**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

**ПРОЕКТНАЯ ДОКУМЕНТАЦИЯ**

Отчёт по лабораторной работе №5

по дисциплине «Объектно-ориентированное программирование»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Выполнил |  |  |  | Д. С. Байкалова |
| Студент группы: О-5КМ11 |  | (подпись, дата) |  |  |
|  |  |  |  |  |
| Проверил преподаватель |  |  |  | А. А. Калентьев |
|  |  | (подпись, дата) |  |  |

Томск 2023

# **ВВЕДЕНИЕ**

Цель лабораторной работы направлена на обучение разработки проектной документации на созданный программный продукт. Для достижения поставленной цели необходимо выполнить следующее:

* Составление UML диаграмму вариантов использования для разработанной программы;
* Составление UML диаграммы классов;
* Описание классов, образующих связь типа «общее-частное» (наследование, реализация);
* Привести дерево ветвлений Git;
* Провести тестирование программы.

# **UML диаграмма вариантов использования**

Диаграмма вариантов использования (англ. Use case diagram) в UML – диаграмма, отражающая отношения между акторами и вариантами использования (ВИ) и являющаяся составной частью модели ВИ, позволяющей описать систему на концептуальном уровне.

ВИ соответствует отдельному сервису системы, определяет один из вариантов её использования и описывает типичный способ взаимодействия пользователя с системой.

Основное назначение диаграммы – это описание функциональности и поведения, позволяющее заказчику, конечному пользователю и разработчику совместно обсуждать проектируемую или существующую систему.

Диаграмма вариантов использования для разработанного ПО приведена на рисунке 1.

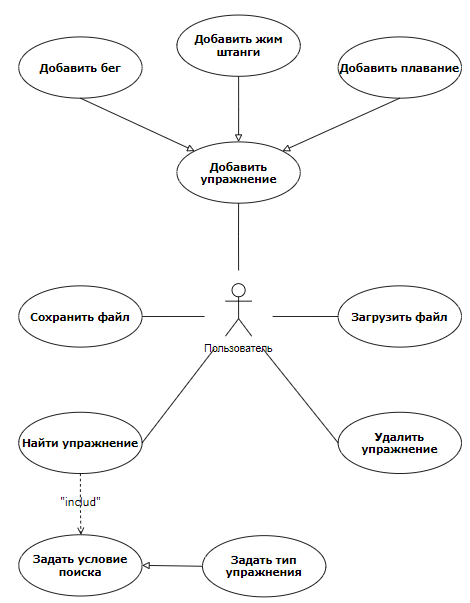


Рисунок 1. Диаграмма вариантов использования

# **UML диаграмма классов**

Диаграмма классов (от англ. «class diagram») предназначена для представления внутренней структуры программы в виде классов и связей между ними. Применяется при проектировании архитектуры, документировании системы, уточнений требований, а также для поддержки системы.

Диаграмма классов представляет описание структуры классов в системе и их взаимосвязи. Она отображает как статические аспекты системы, включая классы, атрибуты и методы, а также динамические аспекты, такие как связи между объектами и выполнение методов во время выполнения программы.

У каждого класса должно быть только одно назначение и уникально осмысленное имя, которое будет связано с этой целью.

Связи между классами на диаграмме могут представлять отношение, такие как ассоциация, наследование и реализацию. Ассоциация показывает отношения между классами, которые могут быть однонаправленными или двунаправленными. Наследование показывает. Как класс наследует свойства другого класса, который является его родительским классом. Реализация показывает, как класс реализует интерфейс или абстрактный класс.

Использование диаграммы классов помогает лучше понимать структуру системы и её компоненты.

Диаграмма классов приведена на рисунке 2.

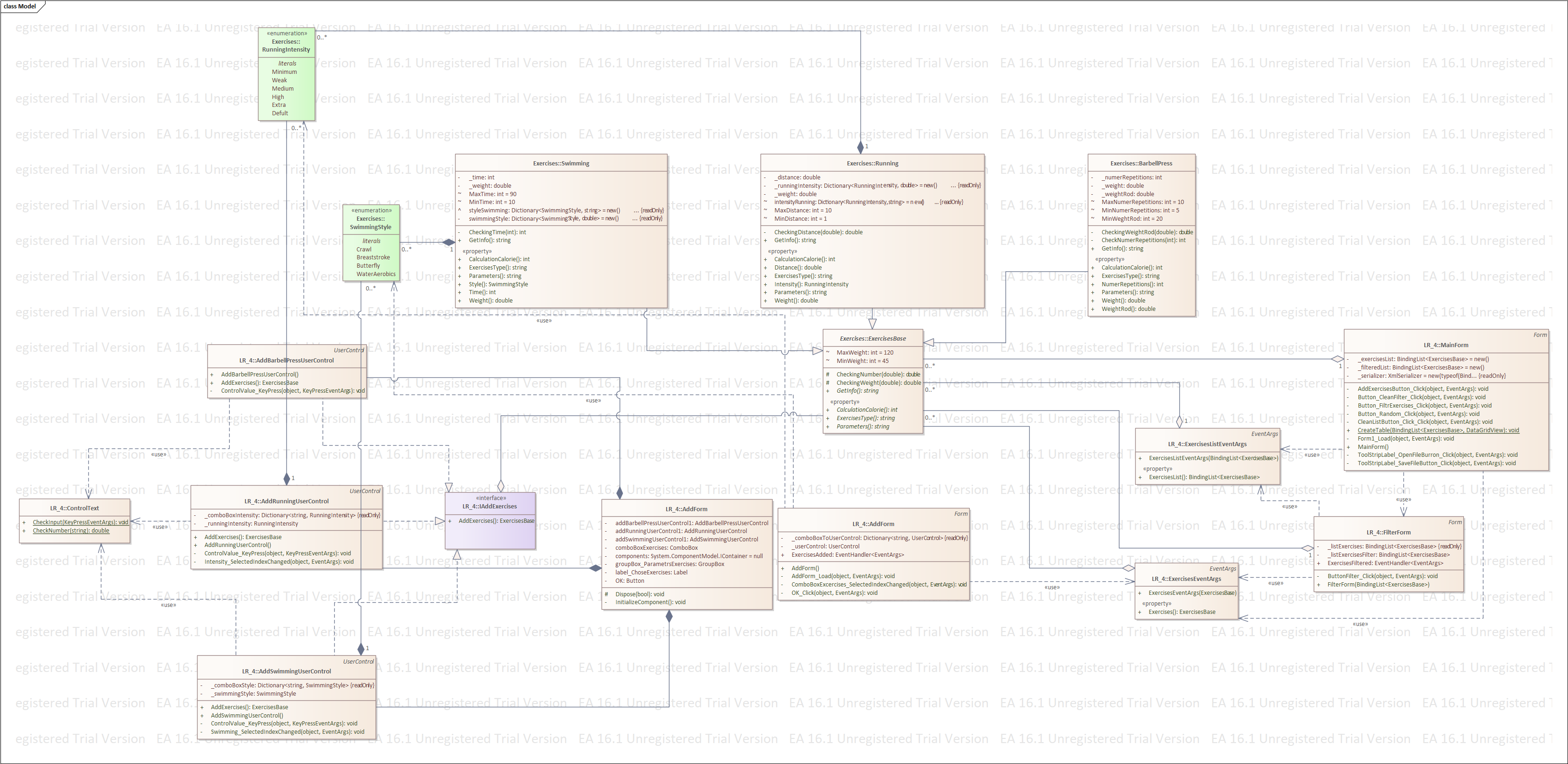


Рисунок 2. Диаграмма классов

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

В таблице 1 приведено описание абстрактного класса *ExercisrsBase* с его имеющимися полями, свойствами и методы класса.

