Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное

учреждение высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра компьютерных систем в управлении и проектировании (КСУП)

ПЛАГИН «ПЕПЕЛЬНИЦА»

ДЛЯ СИСТЕМЫ АВТОМАТИЗИРОВАННОГО ПРОЕКТИРОВАНИЯ

КОМПАС-3D

Пояснительная записка по дисциплине

«Основы разработки САПР»

Студент гр. 589-1

\_\_\_\_\_\_\_\_\_ М. И. Вишняков

\_\_\_\_\_\_\_\_\_

Руководитель

преподаватель каф., к.т.н.

\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_ А.А. Калентьев

(оценка) \_\_\_\_\_\_\_\_\_

Томск 2022

**Реферат**

Учебная работа 35 страниц, 12 таблиц, 21 рисунок, 16 источников.

Ключевые слова: КОМПАС-3D, ПЕПЕЛЬНИЦА, СИСТЕМА АВТОМАТИЗИРОВАННОГО ПРОЕКТИРОВАНИЯ, БИБЛИОТЕКА.

Целью данной работы является разработка библиотеки «Пепельница» для системы автоматизированного проектирования Компас-3D.

В процессе работы должны были пройдены все стадии разработки программного обеспечения: выбор темы, составление технического задания, составление проекта системы, создание первой реализации и её тестирование, создание конечной библиотеки.

Отчет по учебной работе выполнен в текстовом редакторе Microsoft Word 2016.

**Оглавление**

[Введение 4](#_Toc122070126)

[1 ПОСТАНОВКА И АНАЛИЗ ЗАДАЧИ 5](#_Toc122070127)

[1.1 Описание предмета проектирования 5](#_Toc122070128)

[1.2 Описание инструментов и средств реализации 8](#_Toc122070129)

[1.3 Назначение библиотеки 8](#_Toc122070130)

[2 ОБЗОР АНАЛОГОВ 9](#_Toc122070131)

[2.1 Программа построения 3D моделей по заданным значениям в AutoCAD «Лекало» 9](#_Toc122070132)

[2.2 Плагин «Archimesh» для Blender 10](#_Toc122070133)

[3 ОПИСАНИЕ РЕАЛИЗАЦИИ 12](#_Toc122070135)

[4 ОПИСАНИЕ ПРОГРАММЫ ДЛЯ ПОЛЬЗОВАТЕЛЯ 22](#_Toc122070136)

[5 ТЕСТИРОВАНИЕ ПРОГРАММЫ 25](#_Toc122070137)

[5.1 Функциональное тестирование 25](#_Toc122070138)

[5.2 Модульное тестирование 30](#_Toc122070139)

[5.3 Нагрузочное тестирование 31](#_Toc122070140)

[Заключение 34](#_Toc122070141)

[Список использованных источников 35](#_Toc122070142)

# Введение

Автоматизация проектирования имеет огромное значение для развития науки, техники и производства в современном обществе. В настоящее время автоматизация – основной способ повышения производительности и эффективности труда инженерно-технических работников, занимающихся проектированием сложных устройств. Использование автоматизации в проектировании позволяет создавать все более сложные технические объекты и гибко реагировать на появление новых решений и технологий в той или иной области техники.

Практическая реализация методов и идей автоматизированного проектирования происходит в рамках систем автоматизированного проектирования (САПР). В рамках современного «компьютеризированного» общества инженер любой специальности, занимающийся разработкой технических устройств, должен уметь использовать средства автоматизированного проектирования. Это позволяет повысить эффективность проектирования, улучшить его качество, снизить материальные затраты и уменьшить число разработчиков.

Таким образом, целью учебной работы является разработка библиотеки «Пепельница» для системы автоматизированного проектирования Компас-3D. «Компас» – это семейство систем автоматизированного проектирования, универсальная система автоматизированного проектирования, позволяющая в оперативном режиме выпускать чертежи изделий, схемы, спецификации, таблицы, инструкции, расчётно-пояснительные записки, технические условия, текстовые и прочие документы [1].

# 1 ПОСТАНОВКА И АНАЛИЗ ЗАДАЧИ

В рамках учебной дисциплины «Основы разработки САПР» требовалось разработать библиотеку в соответствии с техническим заданием. На основе заданных параметров библиотека, взаимодействуя с САПР «Компас-3D», должна строить трёхмерную модель Пепельницы [2]. Также библиотека должна позволять изменять входные параметры пепельницы.

Изменяемые параметры:

* диаметр внешней верхней части пепельницы;
* высота пепельницы;
* диаметр дна снизу;
* диаметр дна сверху;
* толщина стенок пепельницы;
* толщина дна;

# 1.1 Описание предмета проектирования

Предметом проектирования является пепельница. На рисунке 1.1 представлен чертеж пепельницы.



Рисунок 1.1 – чертеж пепельницы

На рисунке 1.2 представлена 3D-модель пепельницы



Рисунок 1.2 – 3D-модель пепельницы

Параметры звёздного истребителя:

* A - Диаметр внешней верхней части пепельницы (мин. 80 мм, макс. 100 мм);
* B - Высота пепельницы (мин. 35 мм, макс. 50 мм) ;
* C - Диаметр дна снизу (мин. 50 мм, макс. 70 мм). Диаметр дна сверху (D) больше диаметра дна снизу на 30 мм;
* D - Диаметр дна сверху (мин. 80 мм, макс. 100 мм). Отношение диаметра внешней верхней части пепельницы (A) к диаметру дна сверху (E) равно 1 к 1;
* E - Толщина стенок пепельницы (мин. 5 мм, макс. 7 мм);
* F - Толщина дна (мин. 7 мм, макс. 10 мм). Высота пепельницы (B) относится к толщине дна (D) как 5 к 1;

# 1.2 Описание инструментов и средств реализации

Для создания библиотеки используется среда разработки Visual Studio 2019 [3]. Библиотека написана с пользовательским интерфейсом на WindowsForms с использованием .NET Framework 4.7.2 [4]. Инструментом для тестирования является библиотеки NUnit 3.13.3 [5] и NUnit3TestAdapter 4.2.1 [5]. В качестве системы автоматизированного проектирования выбран «Компас-3D» v.20 [1].

# 1.3 Назначение библиотеки

Программа предназначена для автоматизации моделирования детали «Пепельница».

