**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»**

|  |  |  |
| --- | --- | --- |
| Выполнил: | E:\Учеба\3 курс 5 семестр\Документы и всякое\Роспись для документов))).JPG | Соловьев Макар Борисович |
| студент группы:  О-5КМ11 | (подпись, дата) | (Ф.И.О) |

|  |  |  |
| --- | --- | --- |
| Проверил: |  | Калентьев Алексей Анатольевич |
| преподаватель | (подпись, дата) | (Ф.И.О) |

Томск 2023 г.

**Содержание**

[Введение 3](#_Toc137163272)

[1. Основная часть 4](#_Toc137163273)

[1.1. UML диаграмма вариантов использования 4](#_Toc137163274)

[1.2. UML диаграмма классов 5](#_Toc137163275)

[1.3. Описание классов, образующих связь типа «общее-частное» 7](#_Toc137163276)

[1.4. Дерево ветвлений Git 11](#_Toc137163277)

[1.5. Тестирование программы 11](#_Toc137163278)

[1.5.1. Тестовый случай «Add Motion» 12](#_Toc137163279)

[1.5.2. Тестовый случай «Remove Motion» 14](#_Toc137163280)

[1.5.3. Тестовый случай «Filter Motion» 17](#_Toc137163281)

[1.5.4. Тестовый случай «Save Motion» 19](#_Toc137163282)

[1.5.5. Тестовый случай «Load Motion» 19](#_Toc137163283)

[Список используемых источников 21](#_Toc137163284)

[Приложение А 22](#_Toc137163285)

# **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. *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.

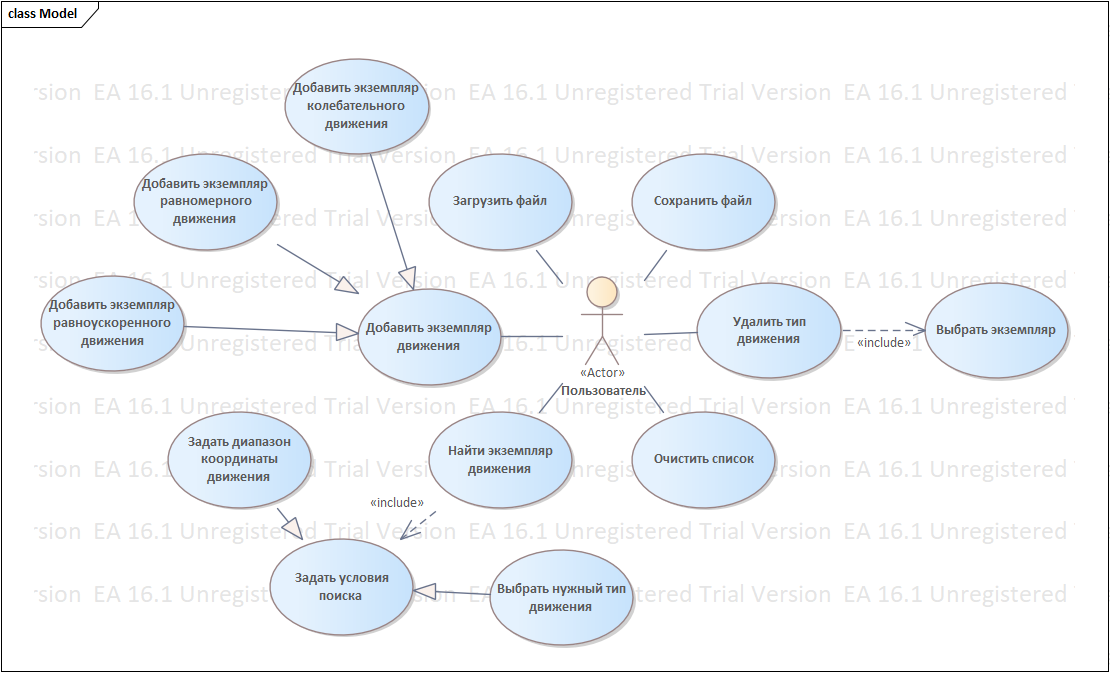


Рисунок 1 – Диаграмма вариантов использования

* 1. *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.

