Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждения

высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра компьютерных систем в управлении и проектировании (КСУП)

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

по дисциплине

«Основы разработки САПР» (ОРСАПР)

Выполнил:

студент гр. 586-2

\_\_\_\_\_\_\_\_\_\_\_\_\_\_Суберляк Е. М.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2020 г.

Проверил:

к.т.н., доцент каф. КСУП

\_\_\_\_\_\_\_\_\_\_\_\_\_\_Калентьев А. А.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2020 г.

Томск 2020 г.

**РЕФЕРАТ**

Пояснительная записка 23 с., 10 рис., 7 таблиц, 7 источников.

КОМПАС-3D, ПЛАГИН, МОДЕЛЬ ЭЛЕКТРИЧЕСКОГО ЧАЙНИКА, АВТОМАТИЗИРОВАННОЕ ПРОЕКТИРОВАНИЕ, САПР.

Целью данной работы является разработка плагина, автоматизирующего построение модели электрического чайника, для системы автоматизированного проектирования КОМПАС-3D v18.1, с помощью интегрированной среды разработки Visual Studio Сommunity 2019.

В процессе работы изучены основные функции системы автоматизированного проектирования «Компас-3D» v 18.1

В результате работы был создан плагин, автоматизирующий построение электрического чайника.

Отчёт по пояснительной записке выполнен в текстовом редакторе Microsoft Word 2016.

Оглавление

[1 Описание САПР 4](#_Toc39408284)

[1.1 Описание программы 4](#_Toc39408285)

[1.2 Описание API 5](#_Toc39408286)

[1.3 Обзор аналогов 10](#_Toc39408287)

[1.3.1 Плагин PDF для САПР КОМПАС-3D 10](#_Toc39408288)

[2 Описание предмета проектирования 11](#_Toc39408289)

[3 Проект программы 12](#_Toc39408290)

[3.1 Описание технических и функциональных аспектов проекта 12](#_Toc39408291)

[3.2 Диаграмма вариантов использования (Use Cases) 12](#_Toc39408292)

[3.3 Диаграмма классов 14](#_Toc39408293)

[3.3 Макет пользовательского интерфейса 15](#_Toc39408294)

[4 Тестирование 16](#_Toc39408295)

[4.1 Функциональное тестирование 16](#_Toc39408296)

[4.2 Модульное тестирование 18](#_Toc39408297)

[4.3 Нагрузочное тестирование 19](#_Toc39408298)

[Заключение 22](#_Toc39408299)

[Список литературы 23](#_Toc39408300)

# 1 Описание САПР

# Описание программы

САПР – Система автоматизированного проектирования автоматизированная система, реализующая информационную технологию выполнения функций проектирования, представляет собой организационно техническую систему, предназначенную для автоматизации процесса проектирования, состоящую из комплекса технических, программных и других средств автоматизации его деятельности [1].

КОМПАС-3D – система трехмерного проектирования, ставшая стандартом для тысяч предприятий, благодаря сочетанию простоты освоения и легкости работы с мощными функциональными возможностями твердотельного и поверхностного моделирования. Ключевой особенностью продукта является использование собственного математического ядра С3D и параметрических технологий, разработанных специалистами АСКОН. КОМПАС-3D обеспечивает поддержку наиболее распространенных форматов 3D-моделей (STEP, ACIS, IGES, DWG, DXF), что позволяет организовывать эффективный обмен данными со смежными организациями и заказчиками, использующими любые CAD / CAM / CAE-системы в работе [2].

# Описание API

API (англ. Application Programming Interface) – описание способов, которыми одна компьютерная программа может взаимодействовать с другой программой.

В КОМПАС-3D существуют API двух версий: API 5 и API 7. Обе версии реализуют различные функции системы и дополняют друг друга. Обе версии программных интерфейсов в равной мере поддерживаются и развиваются с учетом самих изменений в системе. В основном, для создания полноценных подключаемых модулей достаточно методов и свойств интерфейсов API 5.