Плагин позволяет пользователю ввести вышеперечисленные значения через графический интерфейс. В программе предусмотрена проверка корректности введенных данных и сообщение пользователю о неправильно заполненных полях с помощью цветового выделения и всплывающих подсказок.

При запуске моделирования с некорректными значениями программа выводит сообщение об ошибке и отменяет построение модели.

При правильно введенных значениях результатом работы программы будет созданная по ним модель пепельницы. Размеры всех параметров, необходимых при построении, рассчитываются автоматически.

# 2 ОБЗОР АНАЛОГОВ

2.1 Программа построения 3D моделей по заданным значениям в AutoCAD «Лекало»

Данная программа позволяет создавать следующие 3D модели в AutoCAD посредством ввода размеров с клавиатуры [6]:

* металлопрокат;
* механические соединения;
* механические передачи;
* элементы гидро- и пнемвоприводов;
* построение конструктивных элементов.

На рисунке 2.1 представлен пользовательский интерфейс программы «Лекало» для построения втулки.



Рисунок 2.1 – Пользовательский интерфейс программы «Лекало» для построения втулки

# 2.2 Плагин «Archimesh» для Blender

С помощью данного плагина можно создавать различные поверхности для создания комнаты, а также, что будет рассмотрено в данной работе, различные предметы интерьера [7]. Работа данного плагина будет показана на примере лампы. Меню различных предметов, добавляемых Archimesh в блендер представлено на рисунке 2.2.



Рисунок 2.2 – Меню предметов

Концепция плагина во многом похожа с планируемым приложением. Есть набор параметров, представленных на рисунке 1.2 – размеров примитивов, из которых состоит модель. Если будут меняться параметры, то будет меняться и готовый объект. Изменения объекта в зависимости от параметров представлено на рисунках 2.3 и 2.4.



Рисунок 2.3 – Изменяемые параметры лампы

# 

Рисунок 2.4 – Сгенерированная лампа

# 3 ОПИСАНИЕ РЕАЛИЗАЦИИ

**UML (Unified Modeling Language)** – это система обозначений, которую можно применять для объектно-ориентированного анализа и проектирования. Его можно использовать для визуализации, спецификации, конструирования и документирования программных систем. Словарь UML включает три вида строительных блоков: диаграммы, сущности, связи [9].

**Sparx Systems Enterprise Architect** – это инструмент визуального моделирования и проектирования, основанный на OMG UML. Платформа поддерживает: проектирование и построение программных комплексов; моделирование бизнес-процессов; и моделирование отраслевых доменов. Он используется предприятиями и организациями не только для моделирования архитектуры своих систем, но и для обработки реализации этих моделей на протяжении всего жизненного цикла разработки приложений [10].

Класс Parameter предназначен для описания любого параметра пепельницы и его валидацию, класс Ashtray предназначен для объединения всех параметров воедино, класс AshtrayBuilder предназначаен для построения пепельницы, класс MainForm предназначен для описания пользовательского интерфейса, перечисление ParametersType необходимо для валидации параметров и корректной связи их с MainForm, класс KompasWrapper предназначен для взаимодействия с САПР «Компас-3D», в нём содержатся методы для связи с САПР и рисования в ней. На рисунке 3.1 представлена UML-диаграмма классов.