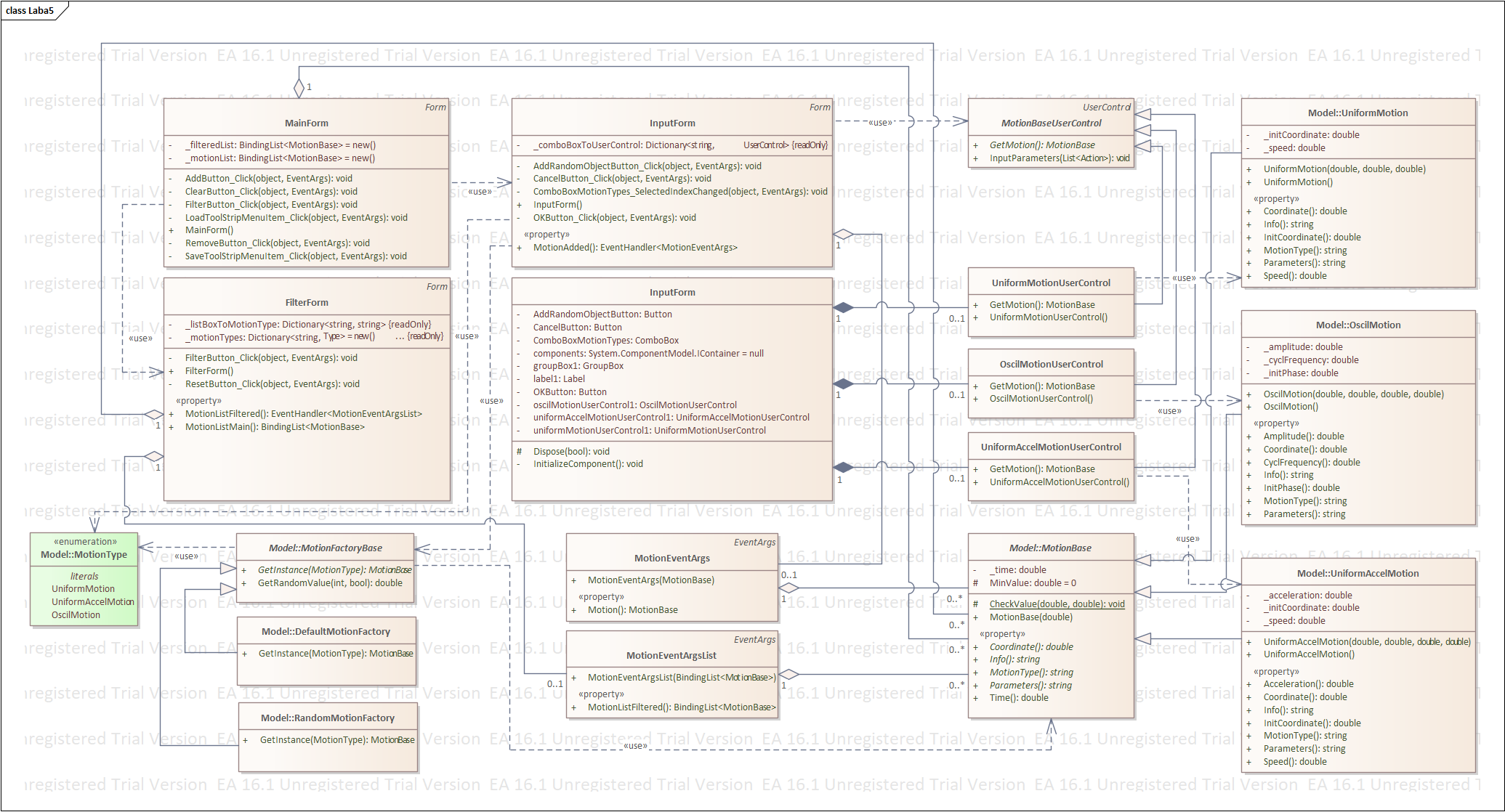


Рисунок 2 – UML диаграмма классов

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

В Таблице 1 приведено описание абстрактного класса MotionBase с его полями, свойствами и методами.

