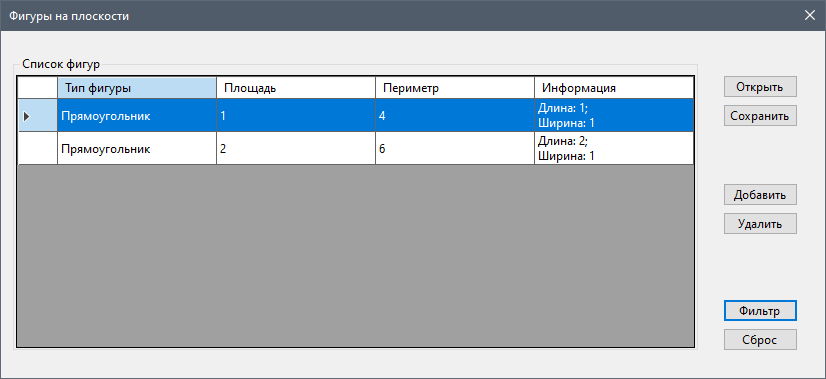


Рисунок 20 – Результат поиск фигуры по типу «Прямоугольник»

* Тип «Треугольник»

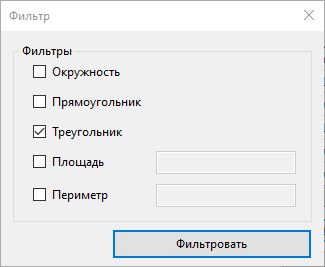


Рисунок 21 – Поиск фигуры по типу «Треугольник»

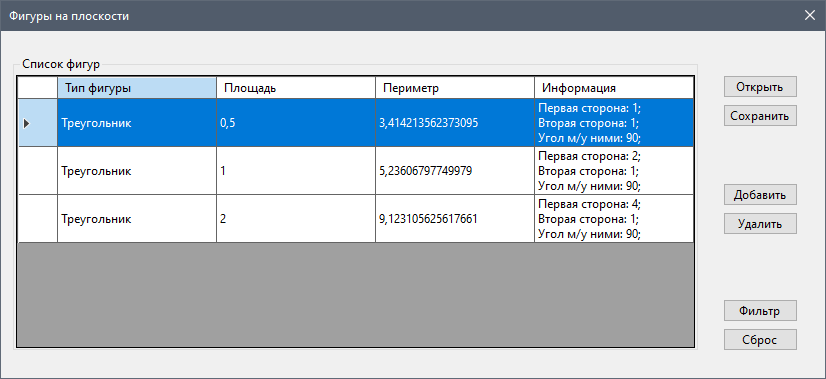


Рисунок 22 – Результат поиск фигуры по типу «Треугольник»

* Значение «Площадь»

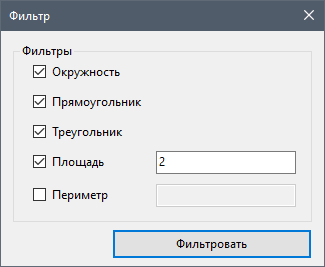


Рисунок 23 – Поиск фигуры по площади

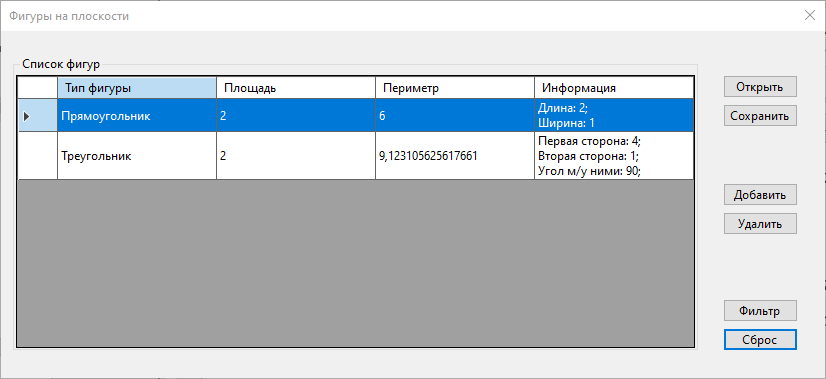


Рисунок 24 – Результат поиска фигуры по площади

* Значение «Периметр»

