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 «ПИВНАЯ КРУЖКА»

ДЛЯ СИСТЕМЫ АВТОМАТИЗИРОВАННОГО ПРОЕКТИРОВАНИЯ

КОМПАС-3D

Пояснительная записка по дисциплине

«Основы разработки САПР»

Студент гр. 589-3

\_\_\_\_\_\_\_\_\_ Дерябин Н.А.

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

Руководитель

преподаватель каф., к.т.н.

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

(оценка) \_\_\_\_\_\_\_\_\_

Томск 2022

**Реферат**

Учебная работа 38 страниц, 9 таблиц, 26 рисунков, 13 источников.

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

Целью данной работы является разработка библиотеки «Пивная кружка» 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 2019.

**Оглавление**

[Введение 4](#_Toc122857626)

[1 ПОСТАНОВКА И АНАЛИЗ ЗАДАЧИ 5](#_Toc122857627)

[1.1 Описание предмета проектирования 5](#_Toc122857628)

[1.2 Описание инструментов и средств реализации 7](#_Toc122857629)

[1.3 Назначение библиотеки 7](#_Toc122857630)

[2 ОБЗОР АНАЛОГОВ 9](#_Toc122857631)

[2.1 Плагин «Archimesh» для Blender 9](#_Toc122857632)

[3 ОПИСАНИЕ РЕАЛИЗАЦИИ 12](#_Toc122857634)

[4 ОПИСАНИЕ ПРОГРАММЫ ДЛЯ ПОЛЬЗОВАТЕЛЯ 21](#_Toc122857635)

[5 ТЕСТИРОВАНИЕ ПРОГРАММЫ 25](#_Toc122857636)

[5.1 Функциональное тестирование 25](#_Toc122857637)

[5.2 Модульное тестирование 31](#_Toc122857638)

[5.3 Нагрузочное тестирование 32](#_Toc122857639)

[Заключение 36](#_Toc122857640)

[Список использованных источников 37](#_Toc122857641)

# 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 «Пивная кружка» 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 пивной кружки [2]. Также библиотека должна позволять изменять входные параметры пепельницы.

Изменяемые параметры:

* Диаметр внешней части горла кружки;
* Толщина стенок кружки;
* Высота от горла кружки до дна;
* Толщина дна;
* Диаметр дна сверху;
* Диаметр дна снизу;

# 1.1 Описание предмета проектирования

Программа предназначена для автоматизации моделирования детали «Пивная кружка»

Плагин позволяет пользователю ввести вышеперечисленные значения через графический интерфейс. В программе предусмотрена проверка корректности введенных данных и сообщение пользователю о неправильно заполненных полях.

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 пивной кружки.

Изображения моделируемого объекта см. рис. 1.1-1.2:

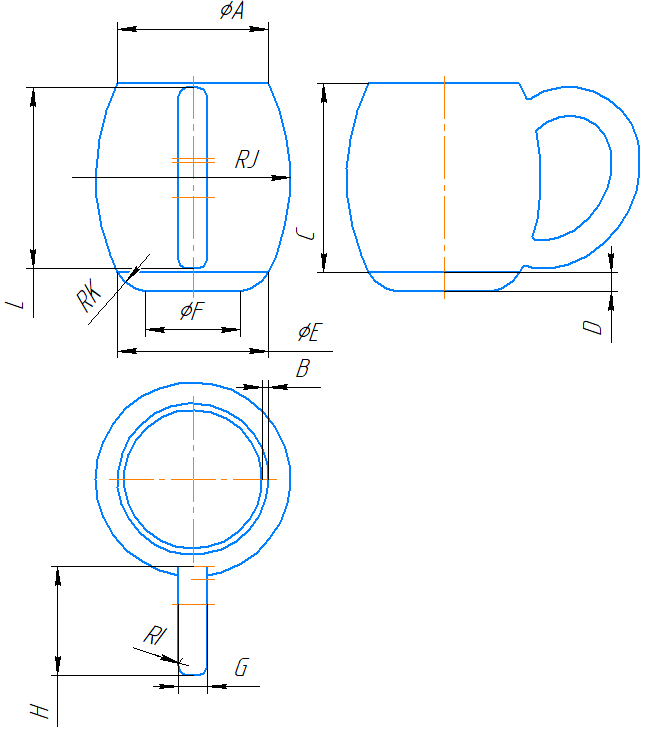


Рисунок 1.1 – Чертёж модели «Пивная кружка»



Рисунок 1.2 – 3D модель пивной кружки

Измеряемые параметры для плагина:

* A – Диаметр внешней части горла кружки (мин. 80 мм, макс. 100 мм);
* B – Толщина стенок кружки (мин. 5 мм, макс. 7 мм);
* C – Высота от горла кружки до дна (мин. 100 мм, макс. 165 мм);
* D – Толщина дна (мин. 10 мм, макс. 16.5 мм). Высота от горла кружки до дна (C) относится к толщине дна (D) как 10 к 1;
* E – Диаметр дна сверху (мин. 80 мм, макс. 100 мм). Отношение диаметра внешней части горла кружки (A) к диаметру дна сверху (E) равно 1 к 1;
* F – Диаметр дна снизу (мин. 50 мм, макс. 70 мм). Диаметр дна сверху (E) больше диаметра дна снизу на 30 мм;

# 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.21 [1].

# 1.3 Назначение библиотеки

Программа предназначена для автоматизации моделирования детали «Пивная кружка»

Плагин позволяет пользователю ввести вышеперечисленные значения через графический интерфейс. В программе предусмотрена проверка корректности введенных данных и сообщение пользователю о неправильно заполненных полях.

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 пивной кружки.

# 2 ОБЗОР АНАЛОГОВ

# 2.1 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 2.1.

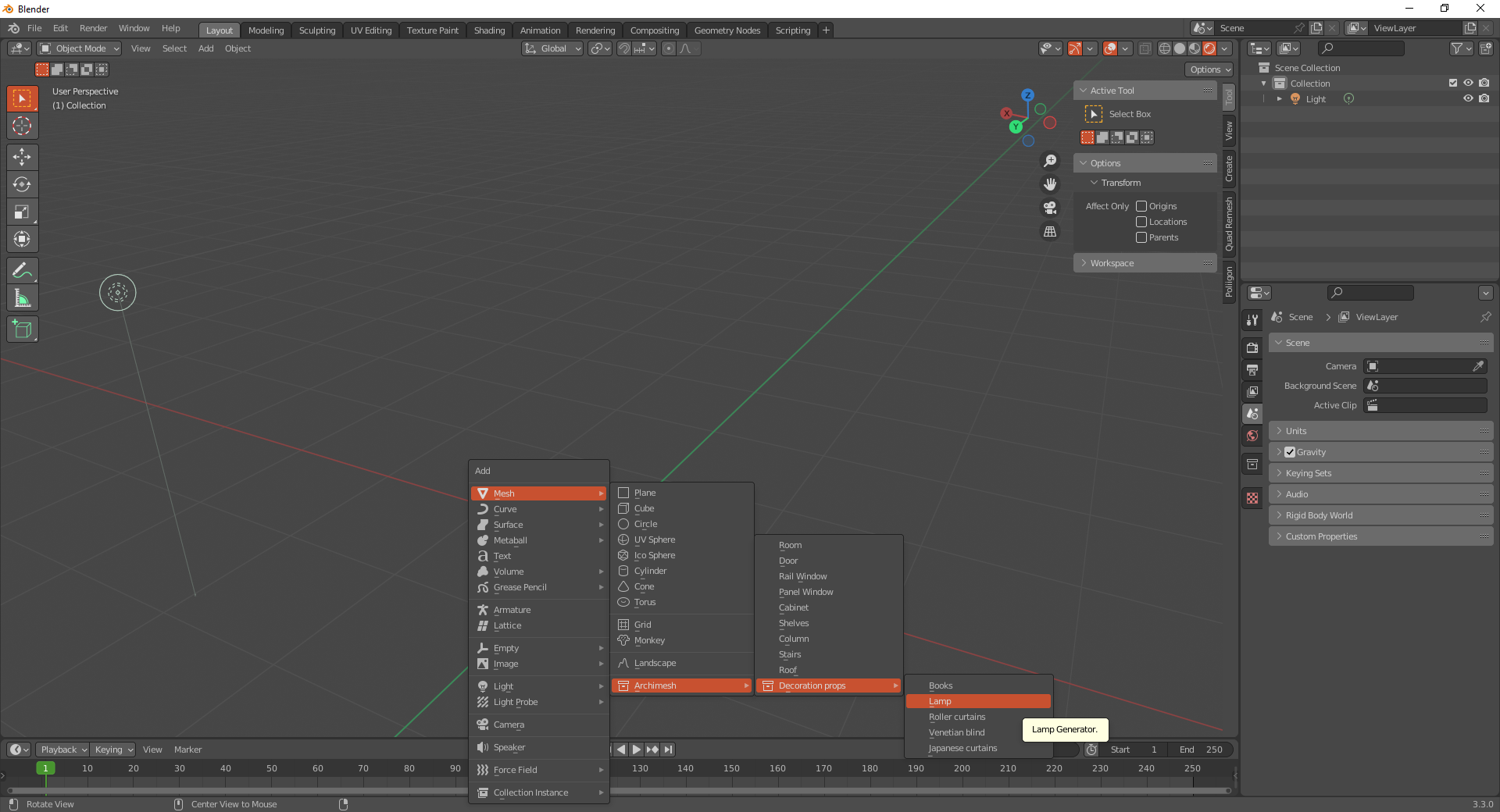


Рисунок 2.1 – Меню предметов

Концепция плагина во многом похожа с планируемым приложением. Есть набор параметров, представленных на рисунке 2.2 – 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.