Главным интерфейсом API системы КОМПАС-3D является KompasObject. Получить указатель на этот интерфейс (на интерфейс приложения API 5) можно при работе под управлением внешнего приложения (контроллера) – после вызова стандартной системной функции. Методы этого интерфейса реализуют наиболее общие функции работы с документами системы, системными настройками, файлами, а также дают возможность получить указатели на другие интерфейсы.

Ниже в таблице 1.1 представлены основные свойства и методы интерфейса KompasObject.

Таблица 1.1 – Методы и свойства интерфейса KompasObject

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Входные параметры | Тип возвращаемых данных | Описание |
| Document3D() |  | ksDocument | Метод для получения указателя на интерфейс трехмерного графического документа (детали или сборки) |

Окончание таблицы 1.1

|  |  |  |  |
| --- | --- | --- | --- |
| GetParamStruct(short structType) | structType – тип интерфейса параметров | StructType2D | Метод для получения указателя на интерфейс графического документа (чертежа или фрагмента) |
| Visible |  | bool | Свойство видимости приложения |
| Quit() |  |  | Метод для закрытия активного окна приложения КОМПАС |

В таблице 1.2 представлены методы интерфейса ksEntity, необходимые для разработки плагина.

Таблица 1.2 – Методы интерфейса ksEntity

|  |  |  |
| --- | --- | --- |
| Название | Тип возвращаемых данных | Описание |
| Create() | bool | Создать объект в модели |
| GetDefinition() | IUnkown | Получить указатель на интерфейс параметров объектов и элементов |
| Update() | bool | Изменить свойства объекта (используя ранее установленные свойства) |

В таблице 1.3 представлены свойства и методы интерфейса ksDocument2D, необходимые для разработки плагина.

Таблица 1.3 – Методы интерфейса ksDocument2D

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Входные параметры | Тип возвращаемых данных | Описание |
| ksRectangle(  ksRectangleParam param, int style) | param – параметры прямоугольника.  style – стиль линии. | int | Получить указатель на прямоугольник на двумерной плоскости либо 0 в случае ошибки |
| ksCircle(  double xc, double yc, double rad, int style) | xc, yc - координаты центра окружности.  rad - радиус окружности.  style – стиль линии. | int | Получить указатель на окружность на двумерной плоскости либо 0 в случае ошибки |

В таблице 1.4 представлены свойства и методы интерфейса ksDocument3D, необходимые для разработки плагина.

Таблица 1.4 – Методы интерфейса ksDocument3D

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Входные параметры | Тип возвращаемых данных | Описание |
| Create (bool invisible, bool \_typeDoc) | invisible – признак режима редактирования документа (true – невидимый режим, false – | bool | Создать документ-модель (деталь или сборку) |

Окончание таблицы 1.4

|  |  |  |  |
| --- | --- | --- | --- |
|  | видимый режим),  typeDoc – тип документа (true – деталь, false – сборка). |  |  |
| GetPart(int type) | type – тип компонента из перечисления Типы компонентов. | ksPart | Получить указатель на интерфейс компонента в соответствии с заданным типом |

В таблице 1.5 представлены методы интерфейса ksPart, необходимые для разработки плагина.

Таблица 1.5 – Свойства и методы интерфейса ksPart.

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Входные параметры | Тип возвращаемых данных | Описание |
| EntityCollection(short objType) | objType – тип объектов, содержащихся в массиве. | ksEnintyCollection | Формирует массив объектов и возвращает указатель на его интерфейс |
| GetDefaultEntity(short objType) | objType – тип объекта | ksEntity | Получить указатель на интерфейс объекта, создаваемого системой по умолчанию |

Окончание таблицы 1.5

|  |  |  |  |
| --- | --- | --- | --- |
| GetPart(int type) | Type – тип компонента | ksPart | Получить указатель на интерфейс компонента в соответствии с заданным типом |
| NewEntity(short objType) | objType – тип объекта | ksEntity | Создать новый интерфейс объекта и получить указатель на него |

В таблице 1.6 представлены типы объектов документа-модели, необходимые для разработки плагина.