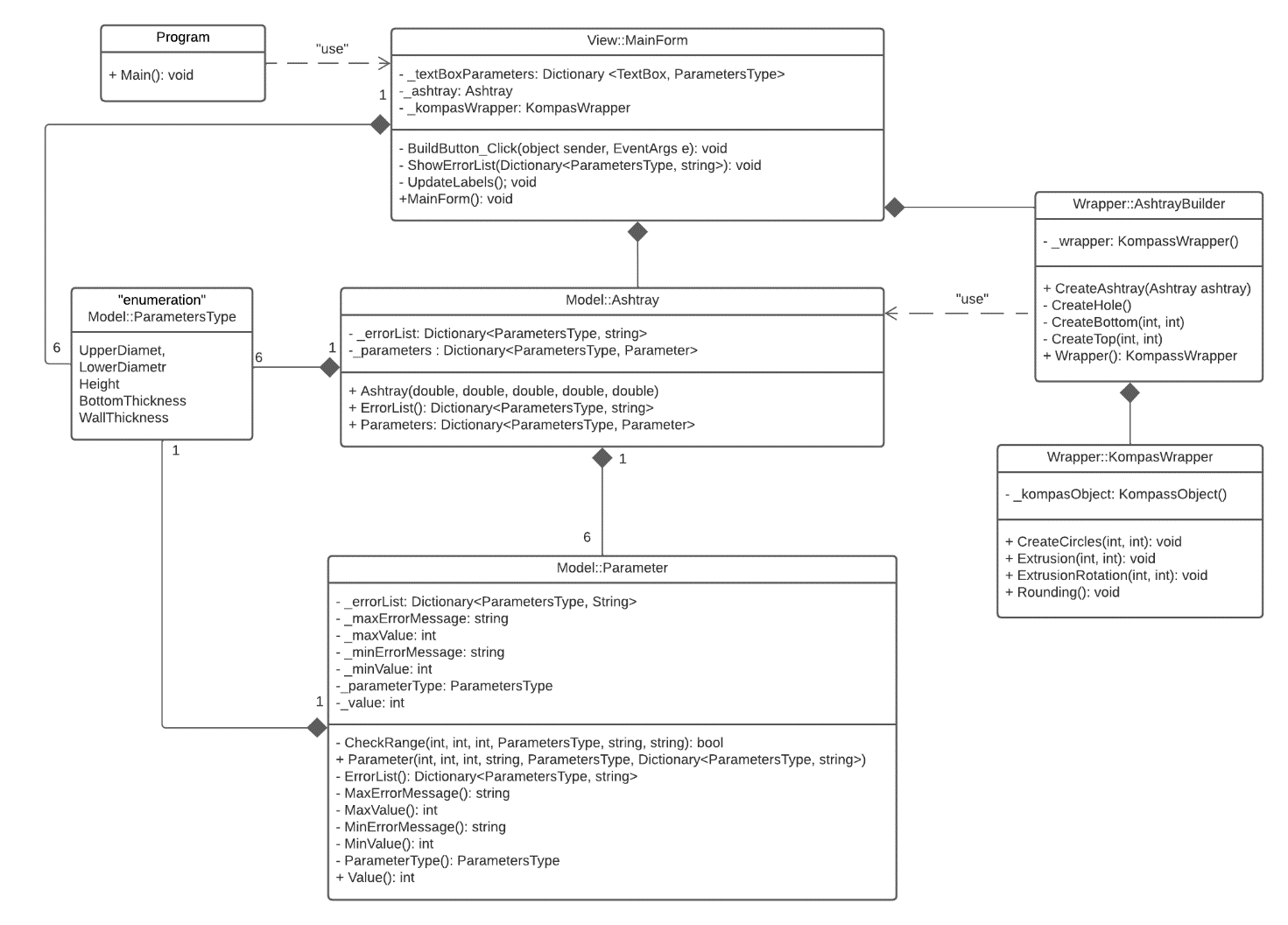


Рисунок 3.1 - UML-диаграмма классов.

Таблица 3.1 – Класс Parameter

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Название** | **Тип** | **Описание** |
| **Поля** | \_value | int | Значение параметра. |
| \_minValue | int | Минимальное допустимое значение параметра. |
| \_maxValue | int | Максимальное допустимое значение параметра. |
| \_minErrorMessage | string | Сообщение о несоблюдении границы минимума. |
| \_maxErrorMessage | string | Сообщение о несоблюдении границы максимума. |
| \_parameterType | ParameterType | Тип параметра. |
| \_errors | Dictionary<  ParameterType, string> | Список ошибок введенного параметра. |

Окончание таблицы 3.1

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Название** | **Тип** | **Описание** |
| **Свойства** | Value | double | Устанавливает и возвращает значение параметра. |
| MinValue | int | Устанавливает и возвращает минимальное допустимое значение параметра. |
| MaxValue | int | Устанавливает и возвращает максимальное допустимое значение параметра. |
| MinErrorMessage | string | Устанавливает и возвращает сообщение о несоблюдении границы минимума. |
| MaxErrorMessage | string | Устанавливает и возвращает сообщение о несоблюдении границы максимума. |
| ParameterType | ParameterType | Устанавливает и возвращает тип параметра. |
| Errors | Dictionary<  ParameterType, string> | Устанавливает и возвращает список ошибок параметра. |
| **Методы** | Parameter | Конструктор | Создает объект класса параметра. |
| CheckRange | bool | Проверка принадлежности диапазону введённого параметра (целое значение). |

Таблица 3.2 – Класс Ashtray

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Название** | **Тип** | **Описание** |
| **Поля** | \_parameters | Dictionary <  ParameterType, Parameter> | Словарь, где ключ: тип параметра пепельницы из перечисления, значение: соответствующий параметр. |

Окончание таблицы 3.2

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Название** | **Тип** | **Описание** |
| **Поля** | \_errorList | Dictionary <  ParameterType, string> | Список ошибок введенного параметра. |
| **Свойства** | Parameters | Dictionary <  ParameterType, Parameter> | Устанавливает и возвращает словарь типов параметров - параметров. |
| Errors | Dictionary< ParameterType, string> | Устанавливает и возвращает список ошибок параметра. |
| **Методы** | AshtrayParameters | Конструктор | Создаёт объект класса пепельницы для построения. |

Таблица 3.3 – Перечисление ParameterType

|  |  |
| --- | --- |
| **Название элемента** | **Описание** |
| UpperDiameter | Верхний диаметр пепельницы. |
| LowerDiameter | Нижний диаметр пепельницы. |
| Height | Высота пепельницы. |
| BottomThickness | Толщина дна пепельницы. |
| WallThickness | Толщина стенок пепельницы. |

Таблица 3.4 – Класс KompasWrapper

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Название** | **Тип** | **Описание** |
| **Поля** | \_kompasObject | KompasObject | Объект Компас API. |
| **Методы** | CreateCircles | void | Создание эскиза. |
| Extrusion | void | Метод выдавливания. |
| ExtrusionRotation | void | Метод выдавливания вращением. |
| Rounding | void | Создание скругления. |

Таблица 3.5 – Класс AshtrayBuilder

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Название** | **Тип** | **Описание** |
| **Поля** | \_wrapper | KompassWrapper | Связь с Компас-3D. |
| **Свойства** | Wrapper | Wrapper | Устанавливает и возвращает связь с Компас-3D. |

Окончание таблицы 3.5

|  |  |  |  |
| --- | --- | --- | --- |
| **Методы** | CreateAshtray | void | Построение детали по заданным параметрам. |
| CreateHole | void | Создание отверстий. |
| CreateBottom | void | Построение нижней части пепельницы. |
| CreateTop | void | Построение верхней части пепельницы. |

Таблица 3.6 – Класс MainForm

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Название** | **Тип** | **Описание** |
| **Поля** | \_ashtrayBuilder | AshtrayBuilder | Объект класса построения детали. |
| \_parameterToTextBox | Dictionary <  ParameterType, string> | Словарь, где ключ: параметр пепельницы, значение: соответствующий тексбокс. |
| \_ashtrayParameters | AshtrayParameters | Объект параметров пепельницы. |
| **Методы** | MainForm | Конструктор | Конструктор основной формы. |
| BuildButton\_Click | void | Построение по введённым параметрам пепельницы. |
| ShowErrorList | void | Демонстрация неправильно введённых параметров. |
| UpdateLabels | void | Обновление диапазонов для параметров пепельницы. |

В процессе реализации библиотеки и её тестирования были изменены поля классов AshtrayForm, AshtrayBuilder, KompasWrapper, Parameter – теперь все приватные поля не имеют свойств. Классу Parameter был добавлен интерфейс IEquatable для возможности сравнивать объекты этого класса.

В KompasWrapper были добавлены новые методы, в том числе для дополнительной функциональности. В главной форме изменились методы для проверки ошибок введённых значений в режиме реального времени.

В таблицах 3.7-3.12 представлены перечисление ParameterType и основные классы Parameter, AshtrayParameters, KompasWrapper, AshtrayBuilder и MainForm, их поля, методы и свойства в итоговой реализации.

