|  |
| --- |
| **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-5КМ11 | |  | \_\_\_\_\_\_\_\_\_\_ | |  | Сухов Е.Р. |
|  | |  |  | |  |  |
|  | |  |  |  | | |
|  | |  |  | |  |  |
| Отчёт принял | доцент, к.т.н. |  | \_\_\_\_\_\_\_\_\_\_ | |  | Калентьев А. А. |
|  |  |  |  | |  |  |
|  |  |  |  | \_\_\_ \_\_\_\_\_\_ | | |

Томск 2023

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc136563218)

[1 Основная часть 4](#_Toc136563219)

[1.1 UML диаграмма вариантов использования 4](#_Toc136563220)

[1.2 UML диаграмма классов 5](#_Toc136563221)

[1.3 Описание классов, образующих связь типа «общее-частное» 7](#_Toc136563222)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 9](#_Toc136563223)

**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.

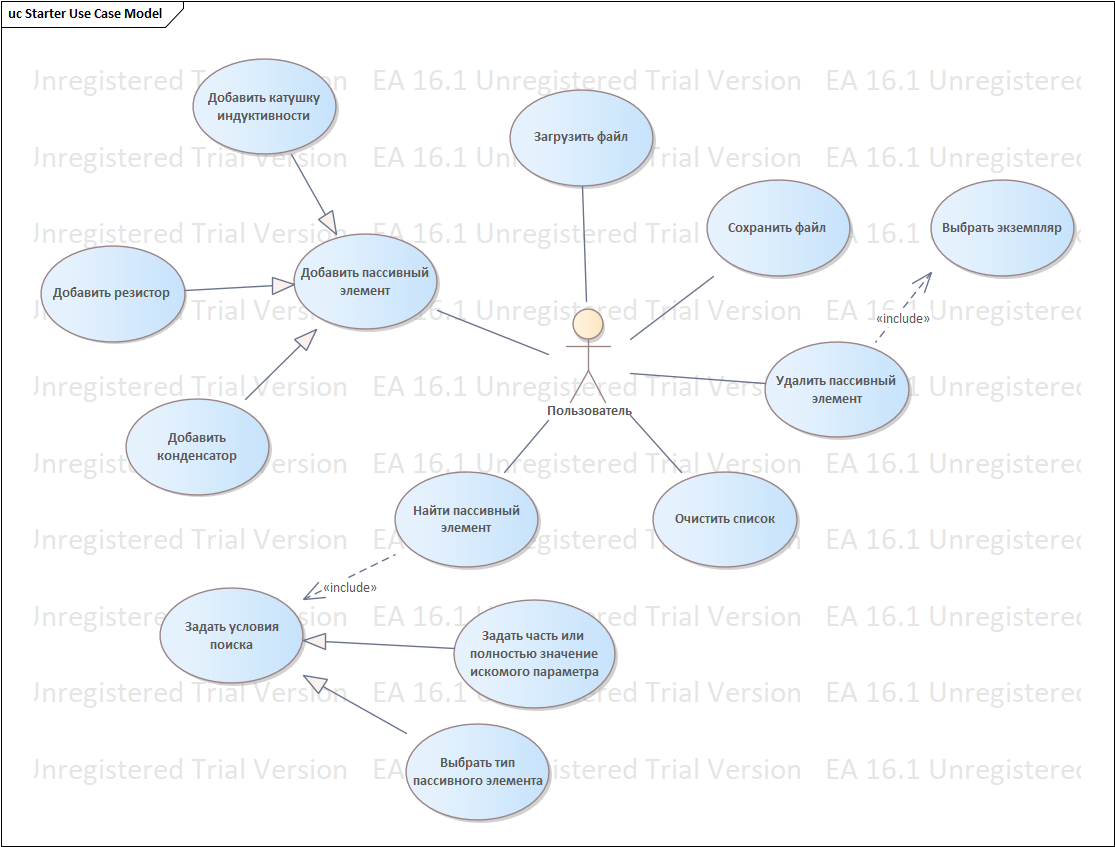


Рисунок 1 – Диаграмма вариантов использования

**1.2 UML диаграмма классов**

Диаграмма классов (англ. class diagram) — структурная диаграмма языка моделирования UML, демонстрирующая общую структуру иерархии классов системы, их коопераций, атрибутов (полей), методов, интерфейсов и взаимосвязей (отношений) 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.