Таблица 1.6 – Некоторые типы объектов документа-модели

|  |  |  |
| --- | --- | --- |
| Идентификатор объекта | Название объекта | Интерфейс параметров |
| o3d\_unknown | Неизвестный (включает все объекты) |  |
| o3d\_planeXOZ | Плоскость XOZ | ksPlaneParam |
| o3d\_planeYOZ | Плоскость YOZ | ksPlaneParam |
| o3d\_planeXOY | Плоскость XOY | ksPlaneParam |
| o3d\_sketch | Эскиз | ksSketchDefinition |
| o3d\_face | Грань | ksFaceDefinition |
| o3d\_baseExtrusion | Базовая операция выдавливания | ksBaseExtrusionDefinition |

# Обзор аналогов

# 1.3.1 Плагин PDF для САПР КОМПАС-3D

Данный плагин позволяет производить экспорт моделей и сборок из КОМПАС-3D в формат PDF формат [5]. Основной особенностью является возможность интерактивного взаимодействия пользователя с сохраненной 3D сценой внутри PDF файла. Например, пользователь может вращать, масштабировать, передвигать детали и сборки внутри 3D PDF файла. Также доступно создание анимации сборки и разборки изделий. Это полезно для подготовки интерактивных сборочных инструкций, создания маркетинговых материалов, презентаций, а также для налаживания взаимодействия между проектировщиками и заказчиками. В подобных ситуациях традиционным подходом являлся экспорт сборки или детали КОМПАС-3D в промежуточный формат и дальнейшее сохранение в формат 3D PDF. Используемый подход в плагине исключает использование промежуточных файлов для осуществления 3D преобразования, что существенно повышает качество выходной 3D модели в формате PDF. Ключевые возможности:

* сохранение деталей и сборок в формате 3D PDF для интерактивного просмотра при помощи бесплатной программы Adobe Reader;
* создание анимаций имитирующих естественный порядок сборки и разборки создание имитации анимации гибки листовых тел;
* вставка в существующие PDF документы, содержащие основной текст, фоновые картинки, таблицы спецификаций, эмблемы, логотипы;

пакетный режим для поочередной конвертации всех файлов.

# 2 Описание предмета проектирования

Электрический чайник — прибор для нагревания питьевой воды, работающий на [электричестве](https://dic.academic.ru/dic.nsf/ruwiki/1308). [4].

Параметры чайника:

* радиус чайника L (от 100 до 140 мм);
* высота чайника H (от 150 до 200 мм);
* длина носика F (от 20 до 25 мм);
* цвет чайника;
* цвет ручки чайника;
* высота носика: D = (1/5)\*L;
* ширина носика: S = (1/5)\*L;
* размер ручки R (от 95 до 125 мм);

Пример модели приведен ниже, на рисунке 2.1.

