|  |
| --- |
| **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-5КМ91 | |  | \_\_\_\_\_\_\_\_\_\_ | |  | Козырев А. М. |
|  | |  |  | |  |  |
|  | |  |  | 06 декабря 2021 г. | | |
|  | |  |  | |  |  |
| Отчёт принял | доцент, к.т.н. |  | \_\_\_\_\_\_\_\_\_\_ | |  | Калентьев А. А. |
|  |  |  |  | |  |  |
|  |  |  |  | \_\_\_ \_\_\_\_\_\_ 2021 г. | | |

Томск 2021

**Содержание**

[**Введение** 3](#_Toc74829062)

[**1 Описание программной системы** 4](#_Toc74829063)

[1.1 UML диаграмма вариантов использования 4](#_Toc74829064)

[1.2 UML диаграмма классов 4](#_Toc74829065)

[1.3 Описание классов, образующих связь типа «общее-частное» 6](#_Toc74829066)

[1.4 Дерево ветвлений Git 9](#_Toc74829067)

[1.5 Тестирование программы 10](#_Toc74829068)

[**Список использованных источников** 22](#_Toc74829069)

[**ПРИЛОЖЕНИЕ А** 23](#_Toc74829070)

**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:

1. Составление технического задания (далее – ТЗ) на разработанную программу (ТЗ представлено в Приложении А);
2. Составление UML диаграммы вариантов использования для разработанной программы;
3. Составление UML диаграммы классов;
4. Описание классов, образующих связь типа «общее-частное»;
5. Привести дерево ветвлений Git;
6. Провести тестирование программы.

**1 Описание программной системы**

**1.1 UML диаграмма вариантов использования**

Диаграмма вариантов использования приведена на рисунке 1.

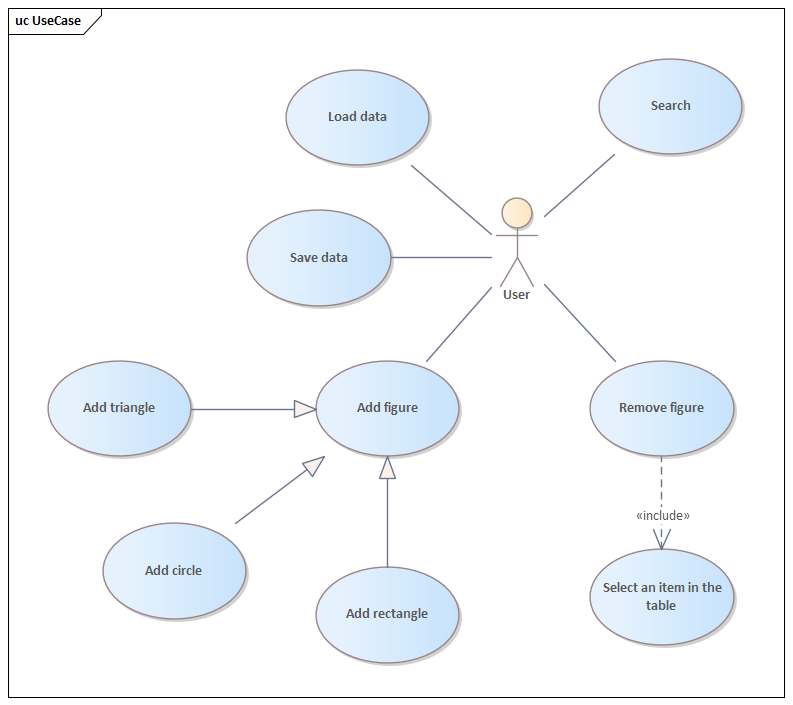


Рисунок 1 – Диаграмма вариантов использования

**1.2 UML диаграмма классов**

Диаграмма классов приведена на рисунке 2.