Таблица 3.7 – Класс Parameter

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Название** | **Тип** | **Описание** |
| **Поля** | \_value | int | Значение параметра. |
| \_minValue | int | Минимальное допустимое значение параметра. |
| \_maxValue | int | Максимальное допустимое значение параметра. |
| \_minErrorMessage | string | Сообщение о несоблюдении границы минимума. |
| \_maxErrorMessage | string | Сообщение о несоблюдении границы максимума. |
| \_parameterType | ParameterType | Тип параметра. |
| \_errors | Dictionary<  ParameterType, string> | Список ошибок введенного параметра. |
| **Свойства** | Value | int | Устанавливает и возвращает значение параметра. |
| **Методы** | Parameter | Конструктор | Создает объект класса параметра. |
| CheckRange | bool | Проверка принадлежности диапазону введённого параметра. |
| Equals | bool | Проверка на равенство объектов класса. |

Таблица 3.8 – Класс AshtrayParameters

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Название** | **Тип** | **Описание** |
| **Свойства** | Parameters | Dictionary <  ParameterType, Parameter> | Устанавливает и возвращает словарь типов параметров - параметров. |

Окончание таблицы 3.8

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Название** | **Тип** | **Описание** |
| **Свойства** | Errors | Dictionary< ParameterType, string> | Устанавливает и возвращает список ошибок параметра. |
| **Методы** | AshtrayParameters | Конструктор | Создаёт объект класса пепельница для построения. |
| CheckParameterEmpty | void | Проверяет является ли параметр пустым. |
| SetParameters | void | Устанавливает необходимые параметры. |
| CheckParametersRelationship | void | Проверка взаимосвязи параметров между собой. |

Таблица 3.9 – Перечисление ParameterType

|  |  |
| --- | --- |
| **Название элемента** | **Описание** |
| UpperDiameter | Верхний диаметр пепельницы. |
| LowerDiameter | Нижний диаметр пепельницы. |
| Height | Высота пепельницы. |
| BottomThickness | Толщина дна пепельницы. |
| WallThickness | Толщина стенок пепельницы. |

Таблица 3.10 – Класс KompasWrapper

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Название** | **Тип** | **Описание** |
| **Поля** | \_kompas | KompasObject | Объект Компас API. |
| \_part | ksPart | Деталь. |
| \_document | ksDocument3D | Документ-модель. |
| \_entitySketch | ksEntity | Эскиз. |
| \_sketchDefinition | ksSketchDefinition | Интерфейс параметров эскиза |
| \_sketchEdit | ksDocument2D | Документ-эскиз. |

Окончание таблицы 3.10

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Название** | **Тип** | **Описание** |
| **Свойства** | KompasObject | KompasObject | Устанавливает и возвращает объект Компас API. |
| Part | ksPart | Устанавливает и возвращает деталь. |
| Document | ksDocument3D | Устанавливает и возвращает документ-модель. |
| EntitySketch | ksEntity | Устанавливает и возвращает эскиз. |
| SketchDefinition | ksSketchDefinition | Устанавливает и возвращает интерфейс параметров эскиза. |
| SketchEdit | ksDocument2D | Устанавливает и возвращает документ-эскиз |
| **Методы** | StartKompas | void | Запуск Компас-3D. |
| CreateFile | void | Создание документа в Компас-3D. |
| CreateSketch | void | Создание эскиза. |
| CreateCurve | void | Создание кривой. |
| ChooseFillet | void | Выбор плоскости для скругления. |
| CreateFillet | void | Создание скругления. |
| CreateHoles | void | Создание отверстий по концентрической сетке. |
| CreateParallelepipeds | void | Построение параллелепипедов. |
| CreateCylinders | void | Построение цилиндров |
| CreateHemispheres | void | Построение полусфер |
| ExtrudeByRotation | ksBaseRotatedDefinition | Выдавливание эскиза вращением |
| ExtrudeSketch | void | Выдавливание эскиза |
| CreateCircularCopy | void | Построение массива по концентрической сетке |

Таблица 3.11 – Класс AshtrayBuilder

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Название** | **Тип** | **Описание** |
| **Методы** | BuildAshtray | void | Построение детали по заданным параметрам. |

Таблица 3.12 – Класс MainForm

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Название** | **Тип** | **Описание** |
| **Поля** | \_ ashtrayBuilder | AshtrayBuilder | Объект класса построения детали. |
| \_parameterToTextBox | Dictionary <  ParameterType, string> | Словарь Тип параметра-TextBox. |
| \_ashtrayParameters | AshtrayParameters | Объект параметров AshtrayParameters. |
| **Методы** | MainForm | Конструктор | Конструктор основной формы. |
| BuildButton\_Click | void | Построение по введённым параметрам звездолёт. |
| BanCharacterInput | void | Запрет ввода символов и больше одной точки в число. |
| FindError | void | Проверка введенных значений в режиме реального времени. |
| CheckEmptyTextBox | bool | Поиск пустых TextBox. |
| UpdateLabels | void | Динамическое обновление диапазона параметров |

Итоговая диаграмма представлена на рисунке 3.6.