Таблица 1 – Описание класса MotionBase

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс MotionBase – абстрактный базовый класс для типов движения. | | |
| Поля | | |
| - \_time | double | Время движения. |
| # MinValue | double = 0 | Минимальное значение. |
| Свойства | | |
| + Coordinate | double | Рассчитанная координата движения.  Абстрактное свойство, переопределяется в производных классах. |
| + Info | string | Информация по элементу движения.  Абстрактное свойство, переопределяется в производных классах. |
| + MotionType | string | Информация по типу движения экземпляра движения для вывода в DataGrid.  Абстрактное свойство, переопределяется в производных классах. |
| + Parameters | string | Информация по параметрам экземпляра движения для вывода в DataGrid.  Абстрактное свойство, переопределяется в производных классах. |
| + Time | double | Свойство для проверки вводимого значения времени. |
| Методы | | |
| # CheckValue(double, double) | void | Метод проверки параметров.  double – вводимый параметр.  double – величина сравнения. |

В Таблицах 2 – 4 приведены описания классов UniformMotion, UniformAccelMotion, OscilMotion, которые наследуются от MotionBase.

Таблица 2 – Описание класса UniformMotion

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс UniformMotion – класс для создания экземпляров равномерного движения. | | |
| Поля | | |
| - \_initCoordinate | double | Начальная координата. |
| - \_speed | double | Скорость. |
| Свойства | | |
| + Coordinate | double | Рассчитанная координата движения. |
| + Info | string | Информация об экземпляре движения. |
| + InitCoordinate | double | Свойство для проверки вводимой начальной координаты. |
| + MotionType | string | Информация о типе экземпляра движения.  Служит для вывода в DataGrid. |
| + Parameters | string | Информация о параметрах экземпляра движения.  Служит для вывода в DataGrid. |
| + Speed | double | Свойство для проверки вводимой скорости. |
| Методы | | |
| UniformMotion(double, double, double) | UniformMotion | Конструктор для создания нового экземпляра равномерного движения.  double – вводимая начальная координата.  double – вводимая скорость.  double – вводимое время. |
| UniformMotion() | UniformMotion | Пустой конструктор для создания экземпляров равномерного движения. |

Таблица 3 – Описание класса UniformAccelMotion

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс UniformAccelMotion – класс для создания экземпляров равноускоренного движения. | | |
| Поля | | |
| - \_acceleration | double | Ускорение. |
| - \_initCoordinate | double | Начальная координата. |
| - \_speed | double | Скорость. |
| Свойства | | |
| + Acceleration | double | Свойство для проверки вводимого ускорения. |
| + Coordinate | double | Рассчитанная координата движения. |
| + Info | string | Информация об экземпляре движения. |
| + InitCoordinate | double | Свойство для проверки вводимой начальной координаты. |
| + MotionType | string | Информация о типе экземпляра движения.  Служит для вывода в DataGrid. |
| + Parameters | string | Информация о параметрах экземпляра движения.  Служит для вывода в DataGrid. |
| + Speed | double | Свойство для проверки вводимой скорости. |
| Методы | | |
| UniformAccelMotion(double, double, double, double) | UniformAccelMotion | Конструктор для создания нового экземпляра равноускоренного движения.  double – вводимое ускорение.  double – вводимая начальная координата.  double – вводимая скорость.  double – вводимое время. |
| UniformAccelMotion() | UniformAccelMotion | Пустой конструктор для создания экземпляров равноускоренного движения. |

Таблица 3 – Описание класса OscilMotion

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс OscilMotion – класс для создания экземпляров колебательного движения. | | |
| Поля | | |
| - \_amplitude | double | Амплитуда. |
| - \_cyclFrequency | double | Циклическая частота. |
| - \_initPhase | double | Начальная фаза. |
| Свойства | | |
| + Amplitude | double | Свойство для проверки вводимой амплитуды. |
| + Coordinate | double | Рассчитанная координата движения. |
| + Info | string | Информация об экземпляре движения. |
| + CyclFrequency | double | Свойство для проверки вводимой циклической частоты. |
| + MotionType | string | Информация о типе экземпляра движения.  Служит для вывода в DataGrid. |
| + Parameters | string | Информация о параметрах экземпляра движения.  Служит для вывода в DataGrid. |
| + InitPhase | double | Свойство для проверки вводимой начальной фазы. |
| Методы | | |
| OscilMotion(double, double, double, double) | OscilMotion | Конструктор для создания нового экземпляра колебательного движения.  double – вводимая амплитуда.  double – вводимая циклическая частота.  double – вводимая начальная фаза.  double – вводимое время. |
| OscilMotion() | OscilMotion | Пустой конструктор для создания экземпляров колебательного движения. |