Далее в таблицах приведены описания классов *BarbellPress*, *Running*, *Swimming*, которые наследуются от *ExercisrsBase*.

Таблица 1. Описание класса *ExercisrsBase*

|  |  |  |
| --- | --- | --- |
| **Название** | **Тип** | **Описание** |
| **Описание класса** | | |
| Класс *ExercisrsBase* – сущность для описания абстрактного упражнения в программе | | |
| **Свойства** | | |
| *+ ExercisesType* | *string* | Тип упражнения |
| *+* *Parameters* | *string* | Параметры упражнения |
| *+* *CalculationCalorie* | *int* | Затраченные калории |
| **Методы** | | |
| *# CheckingNumber(double)* | *double* | Проверка введённого параметра на ноль и NuN;  *double* – любой параметр упражнения. |
| *#* *CheckingWeight(double)* | *double* | Проверяет на корректность введённый вес;  *double* – вес человека. |
| *+* *GetInfo* | *string* | Метод вывода информации об упражнении.  Абстрактный метод, переопределяется в производных классах. |

Таблица 2. Описание класса *BarbellPress*

|  |  |  |
| --- | --- | --- |
| **Название** | **Тип** | **Описание** |
| **Описание класса** | | |
| Класс *BarbellPress* – сущность для описания жима штанги в программе | | |
| **Поля** | | |
| *-* *\_weight* | *double* | Вес человека |
| *- \_weightRod* | *double* | Вес штанги |
| *- \_numerRepetitions* | *int* | Количество повторений |
| **Свойства** | | |
| *+ Weight* | *double* | Вес человека |
| *+* *WeightRod* | *double* | Вес штанги |
| *+ NumerRepetitions* | *int* | Количество повторений |
| *+ ExercisesType* | *string* | Тип упражнения |
| *+* *Parameters* | *string* | Параметры упражнения |
| *+* *CalculationCalorie* | *int* | Затраченные калории |
| **Методы** | | |
| *+* *GetInfo* | *string* | Метод вывода информации об упражнении.  Абстрактный метод, переопределяется в производных классах. |
| *- CheckingWeightRod(double)* | *double* | Проверка веса штанги  *double* – вес штанги. |
| - *CheckNumerRepetitions(double)* | *double* | Проверка на количество повторений;  *double* – количество повторений. |

Таблица 3. Описание класса *Swimming*

|  |  |  |
| --- | --- | --- |
| **Название** | **Тип** | **Описание** |
| **Описание класса** | | |
| Класс *Swimming* – сущность для описания плавания в программе | | |
| **Поля** | | |
| *-* *\_weight* | *double* | Вес человека |
| *- \_time* | *int* | Время плавания |
| **Свойства** | | |
| *+ Weight* | *double* | Вес человека |
| *+* *Time* | *double* | Время плавания |
| *+ Style* | *SwimmingStyle* | Стиль плавания |
| *+ ExercisesType* | *string* | Тип упражнения |
| *+* *Parameters* | *string* | Параметры упражнения |
| *+* *CalculationCalorie* | *int* | Затраченные калории |
| **Методы** | | |
| *+* *GetInfo* | *string* | Метод вывода информации об упражнении.  Абстрактный метод, переопределяется в производных классах. |
| *- CheckingTime (double)* | *int* | Метод проверки времени при плавании.  *double* – ведённая время. |

Таблица 4. Описание класса *Running*

|  |  |  |
| --- | --- | --- |
| **Название** | **Тип** | **Описание** |
| **Описание класса** | | |
| Класс *Running* – сущность для описания бега в программе | | |
| **Поля** | | |
| *-* *\_weight* | *double* | Вес человека |
| *- \_distance* | *double* | Расстояние |
| **Свойства** | | |
| *+ Weight* | *double* | Вес человека |
| *+* *Distance* | *double* | Расстояние |
| *+ Intensity* | *RunningIntensity* | Интенсивность бега |
| *+ ExercisesType* | *string* | Тип упражнения |
| *+* *Parameters* | *string* | Параметры упражнения |
| *+* *CalculationCalorie* | *int* | Затраченные калории |
| **Методы** | | |
| *+* *GetInfo* | *string* | Метод вывода информации об упражнении.  Абстрактный метод, переопределяется в производных классах. |
| *- CheckingDistance (double)* | *double* | Метод проверки дистанции бега.  *double* – ведённая дистанция. |

# **Дерево ветвлений Git**

На рисунке 3 представлено дерево ветвлений Git.

# **Тестирование программы**

Графический интерфейс пользователя представлен на рисунке 4.