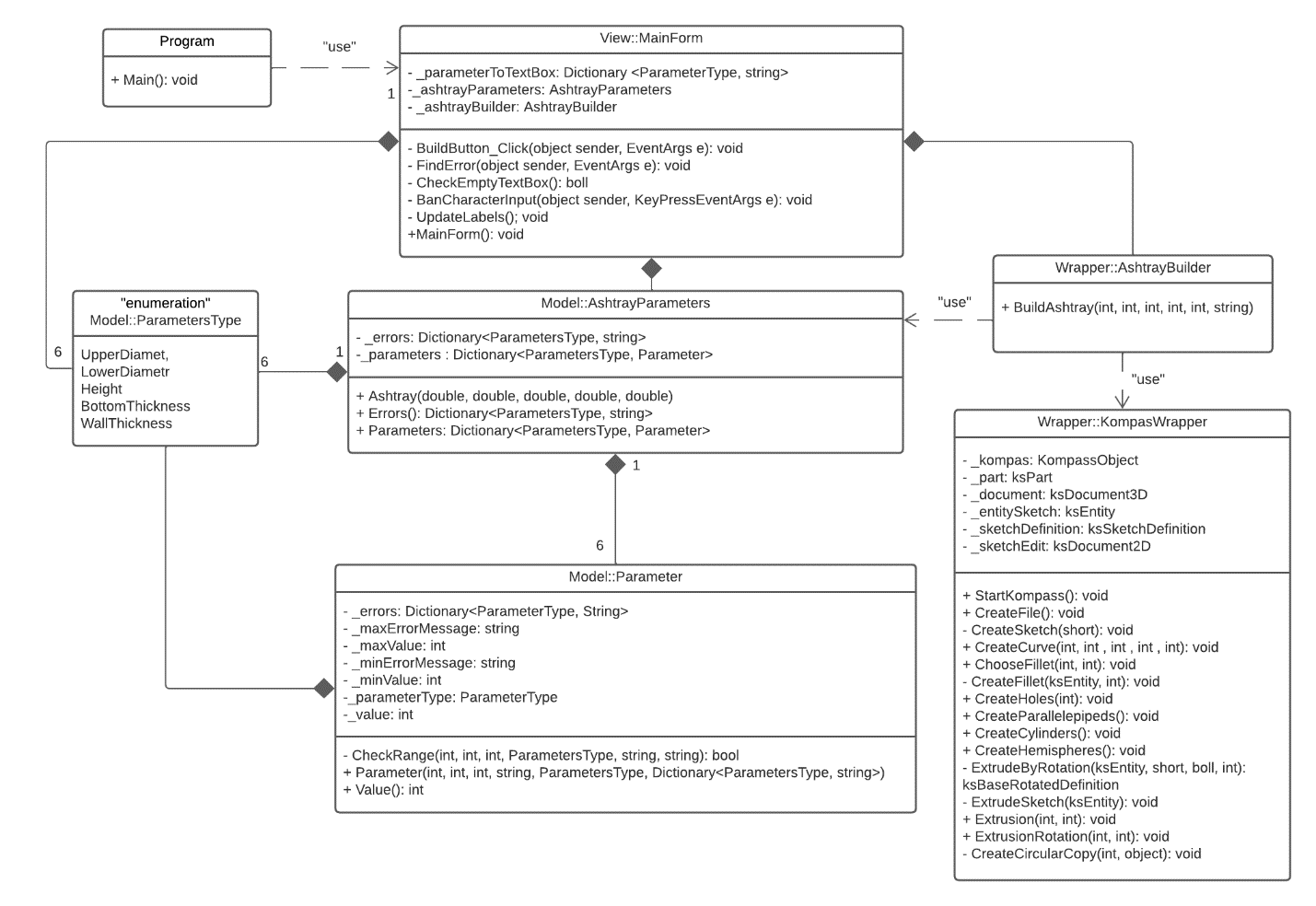


Рисунок 3.6 – Итоговая диаграмма классов

# 4 ОПИСАНИЕ ПРОГРАММЫ ДЛЯ ПОЛЬЗОВАТЕЛЯ

Пользовательский интерфейс (UI) – интерфейс, обеспечивающий передачу информации между пользователем-человеком и программно-аппаратными компонентами компьютерной системы [11].

Плагин представляет собой пользовательскую форму с полями для ввода соответствующих параметров, форму можно увидеть на рисунке 4.1. Посредством кнопки «Построить» осуществляется запуск САПР «Компас-3D», на рабочей области которой построится трехмерная модель детали по заданным параметрам. Если построение детали выполняется несколько раз, то плагин не запускает несколько копий программы «Компас-3D», а создает в ней новый документ. В случае ввода значений параметров, не входящих в допустимый диапазон, поле для ввода окрашивается в розовый цвет и выводится окно в режиме реального времени. При нажатии на кнопку построить появляется окно, информирующее пользователя о некорректности введенного значения. Диапазон зависимых параметров изменяется динамически. Доступен выбор варианта ножек. Пример неправильного ввода представлен на рисунке 4.2, пример вывода списка ошибок представлен на рисунке 4.3.