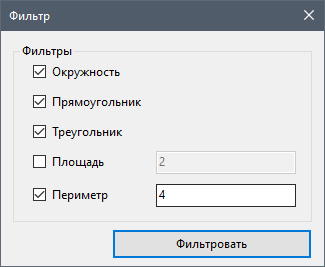


Рисунок 25 – Поиск фигуры по периметру

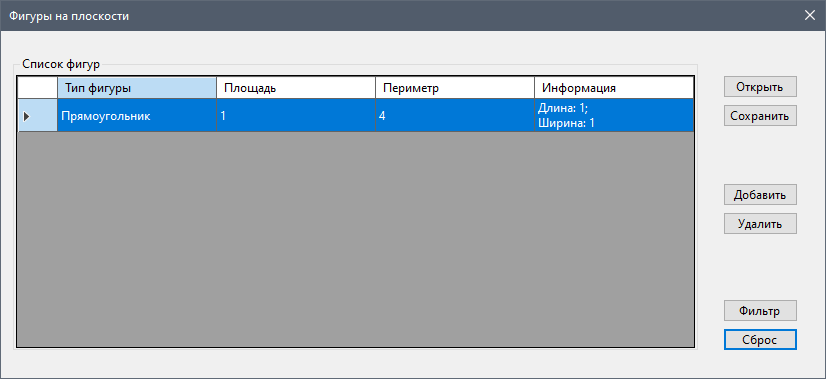


Рисунок 26 – Результат поиска фигуры по периметру

* Значение «Площадь» и «Периметр»

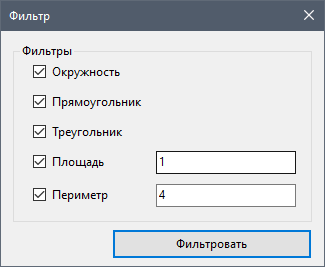


Рисунок 27 – Поиск фигуры по площади и периметру

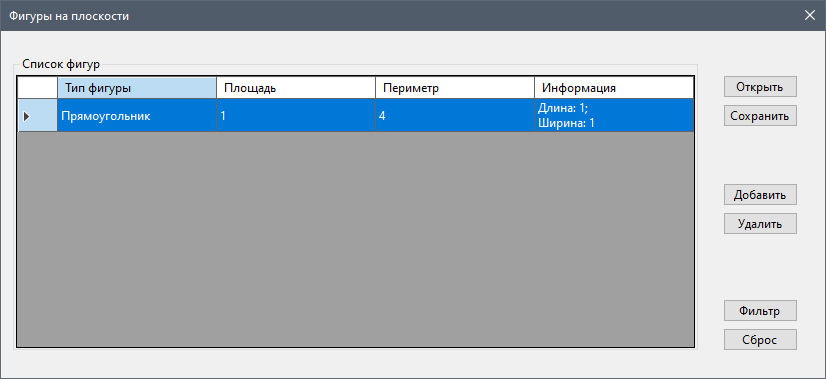


Рисунок 28 – Результат поиска фигуры по площади и периметру

На рисунках 28 и 29 представлена реакция системы на некорректное поведение пользователя.

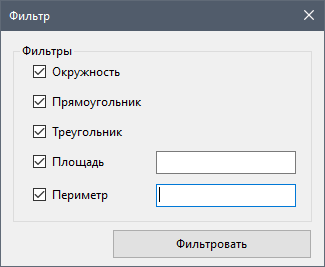


Рисунок 29 – Нажатие кнопки Фильтровать при незаполненных полях Площадь и Периметр

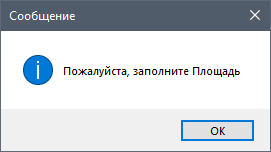


Рисунок 30 – предупреждение пользователя

* 1. **Тестовый случай «Сброс фильтра»**

На рисунках 30 и 31 представлена работа функции сброса фильтра.