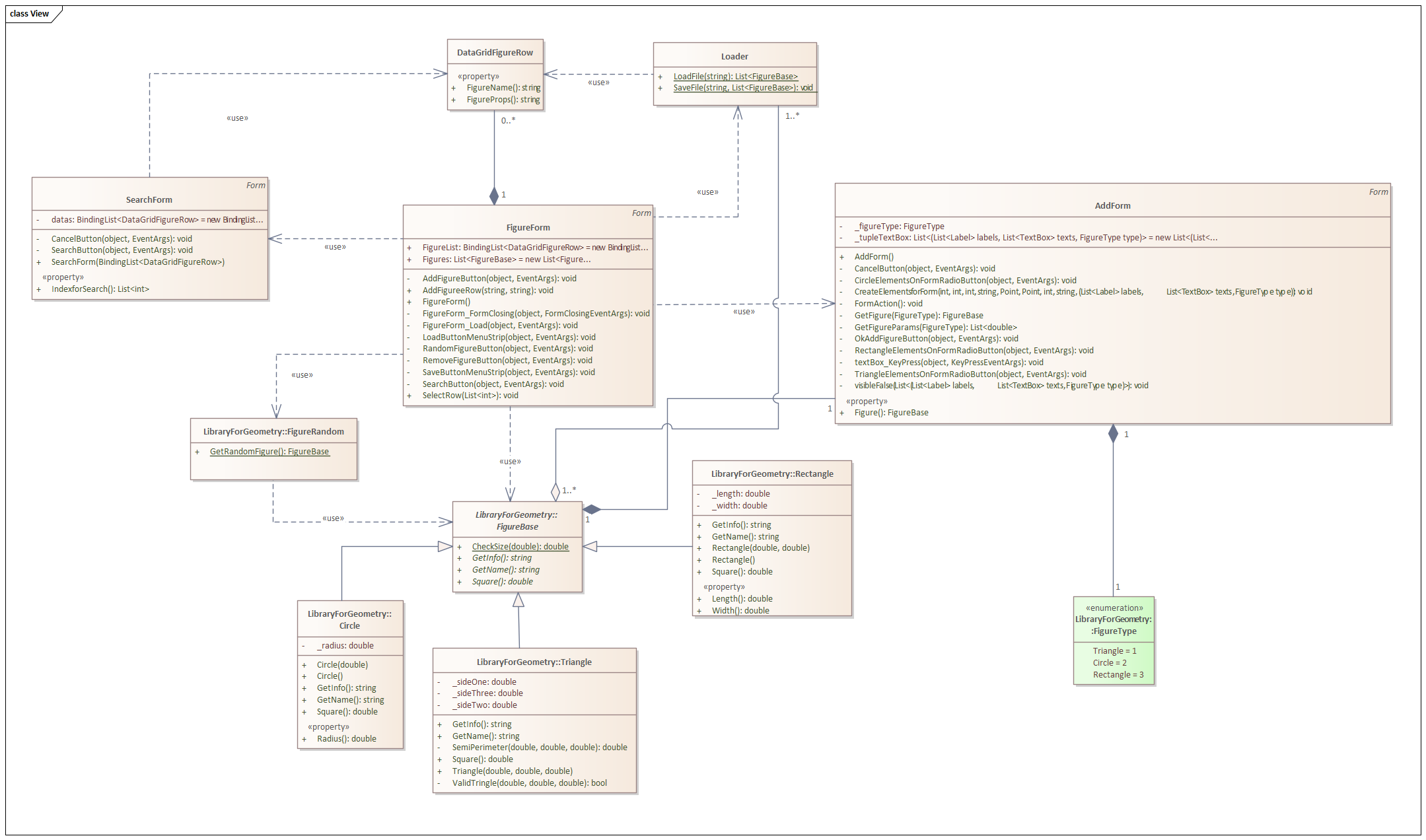


Рисунок 2 – UML диаграмма классов

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

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

Таблица 1 – Описание класса FigureBase

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс FigureBase – абстрактная плоская геометрическая фигура | | |
| Методы | | |
| + Square() | double | Виртуальный метод, для расчета площади абстрактной геометрической фигуры. |
| + CheckSize(double) | double | Метод для проверки числовых значения полей на принадлежность к положительным числам.  Double – сторона фигуры, для которой проводится проверка. |
| + GetInfo() | string | Виртуальный метод, для формирования строки с информацией о фигуре. |
| + GetName | string | Виртуальный метод, для формирования строки с информацией о типе геометрической фигуры |