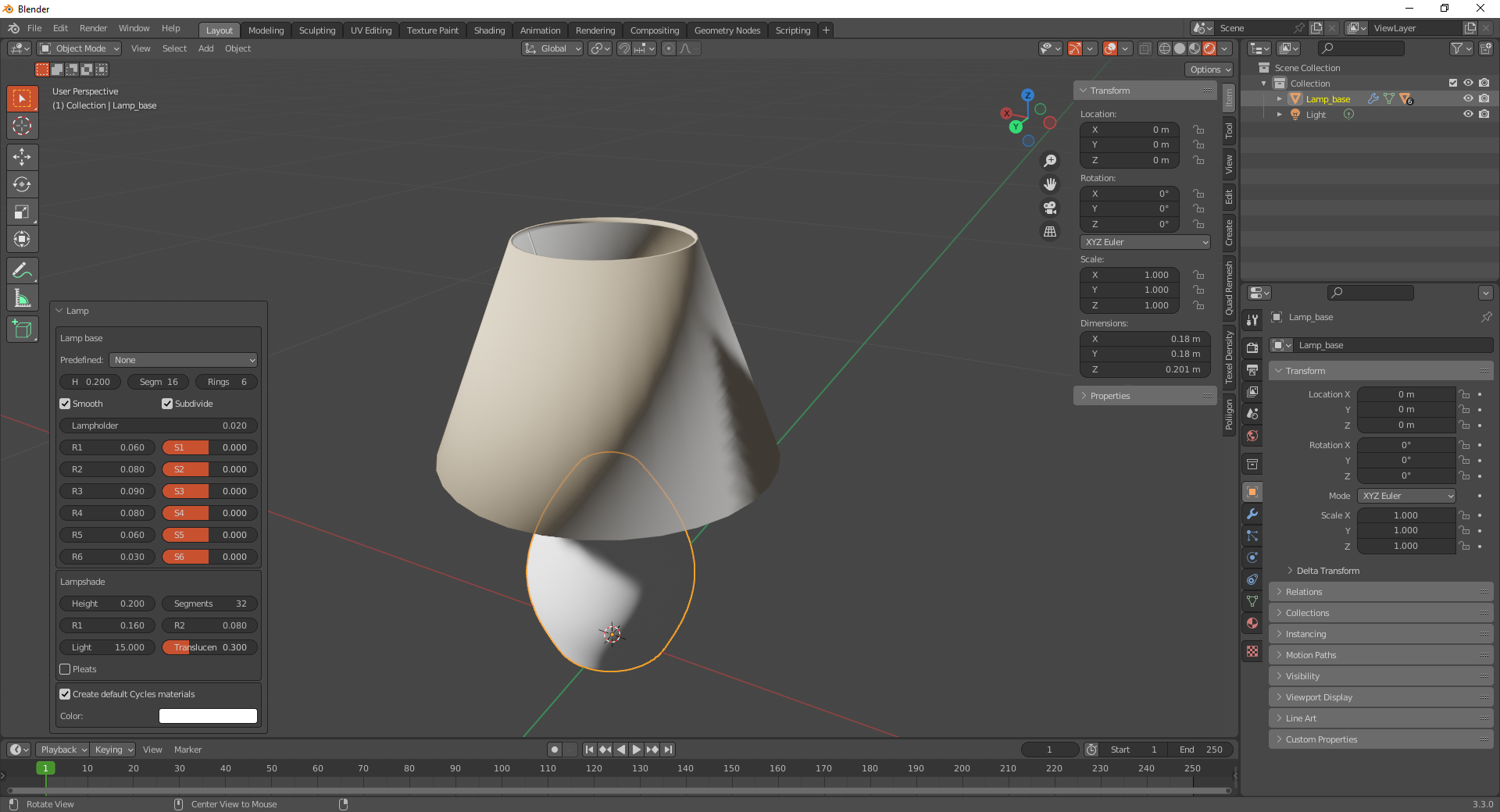


Рисунок 2.2 – Изменяемые параметры лампы

# 

Рисунок 2.3 – Сгенерированная лампа

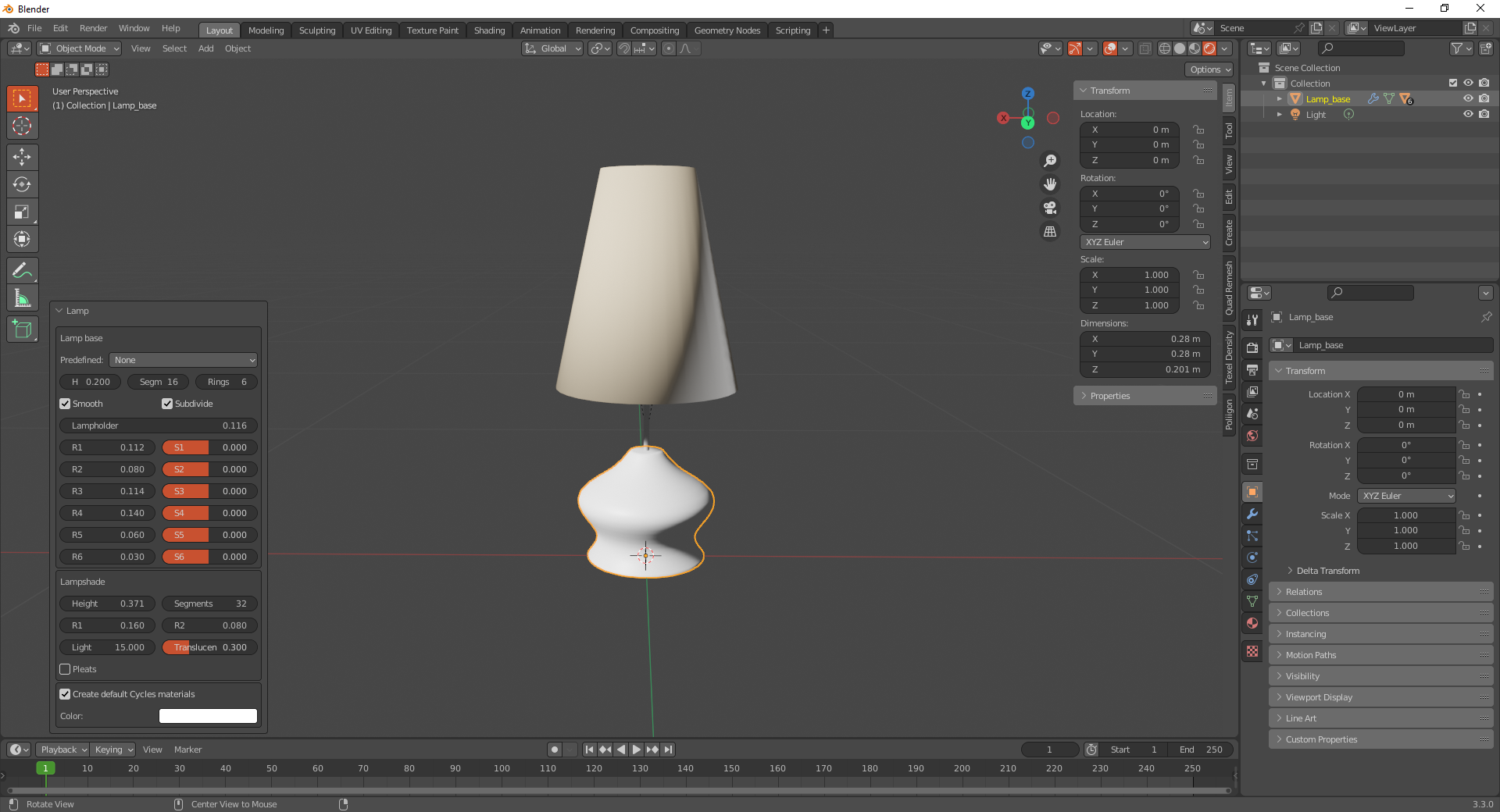


Рисунок 2.4 – Сгенерированная лампа с изменёнными параметрами

# 3 ОПИСАНИЕ РЕАЛИЗАЦИИ

Унифицированный язык моделирования (UML) — это язык моделирования общего назначения, предназначенный для обеспечения стандартного способа визуализации проектирования системы. В 1997 году UML был принят в качестве стандарта Object Management Group (OMG) и с тех пор управляется этой организацией [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 BeerMugParametr предназначен для проверки соответствия параметра минимальным и максимальным значениям, класс BeerMugParametrs предназначен для объединения всех параметров воедино, а также проверки значений и связей. Класс BeerMugBuilder предназначен для построения пивной кружки, класс MainForm предназначен для описания пользовательского интерфейса, перечисление ParametrType необходимо для валидации параметров и корректной связи их с MainForm, класс KompasConnector предназначен для взаимодействия с САПР «Компас-3D», в нём содержатся методы для связи с САПР и рисования в ней. Класс Point2D необходим для валидации параметров точки координат. В классе KompasSketch хранятся все функции для работы с документом «Компас-3D». На рисунке 3.1 представлена UML-диаграмма классов.

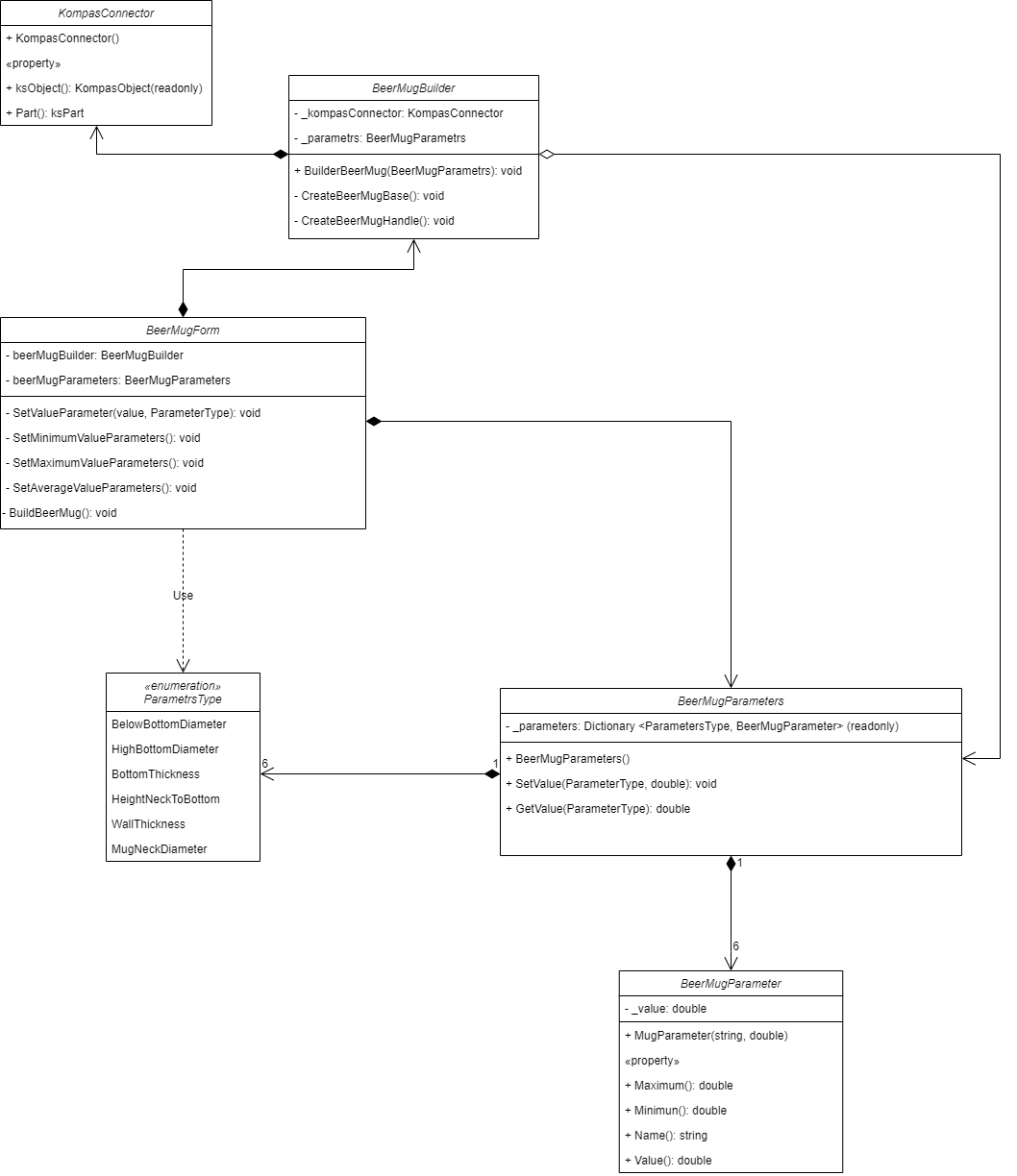


Рисунок 3.1 - UML-диаграмма классов.

Далее в таблицах 3.1 – 3.4 представлено описание классов.

Таблица 3.1 – Описание полей, методов, сущностей класса «BeerMugForm»

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| * beerMugBuilder: BeerMugBuilder |  | Хранит в себе набор методов для построения пивной кружки |
| * beerMugParameters: beerMugParameters |  | Хранит в себе параметры необходимые для построения пивной кружки |
| * SetValueParameter(value, ParameterType) | void | Устанавливает значение параметра |
| - SetMinimumValueParameters() | void | Устанавливает минимальное значение всех параметров |
| * SetMaximumValueParameters() | void | Устанавливает максимальное значение всех параметров |
| - SetAverageValueParameters() | void | Устанавливает среднее значение всех параметров |
| BuildBeerMug() | void | Построение кружки по заданным параметрам |

Таблица 3.2 – Описание полей, методов, сущностей класса «BeerMugParameters»