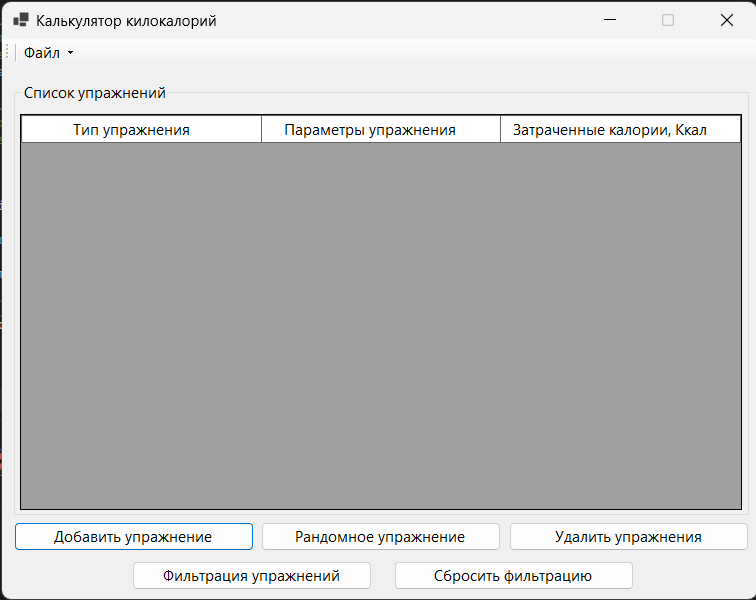


Рисунок 4. Графический интерфейс пользователя

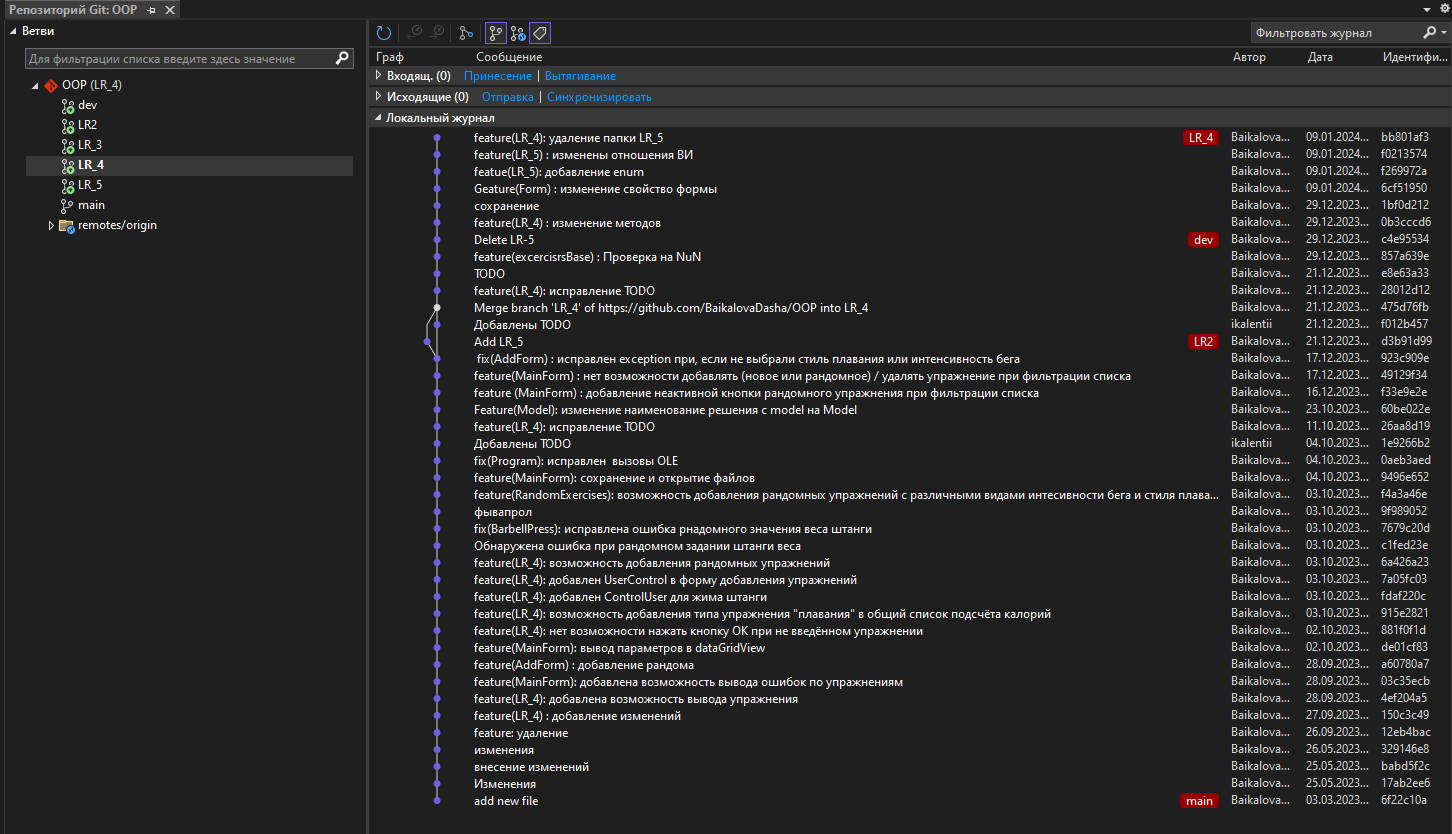


Рисунок 3. Дерево ветвлений Git

## **Добавление упражнение в список**

Для добавления элемента необходимо вызвать соответствующую форму путём нажатия кнопки «Добавить упражнение»

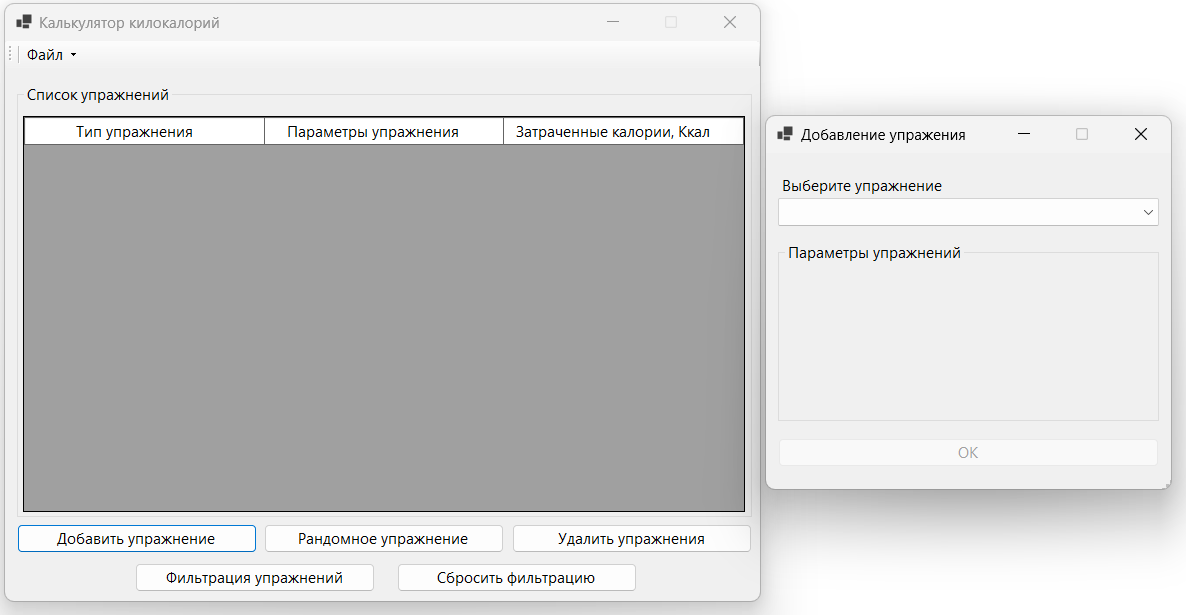
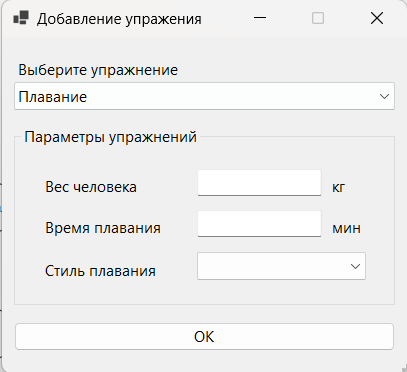
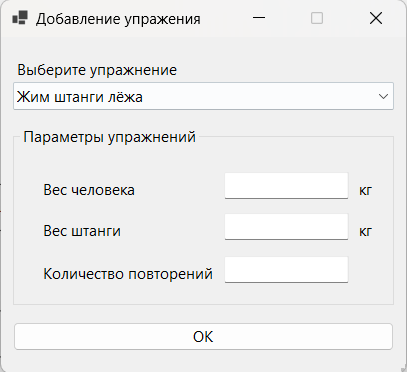


Рисунок 5. Вызов формы добавления элемента

До тех пор, пока не будет выбрано упражнение программа не даёт добавить пустое значение. Можно выбрать 3 вида упражнения (Бег, плавание и жим штанги) в выпадающем меню.