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

Таблица 2 – Описание класса Circle

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс Circle – окружность | | |
| Поля | | |
| – \_radius | double | Радиус |
| Свойства | | |
| + Radius | double | Радиус |
| Методы | | |
| + Square() | double | Перегруженный метод базового класса, для вычисления площади окружности |

Продолжение таблицы 2

|  |  |  |
| --- | --- | --- |
| # Circle(double) |  | Конструктор для создания новой окружности с помощью радиуса.  Double – радиус окружности. |
| # Circle() |  | Пустой конструктор для создания объекта |
| + GetInfo() | string | Перегруженный метод базового класса для формирования строки с информацией об окружности, а именно: радиус, площадь. |
| + GetName() | string | Перегруженный метод базового класса для формирования строки с информацией о типе геометрической фигуры. Формирует строку: «Circle» |

Таблица 3 – Описание класса Triangle

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс Triangle – треугольник | | |
| Поля | | |
| – \_sideOne | double | Первая сторона треугольника |
| – \_sideTwo | double | Вторая сторона треугольника |
| – \_sideThree | double | Третья сторона треугольника |
| Методы | | |
| – SemiPerimeter(double, double, double) | double | Метод для вычисления полупериметра треугольника.  Double – первая сторона треугольника.  Double – вторая сторона треугольника.  Double – третья сторона треугольника. |
| + Square() | double | Перегруженный метод базового класса, для вычисления площади треугольника |
| # Trangle(double, double, double) |  | Конструктор для создания треугольника с помощью всех трех сторон после проверки возможности существования треугольника.  Double – первая сторона треугольника.  Double – вторая сторона треугольника.  Double – третья сторона треугольника. |
| – ValidTringle(double, double, double) | bool | Метод для проверки существования треугольника с заданными сторонами.  Double – первая сторона треугольника.  Double – вторая сторона треугольника.  Double – третья сторона треугольника. |
| + GetInfo() | string | Перегруженный метод базового класса для формирования строки с информацией о треугольнике, а именно: три стороны и площадь. |
| + GetName() | string | Перегруженный метод базового класса для формирования строки с информацией о типе геометрической фигуры. Формирует строку: «Triangle» |

Таблица 4 – Описание класса Rectangle

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс Rectangle – прямоугольник | | |
| Поля | | |
| – \_length | double | Длина прямоугольника |
| – \_width | double | Ширина прямоугольника |
| Свойства | | |
| + Length | double | Длина прямоугольника |
| + Width | double | Ширина прямоугольника |
| Методы | | |
| + Square() | double | Перегруженный метод базового класса, для вычисления площади прямоугольника |
| # Rectangle(double, double) |  | Конструктор для создания прямоугольника с помощью задания длины и ширины.  Double – длина.  Double – ширина. |
| # Rectangle() |  | Пустой конструктор для создания объекта |
| + GetInfo() | string | Перегруженный метод базового класса для формирования строки с информацией о прямоугольнике, а именно: две стороны и площадь. |
| + GetName() | string | Перегруженный метод базового класса для формирования строки с информацией о типе геометрической фигуры. Формирует строку: «Rectangle» |