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| * parameters: Dictionary <ParametersType, BeerMugParameter> |  | Хранит данные о каждом параметре модели из перечисления «BeerMugParameter» |
| + BeerMugParameters() |  | Конструктор для создания экземпляра класса |
| + SetValue(ParameterType, double) | void | Устанавливает значение определённого параметра |
| + GetValue(ParameterType) | double | Возвращает значение определённого параметра |

Таблица 3.3 – Описание полей, методов, сущностей класса «BeerMugParameter»

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| - \_value |  | Поле, хранящее значение |
| + MugParameter(string, double) |  | Конструктор для создания экземпляра класса |
| + Maximum() | double | Возвращает максимально допустимое значения параметра |
| + Minimum() | double | Возвращает минимально допустимое значения параметра |
| + Value() | double | Возвращает и задаёт значение параметра |

Таблица 3.4 – Описание полей, методов, сущностей класса «BeerMugBuilder»

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| * kompasConnector: KompasConnector |  | Хранит в себе методы необходимые для связи с КОМПАС 3D |
| * \_parameters: BeerMugParameters |  | Хранит данные о каждом параметре модели из перечисления «BeerMugParameter» |
| + BuilderBeerMug(BeerMugParameters) | void | Построение пивной кружки по заданным параметрам |
| * CreateBeerMugBase() | void | Построение основы кружки |
| * CreateBeerMugHandle() | void | Построение ручки пивной кружки |

В процессе реализации библиотеки и её тестирования были изменены поля классов MainForm, BeerMugBuilder, KompasConnector, BeerMugParameters, BeerMugParameter

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 BeerMugParameter теперь значения просто проверяются на корректность. В BeerMugParameters хранятся все характеристики кружки, а также находятся сеттеры и геттеры.

В таблицах 3.5-3.9 представлены основные классы BeerMugParameter, BeerMugParameters, KompasConnector, BeerMugBuilder, Point2D и MainForm, их поля, методы и свойства в итоговой реализации.

Таблица 3.5 – Описание новых полей, методов, сущностей класса «BeerMugParameter».

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| + RangeCheck(double, double, double, MugParametersType, Dictionary<MugParametersType, string>) |  | Проверка диапазона входных значений параметров |

Таблица 3.6 – Класс KompasConnector.