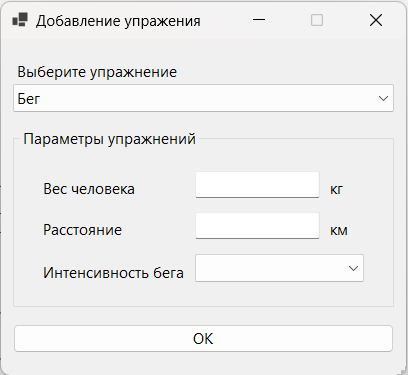


Рисунок 6. Варианты упражнений и их параметры

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

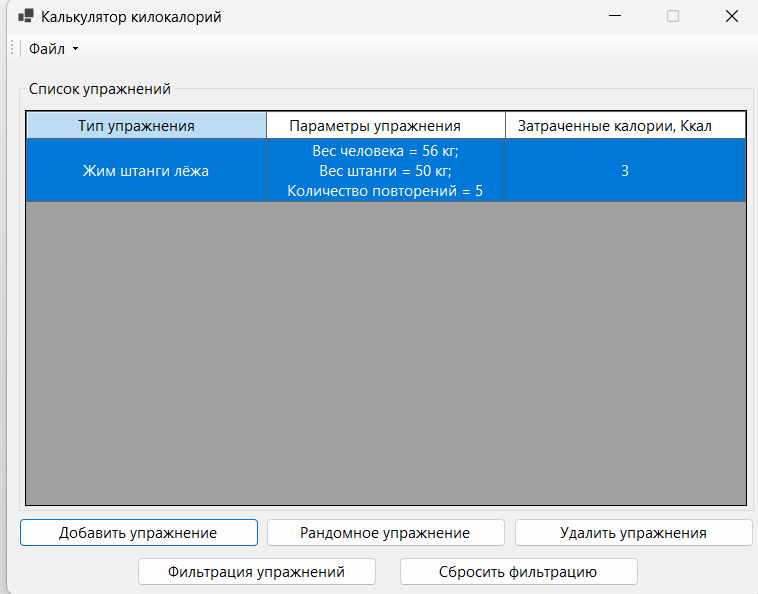


Рисунок 7. Успешное добавление нового элемента

В программе предусмотрена система обработки некорректного ввода данных пользователем.

Для каждого упражнения покажем свой контроль ввода данных.

### **Проверка данных при добавлении Плавания**

При плавании выполняется проверка на вес человека и на время его нахождения в воде, а также каким стелем придерживался.

Вес человека при плавании должен быть меньше 45 и не больше 120, что указано на рисунке 8, а также для того, чтобы начались тратиться, калории необходимо поплавать хотя бы 10 минут. И как ещё одно предостережение для пользователя плавать больше 90 минут не рекомендуется, вредно для здоровья, поэтому также сделана для этого проверка. И без выбора Стиля плавания калькулятор не позволит рассчитать калории выполненного упражнения.

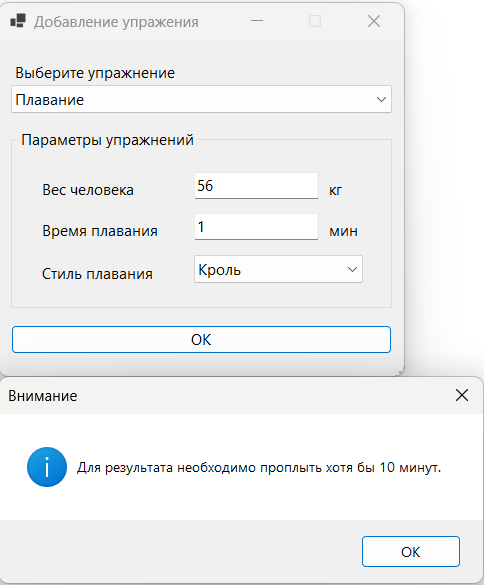
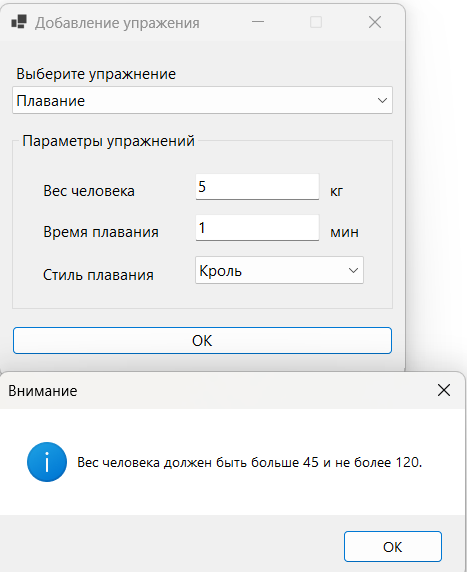
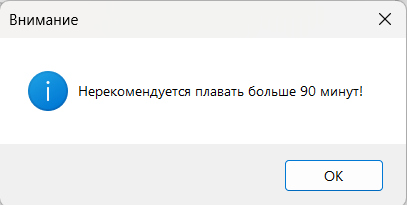
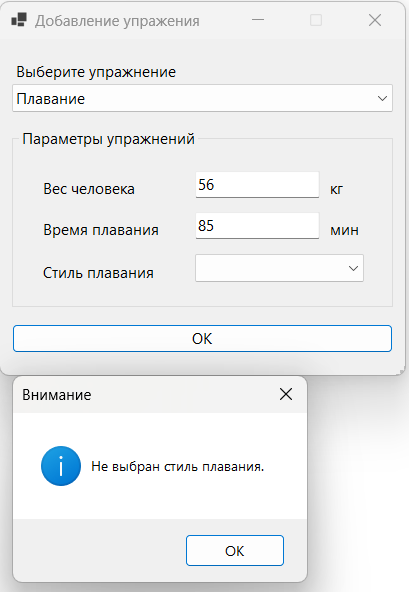


Рисунок 8. Варианты ошибочного ввода пользователем упражнения «плавание» и контроль ввода для него