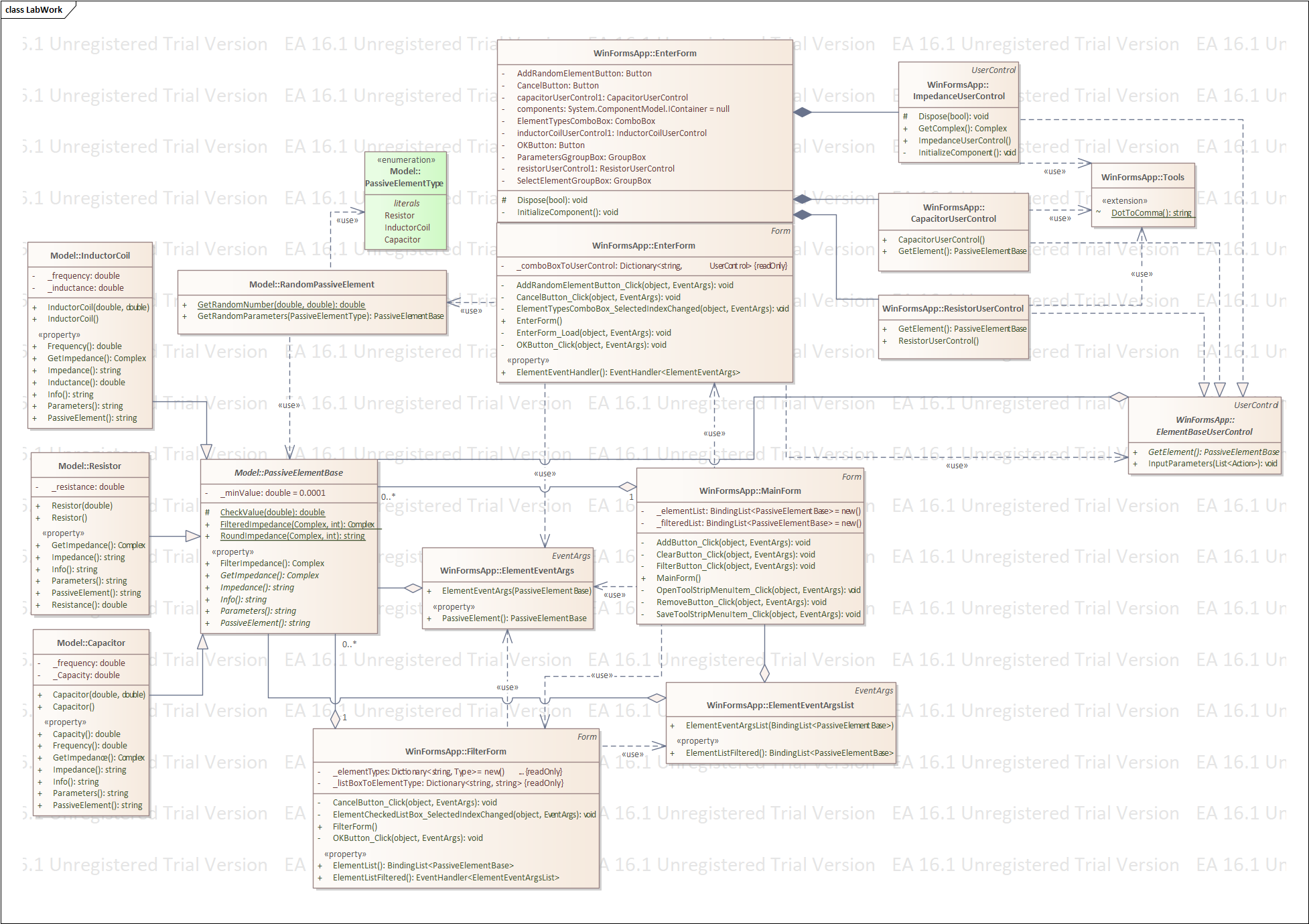


Рисунок 2 – UML диаграмма классов

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

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

Таблица 1 – Описание класса *PassiveElementBase*

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс *PassiveElementBase* – абстрактный базовый класс для пассивных элементов | | |
| Поля | | |
| -\_minValue | double | Минимальное значение |
| Свойства | | |
| + FilterImpedance | Complex | Комплексное сопротивление элемента для фильтра |
| + GetImpedance | Complex | Рассчитанное комплексное сопротивление элемента.  Абстрактное свойство, переопределяется в производных классах. |
| + Impedance | string | Комплексное сопротивление элемента.  Абстрактное свойство, переопределяется в производных классах. |
| + Info | string | Информация об элементе.  Абстрактное свойство, переопределяется в производных классах. |
| + Parameters | string | Параметры элемента.  Абстрактное свойство, переопределяется в производных классах. |
| + PassiveElement | string | Название элемента.  Абстрактное свойство, переопределяется в производных классах. |
| Методы | | |
| # CheckValue(double) | double | Метод проверки параметров  double – Вводимый параметр |
| + FilteredImpedance(Complex, int) | Complex | Метод округления для фильтра  Complex – комплексное сопротивление элемента,  int – степень округления |
| - RoundImpedance (Complex, int) | string | Метод округления для DataGrid  Complex – комплексное сопротивление элемента,  int – степень округления |

В таблицах 2–4 приведены описания классов *Resistor*, *Capacitor*, *InductorCoil*, которые наследуются от *PassiveElementBase*.