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| -KompasObject:\_kompas |  | Объект Компас API. |
| -ksPart: \_part |  | Модель. |
| -ksDocument3D: \_document |  | Документ Компас 3D. |
| -ksEntity: \_entitySketch |  | Эскиз. |
| ksSketchDefinition: |  | Интерфейс параметров эскиза |

Окончание таблицы 3.6.

|  |  |  |
| --- | --- | --- |
| StartKompas() |  | Запуск Компас 3D. |
| Название метода/поля | Возвращаемый тип | Описание |
| CreateDocument() |  | Создание документа в Компас 3D. |
| SetProperties() |  | Установка свойст детали. |
| CreateSketch(int, double) | Скетч. | Создания скетча. |
| Extrude(KompasSketch, double, bool) |  | Выдавливание скетча. |
| CutExtrude(KompasSketch, double, bool) |  | Вырезание выдавливанием. |
| ExtrudeRotation360(KompasSketch) |  | Выдавливание вращением на 360 градусов. |
| ExtrudeRotation180(KompasSketch) |  | Выдавливание вращением на 180 градусов. |
| ExtrudeRotation178(KompasSketch) |  | Выдавливание вращением на 178 градусов. |
| Fillet(double) |  | Скругления на ребрах окружностей. |
| List<ksFaceDefinition> GetCylinderFaces() | Список цилиндрических граней. | Возвращает все цилиндрические грани детали. |
| CutExtrudeRotation(KompasSketch, int) |  | Вырезание выдавливанием вокруг. |

Таблица 3.7 – Класс KompasSketch

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| -ksEntity: Sketch |  | Эскиз. |
| ksSketchDefinition: \_sketchDefinition |  | Интерфейс параметров эскиза. |
| ksDocument2D: \_document2D |  | 2D документ. |

Окончание таблицы 3.7.

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| KompasSketch(ksPart, int, double) |  | Конструктор. |
| EndEdit() |  | Завершить редактирование скетча. |
| CreateCircle(Point2D, double) |  | Создание круга на заданных координатах с заданным радиусом. |
| CreateLineSeg(Point2D, Point2D, int) |  | Создание линии в заданных координатах. |
| ArcBy3Point(Point2D, Point2D, Point2D) |  | Дуга по трём точкам. |
| CreateBezier(Point2D, Point2D) |  | Создание кривых Бизье. |
| ArcByPoint(Point2D, double, Point2D, Point2D) |  | Дуга по точкам и углу. |

Таблица 3.8 – Класс BeerMugBuilder

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| Builder(MugParameters, string) |  | Построение кружки по её параметрам. |
| BuildBottom(double, double, double) |  | Построение основания пивной кружки. |
| BuildBody(double, double, double, double, double) |  | Построение тела пивной кружки. |
| BuildHandle(double, double, double) |  | Создание ручки пивной кружки. |
| BuildHandleRingCap(double, double, double) |  | Построение крышки с ручкой кольцом. |
| BuildCapBase(double, double, double) |  | Построение основы крышки. |

Окончание таблицы 3.8.

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| BuildHandleCircleCap(double, double, double) |  | Построение крышки с ручкой выдавленным кругом. |

Таблица 3.9 – Класс MainForm

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| MugParameters: \_beerMugParametrs |  | Экземпляр класса MugParameters. |
| Color: \_correctColor |  | Цвет корректно заполненного поля. |
| Color: \_incorrectColor |  | Цвет не корректно заполненного поля. |
| Dictionary<TextBox, Action<double>>: \_textBox |  | Словарь, cвязывающий параметр пивной кружки и его текстбокс. |
| TextBoxValidator\_TextChanged(object, EventArgs) |  | Валидация текстбоксов. |
| CheckForCommasAndNumbers\_KeyPress(object, KeyPressEventArgs) |  | Проверка содержания текстбоксом только одной запятой и цифры. |
| IntegerCheck\_KeyPress(object, KeyPressEventArgs) |  | Проверка, чтобы textbox содержал только цифры. |
| MinimumSizeButtom\_Click(object, EventArgs) |  | Обработка нажатия на кнопку Minimum size. |

Окончание таблицы 3.9.

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| MaximumSizeButton\_Click(object, EventArgs) |  | Обработка нажатия на кнопку Maximum size. |
| buildButton\_Click(object, EventArgs) |  | Обработка нажатия на кнопку Build button. |
| MainForm\_Load(object, EventArgs) |  | Ввод значений при загрузке формы. |

Итоговая диаграмма представлена на рисунке 3.2.

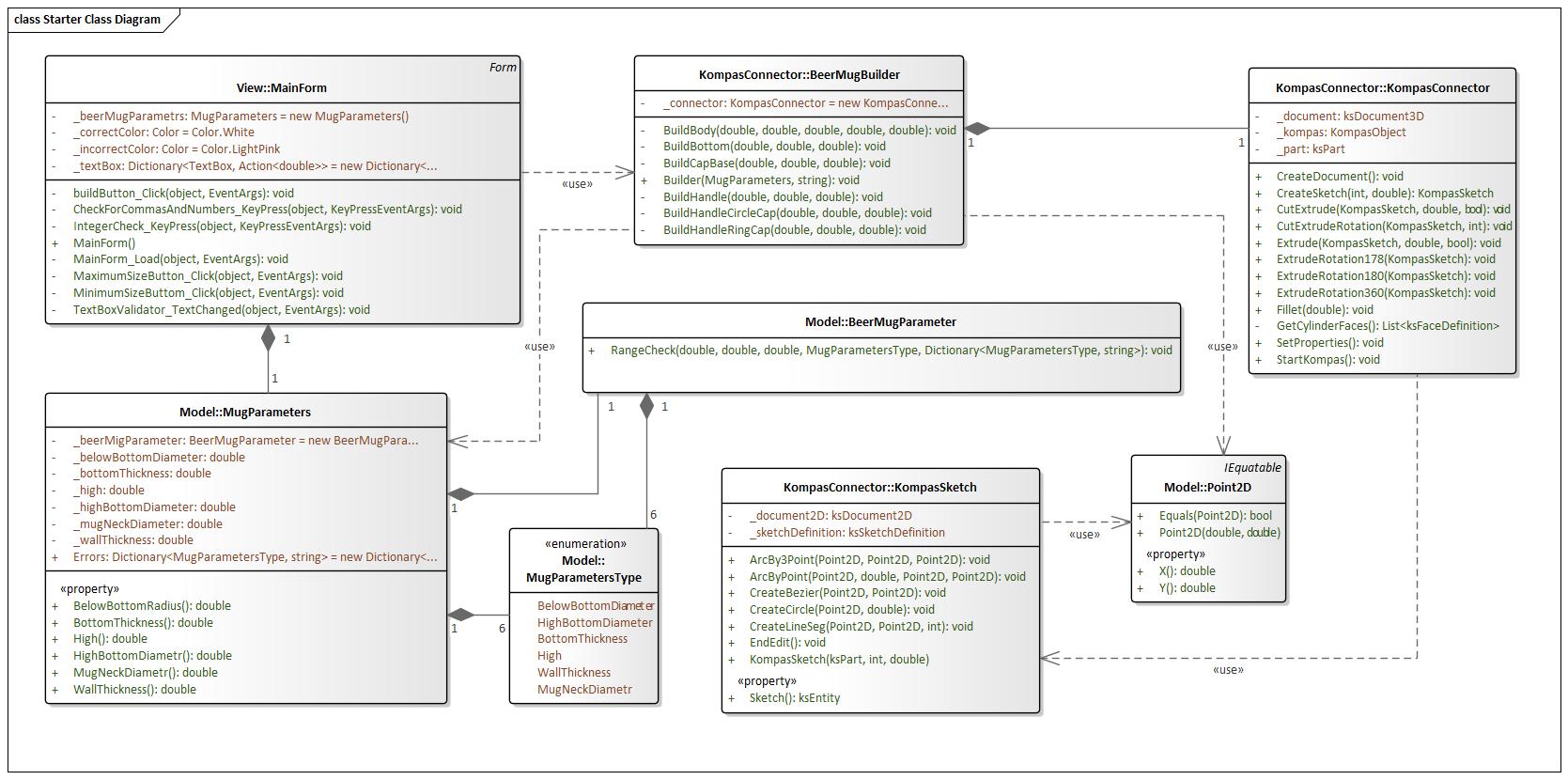


Рисунок 3.2 – Итоговая диаграмма классов

# 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.

С помощью данного окна пользователь может изменять параметры будущей 3D модели пивной кружки, где:

* A – Диаметр внешней части горла кружки (мин. 80 мм, макс. 100 мм);
* B – Толщина стенок кружки (мин. 5 мм, макс. 7 мм);
* C – Высота от горла кружки до дна (мин. 100 мм, макс. 165 мм);
* D – Толщина дна (мин. 10 мм, макс. 16.5 мм). Высота от горла кружки до дна (C) относится к толщине дна (D) как 10 к 1;
* E – Диаметр дна сверху (мин. 80 мм, макс. 100 мм). Отношение диаметра внешней части горла кружки (A) к диаметру дна сверху (E) равно 1 к 1;
* F – Диаметр дна снизу (мин. 50 мм, макс. 70 мм). Диаметр дна сверху (E) больше диаметра дна снизу на 30 мм;
* Cap type – Тип крышки для пивной кружки (дополнительное задание). «Without cap» - крышка построена не будет, «Handle ring» - будет построена крышка у которой ручкой является кольцо, «Handle circle» - будет построена крышка у которой ручкой является вытянутый цилиндр.