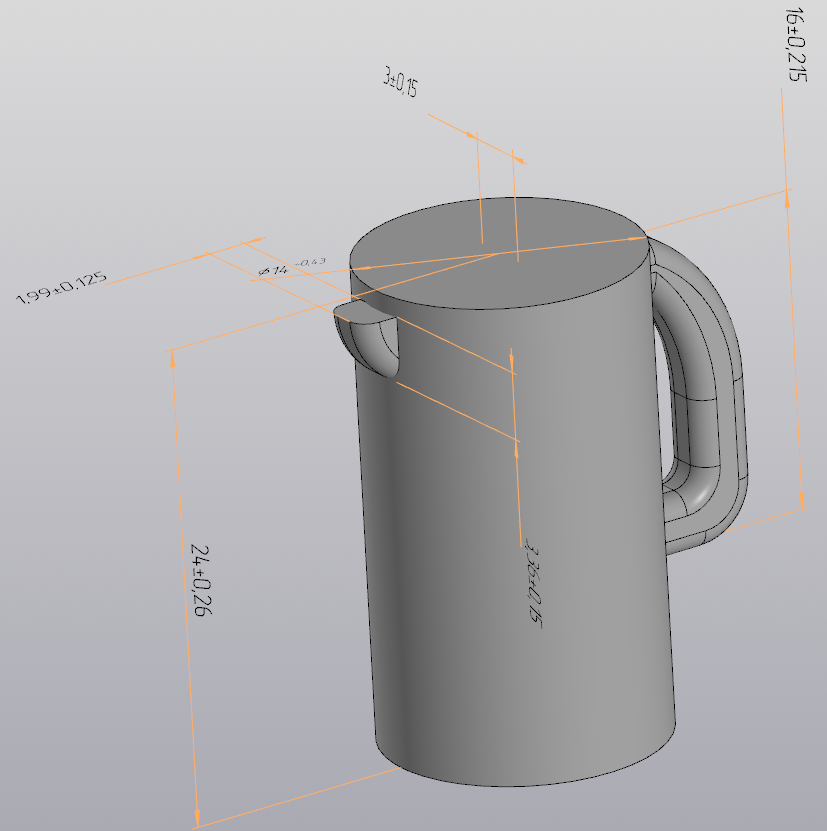


Рисунок 2.1 – Модель электрического чайника

# 3 Проект программы

# 3.1 Описание технических и функциональных аспектов проекта

Для графического описания абстрактной модели проекта, а также пользовательского взаимодействия (сценарии действия) использован стандарт UML.

UML язык графического описания для объектного моделирования в обрасти разработки программного обеспечения. UML является языком широкого профиля, это – открытый стандарт, использующий графические обозначения для создания абстрактной модели системы, называемой UML – моделью. UML был создан для определения, визуализации, проектирования и документирования, в основном, программных систем. UML не является языком программирования, но на основании UML возможна генерация кода и наоборот [5].

При использовании UML были простроены: диаграмма использования и диаграмма классов.

# 3.2 Диаграмма вариантов использования (Use Cases)

Диаграмма вариантов использования (англ. *use case diagram)* в UML – диаграмма, отражающая отношения между актерами и прецедентами и являющаяся составной частью модели прецедентов, позволяющей описать систему на концептуальном уровне.

Назначение диаграммы – описание функциональности и поведения, позволяющее заказчику, пользователю, а также разработчику обсуждать проектируемую или существующую систему. Работа над диаграммой может начаться с текстового описания, полученного при работе с заказчиком. При этом нефункциональные требования (например, конкретный язык или система программирования) при составлении модели прецедентов опускаются (для них составляется другой документ) [5].

На рисунке 3.1 представлена диаграмма вариантов использования.