Таблица 2 – Описание класса *Resistor*

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс *Resistor* – Класс резистора | | |
| Поля | | |
| –\_resistance | double | Сопротивление резистора |
| Свойства | | |
| + GetImpedance | Complex | Рассчитанное комплексное сопротивление элемента. |
| + Impedance | string | Комплексное сопротивление элемента. |
| + Info | string | Информация об элементе. |
| + Parameters | string | Параметры элемента. |
| + PassiveElement | string | Название элемента. |
| + Resistance | double | Свойство для проверки вводимого сопротивления. |
| Методы | | |
| + Resistor(double) |  | Конструктор для создания нового резистора на основе его сопротивления  double - Сопротивление резистора |
| + Resistor() |  | Пустой конструктор для создания нового резистора |

Таблица 3 – Описание класса *Capacitor*

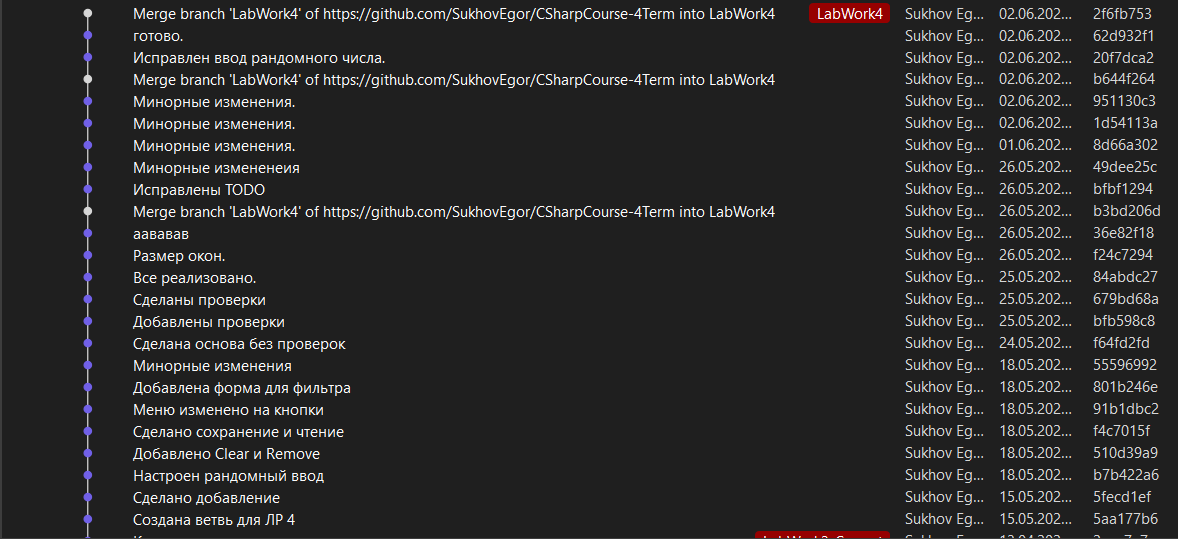
|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс *Capacitor* – Класс конденсатора | | |
| Поля | | |
| –\_capacity | double | Ёмкость конденсатора |
| –\_frequency | double | Частота конденсатора |
| Свойства | | |
| + GetImpedance | Complex | Рассчитанное комплексное сопротивление элемента. |
| + Impedance | string | Комплексное сопротивление элемента. |
| + Info | string | Информация об элементе. |
| + Parameters | string | Параметры элемента. |
| + PassiveElement | string | Название элемента. |
| + Сapacity | double | Свойство для проверки вводимой ёмкости. |
| + Frequency | double | Свойство для проверки вводимой частоты. |
| Методы | | |
| + Capacitor(double, double) |  | Конструктор для создания нового конденсатора на основе его ёмкости и частоты.  double – ёмкость конденсатора,  double – частота конденсатора |
| + Capacitor() |  | Пустой конструктор для создания нового конденсатора |

Таблица 4 – Описание класса *InductorCoil*

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс *InductorCoil* – Класс конденсатора | | |
| Поля | | |
| –\_ inductance | double | Индуктивность конденсатора |
| –\_frequency | double | Частота конденсатора |
| Свойства | | |
| + GetImpedance | Complex | Рассчитанное комплексное сопротивление элемента. |
| + Impedance | string | Комплексное сопротивление элемента. |
| + Info | string | Информация об элементе. |
| + Parameters | string | Параметры элемента. |
| + PassiveElement | string | Название элемента. |
| + Inductance | double | Свойство для проверки вводимой индуктивности. |
| + Frequency | double | Свойство для проверки вводимой частоты. |
| Методы | | |
| + InductorCoil(double, double) |  | Конструктор для создания новой катушки индуктивности на основе ее индуктивности и частоты.  double – индуктивность катушки индуктивности,  double – частота катушки индуктивности |
| + InductorCoil() |  | Пустой конструктор для создания новой катушки индуктивности |

