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)

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

по дисциплине

«Основы разработки САПР»

Выполнил:

Студент гр. 589-3

\_\_\_\_\_\_\_\_\_ Высоцкая О.В.

Руководитель

к.т.н., доцент каф. КСУП

\_\_\_\_\_\_\_\_\_ Калентьев А.А.

(оценка) \_\_\_\_\_\_\_\_\_

Томск 2022 г

**Оглавление**

[1 Введение 3](#_Toc122902089)

[2 Постановка и анализ задачи 4](#_Toc122902090)

[2.1 Описание предмета проектирования 5](#_Toc122902091)

[2.2 Выбор инструментов и средств реализации 7](#_Toc122902092)

[2.3 Назначение плагина 7](#_Toc122902093)

[3 Обзор аналогов 8](#_Toc122902094)

[4 Описание реализации 9](#_Toc122902095)

[4.1 Диаграмма классов 9](#_Toc122902096)

[5 Описание программы для пользователя 15](#_Toc122902097)

[6 Тестирование программы 17](#_Toc122902098)

[6.1 Функциональное тестирование 17](#_Toc122902099)

[6.2 Модульное тестирование 19](#_Toc122902100)

[6.3 Нагрузочное тестирование 22](#_Toc122902101)

[Заключение 25](#_Toc122902102)

[Список использованных источников 26](#_Toc122902103)

# 1 Введение

Автоматизация моделирования 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 моделированием 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. Она позволяет значительно повысить точность расчетов, выбрать наилучшие варианты для реализации на основе строгого математического анализа всех или большинства вариантов проекта с оценкой технических, технологических и экономических характеристик производства и эксплуатации проектируемого объекта, значительно повысить качество конструкторской документации, существенно сократить сроки проектирования и передачи конструкторской документации в производство, эффективнее использовать технологическое оборудование с программным управлением [1].

Таким образом, целью данной работы является разработка плагина, автоматизирующего построение модели «Корпуса спиннера» для системы автоматизированного проектирования Компас 3D с помощью интегрированной среды разработки Visual Studio 2019 Сommunity [2].

Интегрированная среда разработки Visual Studio — это стартовая площадка для написания, отладки и сборки кода, а также последующей публикации приложений. Интегрированная среда разработки (IDE) представляет собой многофункциональную программу, которую можно использовать для различных аспектов разработки программного обеспечения.

# 2 Постановка и анализ задачи

В рамках лабораторных работ в соответствии с технически заданием требовалось разработать плагин, который на основе входных параметров, интегрируя с системой САПР Kompas 3D, строит модель корпуса спиннера [3]. Необходимо чтобы плагин позволял задавать параметры по умолчанию, а также изменять входные параметры корпуса спиннера, такие как:

* диаметр внутренних колец ***I***;
* длина спиннера ***А***;
* толщина корпуса спинера ***С***;
* диаметр центрального кольца спинера ***D***;
* радиус внешних колец ***R***;
* скругление (радиус) корпуса спинера и его колец ***Z***.

# 2.1 Описание предмета проектирования

Спиннер — развлекательная вращающаяся игрушка. В общих чертах данный корпус спиннера имеет центральное кольцо по обеим сторонам которого, симметрично расположены два кольца [4].

На рисунке 2.1 представлен чертеж корпуса спиннера.

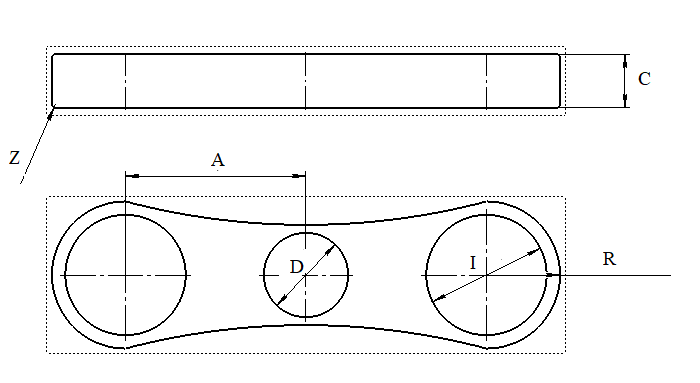


Рисунок 2.1 – Чертеж корпуса спиннера

Параметры корпуса спиннера:

