|  |
| --- |
| **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-5КМ01 | |  | \_\_\_\_\_\_\_\_\_\_ | |  | Чернобров М.Е. |
|  | |  |  | |  |  |
|  | |  |  | \_\_\_ \_\_\_\_\_\_ 2022 г. | | |
|  | |  |  | |  |  |
| Проверил: | доцент, к.т.н. |  | \_\_\_\_\_\_\_\_\_\_ | |  | Калентьев А. А. |
|  |  |  |  | |  |  |
|  |  |  |  | \_\_\_ \_\_\_\_\_\_ 2022 г. | | |

Томск 2022

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc74956671)

[1 Основная часть 4](#_Toc74956672)

[1.1 UML диаграмма вариантов использования 4](#_Toc74956673)

[1.2 UML диаграмма классов 6](#_Toc74956674)

[1.3 Описание классов, образующих связь типа «общее-частное» 8](#_Toc74956675)

[1.4 Дерево ветвлений Git 9](#_Toc74956676)

[1.5 Тестирование программы 10](#_Toc74956677)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 21](#_Toc74956678)

[ПРИЛОЖЕНИЕ А 22](#_Toc74956679)

**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 лабораторной работы:

Обучение разработки 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]. Диаграмма 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.

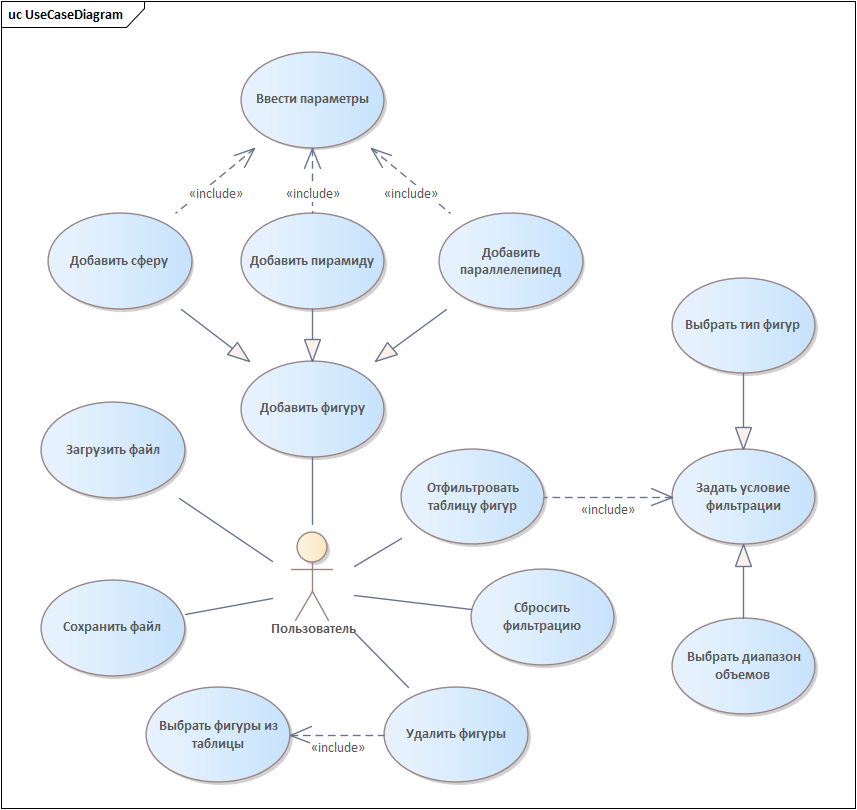


Рисунок 1 – Диаграмма вариантов использования

**1.2 UML диаграмма классов**

Диаграмма классов — это центральная методика моделирования, которая используется практически во всех объектно-ориентированных методах. Эта диаграмма описывает типы объектов в системе и различные виды статических отношений, которые существуют 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.