### **Проверка данных при добавлении Бега**

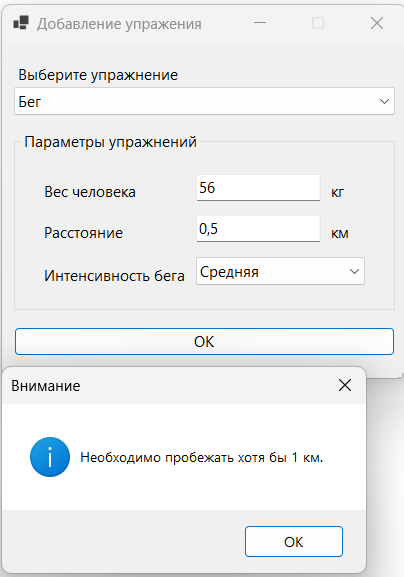
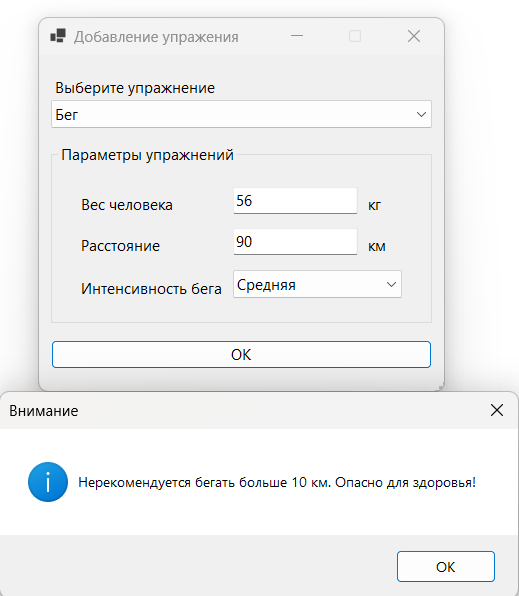
Аналогичная ситуация и при добавлении бега, выполняется проверка веса и выбора интенсивности бега. Для достижения результатов необходимо пробежать хотя бы больше 1 км и бегать больше 10 км не рекомендуется, опасно для здоровья.

Рисунок 9. Варианты ошибочного ввода пользователем упражнения «бег» и контроль ввода для него

### **Проверка данных при добавлении Жима штанги**

При выполнении «Жима штанги» не рекомендуется поднимать штангу больше своего веса (рис. 10).

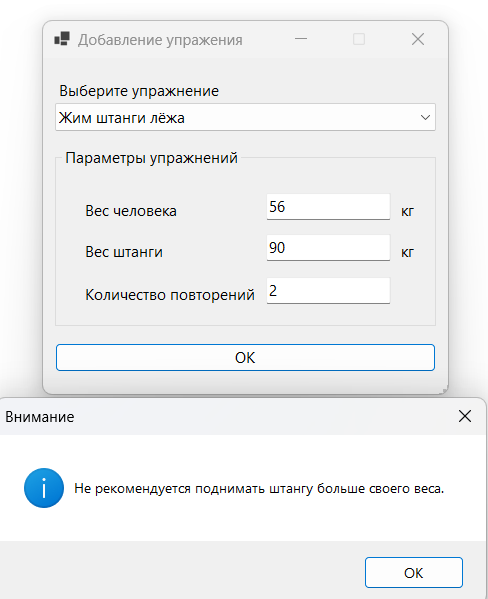


Рисунок 10. Ошибочный ввод веса штанги

А также для достижения результатов начать нужно хотя бы с 5 повторений (рис. 11).

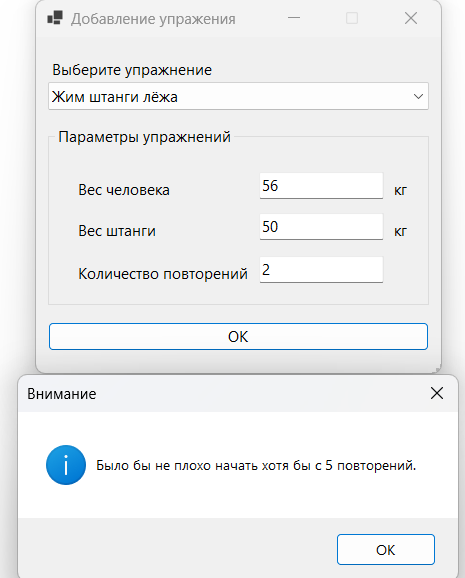


Рисунок 11. Неверный ввод количества повторений

Так же есть контроль ввода на тип значение, то есть в поля параметров упражнения можно вводить только цифры и определённые символы. Отрицательные, нулевые и пустые значения считаются неверным вводом и на каждый неверный ввод выскакивает своё предупреждение.

## **Удаление упражнения из списка**

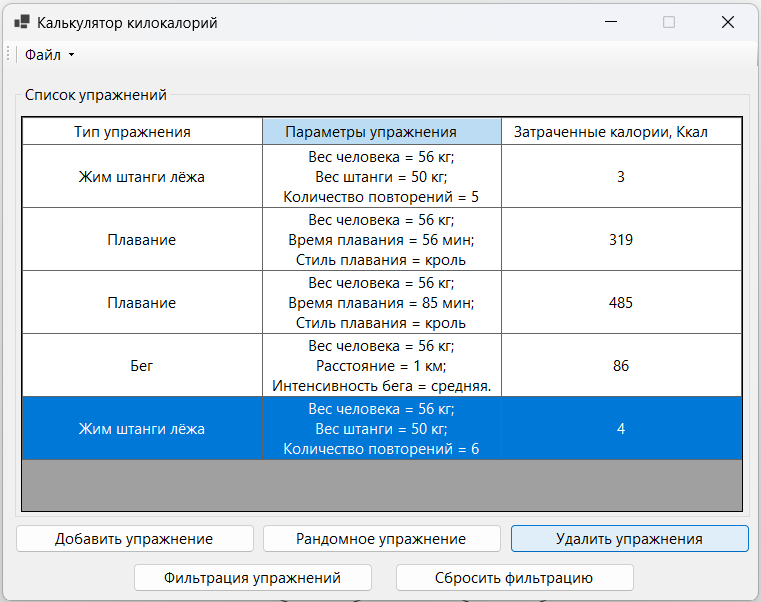
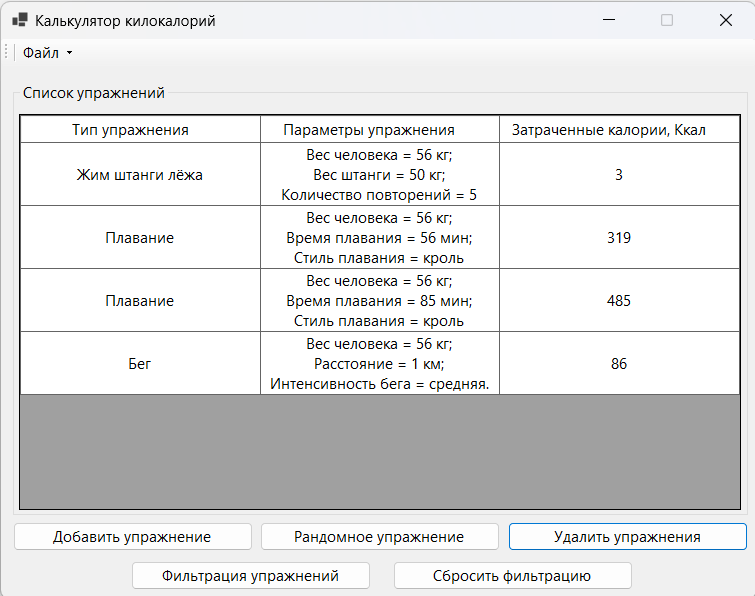
Для удаления одного или нескольких элементов необходимо выбрать их в таблице и нажать на кнопку «Удалить упражнение» (рисунок 12).

Рисунок 12. Выбор элемента и результаты нажатия кнопки «Удалить упражнение»

## **Выполнение фильтрации по списку**

Для поиска упражнений в списке необходимо нажать кнопку «Фильтрация упражнений». Откроется окно «Фильтр», где можно выбрать тип упражнения (рис. 13).