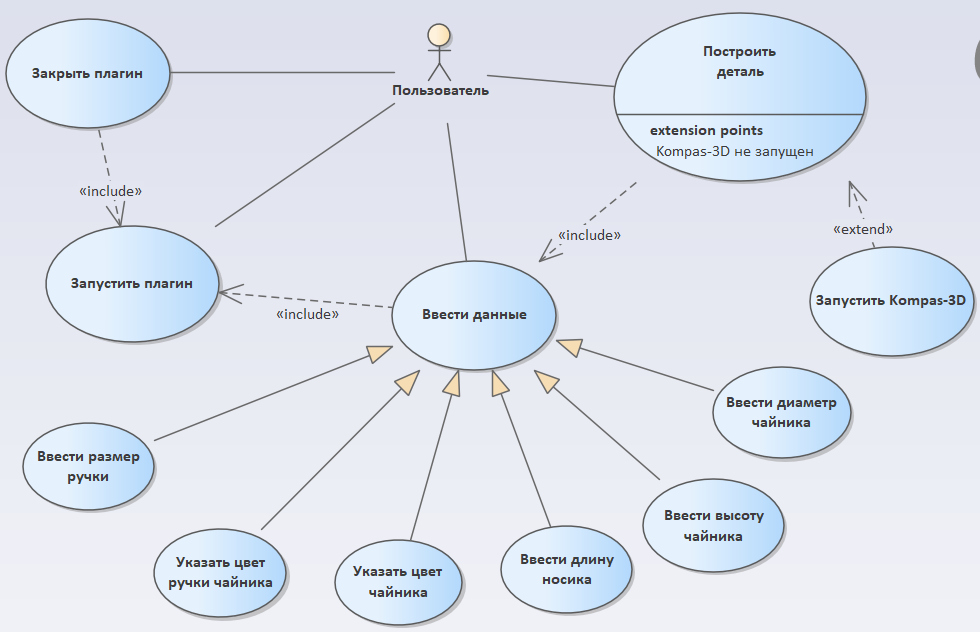


Рисунок 3.1– Диаграмма вариантов использования.

# 3.3 Диаграмма классов

Диаграмма классов – структурная диаграмма языка моделирования UML, демонстрирующая общую структуру иерархии классов системы, их коопераций, атрибутов (полей), методов, интерфейсов и взаимосвязей между ними.[5]

На рисунке 3.2 представлена диаграмма классов.

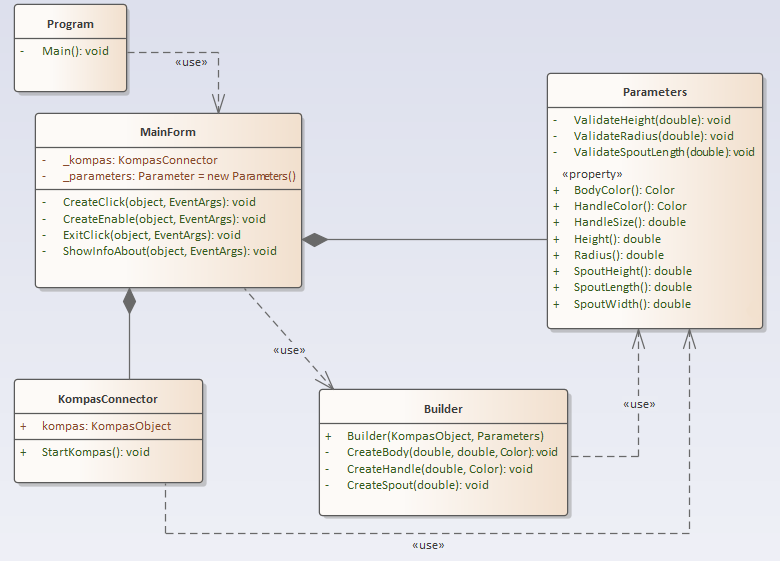


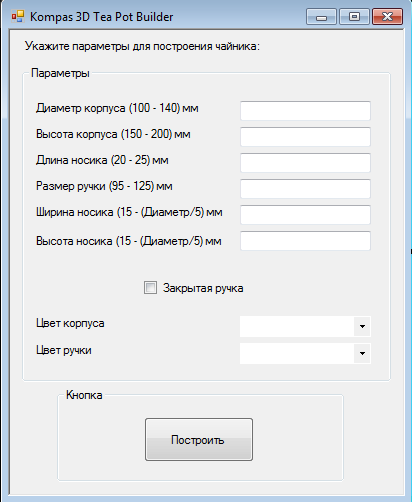
Рисунок 3.2 – Диаграмма классов

Класс «Program», использует «MainForm» для обработки действий в графическом интерфейсе. «DetailCreator» содержит в себе методы создания 3D модели в «Компас 3D», класс «Validator» проверяет входные данные, введенные в графическом интерфейсе.