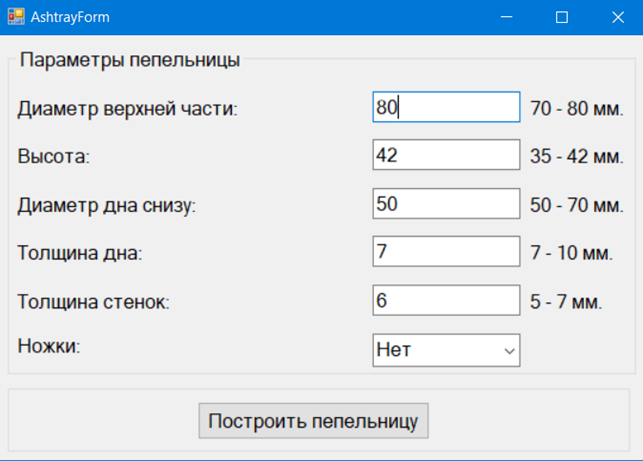


Рисунок 4.1 – Макет пользовательского интерфейса

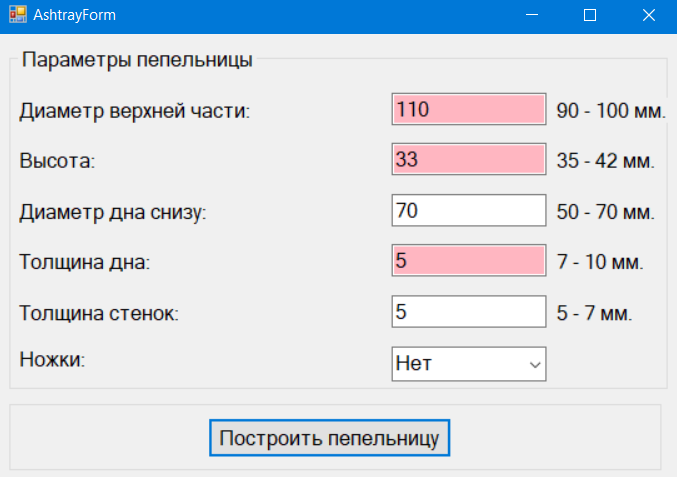


Рисунок 4.2 – Реакция приложения на ввод некорректных значений

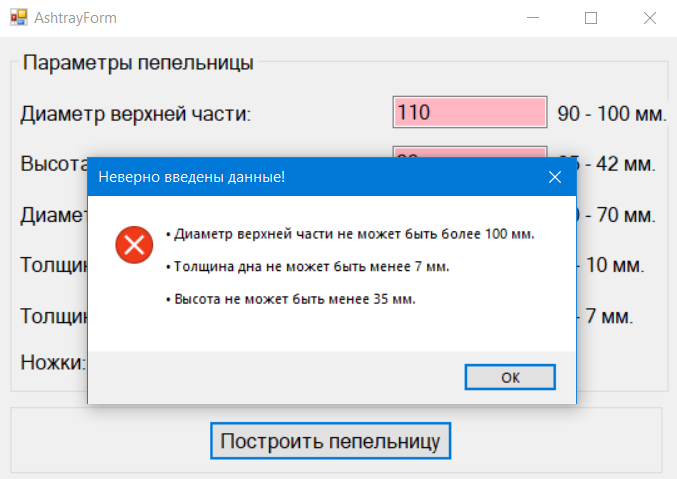


Рисунок 4.3 – Список ошибок на ввод некорректных значений

Если пользователь ввёл правильные значения, при нажатии кнопки «Построить» происходит построение детали. Трёхмерная модель пепельницы, построенная в системе автоматизированного проектирования Компас-3D с помощью разработанной библиотеки представлена на рисунке 4.5

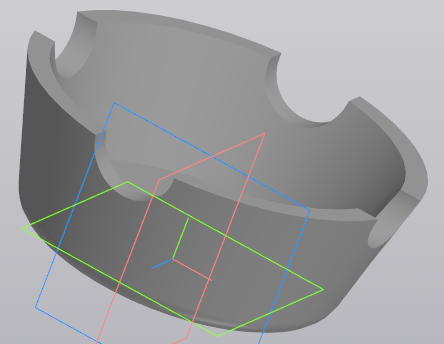


Рисунок 4.5 – Трёхмерная модель пепельницы

# 5 ТЕСТИРОВАНИЕ ПРОГРАММЫ

# 5.1 Функциональное тестирование

Функциональное тестирование – это тип тестирования программного обеспечения, при котором тестовые сценарии выполняются тестировщиком вручную без использования автоматизированных инструментов. Целью ручного тестирования является выявление ошибок, проблем и дефектов в программном приложении. Функциональное тестирование программного обеспечения – это самый примитивный метод из всех видов тестирования. Концепции ручного тестирования не требуют знания какого-либо инструмента тестирования [12].

В рамках функционального тестирования будет проверка библиотеки при введённых минимальных, средних и максимальных параметрах.

Тестирование при минимальных параметрах:

* верхний диаметр равен 70 мм;
* высота равна 35 мм;
* диаметр дна снизу равен 50 мм;
* толщина дна равна 7 мм;
* толщина стенок равна 5 мм;

Результат построения при заданных параметрах представлен на рисунках 5.1-5.2.

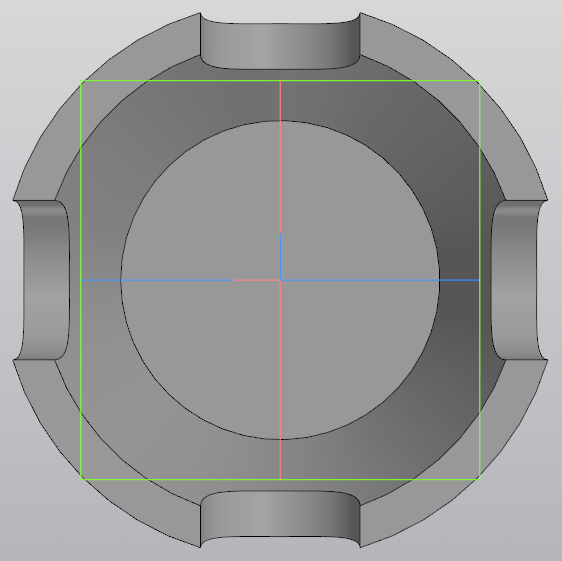


Рисунок 5.1 – Вид на модель сверху

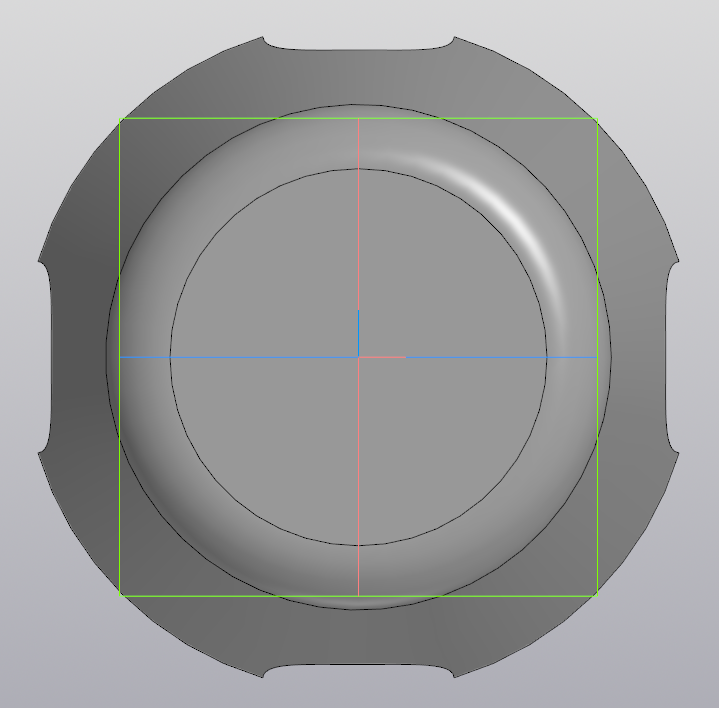


Рисунок 5.2 – Вид на модель снизу

Тестирование при средних параметрах:

* верхний диаметр равен 85 мм;
* высота равна 48 мм;
* диаметр дна снизу равен 60 мм;
* толщина дна равна 8 мм;
* толщина стенок равна 6 мм;

Результат построения при заданных параметрах представлен на рисунках 5.3-5.4.