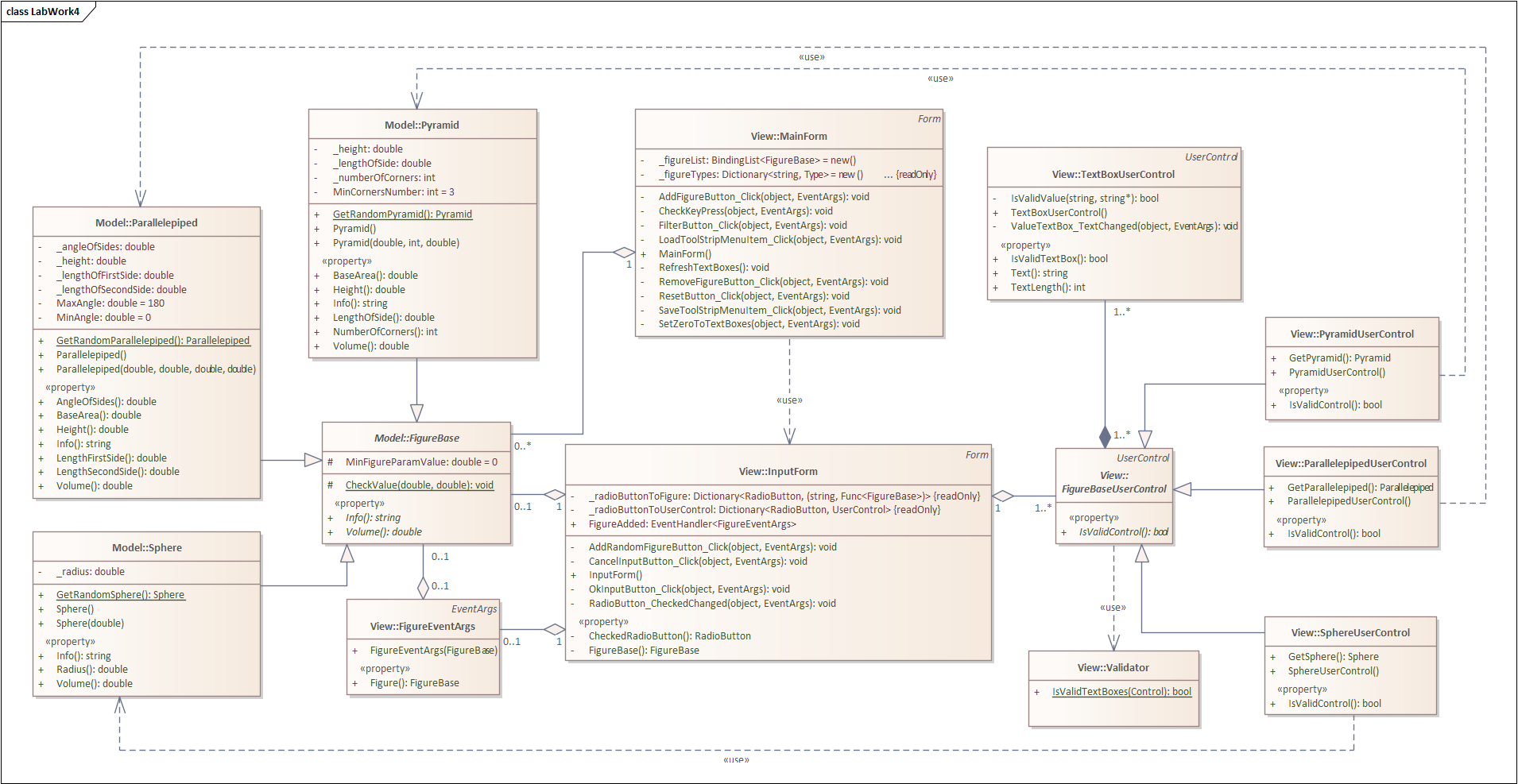


Рисунок 2 – UML диаграмма классов

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

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

Таблица 1 – Описание класса *FigureBase*

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс *FigureBase* – абстрактный базовый класс для объемных фигур | | |
| Поля | | |
| # MinFigureParamValue | double = 0 | Минимальное значение любого из параметров фигуры |
| Свойства | | |
| + Info | string | Информации о фигуре.  Абстрактное свойство, переопределяется в производных классах |
| + Volume | double | Объем фигуры.  Абстрактное свойство, переопределяется в производных классах |
| Методы | | |
| + CheckValue(double, double) | void | Проверяет корректность ввода параметров фигуры |

В таблицах 2–4 приведены описания классов Sphere, Pyramid и Parallelepiped, которые наследуются от *FigureBase*.

Таблица 2 – Описание класса Sphere

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс Sphere – сфера | | |
| Поля | | |
| – \_radius | double | Радиус сферы |
| Свойства | | |
| + Info | string | Информации о сфере |
| + Volume | double | Объем сферы |
| + Radius | double | Радиус сферы |
| Методы | | |
| + GetRandomSphere() | Sphere | Статический метод получения сферы c произвольными параметрами |

Таблица 3 – Описание класса Pyramid

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс Pyramid – пирамида | | |
| Поля | | |
| – \_height | double | Высота пирамиды |
| – \_lengthOfSide | double | Длина стороны основания пирамиды |
| – \_numberOfCorners | int | Количество углов основания пирамиды |
| – MinCornersNumber | int = 3 | Минимальное количество углов основания пирамиды |
| Свойства | | |
| + BaseArea() | double | Площадь основания |
| + Height() | double | Высота пирамиды |
| + Info() | double | Информация о пирамиде |
| + LenghtOfSides() | double | Длина стороны основания пирамиды |
| + NumberOfCorners() | int | Количество углов основания пирамиды |
| + Volume() | double | Объем пирамиды |
| Методы | | |
| + GetRandomPyramid() | Pyramid | Статический метод получения пирамиды c произвольными параметрами |

Таблица 4 – Описание класса Parallelepiped

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс Parallelepiped – параллелепипед | | |
| Поля | | |
| – \_angleOfSides | double | Угол между сторонами основания параллелепипеда |
| – \_height | double | Высота параллелепипеда |
| – \_lenghtOfFirstSide | double | Длина первой стороны основания |
| – \_lenghtOfSecondSide | double | Длина второй стороны основания |
| – MaxAngle | double = 180 | Максимальный угол между сторонами основания |
| – MinAngle | double = 0 | Минимальный угол между сторонами основания |
| Свойства | | |
| + AngleOfSides() | double | Угол между сторонами основания параллелепипеда |
| + BaseArea() | double | Площадь основания параллелепипеда |
| + Height() | double | Высота параллелепипеда |
| + Info() | double | Информация о параллелепипеде |
| + LenghtOfFirstSide() | double | Длина первой стороны основания |
| + LenghtOfSecondSide() | double | Длина второй стороны основания |
| + Volume() | double | Объем параллелепипеда |
| Методы | | |
| + GetRandomParallelepiped() | Parallelepiped | Статический метод получения параллелепипеда c произвольными параметрами |