**1.4 Дерево ветвлений Git**

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

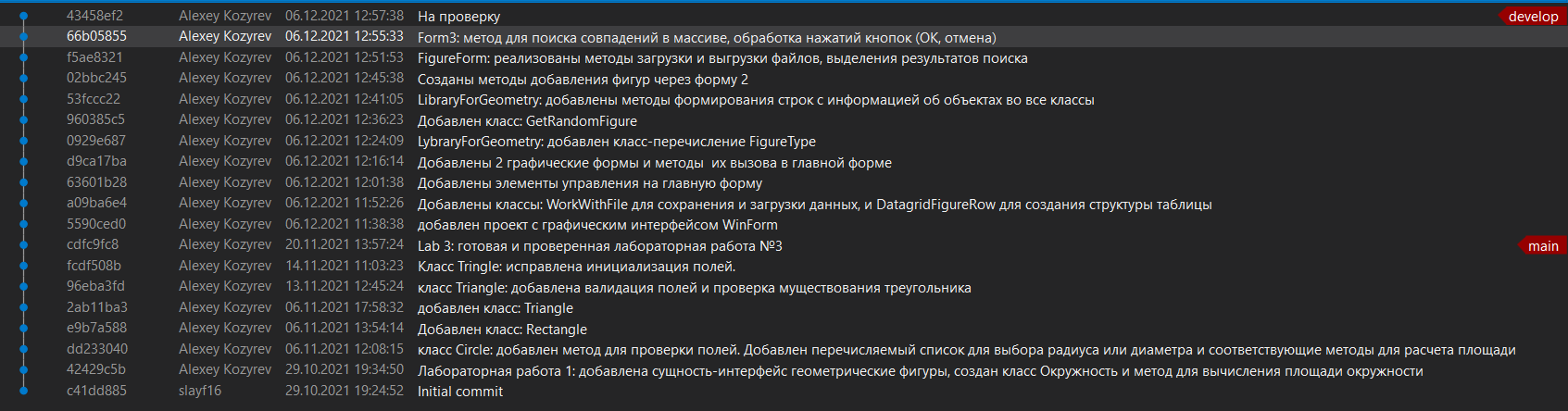


Рисунок 3 – Дерево ветвлений Git

**1.5 Тестирование программы**

Далее приводится процесс функционального тестирования программы.

Графический интерфейс пользователя представлен на рисунке 4.

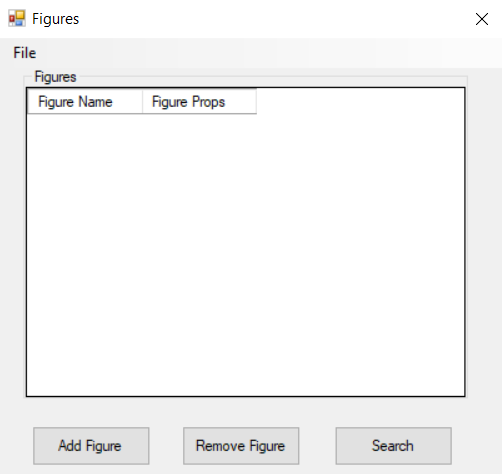


Рисунок 4 – Графический интерфейс пользователя

**1.5.1 Тестовый случай «AddFigure»**

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

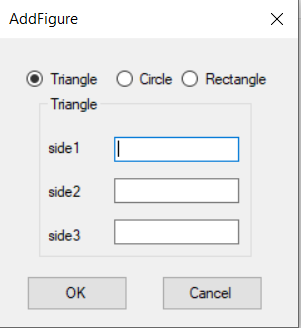


Рисунок 5 – Форма для добавления элемента «Треугольник»

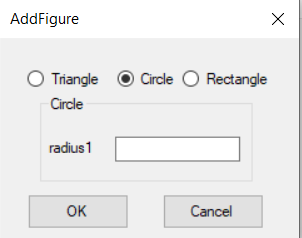


Рисунок 6 – Форма для добавления элемента «Окружность»

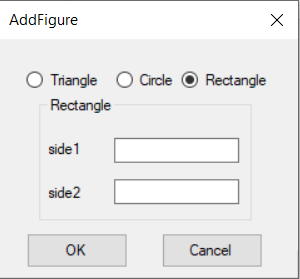


Рисунок 7 – Форма для добавления элемента «Прямоугольник»

После ввода данных необходимо нажать кнопку «ОК», элемент появится в таблице главной формы (рисунки 8 и 9).