# 3.3 Макет пользовательского интерфейса

Макет пользовательского интерфейса представляет собой форму для ввода параметров электрического чайника. Построение модели осуществляется путем нажатия на кнопку «Построить». При попытке ввода недопустимых символов, они не будут вводиться в строку (если необходимо ввести цифры, то невозможно будет ввести другие символы). Если в поля ввода, будут введены некорректные значения или зависимые параметры будут противоречить друг другу, то поля этих параметров будут подсвечиваться красный цветом, а также кнопка «Построить» будет недоступна до тех пор, пока пользователь не введет корректные значения.

На рисунке 3.3 представлен макет пользовательского интерфейса.

  
Рисунок 3.3 – Макет пользовательского интерфейса

Для построения модели «Электрический чайник» необходимо:

* Ввести параметры чайника;
* Запустить построение модели (нажать кнопку «Построить»).

# 4 Тестирование

# 4.1 Функциональное тестирование

Функциональное тестирование — тестирование функциональности объекта, т.е. правильно ли объект выполняет свои функции. Фактически, выполняется проверка правильности выходных данных при соответствующих входных[6]. Проведем тестирование при максимальных и минимальных параметрах модели.

Модель с минимальными входными параметрами представлена на рисунке 4.1.

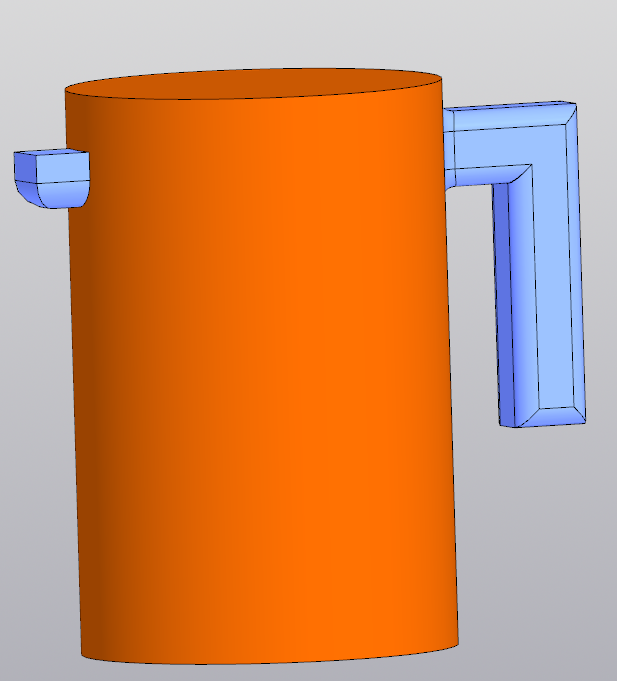


Рисунок 4.1 — Модель чайника с минимальными входными параметрами

Модель с максимальными входными параметрами представлена на рисунке 4.2.

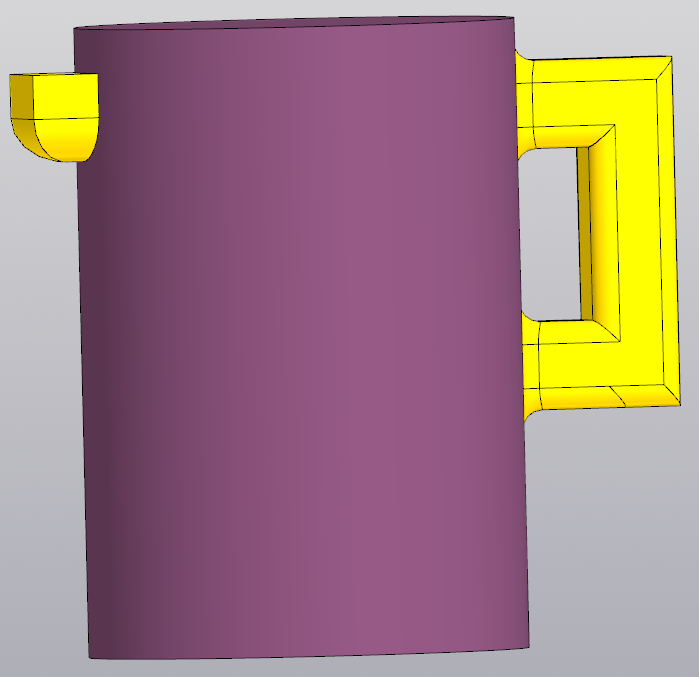


Рисунок 4.2 — Модель чайника с максимальными входными параметрами и закрытой ручкой

При выходе из зоны допустимых значений параметров поле подсвечивается красным цветом, появится сообщение о том, что введены не корректные данные, а также кнопка «Построить» становится недоступной.