**1.4 Дерево ветвлений Git**

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

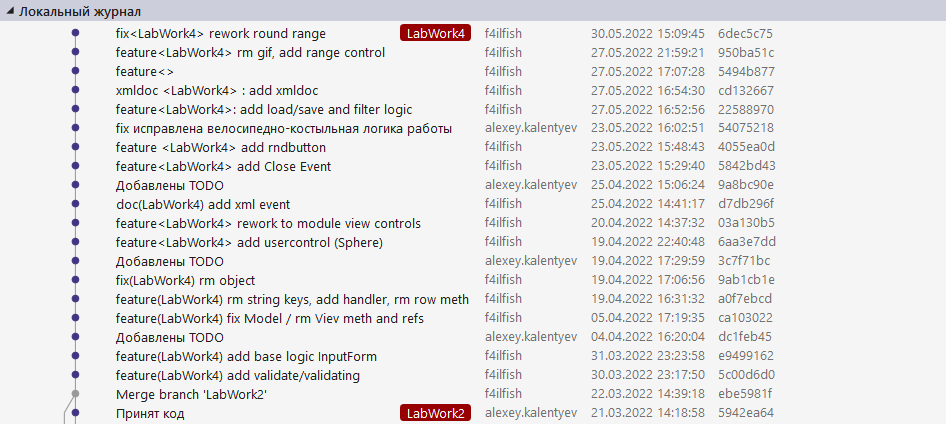


Рисунок 3 – Дерево ветвлений Git

**1.5 Тестирование программы**

Далее приводится процесс функционального тестирования программы. Графический интерфейс пользователя представлен на рисунке 4.

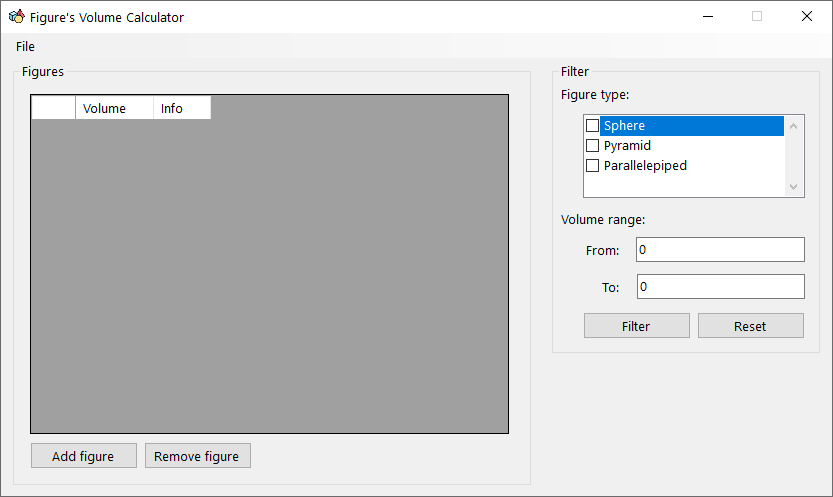


Рисунок 4 – Графический интерфейс пользователя

**1.5.1 Тестовый случай «Добавить элемент»**

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

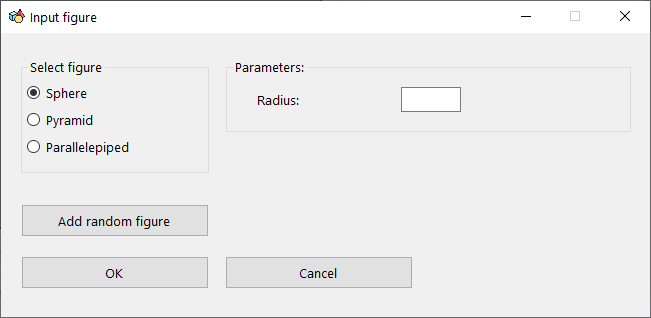


Рисунок 5 – Форма для добавления элемента

Параметры фигуры (сферы, пирамиды или параллелепипеда) можно ввести, предварительно выбрав соответствующий переключатель (радиобаттон).