* 1. *Дерево ветвлений Git*

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

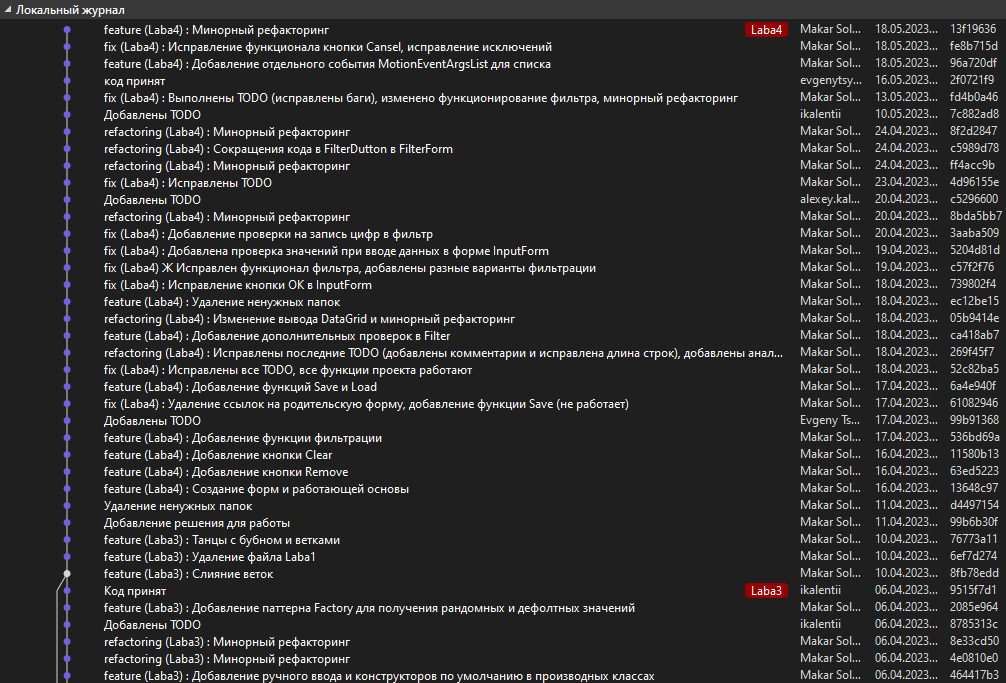


Рисунок 3 – Дерево ветвлений Git

* 1. *Тестирование программы*

Далее приводится процесс функционального тестирования программы.

Графический интерфейс пользователя представлен на Рисунке 4.

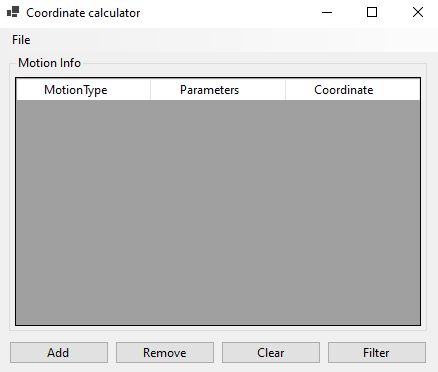


Рисунок 4 – Графический интерфейс пользователя

* + 1. *Тестовый случай «Add Motion»*

Для добавления экземпляра движения необходимо вызвать соответствующую форму путём нажатия кнопки «Add» (Рисунок 5).

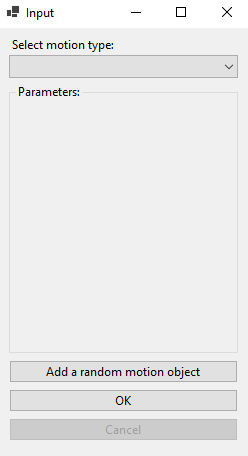


Рисунок 5 – Форма для добавления элемента

Выбрать тип движения можно указать в выпадающем меню.