Пример приведен на рисунке 4.3. Рамками выделены сообщение об ошибке и поле, в котором значение задано не корректно.

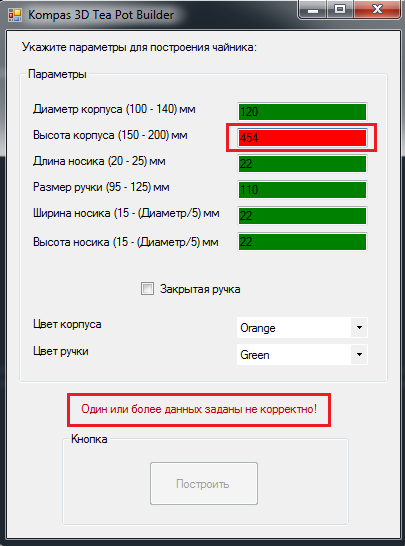


Рисунок 4.3 – Ошибка при вводе некорректных параметров.

# 4.2 Модульное тестирование

Юнит-тестирование (блочное тестирование, «unit-testing») —тестирование отдельного элемента изолированно от остальной системы[6].

Список тестовых сценариев для модульного тестирования граничных значений входных параметров представлен в таблице 4.1.

Таблица 4.1 – Список тестовых сценариев.

|  |  |
| --- | --- |
| Название тестового метода | Описание |
| TestGet\_CorrectValues(double Diameter, double Height, double SpoutLength, double SpoutWidth, double SpoutHeight, double HandleSize, TeaPotParams.TheColor theColor, TeaPotParams.TheColor theColor1) | Тестирование при вводе корректных значений чайника |

Окончание таблицы 4.1

|  |  |
| --- | --- |
| TestGet\_BadValues(double Diameter, double Height, double SpoutLength, double SpoutWidth, double SpoutHeight, double HandleSize, TeaPotParams.TheColor theColor,  TeaPotParams.TheColor theColor1) | Тестирование при вводе не корректных значений чайника |

Результаты успешного прохождения всех модульных тестов приведены на рисунке 4.4.

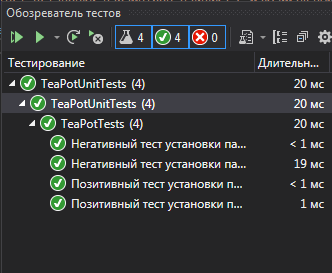


Рисунок 4.4 - Результаты модульных тестов

# 4.3 Нагрузочное тестирование

Нагрузочное тестирование — 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)[7].

После построения 207 деталей с минимальными параметрами программа завершилась со сбоем из-за нехватки памяти. На момент сбоя количество потребляемой памяти было около 724 МБ.

После построения 158 деталей с максимальными параметрами программа также завершилась со сбоем. Но момент сбоя количество потребляемой памяти было около 803 МБ.

На представленных графиках в текущей главе ось «X» – время в минутах, ось «Y» – количество построенных деталей.