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

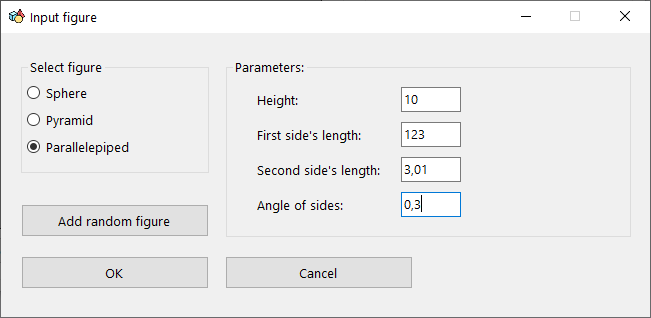


Рисунок 6 – Заполнение полей

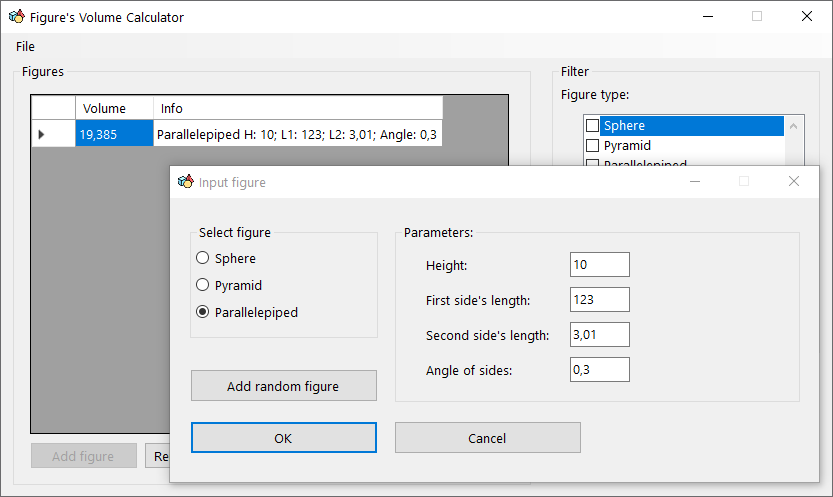


Рисунок 7 – Успешное добавление нового элемента

В программе предусмотрена система обработки некорректного ввода данных пользователем. Например, при попытке ввести отрицательное число, символы, NaN, Null, число в ином формате разделителя (в зависимости от настроек культуры) или просто оставить пустым ячейку – рядом будет выведено соответствующее уведомление о некорректности ввода.

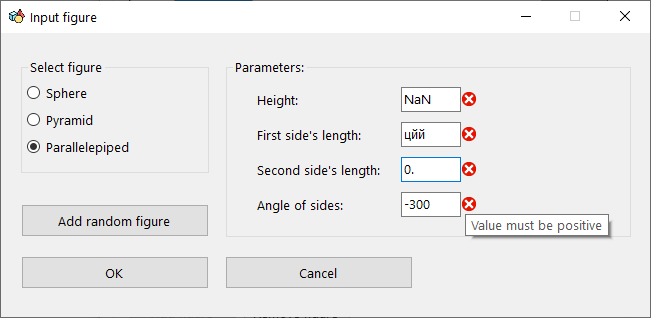


Рисунок 8 – Некорректный ввод (отрицательное число)

Если попытаться нажать кнопку «ОК» при имеющихся ошибках, то будет выведено сообщение об ошибке.

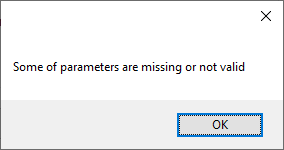


Рисунок 9 – Сообщение об ошибке

Подобная обработка предусмотрена для всех параметров транспорта.

При этом наличие ошибок при вводе параметров одного типа фигуры не блокирует возможность сменить тип фигуры и начать вводить новые параметры.

**1.5.2 Тестовый случай «Удалить элемент»**

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

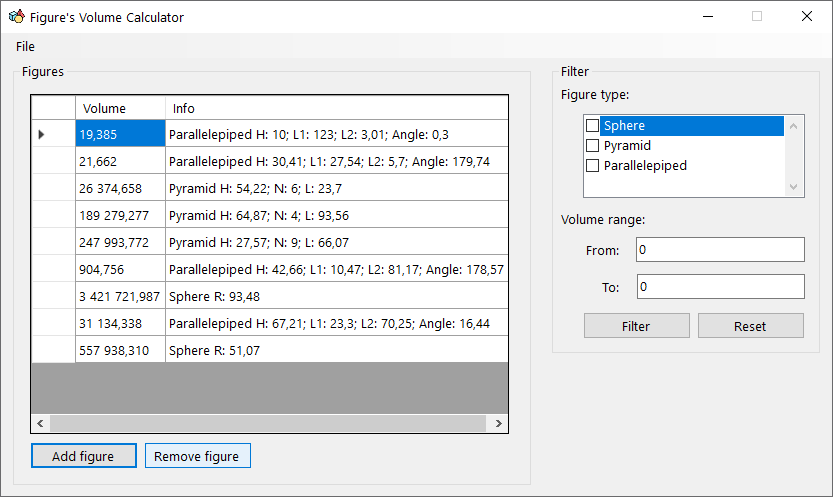


Рисунок 10 – Выбор элемента в таблице

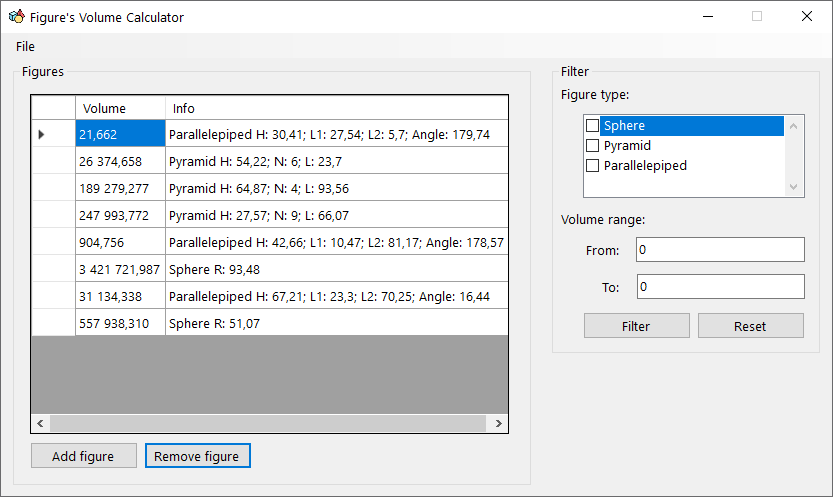


Рисунок 11 – Результат нажатия кнопки «Remove figure»