**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.

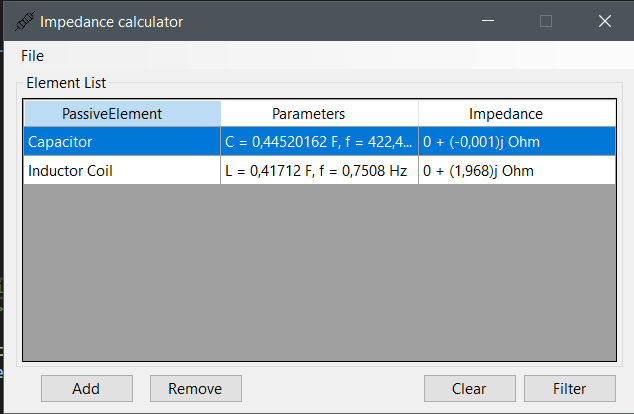


Рисунок 4 – Графический интерфейс пользователя

**1.5.1 Тестовый случай «Add Element»**

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

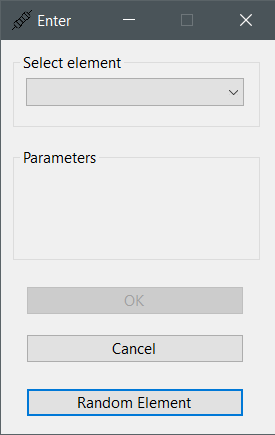


Рисунок 5 – Форма для добавления элемента

Параметры элементы (сопротивление, индуктивность частота, ёмкость) можно указать в выпадающем меню.

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

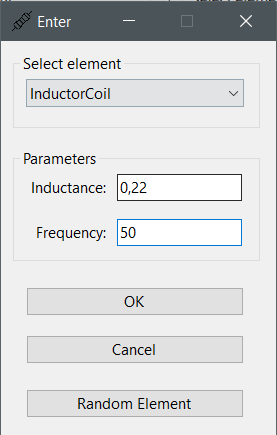
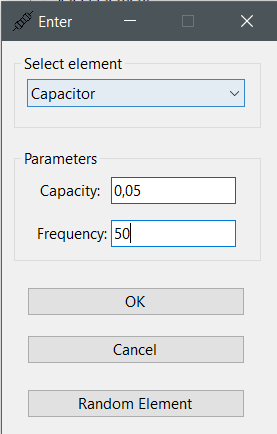
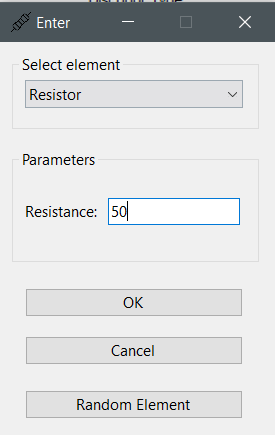


Рисунок 6 – Заполнение полей

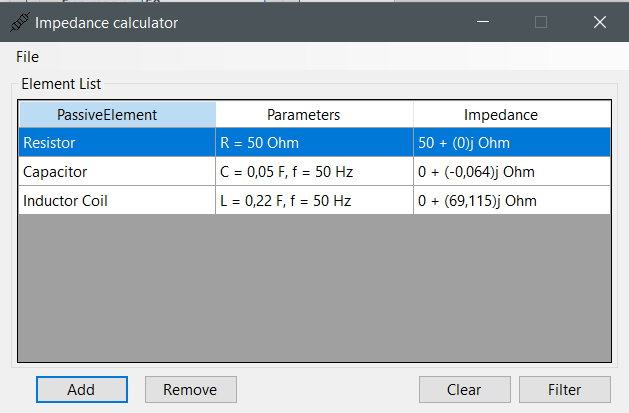


Рисунок 7 – Успешное добавление нового элемента

В программе предусмотрена система обработки некорректного ввода данных пользователем. Например, при попытке ввести число вне заранее определенного диапазона, появится соответствующее сообщение об ошибке (рисунки 8-9).