Слева от полей ввода находится название компонента, за который поле отвечает, корректные размеры – минимальный и максимальный, а также в случае наличия взаимосвязей указаны соотношения.

При нажатии на кнопку «Minimum size» будет создана 3D модель с минимальными корректными размерами. При нажатии на кнопку «Maximum size» будет создана 3D модель с максимальными корректными размерами. При загрузке формы в поля введены средние значения параметров. При нажатии на кнопку «Build model» будет построена 3D модель по заданным параметрам. Чертёж модели справа необходим для лучшего понимания расположения вводимых размеров.

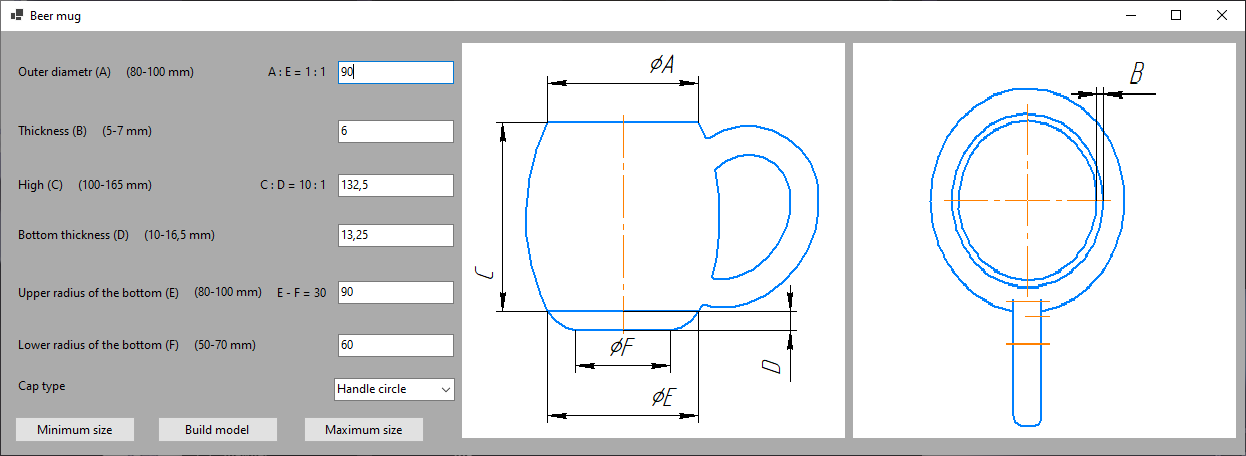


Рисунок 4.1 – Макет пользовательского интерфейса

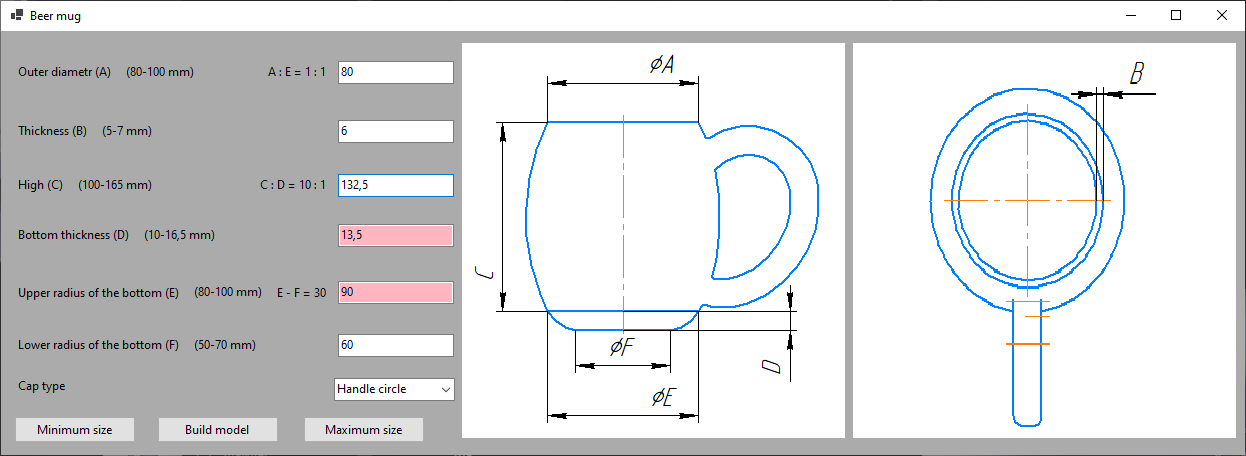


Рисунок 4.2 – Реакция приложения на ввод некорректных значений

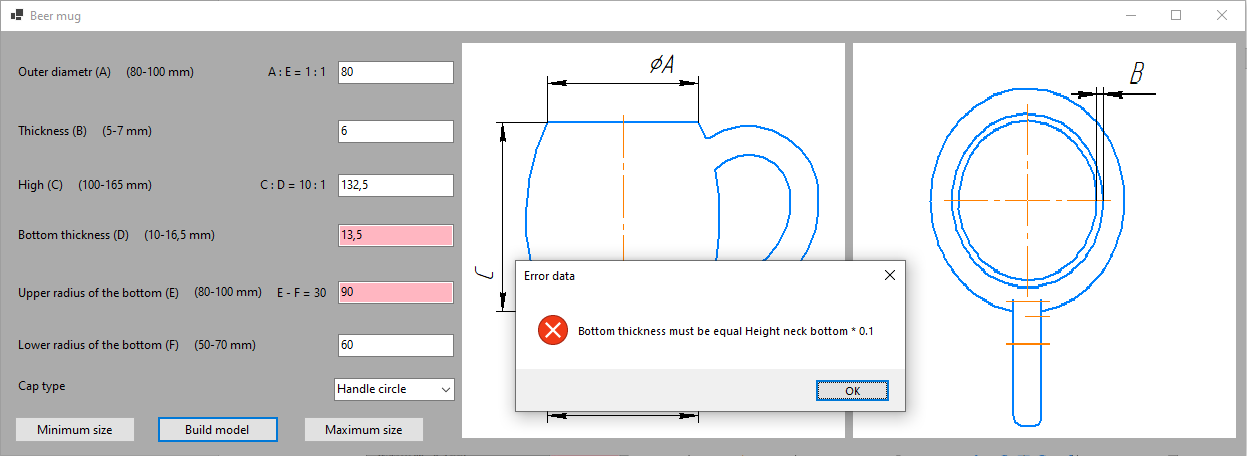


Рисунок 4.3 – Ошибка на ввод некорректных значений

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

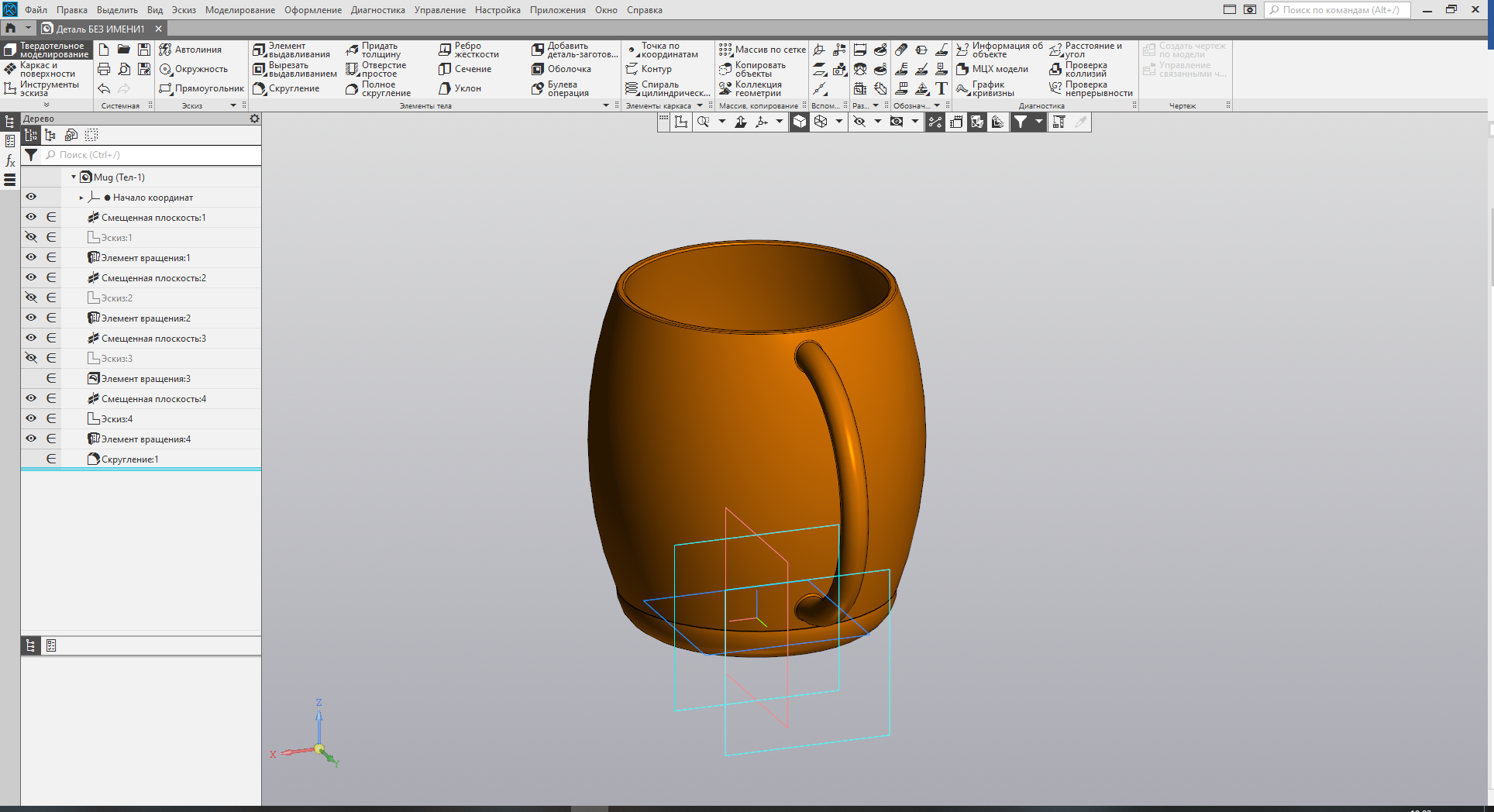


Рисунок 4.5 – Трёхмерная модель пивной кружки

Также была добавлена дополнительная функциональность, а именно 2 типа крышек. Принцип описан выше. На рисунках 4.6 – 4.7 представлен внешний вид пивной кружки с разными крышками.