До тех пор, пока находится в отфильтрованном списке нельзя добавить или удалить упражнение.

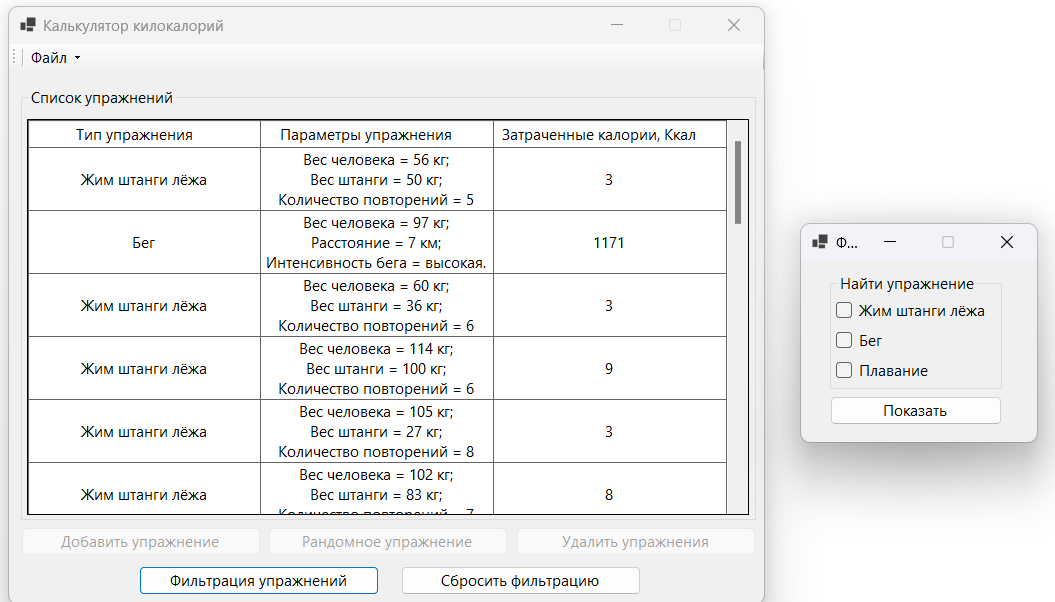


Рисунок 13. Выбор фильтра

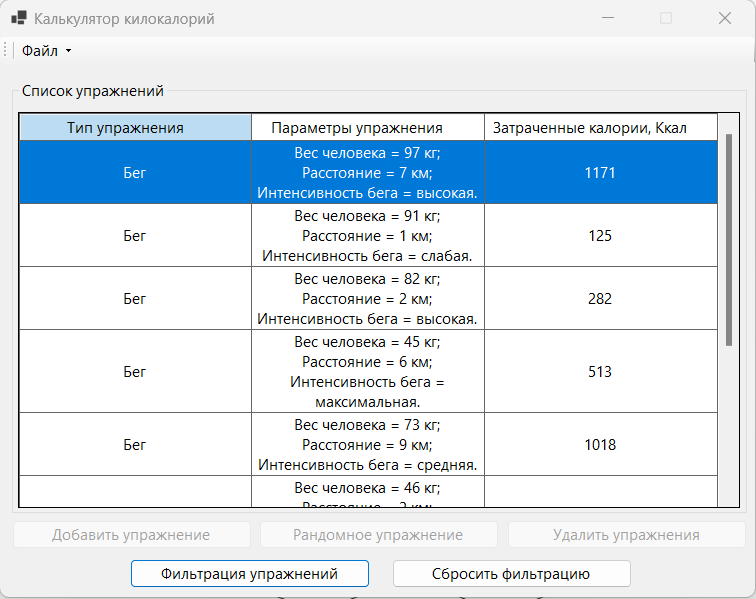


Рисунок 14. Отфильтрованный список

## **Сохранение данных**

Для сохранения данных в таблице необходимо нажать на панели «Файл/Сохранить как» (рис. 16). Откроется системный диалог сохранения файла, где пользователь выбирает директорию и указывает имя файла (рисунок 17).