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

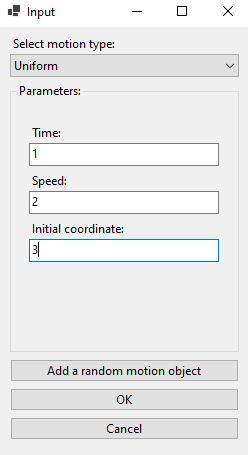
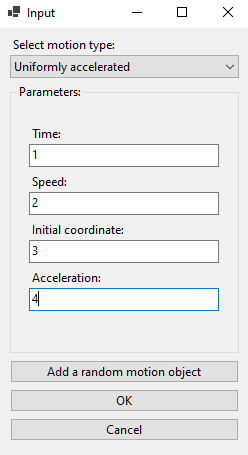
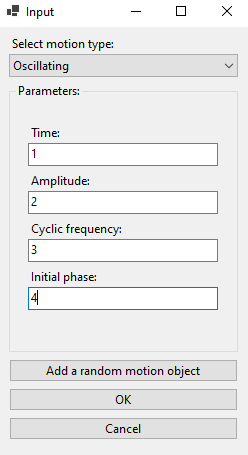
  

Рисунок 6 – Заполнение полей

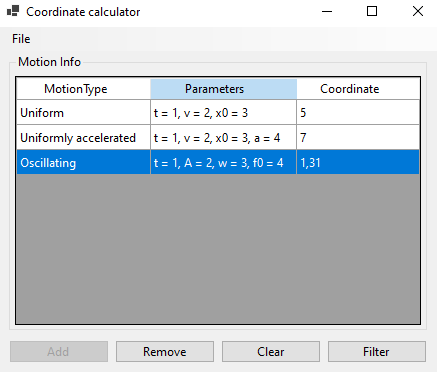


Рисунок 7 – Успешное добавление новых экземпляров

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).

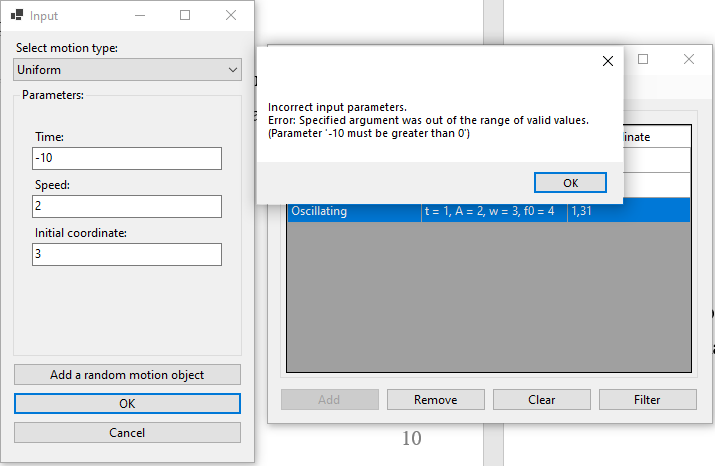


Рисунок 8 – Некорректный ввод (отрицательное число)

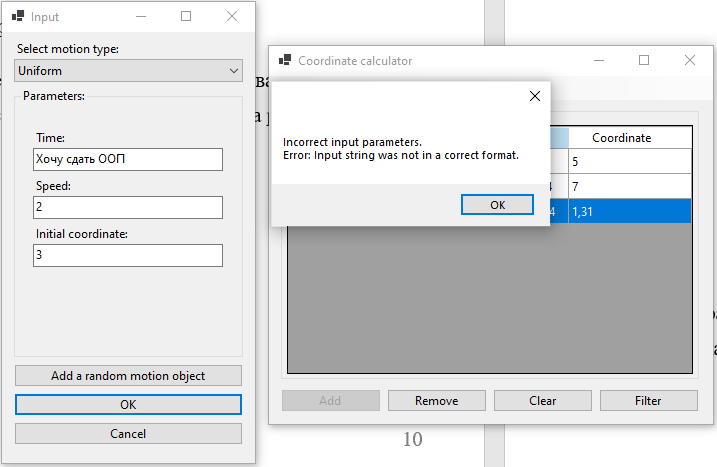


Рисунок 9 – Некорректный ввод (ввод недопустимых символов)

Подобная обработка предусмотрена для всех параметров элементов.