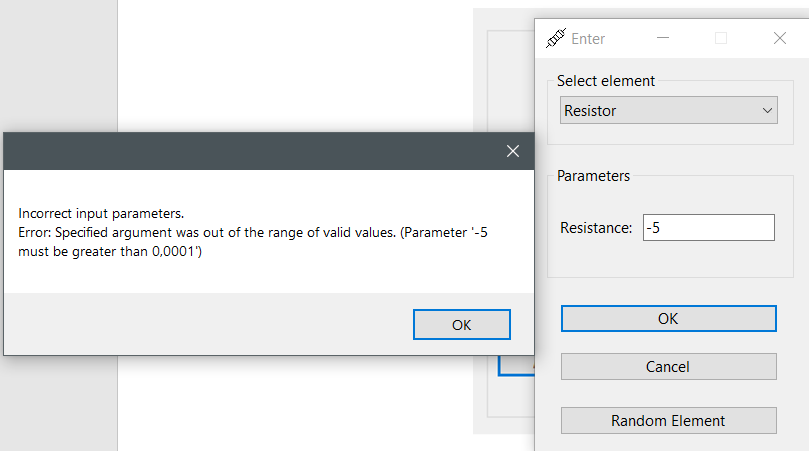


Рисунок 8 – Некорректный ввод (отрицательное число)

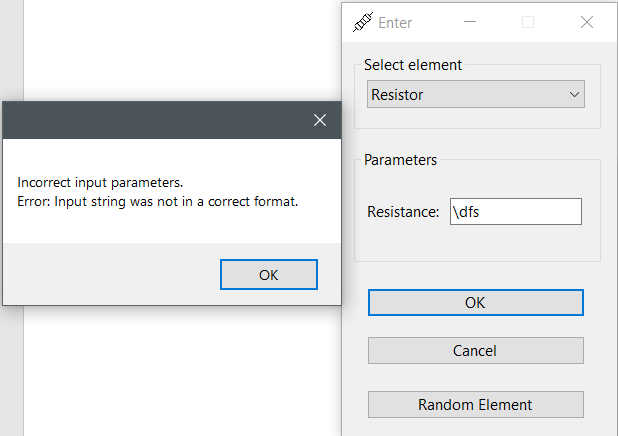


Рисунок 9 – Некорректный ввод (ввод недопустимых символов)

Подобная обработка предусмотрена для всех параметров элементов.

**1.5.2 Тестовый случай «Удалить элемент»**

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

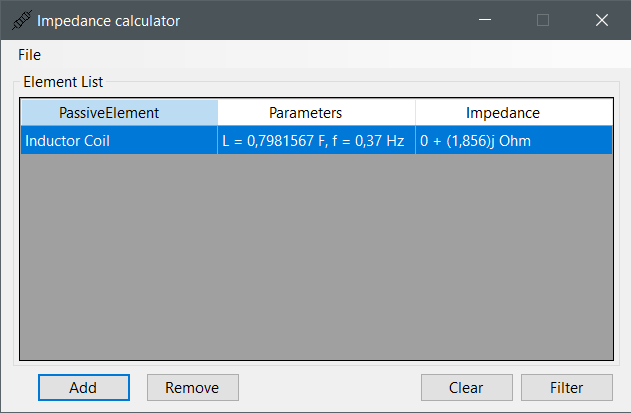


Рисунок 10 – Выбор элемента в таблице

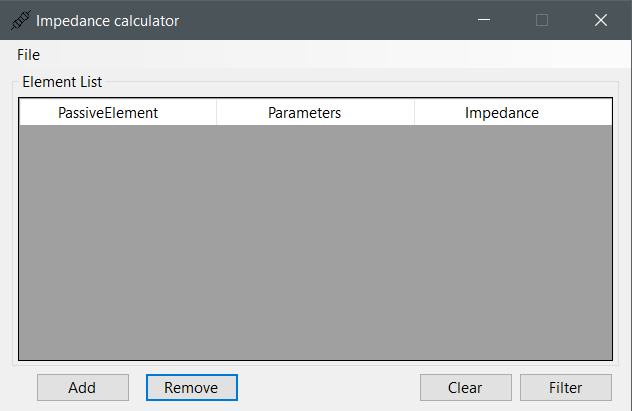


Рисунок 11 – Результат нажатия кнопки «Remove»