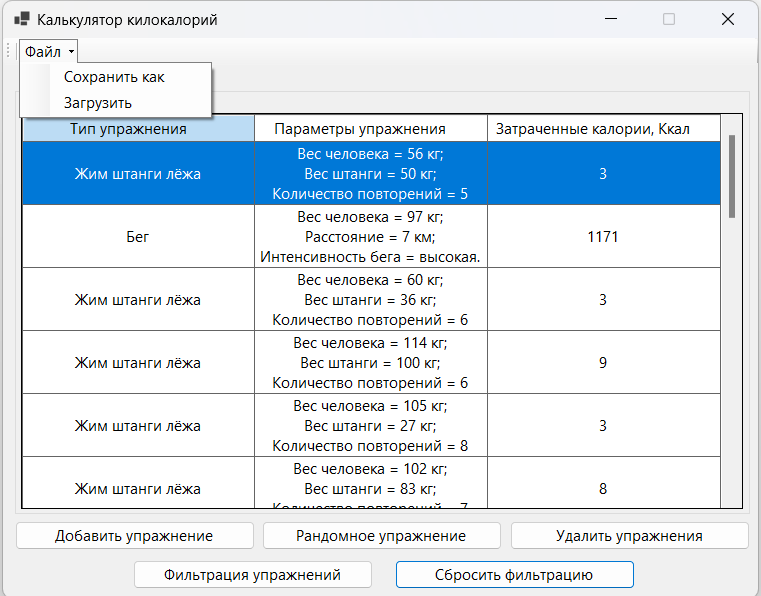


Рисунок 16. Сохранение данных

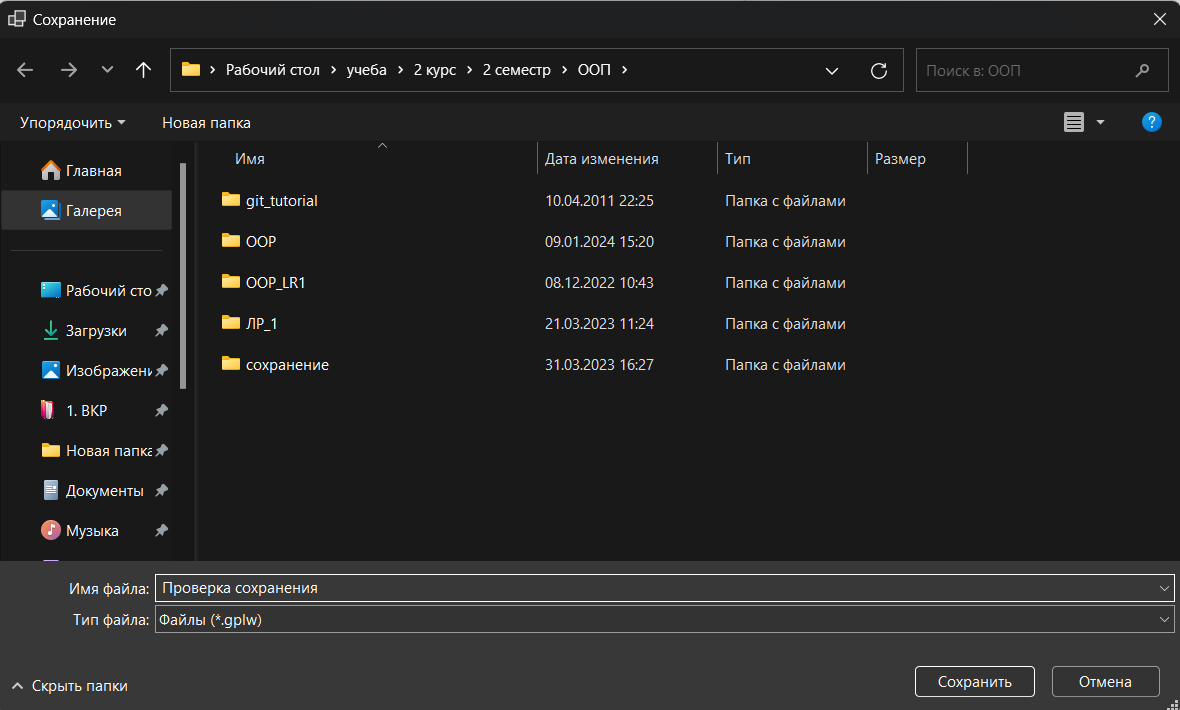


Рисунок 17. Сохранение файла

После сохранения данных в файл появляется сообщение об успешном сохранении.

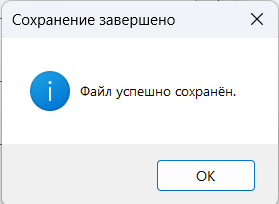


Рисунок 18. Успешное сохранение

Если таблица пустая, то данные для сохранения отсутствуют (рис. 19).

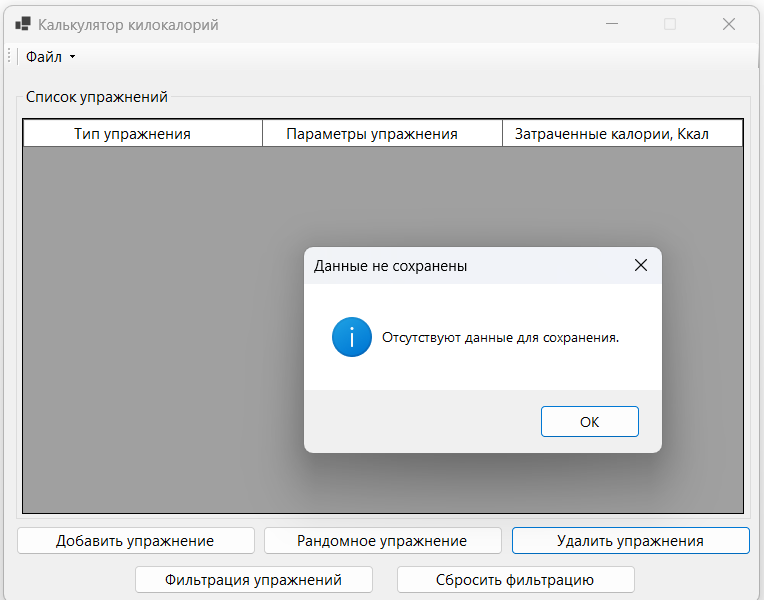


Рисунок 19. Результат нажатия на кнопку «Сохранить как» при пустой таблице

## **Загрузка данных**

Для загрузки данных в таблицу необходимо нажать на панели «Файл/ Загрузить» (рис. 20). Откроется системный диалог загрузки файла, где пользователь выбирает директорию и указывает файл (рисунок 21).