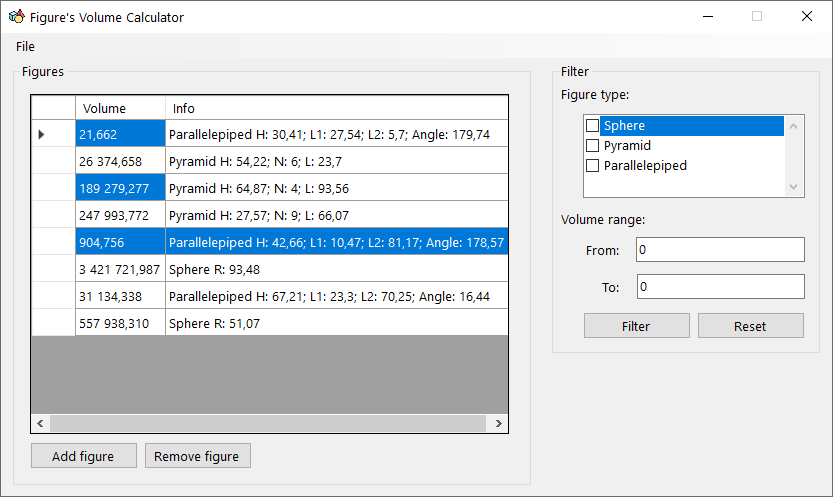


Рисунок 12 – Выбор нескольких элементов для удаления

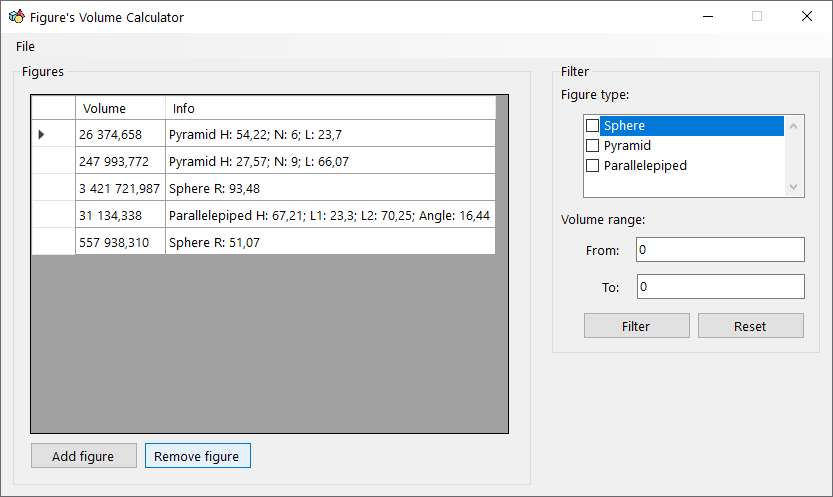


Рисунок 13 – Результат удаления выбранных элементов

**1.5.3 Тестовый случай «Отфильтровать фигуры»**

Для фильтрации фигур в таблице (рисунок 14) необходимо:

* выбрать необходимые типы фигур посредством флажков (чекбоксов);
* задать диапазон объемов в ячейках «From» и «To».

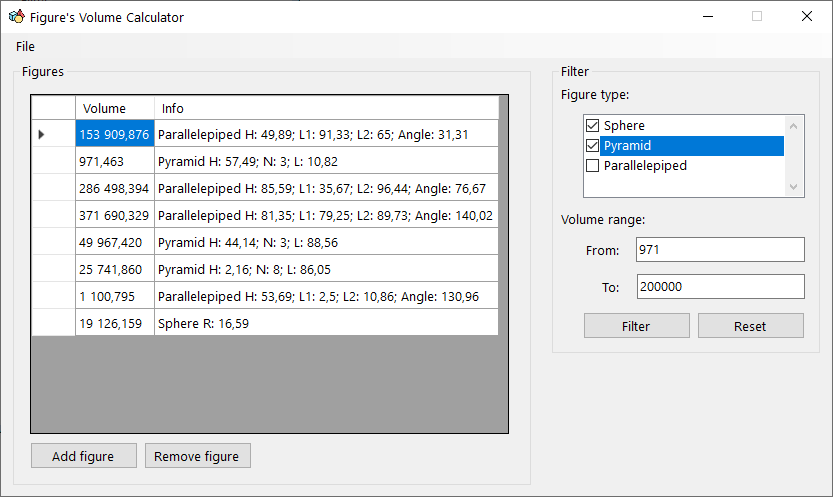


Рисунок 14 – Задание условий фильтрации

После ввода значения по нажатию кнопку «Filter» осуществляется фильтрация таблицы с фигурами (рисунок 15).