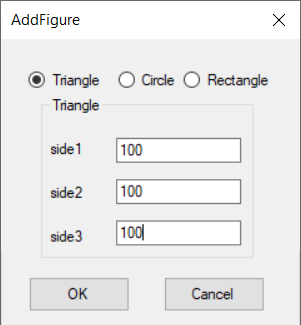


Рисунок 8 – Заполнение полей

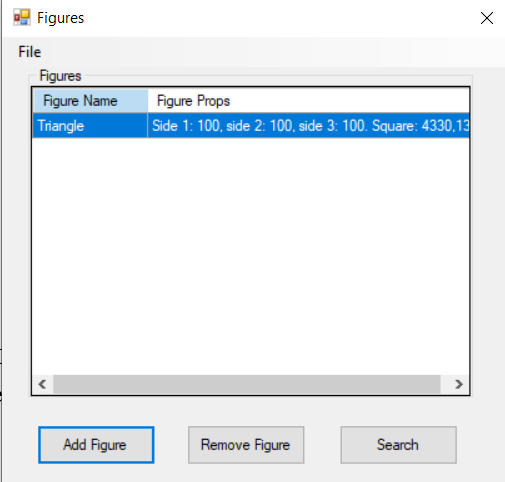


Рисунок 9 – Успешное добавление нового элемента

Система не позволяет ввести пользователю ввести некорректные данные, в соответствующие элементы ввода можно ввести только положительные числа.

**1.5.2 Тестовый случай «Remove»**

Для удаления элемента необходимо выбрать его в таблице и нажать на кнопку «Remove Figure» (рисунки 10 и 11).

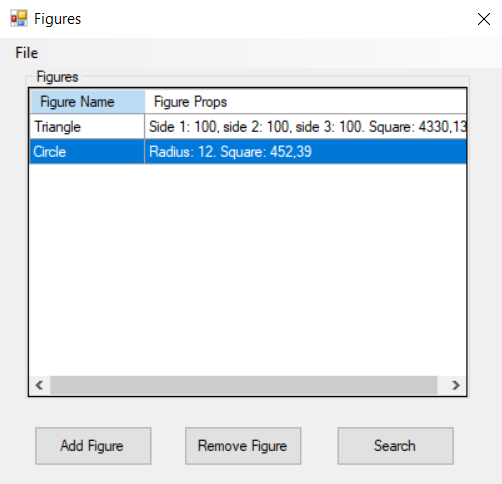


Рисунок 10 – Выбор элемента в таблице

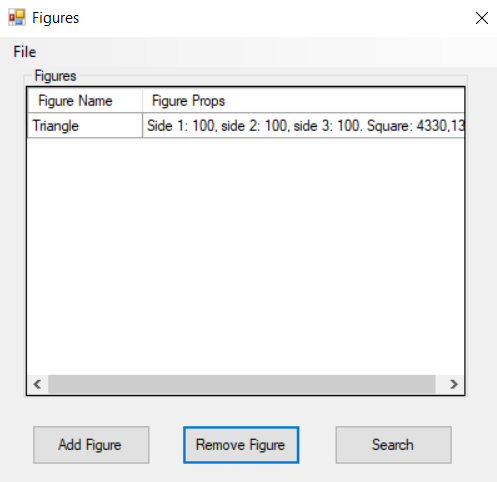


Рисунок 11 – Результат нажатия кнопки «Удалить элемент»

**1.5.3 Тестовый случай «Найти элемент»**

Для поиска элементов в списке необходимо нажать кнопку «Найти элемент» (рисунок 12). Откроется соответствующая форма для поиска (рисунок 13).