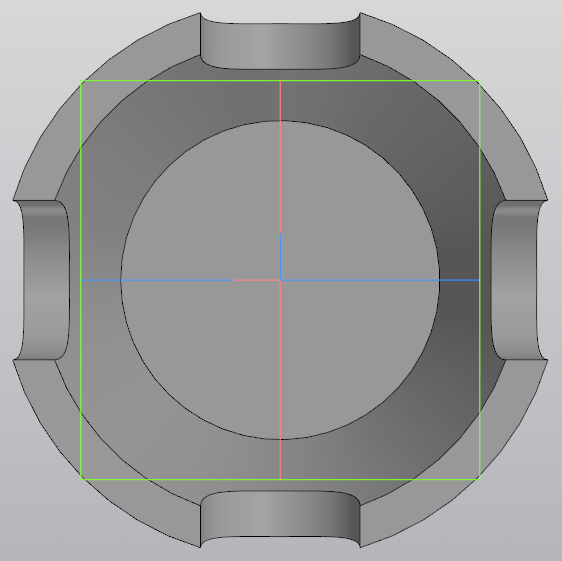


Рисунок 5.3 – Вид на модель сверху

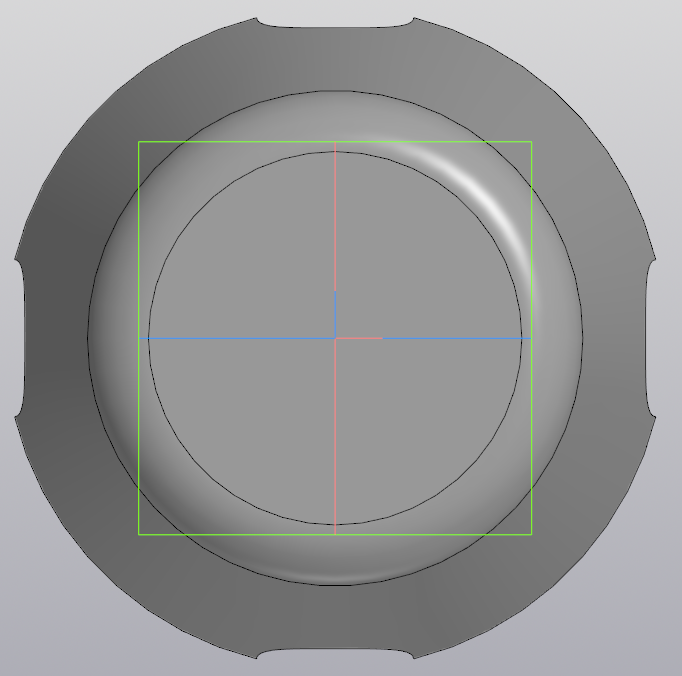


Рисунок 5.4 – Вид на модель снизу

Тестирование при максимальных параметрах:

* верхний диаметр равен 100 мм;
* высота равна 60 мм;
* диаметр дна снизу равен 70 мм;
* толщина дна равна 10 мм;
* толщина стенок равна 7 мм;

Результат построения при заданных параметрах представлен на рисунках 5.5-5.6.

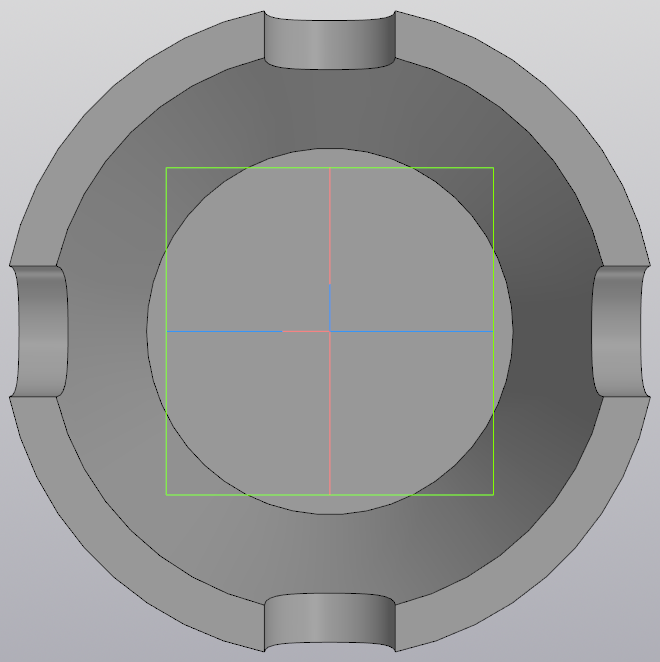


Рисунок 5.5 – Вид на модель в сверху

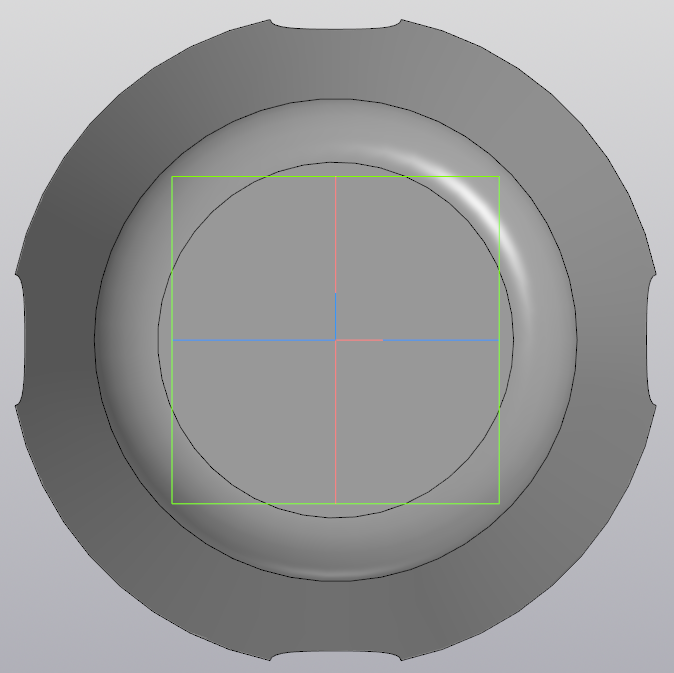


Рисунок 5.6 – Вид на модель снизу

# 5.2 Модульное тестирование

**Модульное тестирование (Unit Testing)**– это тип тестирования программного обеспечения, при котором тестируются отдельные модули или компоненты программного обеспечения. Его цель заключается в том, чтобы проверить, что каждая единица программного кода работает должным образом. Данный вид тестирование выполняется разработчиками на этапе кодирования приложения. Модульные тесты изолируют часть кода и проверяют его работоспособность. Единицей для измерения может служить отдельная функция, метод, процедура, модуль или объект [13].