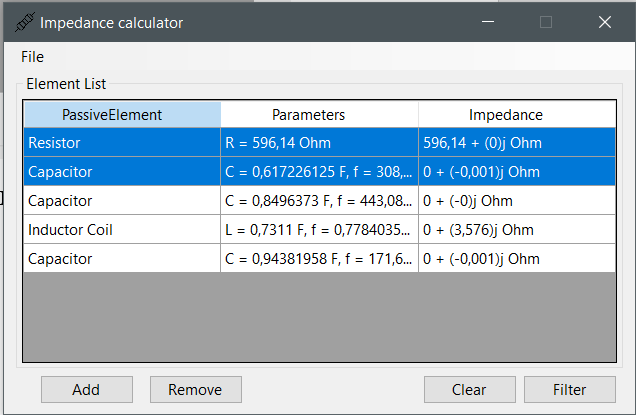


Рисунок 12 – Выбор нескольких элементов для удаления

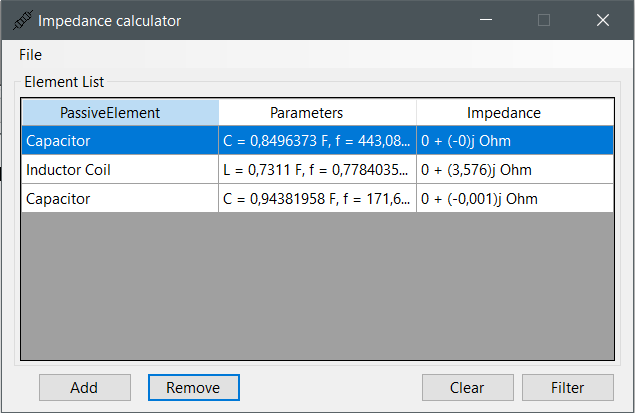


Рисунок 13 – Результат удаления выбранных элементов

**1.5.3 Тестовый случай «Search»**

Для поиска элементов предусмотрена форма поиска (рисунок 14)

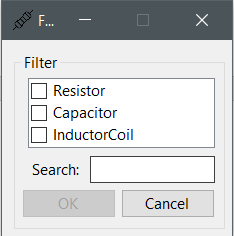


Рисунок 14 – Настройка параметров для поиска элементов

Пользователь выбирает параметры, по которым требуется найти элемент, вводит значения этих параметров и нажимает кнопку «OK» (рисунки 15 и 16).

Для того, чтобы сбросить фильтр, предусмотрена соответствующая кнопка «Cancel» (рисунок 17).