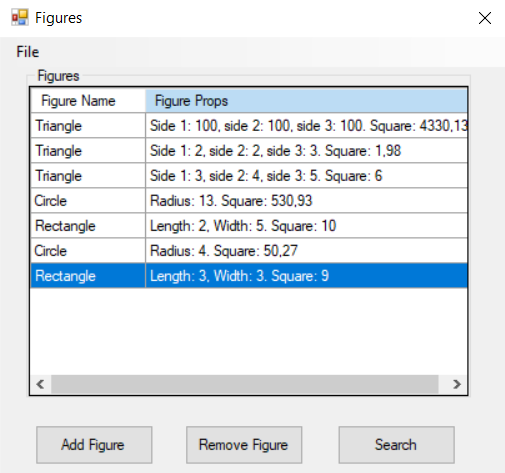


Рисунок 12 – Вызов формы для поиска элементов

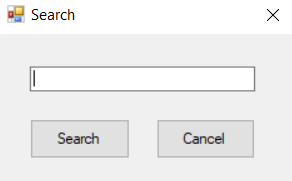


Рисунок 13 – Форма для поиска

Далее пользователь вводит строку, которую хочет найти в таблице (рисунок 14):

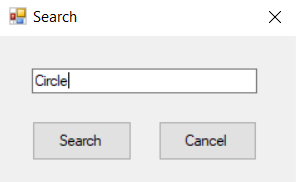


Рисунок 14 – Поиск элемента по названию

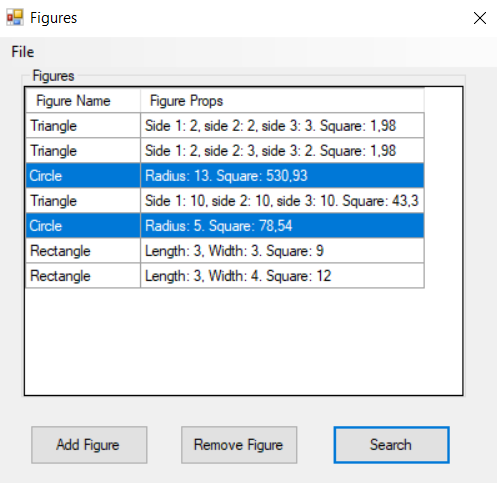


Рисунок 15 – Результат поиска элемента

Все строки, в которых найдено совпадения, выделяются. Также, после выполнения поиска происходит автоматическое закрытие формы поиска.

**1.5.4 Тестовый случай «Сохранить данные»**

Для сохранения данных в таблице необходимо в меню нажать на соответствующую кнопку (рисунок 16). Откроется системный диалог сохранения файла, где пользователь выбирает директорию и указывает имя файла (рисунок 17).