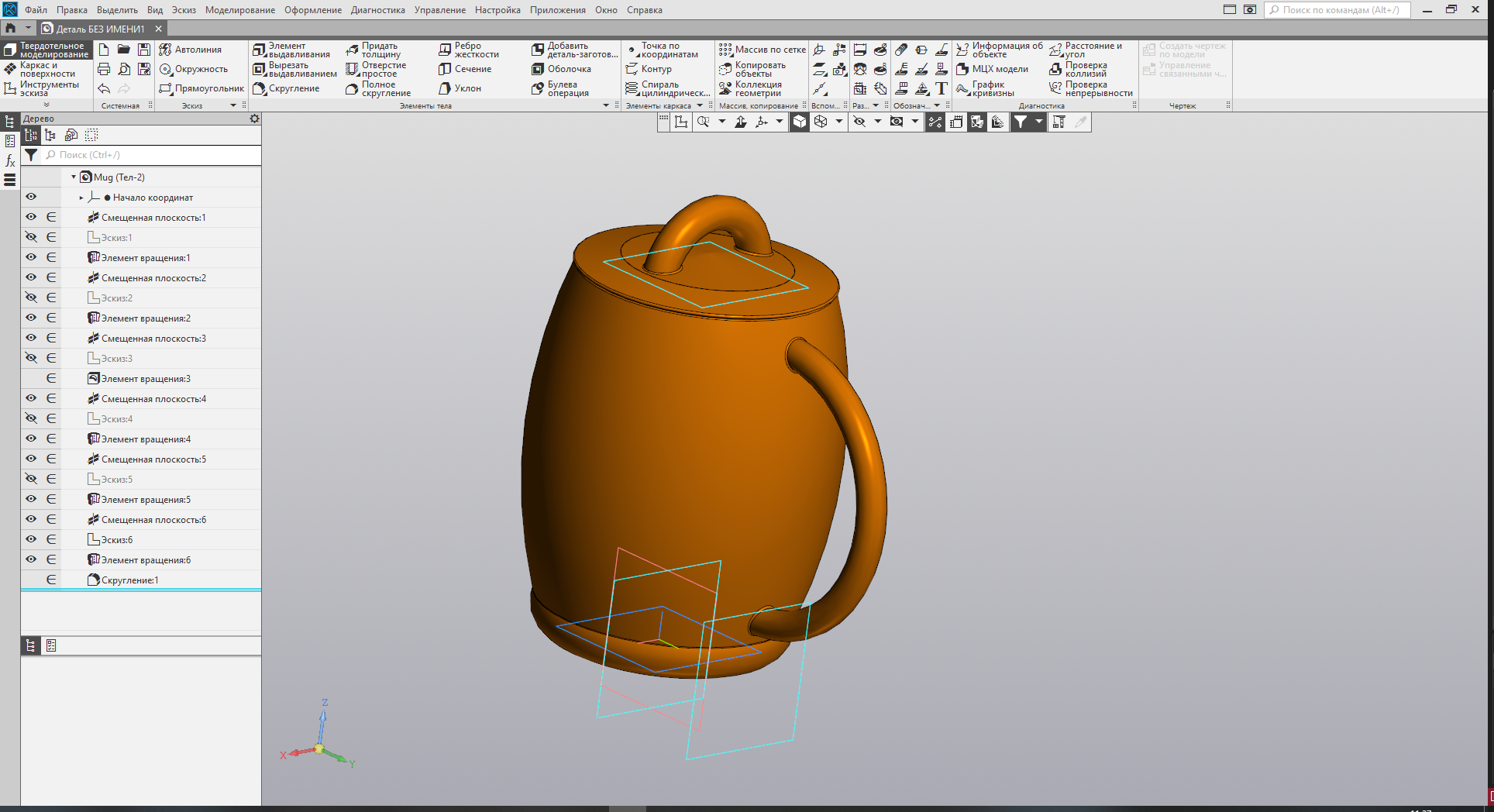


Рисунок 4.6 – Трёхмерная модель пивной кружки с крышкой, у которой ручка кольцо

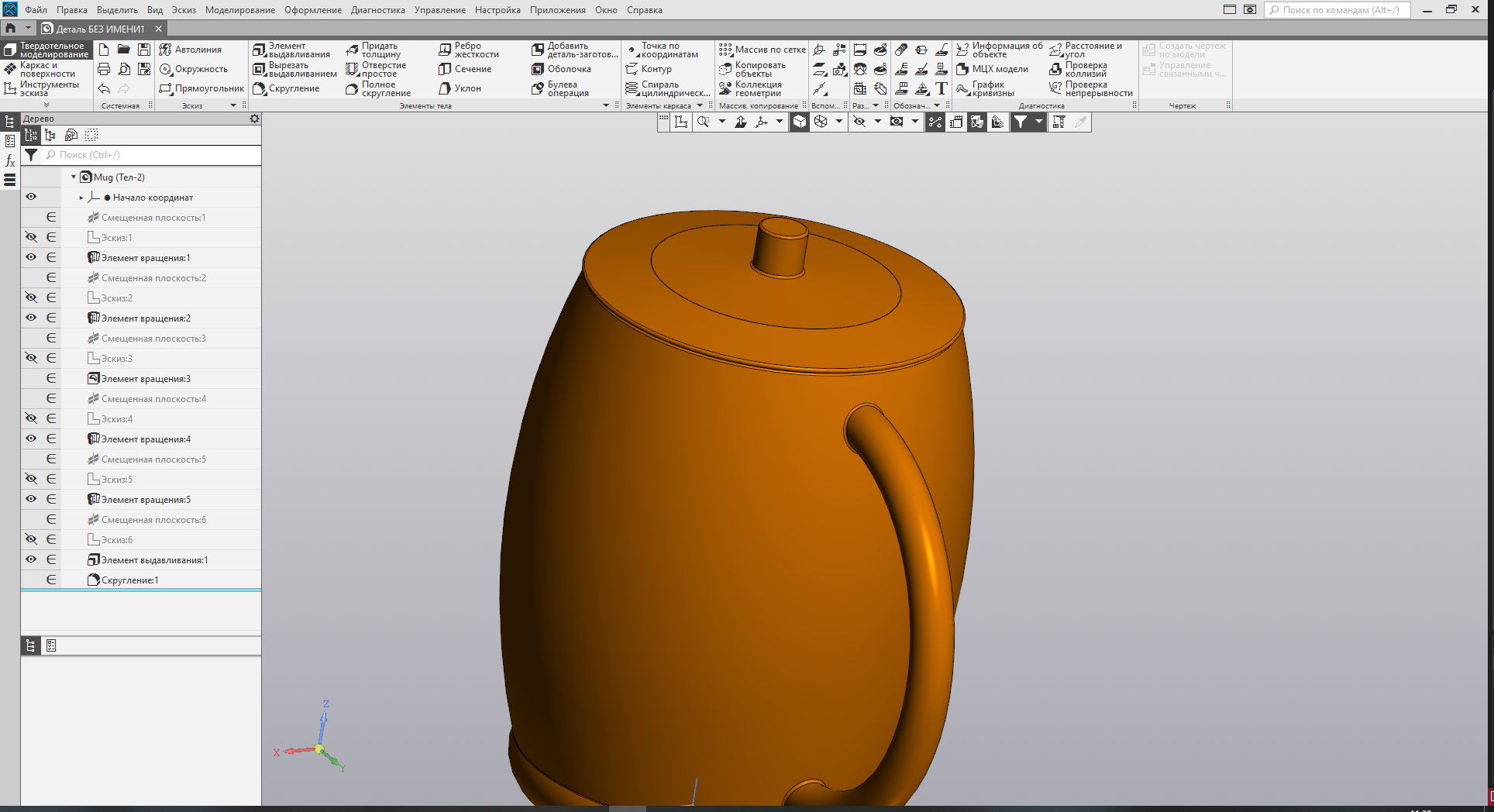


Рисунок 4.6 – Трёхмерная модель пивной кружки с крышкой, у которой ручка цилиндр

# 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:

* Диаметр горла кружки равен 80 мм;
* Толщина стенок равна 5 мм;
* Высота кружки равна 100 мм;
* Толщина дна равна 10 мм;
* Верхний радиус дна равен 80 мм;
* Нижний радиус дна равен 50 мм;

Результат построения при заданных параметрах представлен на рисунках 5.1-5.3.