* + 1. *Тестовый случай «Remove Motion»*

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

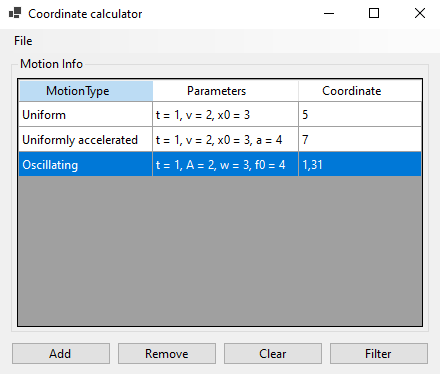


Рисунок 10 – Выбор элемента в таблице

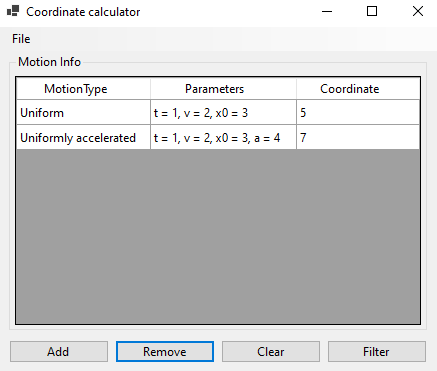


Рисунок 11 – Результат нажатия кнопки «Remove»

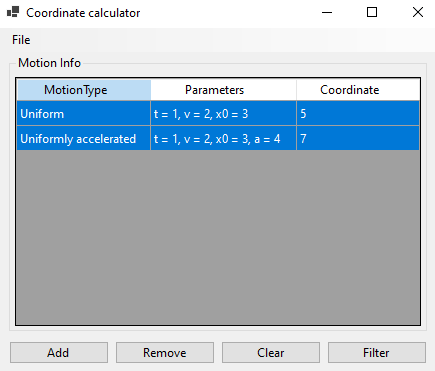


Рисунок 12 – Выбор нескольких элементов для удаления

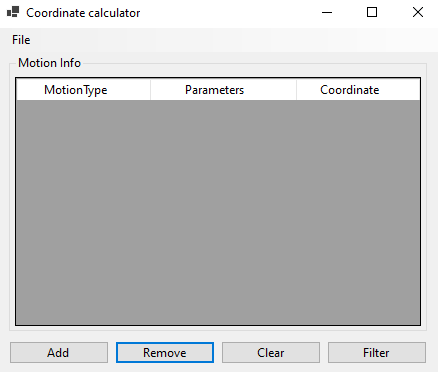


Рисунок 13 – Результат удаления выбранных элементов

* + 1. *Тестовый случай «Filter Motion»*

Для поиска элементов предусмотрена форма поиска (Рисунок 14).

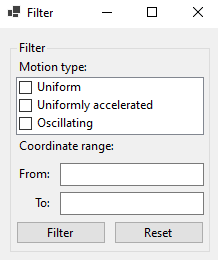


Рисунок 14 – Настройка параметров для поиска элементов

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).