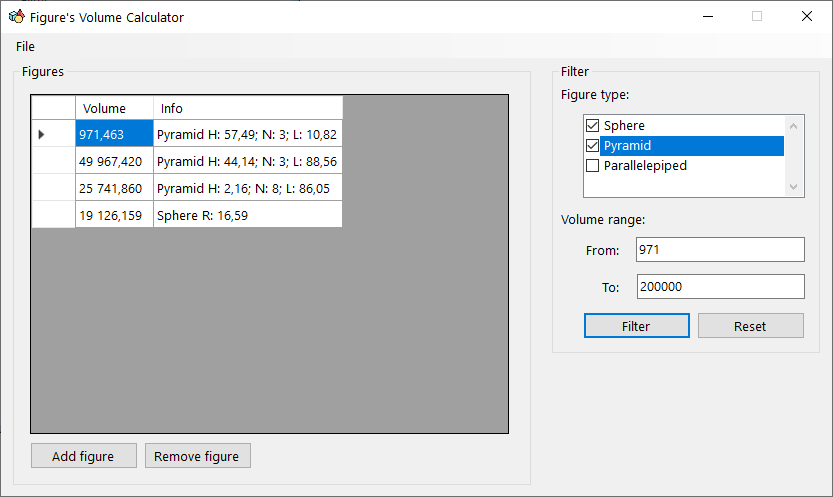


Рисунок 15 – Применение фильтрации

Для того, чтобы сбросить фильтр, предусмотрена соответствующая кнопка «Reset» (рисунок 16). Также кнопка автоматически устанавливает в диапазоны минимальный и максимальный объем фигур из таблицы.

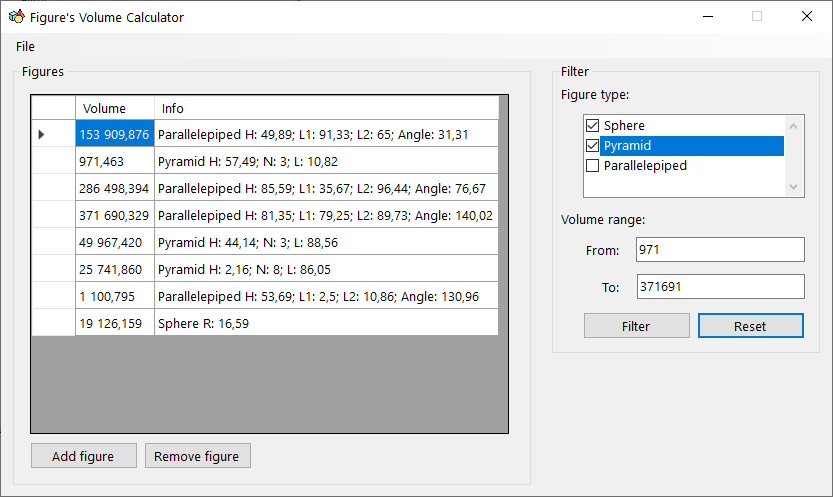


Рисунок 16 – Сброс фильтра

**1.5.4 Тестовый случай «Сохранить данные»**

Для сохранения данных из таблицы необходимо в главном меню перейти к «File» → «Save» (рисунок 17). Откроется системный диалог сохранения файла, где пользователь выбирает директорию и указывает имя файла (рисунок 18).

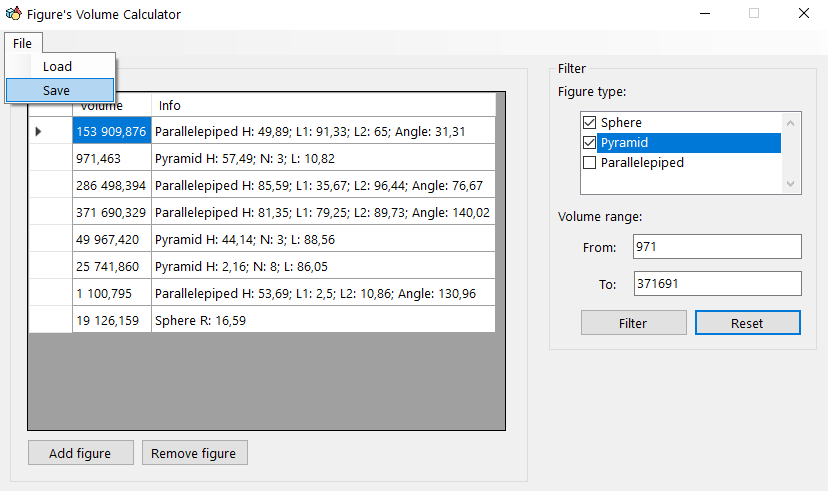


Рисунок 17 – Вызов контекстного меню «File» → «Save»