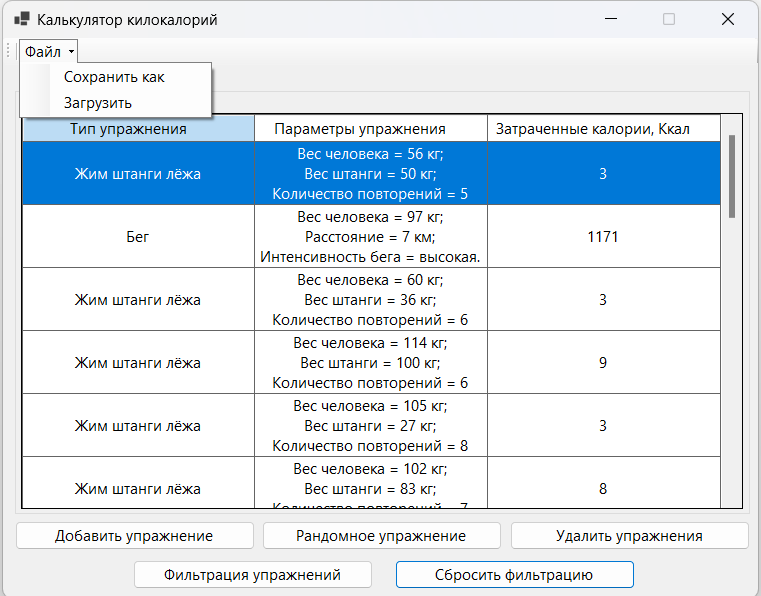


Рисунок 20. Сохранение данных

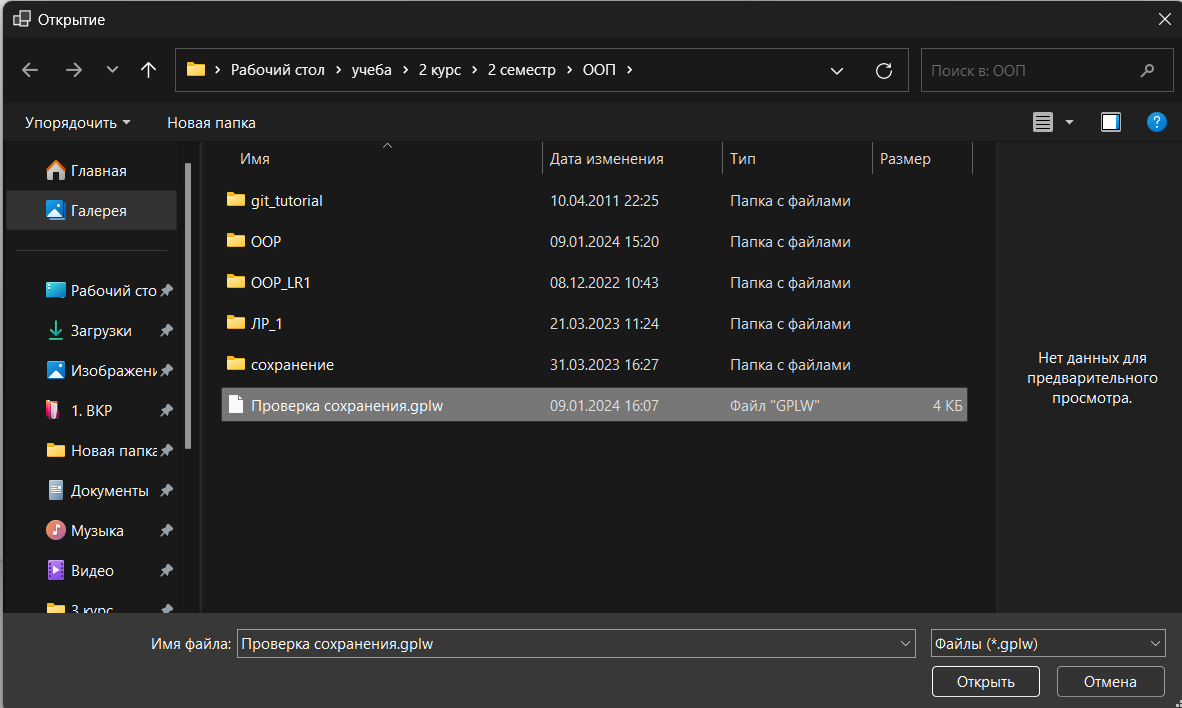


Рисунок 21. Сохранение файла

После загрузки данных в таблицу появляется сообщение об успешной загрузке.

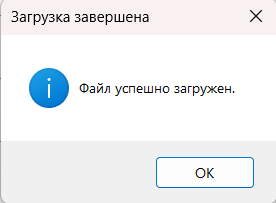


Рисунок 22. Успешная загрузка

В случае, если структура файла не соответствует установленному формату, появится соответствующее сообщение (рисунок 23).

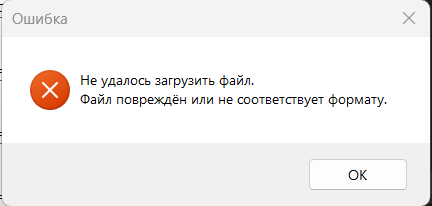


Рисунок 23. Загрузка повреждённого файла

**ПРИЛОЖЕНИЕ А**

**Техническое задание на создание автоматизированной системы**

Программа для расчёта затраченных калорий в зависимости от упражнения

Разработчик: студент гр. О-5КМ11 НИ ТПУ Байкалова Д.С.

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 расчёта калорий».

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 Байкалова Д.С.

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

Начало работ: 09 апреля 2023 г.

Окончание работ: 09 июня 2023 г.

**2 Назначение и цели создания системы**

**2.1 Назначение системы**

Система предназначена для расчёта затраченных калорий в зависимости от упражнения: бег, плавание и жим штанги.

**2.2 Цели создания системы**

Система создаётся в целях сокращения трудозатрат человека при расчёте затраченных калорий в зависимости от упражнения.

**3 Характеристика объектов автоматизации**

Система рассчитывает затраченные калории при выполнении упражнений: жим штанги, плавание, бег. Каждый из данных типов упражнений характеризуется различным набором информации, необходимой для осуществления расчета.

Для жима штанги необходимо указать количество повторений, вес штанги и вес человека. Для плавания должно указываться вес человека, время, уделённое тренировке и стиль плавания. Для бега указывается также вес человека, преодолённое расстояние и интенсивность бега.

**4 Требования к системе**

Таблица 4.1. Префиксы 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 упражнения должны храниться в XML-файле с расширением .gplw, структура которого представлена на рисунке А.1

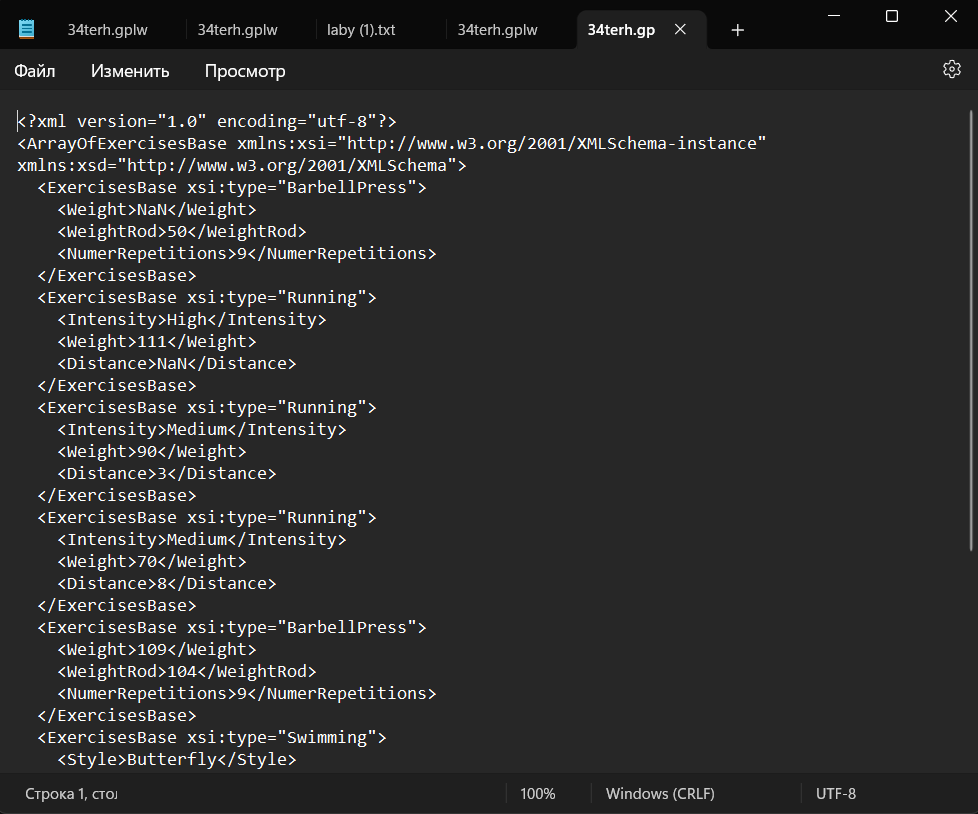


Рисунок А.1. Структура поддерживаемого XML-файла с расширением . gplw

**4.3 Функциональные требования**

**F01.** Система должна рассчитывать калории при выполнении упражнений:

* бег;
* плавание;
* жим штанги

**F01.01.** Калории, затраченные при выполнении жима штанги, рассчитываются:

где – масса штанги, кг;

–количество повторений, шт;

**F01.02.** Калории, затраченные при выполнении бега, рассчитываются:

где – расстояние после пробежки, км;

– интенсивность бега;

**F01.03.** Калории, затраченные при выполнении плавания, рассчитываются:

где – время плавания, мин;

– стиль плавания;

**F02.** В системе должен быть реализован список элементов затраченных калорий

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

* тип упражнения;

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

**F04.** В системе должна присутствовать функция удаления упражнений 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 11 версией не ниже 22H2. Работоспособность на других выпусках и версиях не гарантируется.

**C02.** На рабочей станции должен быть установлен .NET Framework версии 7.0.

**4.6 Требования к аппаратному обеспечению**

**C03.** Процессор – тактовая частота не менее 2,4 ГГц.

**C04.** ОЗУ: не менее 2 ГБ для 32-разрядных систем или не менее 4 ГБ для 64-разрядных систем

**C05.** Объем жесткого диска: не менее 10 МБ