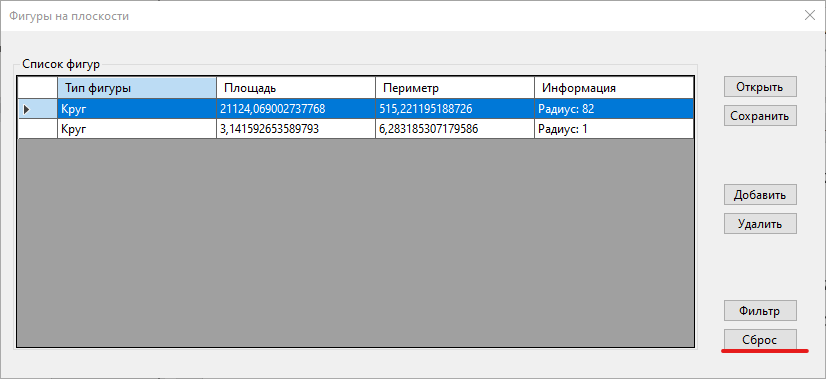


Рисунок 31 – Нажатие кнопки Сброс

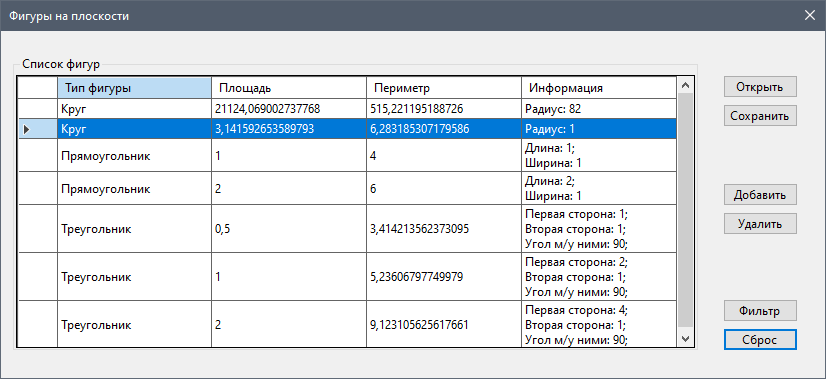


Рисунок 32 – Отображение исходного списка после сброса фильтра

* 1. **Тестовый случай «Сохранить файл»**

Для сохранения данных в файл необходимо нажать на кнопку «Сохранить», рисунок 32. Откроется системный диалог сохранения файла, где пользователь выбирает директорию и указывает имя файла, рисунок 33.

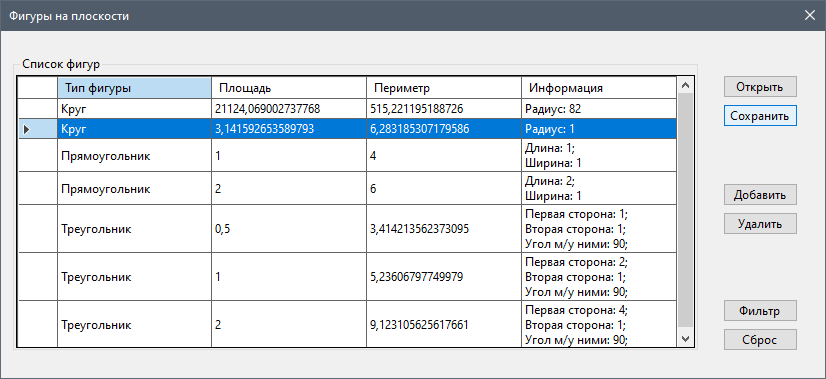


Рисунок 33 – Сохранение данных

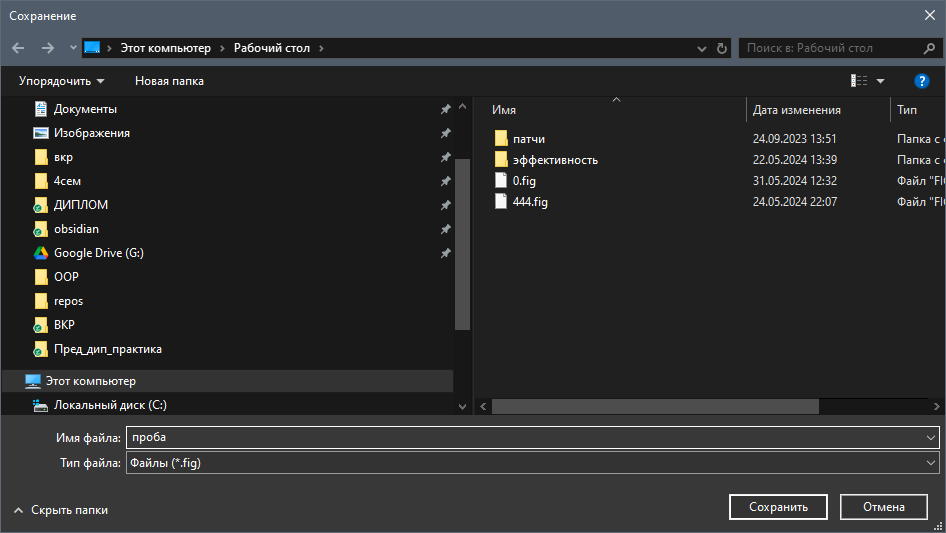


Рисунок 34 – Сохранение файла

В случае, если таблица пуста, сохранение не производится, рисунок 34.