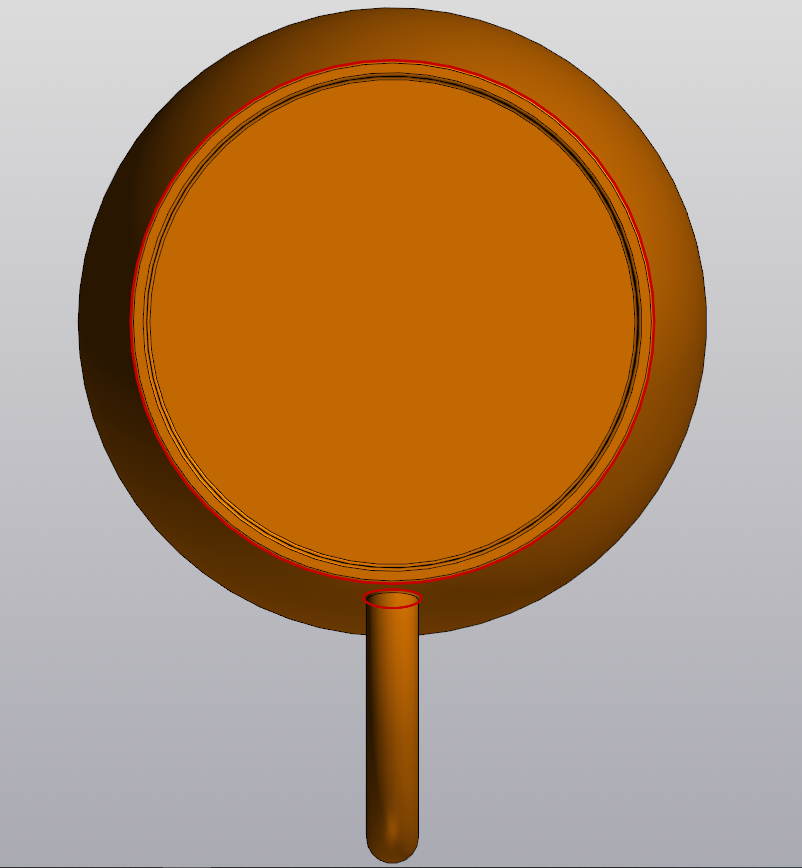


Рисунок 5.1 – Вид на модель сверху

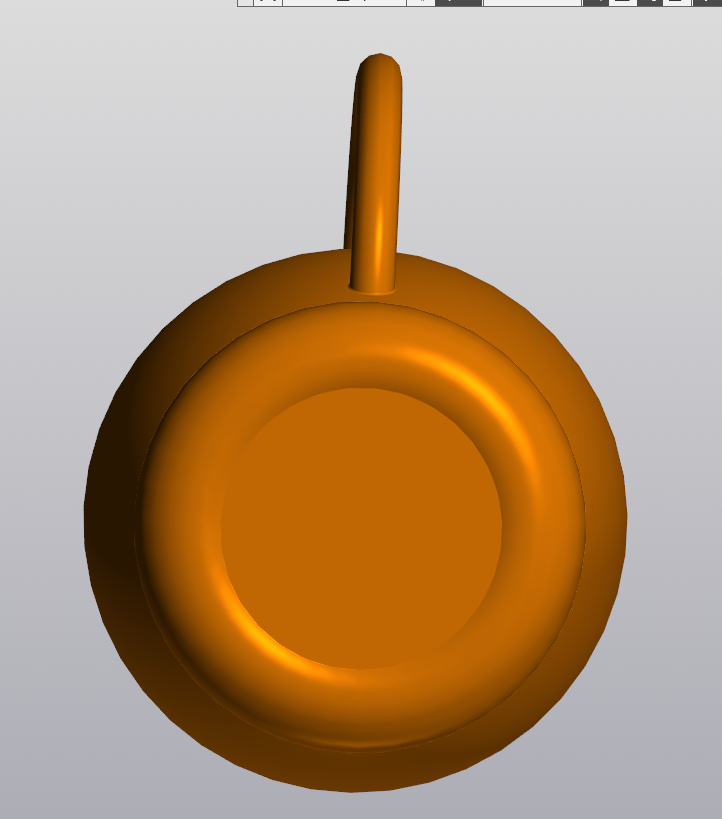


Рисунок 5.2 – Вид на модель снизу



Рисунок 5.3 – Вид на модель снизу

Тестирование при средних параметрах:

* Диаметр горла кружки равен 90 мм;
* Толщина стенок равна 6 мм;
* Высота кружки равна 132,5 мм;
* Толщина дна равна 13,25 мм;
* Верхний радиус дна равен 90 мм;
* Нижний радиус дна равен 60 мм;

Результат построения при заданных параметрах представлен на рисунках 5.4-5.7.