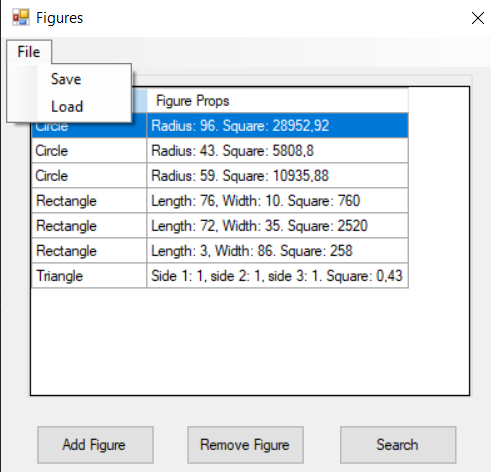


Рисунок 16 – Сохранение данных

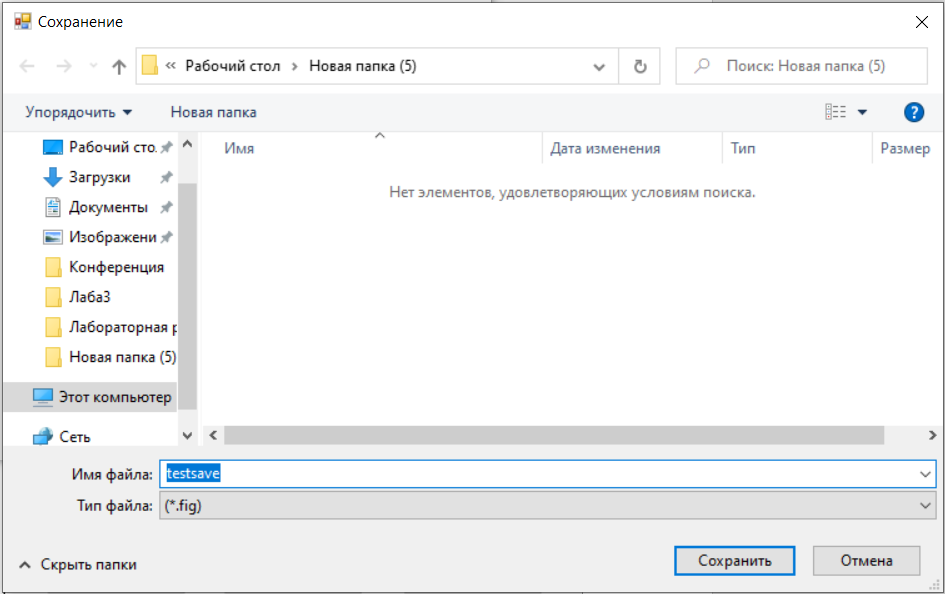


Рисунок 17 – Сохранение файла

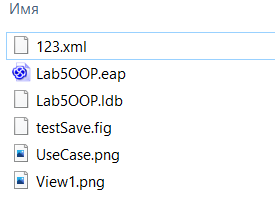


Рисунок 18 – Результат сохранения файла

В случае, если таблица пуста, сохранение не производится (рисунок 19).

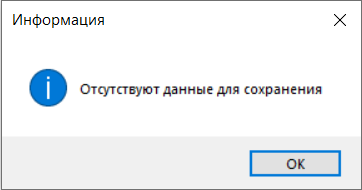


Рисунок 19 – Результат нажатия на кнопку «Сохранить данные» при пустой таблице

**1.5.5 Тестовый случай «Загрузить данные»**

Для загрузки данных в таблицу необходимо нажать на соответствующую кнопку (рисунок 20).

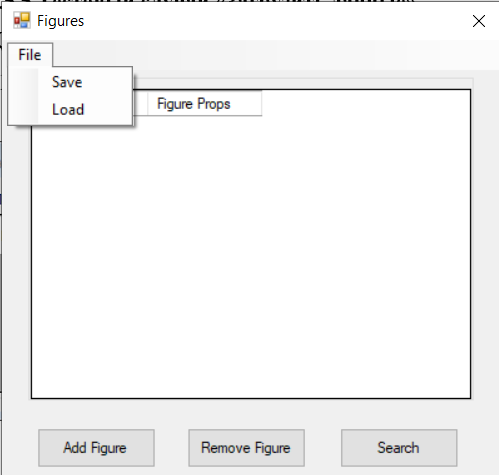


Рисунок 20 – Загрузить данные в таблицу

Далее откроется системный диалог загрузки файла (рисунок 21). Результат на рисунке 22.