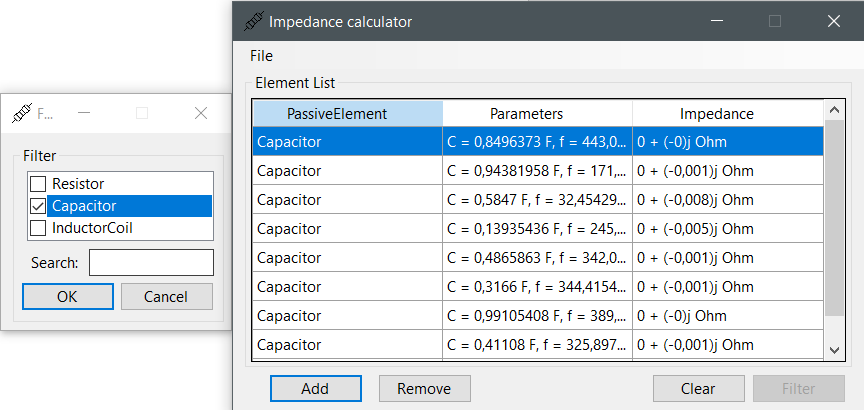


Рисунок 15 – Поиск по типу элемента

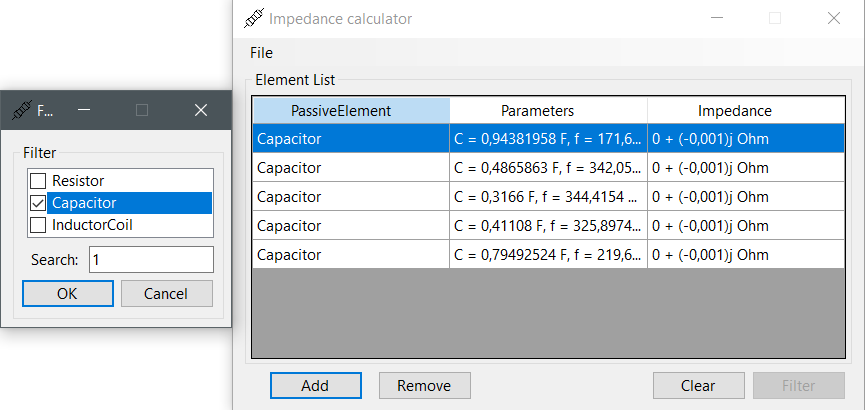


Рисунок 16 – Поиск по типу элемента и значению

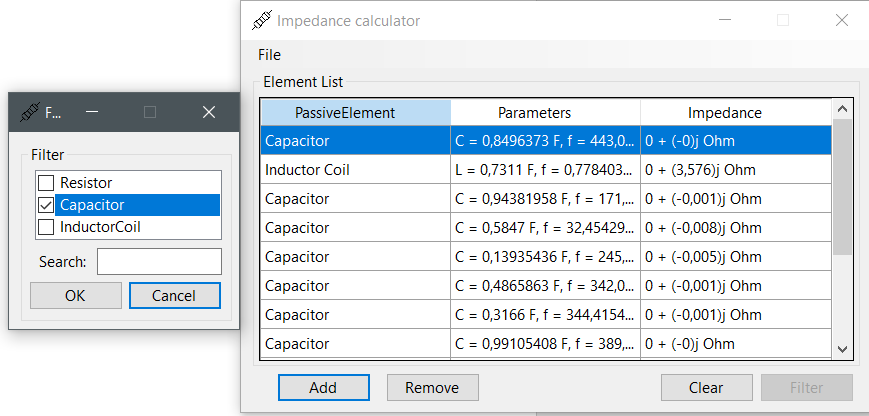


Рисунок 17 – Сброс условий поиска

**1.5.4 Тестовый случай «Save»**

Для сохранения данных в таблице необходимо нажать на кнопку «Save». Откроется системный диалог сохранения файла, где пользователь выбирает директорию и указывает имя файла (рисунок 18).

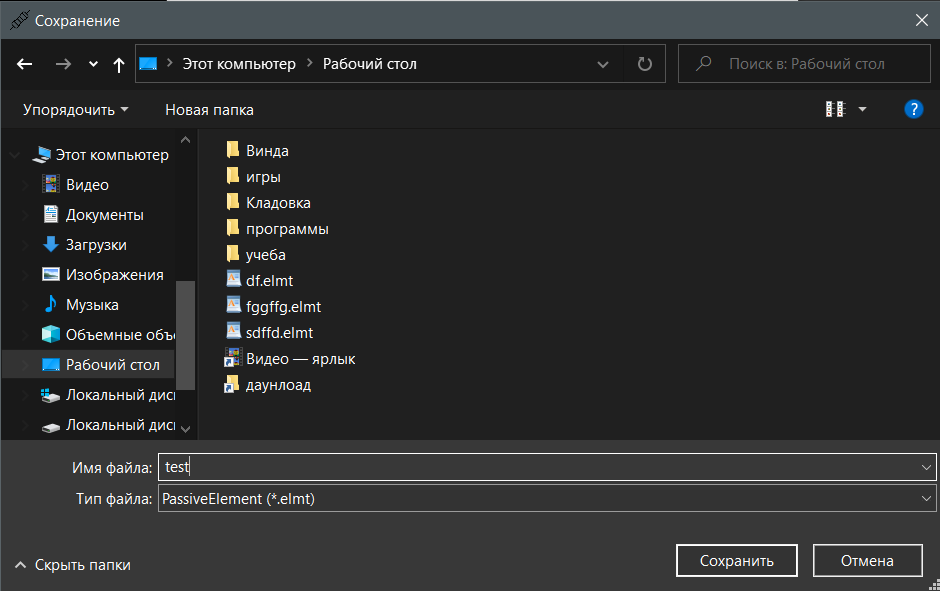


Рисунок 18 – Сохранение файла

**1.5.5 Тестовый случай «Загрузить данные»**

Для загрузки данных в таблицу необходимо нажать на соответствующую кнопку «Load».

Далее откроется системный диалог загрузки файла (рисунок 19).