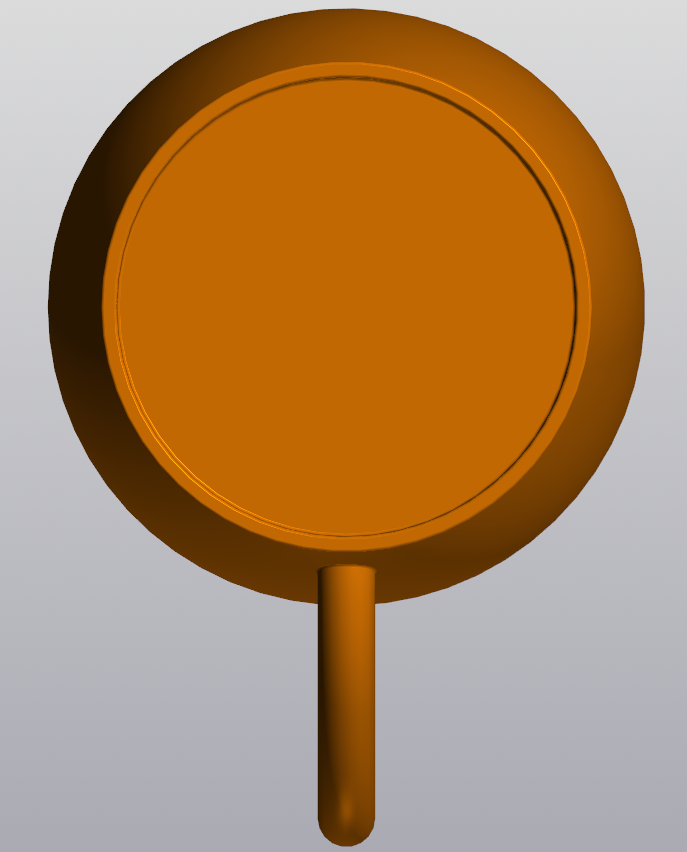


Рисунок 5.4 – Вид на модель сверху

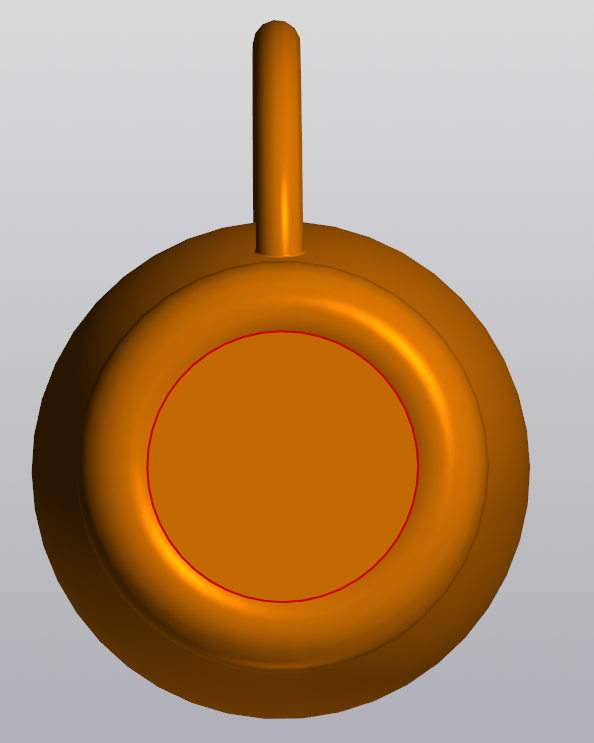


Рисунок 5.5 – Вид на модель снизу



Рисунок 5.6 – Вид на модель сбоку

Тестирование при максимальных параметрах:

* Диаметр горла кружки равен 100 мм;
* Толщина стенок равна 7 мм;
* Высота кружки равна 165 мм;
* Толщина дна равна 16,5 мм;
* Верхний радиус дна равен 100 мм;
* Нижний радиус дна равен 70 мм;

Результат построения при заданных параметрах представлен на рисунках 5.7-5.9.



Рисунок 5.7 – Вид на модель в сверху

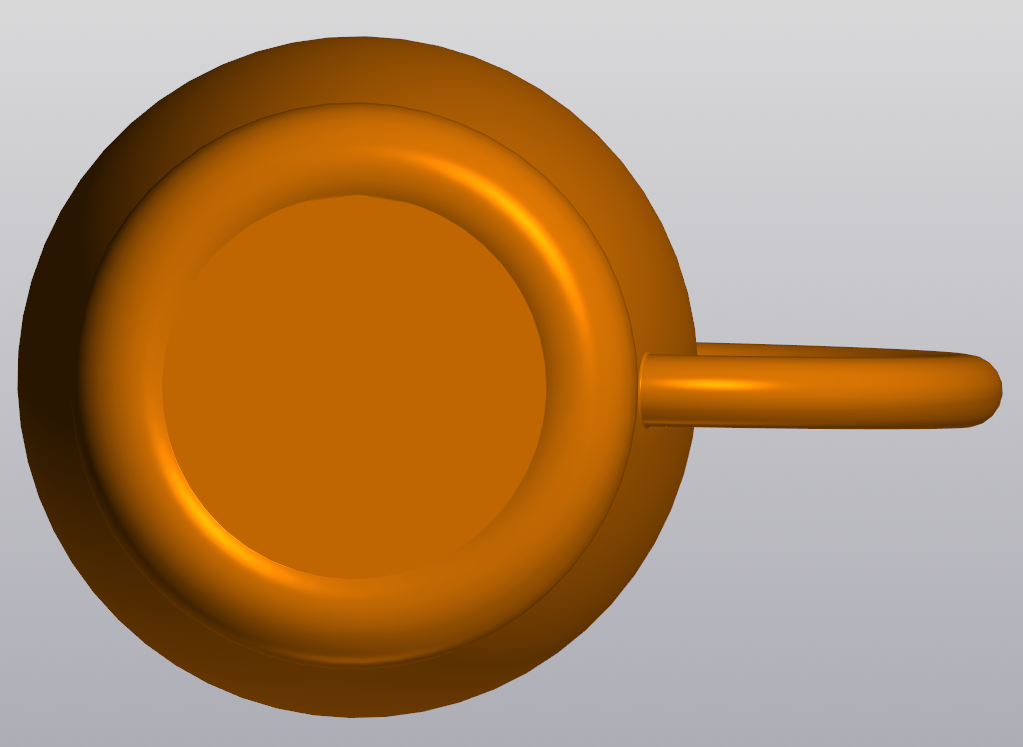


Рисунок 5.8 – Вид на модель снизу



Рисунок 5.9 – Вид на модель сбоку

# 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 30 тестов. На рисунке 5.10 представлено тестирование логики.

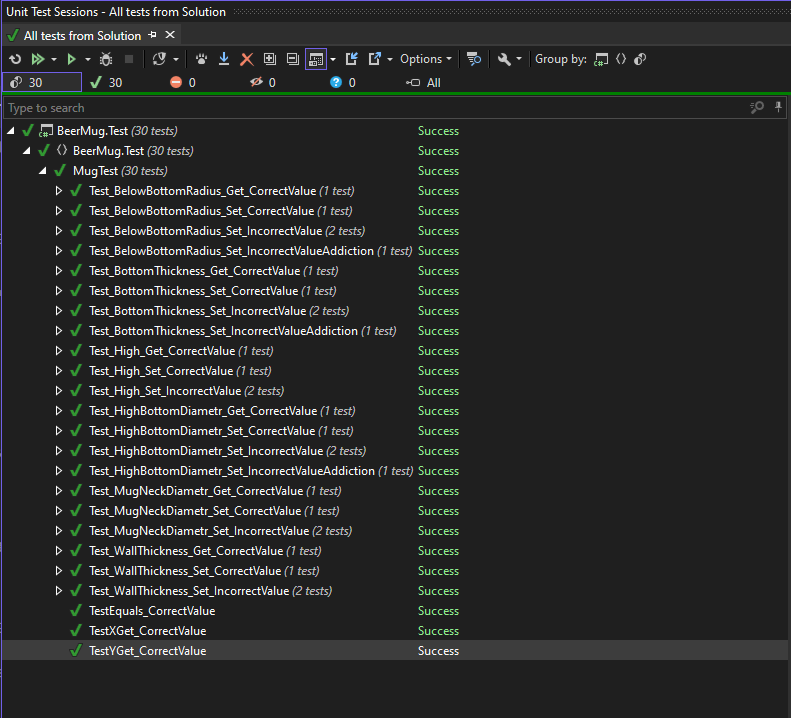


Рисунок 5.10 – Тестирование логики программы

# 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 AMD Ryzen 7 5700G with Radeon Graphics 3.8 GHz;
* оперативная память 32 ГБ;
* видеокарта NVIDIA GeForce GTX 3050.

Для тестирования будет зациклено построение модели со следующими параметрами:

* Диаметр горла кружки равен 90 мм;
* Толщина стенок равна 6 мм;
* Высота кружки равна 132,5 мм;
* Толщина дна равна 13,25 мм;
* Верхний радиус дна равен 90 мм;
* Нижний радиус дна равен 60 мм;

В результате тестирования было построено 269 деталей. По полученным измерениям был рассчитан средний результат, по которому были построены графики. На рисунке 5.11 показана зависимость количества используемой оперативной памяти в гигабайтах от количества построенных деталей.

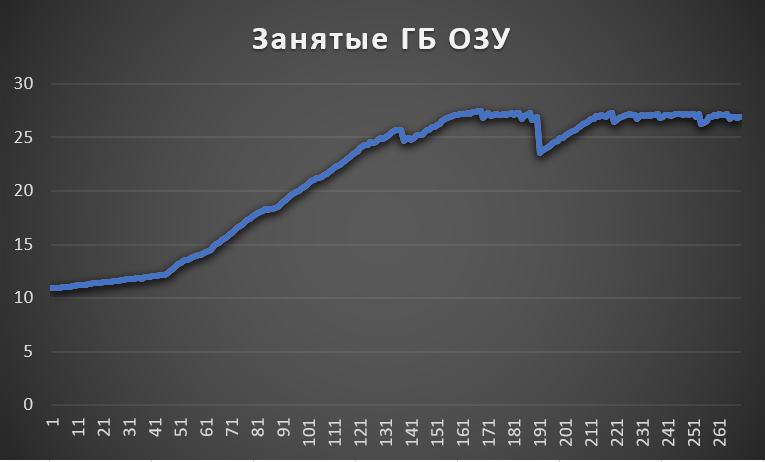


Рисунок 5.11 – Зависимость оперативной памяти от количества деталей

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 5.12 показана зависимость потраченного времени от количества построенных деталей.



Рисунок 5.12 – 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)