* диаметр внутренних колец спинера ***I*** (мин – 30 мм, макс – 60 мм);
* толщина корпуса спинера ***С*** (мин – 10 мм, макс – 30 мм);
* длина спинера ***A*** (зависит от диаметра внутренних колец: мин – I (мин) \* 1.5, макс – I (макс) \* 1.5);
* диаметр центрального кольца спинера ***D*** (зависит от диаметра внутренних колец: мин – I (мин) \* 0.7, макс – I (макс) \* 0.7);
* радиус внешних колец ***R*** (зависит от диаметра внутренних колец: мин – (I (мин) + 10) / 2, макс – (I (макс) + 10) / 2);
* скругление (радиус) корпуса спинера и его колец ***Z*** (мин – 0.5 мм, макс – 2 мм).

# 2.2 Выбор инструментов и средств реализации

На основе требований к техническому заданию программа выполнена на языке программирования C# в среде Microsoft Visual Studio 2019 с использованием .NET Framework 4.7.2 [2], библиотеки для Kompas 3D [5].

Инструментом тестирования и создания модульных тестов был выбран тестовый фреймворк NUnit [6] версии 3.13.3.

Для реализации пользовательского интерфейса использовалась система для построения настольных приложений WindowsForms [7].

# 2.3 Назначение плагина

Назначение разрабатываемого плагина обусловлено быстрым моделированием корпуса спиннера различных размеров. Благодаря данному расширению, любой желающий может наглядно рассмотреть спроектированную модель, при необходимости перестроить под необходимые им параметры.

# 3 Обзор аналогов

SolidWokrs (cолидворкc, от англ. solid — твёрдое тело и англ. works — работать) — программный комплекс САПР для автоматизации работ промышленного предприятия на этапах конструкторской и технологической подготовки производства. Обеспечивает разработку изделий любой степени сложности и назначения [8].

Программу начали разрабатывать в 1993 году, она начала продаваться в 1995 и составила конкуренцию таким продуктам, как AutoCAD и Autodesk Mechanical Desktop, SDRC I-DEAS, Компас-3D и Pro/ENGINEER [4].

Принцип построение корпуса спиннера и в Компас-3D, и в SolidWorks очень схожи.

Принцип построение корпуса спиннера в SolidWorks показан на рисунке 3.1.

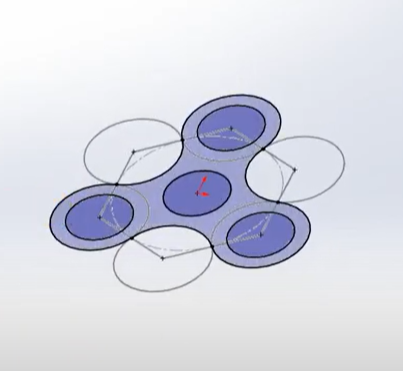


Рисунок 3.1 – Принцип построение корпуса спиннера в SolidWorks

# 4 Описание реализации

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

UML язык графического описания для объектного моделирования в области 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 [9].

При использовании UML были простроена диаграмма классов.

# 4.1 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 [9].

На рисунке 4.1 представлена диаграмма классов.