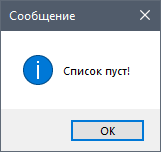


Рисунок 35 – Результат нажатия на кнопку «Сохранить» при пустой таблице

* 1. **Тестовый случай «Загрузить файл»**

Для загрузки данных в таблицу необходимо нажать на кнопку «Открыть», рисунок 35.

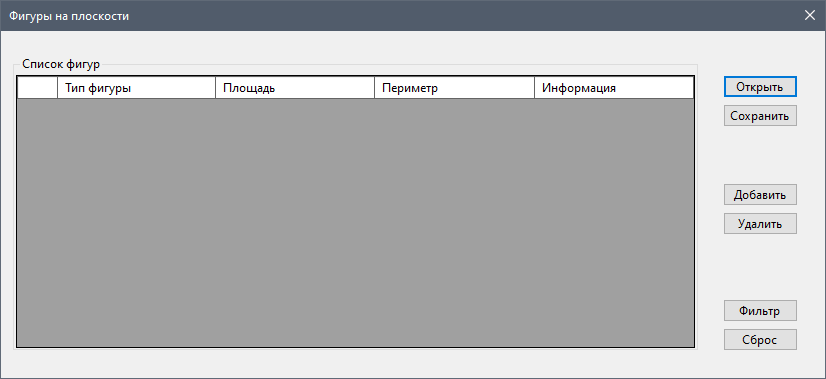


Рисунок 36 – Загрузить данные в таблицу

Далее откроется системный диалог загрузки файла, рисунок 36.

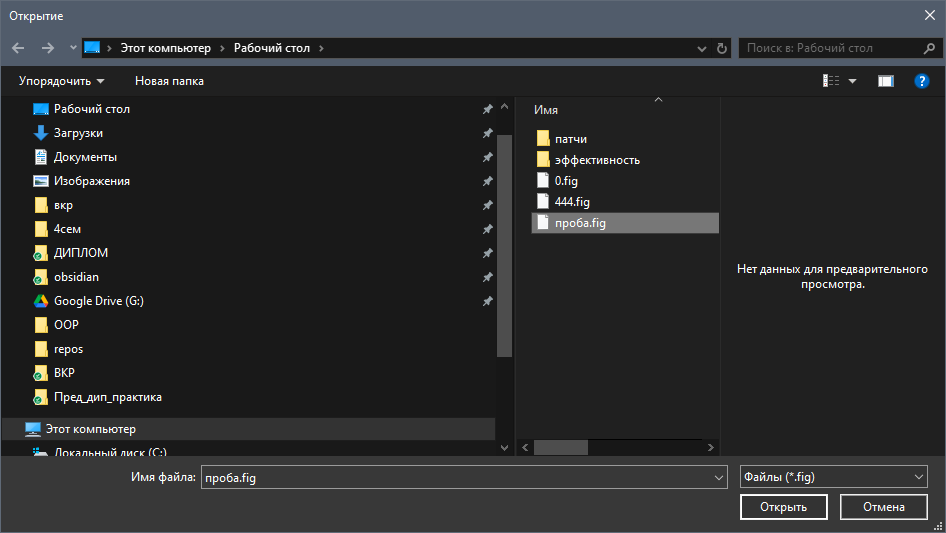


Рисунок 37 – Выбор файла для загрузки

В случае, если структура загружаемого файла не распознана, в случае если в файле присутствуют некорректные значения параметров или значения отсутствуют, появится соответствующее сообщение, представленное на рисунке 37.

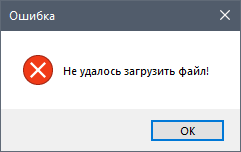


Рисунок 38 – Загрузка некорректного файла

**Список 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.
2. git [Электронный ресурс]. — URL : http://git-scm.com/ (Дата обращения: 11.06.2024)

**ПРИЛОЖЕНИЕ А**

**Техническое задание на создание автоматизированной системы**

Программа для расчёта фигур на плоскости

Разработчик: студент гр. [О-5КМ21](https://iso.tpu.ru/magistracy/rasp/rasp_5am0r/) НИ ТПУ Кичигин Е.Е.