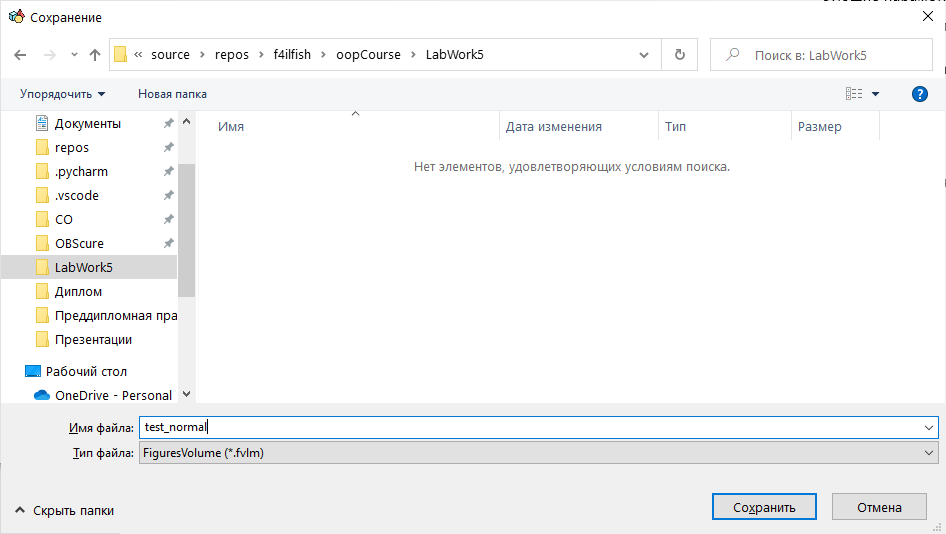


Рисунок 18 – Системный диалог сохранения файла

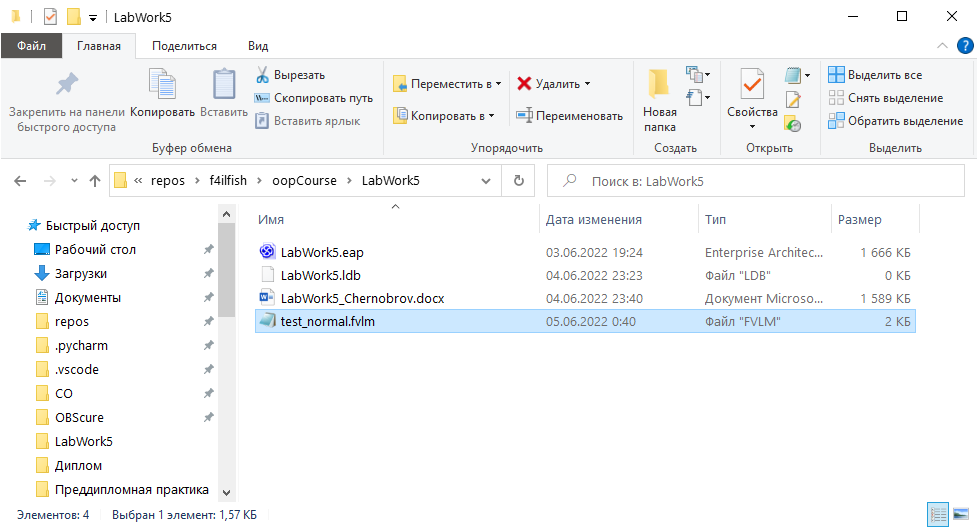


Рисунок 19 – Результат сохранения файла

Также функция позволяет осуществлять сохранение содержимое предварительно отфильтрованной таблицы.

**1.5.5 Тестовый случай «Загрузить данные»**

Для загрузки данных в таблицу необходимо в главном меню перейти к «File» → «Load» (рисунок 20). Откроется системный диалог загрузки файла, где пользователь выбирает директорию и файла (рисунок 21).

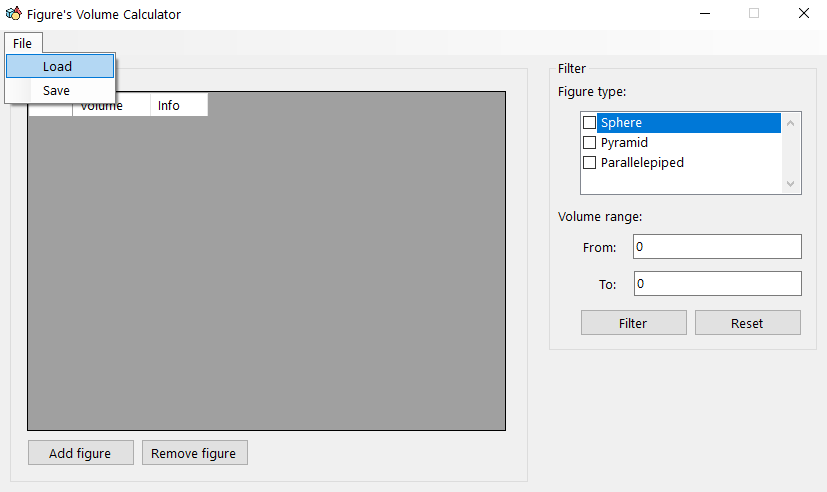


Рисунок 20 – Вызов контекстного меню «File» → «Load»