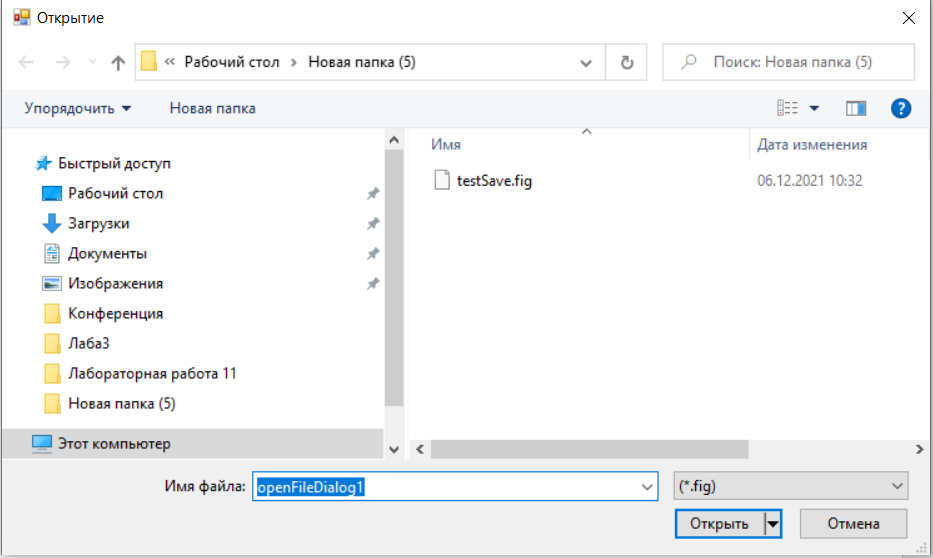


Рисунок 21 – Выбор файла для загрузки

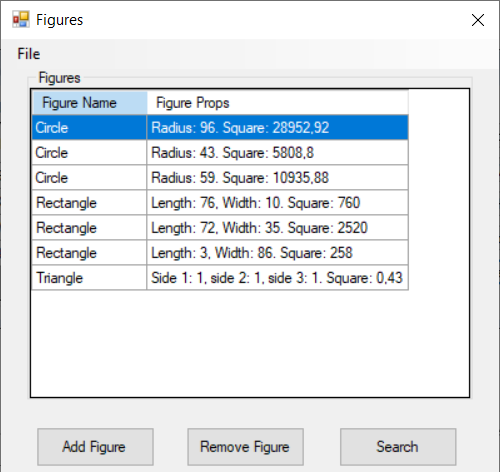


Рисунок 22 – Результат загрузки данных

В случае, если файл поврежден, то появится следующее сообщение (рисунок 23).

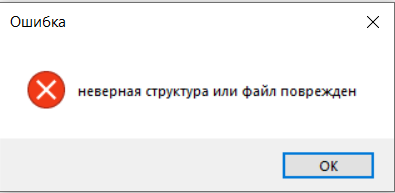


Рисунок 23 – Ошибка при попытке загрузки поврежденного файла

**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 площади окружности, треугольника и прямоугольника

Разработчик: студент гр. О-5КМ91 НИ ТПУ Козырев А. М.

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

Томск, 2021

**1 Общие сведения**

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

Полное наименование: «Программа для расчёта площади окружности, треугольника и прямоугольника»

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-5КМ91 НИ ТПУ Козырев А. М.

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

Начало работ: 01 сентября 2021 г.

Окончание работ: 05 декабря 2021 г.

**2 Назначение и цели создания системы**

**2.1 Назначение системы**

Система предназначена для расчёта площади основных фигур планиметрии, а именно: окружности, прямоугольника и треугольника.

**2.2 Цели создания системы**

Система создаётся для автоматизации расчетов площади фигу при решении планиметрических задач.

**3 Характеристика объектов автоматизации**

Расчет площадей фигур является одной из основных задач планиметрии. Поскольку, данный расчет выполняется математиками практически в любой планиметрической задаче целесообразно автоматизировать данный процесс.

**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 Функциональные требования**

**F01.** Система должна рассчитывать площади следующих фигур:

* окружность;
* треугольник;
* прямоугольник.

**F01.01.** Площадь окружности должна определяться следующим образом:



**F01.02.** Площадь треугольника должна определяться по формуле:



где *a, b, c* – стороны треугольника;

*p* – полупериметр треугольника, определяемый по формуле:



**F01.03.** Площадь прямоугольника должна определяться по формуле:



где *a, b* – стороны прямоугольника

**F02.** В системе должен быть реализован список фигур с их параметрами

**F02.01.** Список должен иметь следующую информацию:

* наименование фигуры;
* параметры фигуры: площадь и стороны, определяющие фигуру.

**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.

**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 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.** Процессор – не менее 1 ГГц.

**C04.** ОЗУ – не менее 2 ГБ.