На рисунке 4.5 представлено тестирование зацикленного построения фигуры с минимальными параметрами:

* Диаметр корпуса 100 мм;
* Высота корпуса 150 мм;
* Длина носика 20 мм;
* Размер ручки 95 мм;
* Ширина носика 15 мм;
* Высота носика 15 мм;

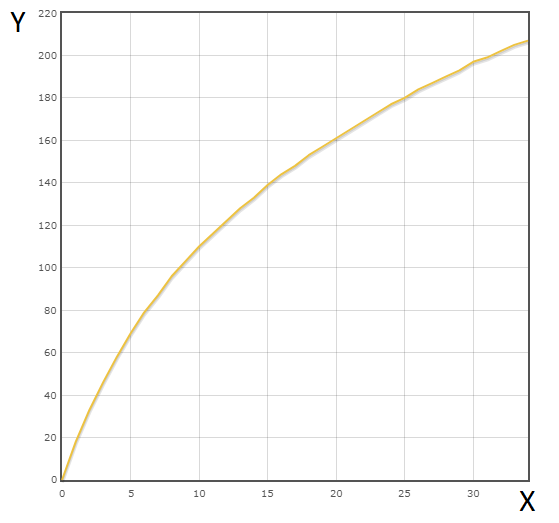


Рисунок 4.5 – График зависимости времени от количества построенных фигур с минимальными параметрами

На рисунке 4.6 представлено тестирование зацикленного построения фигуры с максимальными параметрами:

* Диаметр корпуса 140 мм;
* Высота корпуса 200 мм;
* Длина носика 25 мм;
* Размер ручки 125 мм;
* Ширина носика 28 мм;
* Высота носика 28 мм;

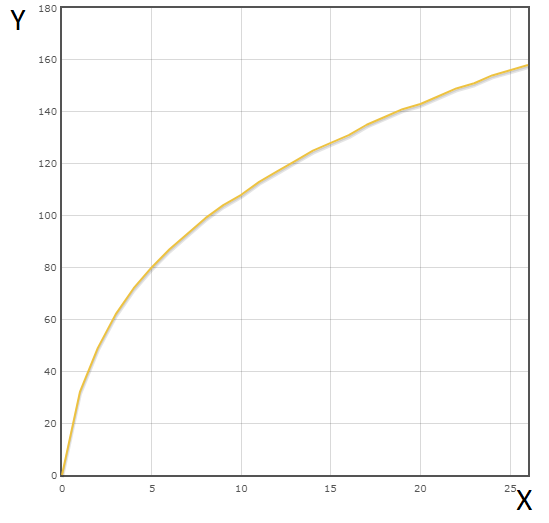


Рисунок 4.6 – График зависимости времени от количества построенных фигур с максимальными параметрами

По графикам можно определить, что при увеличении количества деталей, постепенно спадает производительность.

# Заключение

В ходе выполнения лабораторных работ были изучены предметная область проектирования, предмет проектирования, аналоги предмета проектирования, API и на основании полученных данных были спроектированы UML диаграммы классов, разработан плагин для создания 3D моделей «Электрический чайник» в САПР «Компас 3D».

# Список литературы

САПР – System Enginering Thinking Wiki. [Электронный ресурс]. – Режим доступа: <http://sewiki.ru/%D0%A1%D0%90%D0%9F%D0%A0> (дата обращения 22.02.2020)

КОМПАС-3D: О программе. [Электронный ресурс]. – Режим доступа: <http://kompas.ru/kompas-3d/about/> (дата обращения 22.02.2020)

Плагин PDF [Электронный ресурс]. — Режим доступа: <http://gkmsoft.ru/ru/#/pdf3d_kompas> (дата обращения 02.03.2020).

Электрический чайник. [Электронный ресурс]. — Режим доступа: <https://dic.academic.ru/dic.nsf/ruwiki/703986> (дата обращения 22.02.2020)

UML. [Электронный ресурс]. – Режим доступа: <http://www.uml.org/> (дата обращения 02.03.2020)

Новые технологии в программировании: учебное пособие / А.А. Калентьев, Д. В. Гарайс, А. Е. Горяинов. – Томск, 2014. − ­ 176 стр. (дата обращения: 01.05.2020).

Виды тестирования [электронный ресурс]. – режим доступа: <https://qa-academy.by/qaacademy/news/klassifikaciya-vidov-testirovaniya/> (дата обращения: 02.05.2020)