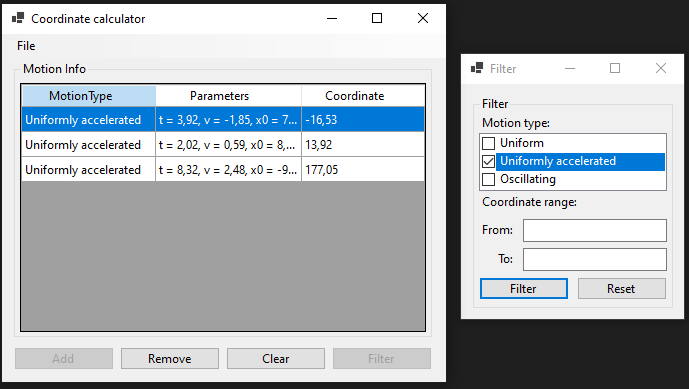


Рисунок 15 – Поиск по типу движения

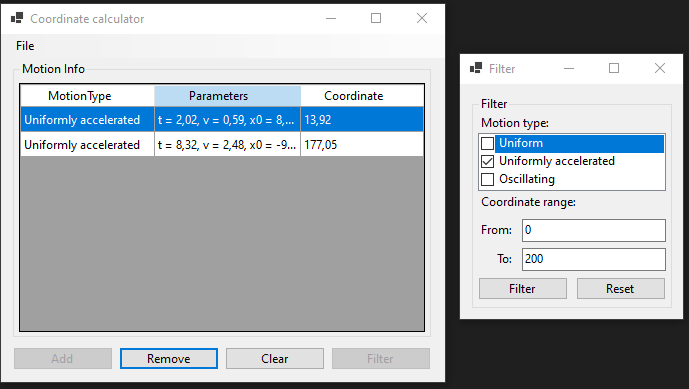


Рисунок 16 – Поиск по типу движения и величине координаты

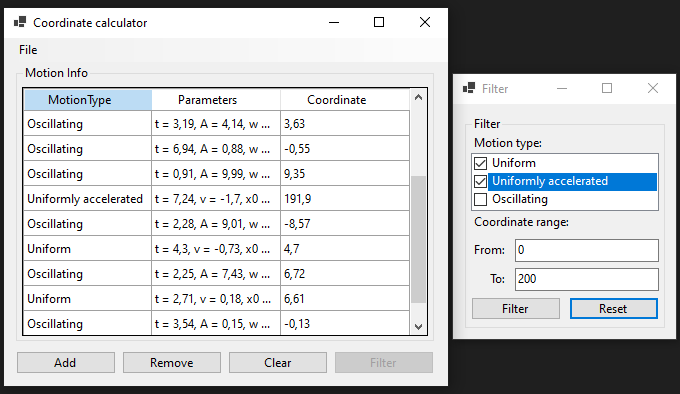


Рисунок 17 – Сброс условий поиска

* + 1. *Тестовый случай «Save Motion»*

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).

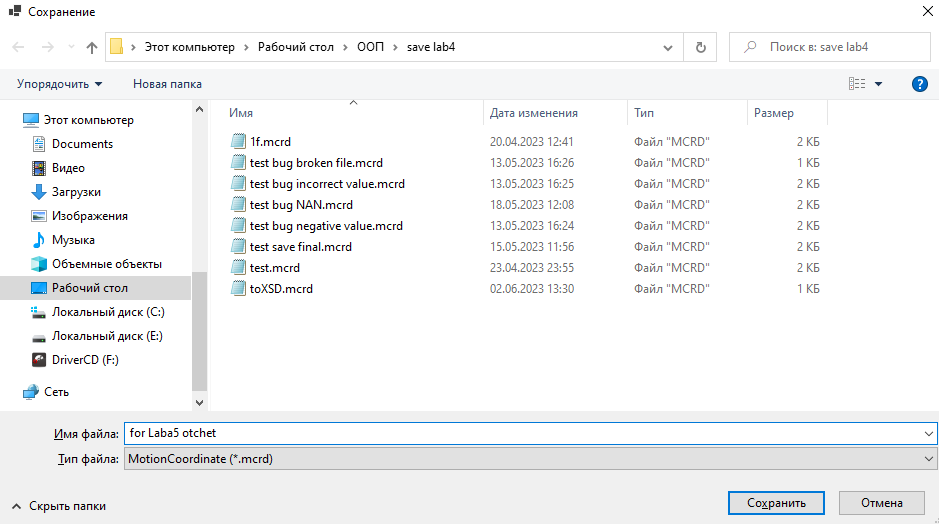


Рисунок 18 – Сохранение файла

* + 1. *Тестовый случай «Load Motion»*

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

Далее откроется системный диалог загрузки файла (Рисунок 19).