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

Томск 2024

**1 Общие сведения**

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

Полное наименование: «Программа для расчёта фигур на плоскости».

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-5КМ21](https://iso.tpu.ru/magistracy/rasp/rasp_5am0r/) НИ ТПУ Кичигин Е.Е.

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

Начало работ: 10 мая 2024 г.

Окончание работ: 20 июня 2024 г.

**2 Назначение и цели создания системы**

**2.1 Назначение системы**

Система предназначена для расчёта площадей плоских фигур: окружности, прямоугольника, треугольника.

**2.2 Цели создания системы**

Система создаётся в целях сокращения трудозатрат людей, которым необходимо быстро рассчитывать площади плоских поверхностей~~.~~

**3 Характеристика объектов автоматизации**

Вычисление площади различных фигур может пригодиться в любой сфере жизнедеятельности человека.

Поскольку такие расчёты выполняются повсеместно и довольно часто, представляется целесообразным автоматизировать этот процесс.

**4 Требования к системе**

Таблица А.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 параметрах элементов электрических схем должны храниться в XML-файле с расширением \*.fig.

Структура XML-документа представляет собой иерархическую организацию элементов, начиная от корневого элемента и включая вложенные элементы. В случае данного XML-документа структура будет следующей:

**<ArrayOfGeometricFigureBase>** <!-- Корневой элемент -->

**<GeometricFigureBase>** <!-- Элемент представляющий геометрическую фигуру -->

<!-- Дополнительные элементы, зависящие от типа геометрической фигуры -->

**</GeometricFigureBase>**

<!-- Возможно, другие элементы GeometricFigureBase, представляющие другие геометрические фигуры -->

**</ArrayOfGeometricFigureBase>**

Для каждого типа геометрической фигуры внутри элемента *GeometricFigureBase* будут различные дочерние элементы, содержащие информацию о характеристиках каждой фигуры.

* Для типа*Circle*:

**<GeometricFigureBase xsi:type="Circle">**

**<Radius>** <!-- Радиус круга --> **</Radius>**

**</GeometricFigureBase>**

* Для типа*Rectangle*:

**<GeometricFigureBase xsi:type="Rectangle">**

**<Length>** <!-- Длина прямоугольника --> **</Length>**

**<Width>** <!-- Ширина прямоугольника --> **</Width>**

**</GeometricFigureBase>**

* Для типа*Triangle*:

**<GeometricFigureBase xsi:type="Triangle">**

**<FirstSide>** <!-- Длина первой стороны треугольника --> **</FirstSide>**

**<SecondSide>** <!-- Длина второй стороны треугольника --> **</SecondSide>**

**<Angle>** <!-- Угол треугольника -->

**<Degrees>** <!-- Угол в градусах --> **</Degrees>**

**<Radians>** <!-- Угол в радианах --> **</Radians>**

**</Angle>**

**</GeometricFigureBase>**

**4.3 Функциональные требования**

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

* окружность;
* прямоугольник;
* треугольник.

**F01.01.** Площадь окружности должна определяться по выражению

где *π* – число ПИ о.е.;

*r* – радиус окружности, м, в диапазоне от 0 до 1,7 × 10**308**;

**F01.02.** Площадь прямоугольника должна определяться по выражению

где *a* – длина прямоугольника, м, в диапазоне от 0 до 1,7 × 10**308**;

*b* – ширина прямоугольника, м, в диапазоне от 0 до 1,7 × 10**308**.

**F01.03.** Площадь треугольника должна определяться по выражению

где *a* – первая сторона треугольника, м, в диапазоне от 0 до 1,7 × 10**308**;

*b* – вторая сторона треугольника, м, в диапазоне от 0 до 1,7 × 10**308**;

*δ –* угол между двумя сторонами треугольника от 0 до 180.

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

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

* тип фигуры;
* площадь.

**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 64-разрядной операционной системе Windows 10 версии 22H2 и выше. Работоспособность на других выпусках и версиях не гарантируется.

**C02.** На рабочей станции должен быть установлен .NET Framework версии 8.0.

**4.6 Требования к аппаратному обеспечению**

**C03.** Частота процессора не менее 2500 МГц.

**C04.** Объём оперативной памяти не менее 2 ГБ.

**C05.** Объём HDD/SSD памяти не менее 1 ГБ.