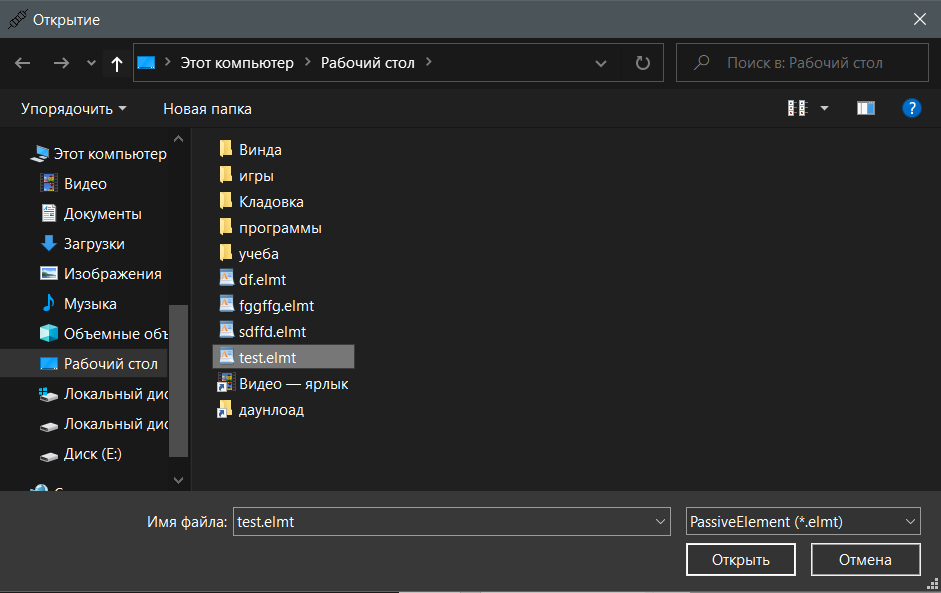


Рисунок 19 – Выбор файла для загрузки

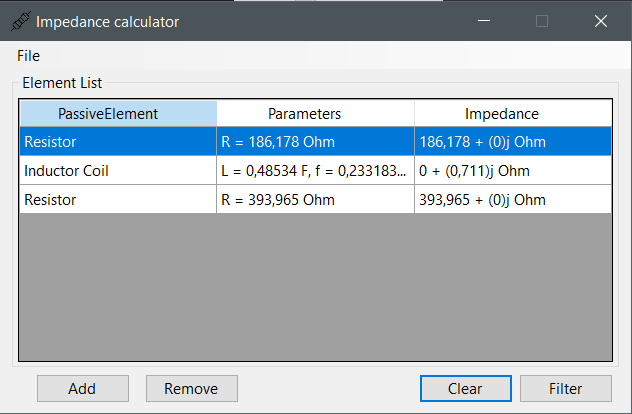


Рисунок 20 – Результат загрузки данных

В случае, если схема файла не соответствует установленному формату, появится соответствующее сообщение (рисунок 21).

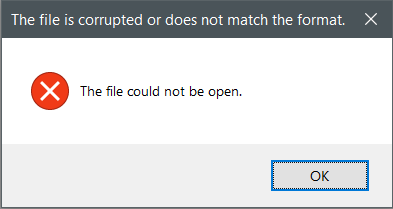


Рисунок 21 – Загрузка повреждённого файла

**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 просмотра списка работников

Разработчик: студент гр. О-5КМ11 Сухов Е.Р.

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

Томск 2023

**1 Общие сведения**

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

Полное наименование: «Программа для расчета комплексного сопротивления пассивных элементов».

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-5КМ11 НИ ТПУ Сухов Е.Р.

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

Начало работ: 15 апреля 2023 г.

Окончание работ: 2 июня 2023 г.

**2 Назначение и цели создания системы**

**2.1 Назначение системы**

Система предназначена для расчета комплексного сопротивления пассивных элементов, просмотра существующих элементов и их сопротивлений и добавления новых.

**2.2 Цели создания системы**

Система создается в целях автоматизации расчета комплексного сопротивления пассивных элементов и формирования общего списка.

**3 Характеристика объектов автоматизации**

Раньше лаборантам необходимо было вручную проводить расчет комплексного сопротивления пассивных элементов в схемах и запоминать их характеристики. Этот монотонный труд мог приводить к ошибкам, которые приводят к экономическим потерям и коротким замыканиям. С целью ликвидации данных ситуаций разрабатывается данная система.

**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 параметрах элементов электрических схем должны храниться в XML-файле с расширением \*.xml.

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

<?xml version="1.0" encoding="utf-8"?><ArrayOfPassiveElementBase xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"><PassiveElementBase xsi:type="Resistor">

<Resistance>[Resistance]</Resistance>

</PassiveElementBase>

<PassiveElementBase xsi:type="Capacitor">

<Capacity>[Capacity]</Capacity>

<Frequency>[Frequency]</Frequency>

</PassiveElementBase>

<PassiveElementBase xsi:type="InductorCoil">

<Inductance>[Inductance]</Inductance>

<Frequency>[Frequency]</Frequency>

</PassiveElementBase>

</ArrayOfPassiveElementBase>

**4.3 Функциональные требования**

**F01.** В системе должен быть реализован список элементов.

**F01.01.** В зависимости от типа элемента необходимо заполнить следующие параметры:

Резистор:

- Сопротивление

Конденсатор:

- Ёмкость

- Частота

Катушка индуктивности:

- Индуктивность

- Частота

**F02.** В системе должна присутствовать функция добавления элементов в список.

**F03.** В системе должна присутствовать функция удаления элементов из списка.

**F04.** В системе должна присутствовать функция поиска элементов по результатам вычисления указанных параметров в **F01.01**.

**F05.** В системе должна присутствовать функция сохранения списка элементов в файл (**D01**).

**F06.** 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 Home. Работоспособность на других выпусках и версиях не гарантируется.

**C02.** На рабочей станции должен быть установлен .NET Framework версии 4.8

**4.6 Требования к аппаратному обеспечению**

**C03.** Процессор: процессор с частотой 1 гигагерц (ГГц) или выше

**C04.** ОЗУ: 1 ГБ для 32-разрядных систем или 2 ГБ для 64-разрядных систем

**C05.**Место на жестком диске: 16 ГБ для 32-разрядных систем или 32 ГБ для 64-разрядных систем