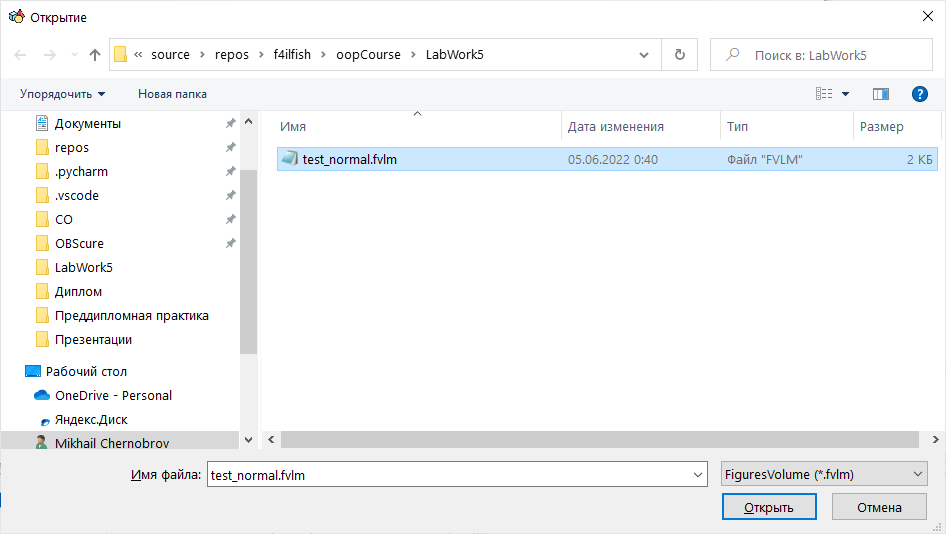


Рисунок 21 – Системный диалог загрузки файла

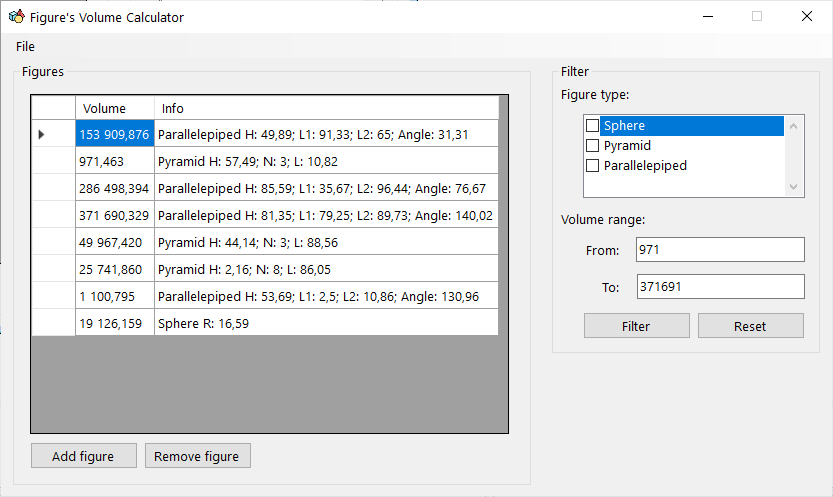


Рисунок 22 – Результат загрузки файла

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

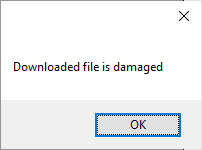


Рисунок 23 – Загрузка повреждённого файла

**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 расчёта объема трехмерных фигур

Разработчик: студент гр. О-5КМ01 НИ ТПУ Чернобров М.Е.

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

Томск 2022

**1 Общие сведения**

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

Полное наименование: «Программа для расчёта трехмерных фигур».

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-5КМ01 НИ ТПУ Чернобров М.Е.

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

Начало работ: 22 марта 2022 г.

Окончание работ: 9 июня 2022 г.

**2 Назначение и цели создания системы**

**2.1 Назначение системы**

Система предназначена для расчёта объёмов трёхмерных фигур: сферы, пирамиды, параллелепипеда.

**2.2 Цели создания системы**

Система создаётся в учебных целях для сокращения трудозатрат при расчете объема трехмерных фигур.

**3 Характеристика объектов автоматизации**

Вычисление объема трехмерных фигур может быть применено научно-педагогическим составом при подготовке учебно-методического материала или проверке результатов работ, выполненных обучающимися.

**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 объемных фигур должны храниться в XML-файле с расширением \*.fvlm.

**D01.01.** Формат XML-файла:

<?xml version="1.0"?>

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

<FigureBase xsi:type="Sphere">

<Radius></Radius>

</FigureBase>

<FigureBase xsi:type="Pyramid">

<Height></Height>

<NumberOfCorners></NumberOfCorners>

<LengthOfSide></LengthOfSide>

</FigureBase>

<FigureBase xsi:type="Parallelepiped">

<Height></Height>

<LengthFirstSide></LengthFirstSide>

<LengthSecondSide></LengthSecondSide>

<AngleOfSides></AngleOfSides>

</FigureBase>

</ArrayOfFigureBase>

**4.3 Функциональные требования**

**F01.** Система должна рассчитывать объем для следующих типов фигур:

* сфера;
* пирамида;
* параллелепипед.

**F01.01.** Объём сферы должен определяться по выражению:



где *R* – радиус сферы;

**F01.02.** Объём пирамида должен определяться по выражению:



где *S* – площадь основания пирамиды;

*h* – высота пирамида.



где *N* – количество углов основания пирамиды;

*a* – длина стороны основания пирамиды.

**F01.03.** Объём параллелепипеда должен определяться по выражению:



где *S* – площадь основания параллелепипеда;

*h* – высота параллелепипеда.



где *a* – длина первой стороны основания параллелепипеда;

*b* – длина второй стороны основания параллелепипеда;

*α* – угол между сторонами основания параллелепипеда, градусы.

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

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

* объем;
* информацию (тип фигуры и значения параметров, на основе которых осуществляется расчет объема);

**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 фильтрации 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 10 Pro 20H1, 20H2, 21H1, 21H2. Работоспособность на других выпусках и версиях не гарантируется.

**C02.** На рабочей станции должен быть установлен .NET Runtime 5.0.17.

**4.6 Требования к аппаратному обеспечению**

**C03.** Для работы системы необходим ПК/ноутбук.

**C04.** Не менее 1 гигагерц (GHz).

**C05.** ОЗУ – не менее 1 ГБ для 32-разрядной системы и 2 ГБ для 64-разрядной.

**C06.** Накопитель – 20 ГБ свободного пространства для 64-разрядных и 16 ГБ свободного пространства для 32-разрядных систем.

**C07.** Разрешение экрана – минимум 800 х 600 пикселей.

**C08.** Видеоадаптер DirectX 9 или более поздняя версия