Для тестирования используется библиотеки NUnit 3.13.3 [5] и NUnit3TestAdapter 4.2.1 [5]. Тестами покрыта вся логика программы. Всего было написано 10 тестов. На рисунке 5.7 представлено тестирование логики.

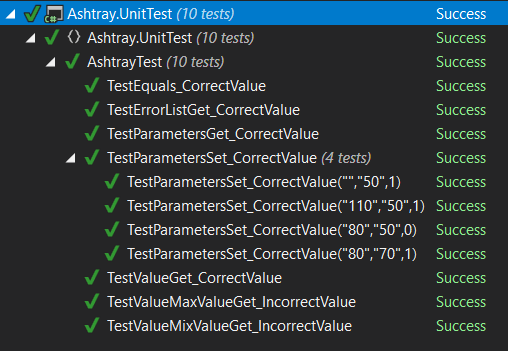


Рисунок 5.7 – Тестирование логики программы

# 5.3 Нагрузочное тестирование

Нагрузочное тестирование – это подвид тестирования производительности, сбор показателей и определение производительности и времени отклика программно-технической системы или устройства в ответ на внешний запрос с целью установления соответствия требованиям, предъявляемым к данной системе (устройству) [14].

Нагрузочное тестирование проводилось на персональном компьютере со следующей конфигурацией:

* процессор Intel(R) Core(TM) i5-8250U CPU @ 1.60GHz;
* оперативная память 8,0 ГБ;
* видеокарта NVIDIA GeForce MX130.

Для тестирования будет зациклено построение модели со следующими параметрами:

* верхний диаметр равен 85 мм;
* высота равна 48 мм;
* диаметр дна снизу равен 60 мм;
* толщина дна равна 8 мм;
* толщина стенок равна 6 мм;

В результате тестирования было построено 36 деталей. По полученным измерениям был рассчитан средний результат, по которому были построены графики. На рисунке 5.13 показана зависимость количества используемой оперативной памяти в гигабайтах от количества построенных деталей.



Рисунок 5.13 – Зависимость оперативной памяти от количества деталей

Из графика видно, что зависимость является линейной пока оперативная память на загружена практически полностью. На графики имеются скачки вниз, то есть происходит уменьшение нагрузки на оперативную память. Это обусловлено устройством оперативной памяти, что для её регенерации периодически приостанавливается обращение, это снижает среднюю скорость обмена и понижает нагрузку [12]. Также в середине графика видно, что нагрузка на память не увеличивается. Это связано с тем, что операционная система имеет файл подкачки, в который выгружаются неактивные и неиспользуемые данные, снимая нагрузку с оперативной памяти [13].

На рисунке 5.14 показана зависимость потраченного времени от количества построенных деталей.



Рисунок 5.14 – Зависимость времени от количества построенных деталей

Из графика видно, что зависимость количества необходимого времени от числа деталей является линейной с изменением угла наклона в течение теста. Это связано с тем, что при большой загрузке оперативной памяти и центрального процессора падает скорость построения одной детали. Поэтому увеличивается коэффициент наклона прямой.

# Заключение

В результате выполнения данной учебной работы были пройдены все стадии разработки программного обеспечения: выбор темы, составление технического задания, составление проекта системы, создание первой реализации и её тестирование, создание конечной библиотеки. Был изучен API приложения Компас-3D. Также были найдены аналоги разрабатываемого плагина. Были спроектированы UML диаграммы классов и было проведено функциональное, модульное и нагрузочное тестирования созданной библиотеки.

# Список использованных источников

1. Официальный сайт системы автоматизированного проектирования «Компас-3D» [Электронный ресурс] – Режим доступа: https://kompas.ru/kompas-3d/about/ (дата обращения: 30.09.2022)
2. Компас (САПР) [Электронный ресурс] – Режим доступа: https://ru.wikipedia.org/wiki/Компас\_(САПР) (дата обращения: 30.09.2022)
3. Заметки о выпуске Visual Studio 2019 версии 16.10 2019 [Электронный ресурс] – Режим доступа: https://learn.microsoft.com/ru-ru/visualstudio/releases/2019/release-notes-v16.10 (дата обращения: 19.11.2022)
4. Платформа .NET Framework 4.7.2 для Windows [Электронный ресурс] – Режим доступа: https: //support.microsoft.com/ru-ru/topic/автономный-установщик-microsoft-платформа-net-framework-4-7-2-для-windows-05a72734-2127-a15d-50cf-daf56d5faec2 (дата обращения: 19.11.2022)
5. Фреймворк NUnit [Электронный ресурс] – Режим доступа: https://nunit.org/ (дата обращения: 19.11.2022)
6. Программа автоматического построения 3D моделей и разверток по заданным значениям в AutoCAD «Лекало». Расчет и построение механических передач [Электронный ресурс] – Режим доступа: https://www.2d-3d.ru/3d-galereia/autocad/811-programma-dlya-autocad-lekalo.html (дата обращения: 07.10.2022)
7. Blender manual Add-ons // Blender 3.3 manual URL: <https://docs.blender.org/manual/en/latest/editors/preferences/addons.html> (дата обращения: 20.10.2022).
8. UML-диаграммы классов [Электронный ресурс] – Режим доступа: http://www.uml.org/ (дата обращения: 06.10.2022)
9. Enterprise Architect (software) [Электронный ресурс] – Режим доступа: https://sparxsystems.com/ (дата обращения: 06.10.2022)
10. Интерфейс пользователя [Электронный ресурс] – Режим доступа: https://ru.wikipedia.org/wiki/Интерфейс\_пользователя (дата обращения: 15.12.2022)
11. Куликов С. Тестирование программного обеспечения. Базовый курс. 3-е издание, 301 с
12. Оперативная память [Электронный ресурс] – Режим доступа: https://ru.wikipedia.org/wiki/Оперативная\_память (дата обращения: 15.12.2022)
13. Файл подкачки [Электронный ресурс] – Режим доступа: https://www.zeluslugi.ru/info-czentr/it-glossary/term-file-podkachki (дата обращения: 15.12.2022)