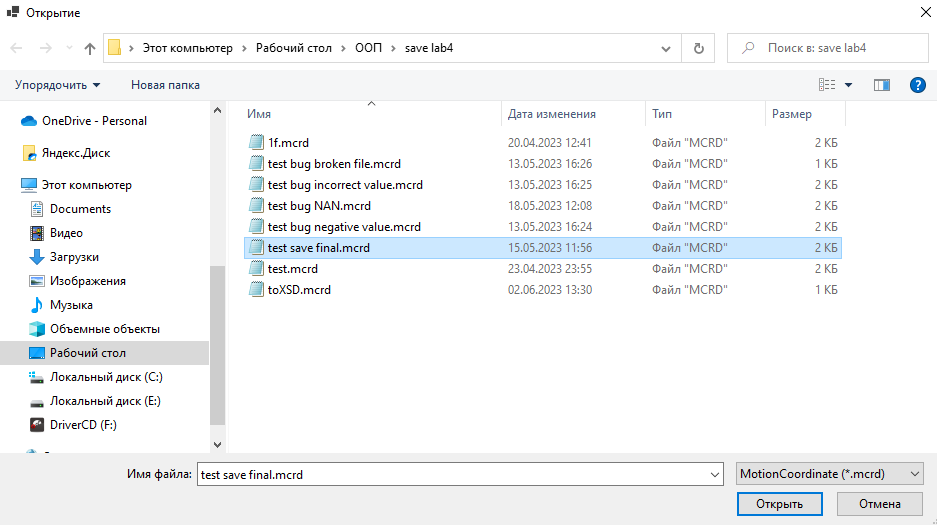


Рисунок 19 – Выбор файла для загрузки

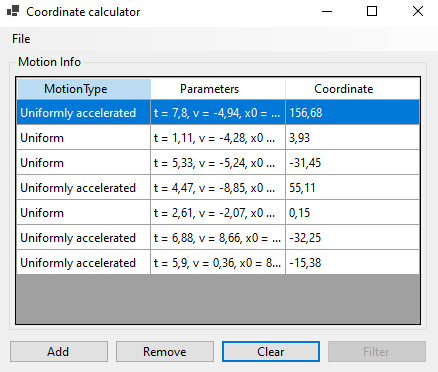


Рисунок 20 – Результат загрузки данных

В случае, если структура данных файла не соответствует установленному формату, появится соответствующее сообщение (Рисунок 21).

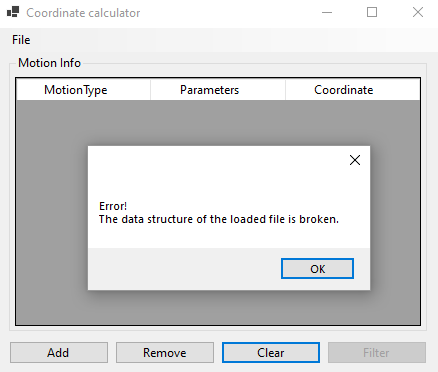


Рисунок 21 – Загрузка повреждённого файла

# **Список используемых 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 определения координаты для различных типов движения

Разработчик: студент гр. О-5КМ11 Соловьев М.Б..

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 определения координаты для различных типов движения».

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 Соловьев М.Б.

**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 определения координаты для различных типов движения, просмотра существующих экземпляров движений и их параметров и добавления новых.

**2.2 Цели создания системы**

Система создается в целях автоматизации расчета координаты для различных типов движения и формирования общего списка.

**3 Характеристика объектов автоматизации**

Раньше лаборантам необходимо было вручную проводить расчет координаты для различных типов движений при расчете траекторий движений и запоминать их параметры. Этот монотонный труд мог приводить к ошибкам, которые приводят к неточным расчетам. 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 следующему:

<?xml version="1.0" encoding="utf-8"?>

<ArrayOfMotionBase xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">

<MotionBase xsi:type="OscilMotion">

<Time></Time>

<Amplitude></Amplitude>

<CyclFrequency></CyclFrequency>

<InitPhase></InitPhase>

</MotionBase>

<MotionBase xsi:type="UniformMotion">

<Time></Time>

<Speed></Speed>

<InitCoordinate></InitCoordinate>

</MotionBase>

<MotionBase xsi:type="UniformAccelMotion">

<Time></Time>

<Speed></Speed>

<InitCoordinate></InitCoordinate>

<Acceleration></Acceleration>

</MotionBase>

</ArrayOfMotionBase>

**4.3 Функциональные требования**

**F01.** В системе должен быть реализован список экземпляров типов движений.

**F01.01.** В зависимости от типа элемента необходимо заполнить параметры для следующих типов движения: равномерное (начальная координата, скорость, время), равноускоренное (ускорение, начальная координата, скорость, время) и колебательное (амплитуда, циклическая частота, начальная фаза, время).

**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 Pro 20H2. Работоспособность на других выпусках и версиях не гарантируется.

**C02.** На рабочей станции должен быть установлен .NET Framework версии 6.

**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.