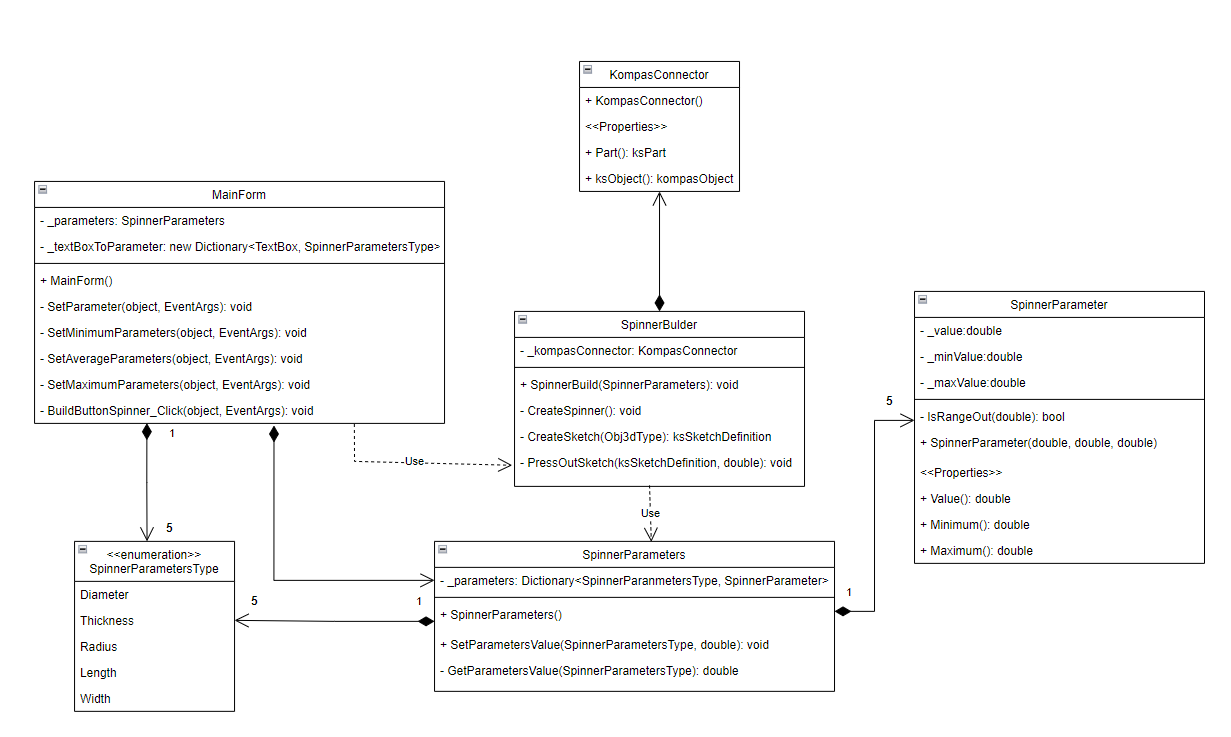


Рисунок 4.1 – Изначальная диаграмма классов

Далее в таблицах 3.1 – 3.4 представлено описание классов.

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

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| \_parameters |  | Хранит в себе набор методов для построения корпуса спиннера |
| \_textBoxToParameter |  | Хранит в себе набор TextBox и соответствующий ему тип параметра из перечисления «SpinnerParametersType» |
| SetParameter(object, EventArgs) | void | Устанавливает значение параметра |
| SetMinParameters() | void | Устанавливает минимальное значение всех параметров |
| * SetMaxParameters() | void | Устанавливает максимальное значение всех параметров |
| - SetAvgParameters() | void | Устанавливает среднее значение всех параметров |
| Build() | void | Строит корпус спиннера по заданным параметрам |

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

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| \_parameters |  | Хранит данные о каждом параметре модели из перечисления «SpinnerParametersType» |
| SpinnerParameters() |  | Конструктор для создания экземпляра класса |
| SetParameterValue(ParameterType, double) | void | Устанавливает значение определённого параметра |
| GetParameterValue(ParameterType) | double | Возвращает значение определённого параметра |

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

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| \_value |  | Хранит текущее значение |
| SpinnerParameter(double, double, double) |  | Конструктор для создания экземпляра класса |
| Maximum() | double | Возвращает максимальное допустимое значение параметра |
| Minimum() | double | Возвращает минимальное допустимое значение параметра |
| Value() | double | Возвращает текущее значение параметра. Задает новое значение параметра |

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

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| kompasWrapper |  | Хранит в себе методы необходимые для связи с КОМПАС 3D |
| \_parameters |  | Хранит данные о каждом параметре модели из перечисления «SpinnerParameter» |
| SpinnerBuild(SpinnerParameters) | void | Построение корпуса спиннера по заданным параметрам |
| CreateSpinner() | void | Построение корпуса спиннера |
| CreateSketch(obj3dType) | ksSketchDefinition | Возвращает интерфейс параметров эскиза |
| PressOutSketch(ksSketchDefinition, double) | void | Выдавливает эскиз |

В итоговом проекте созданы следующие классы и методы, которые отображены на итоговой диаграмме классов (рисунок 4.2).

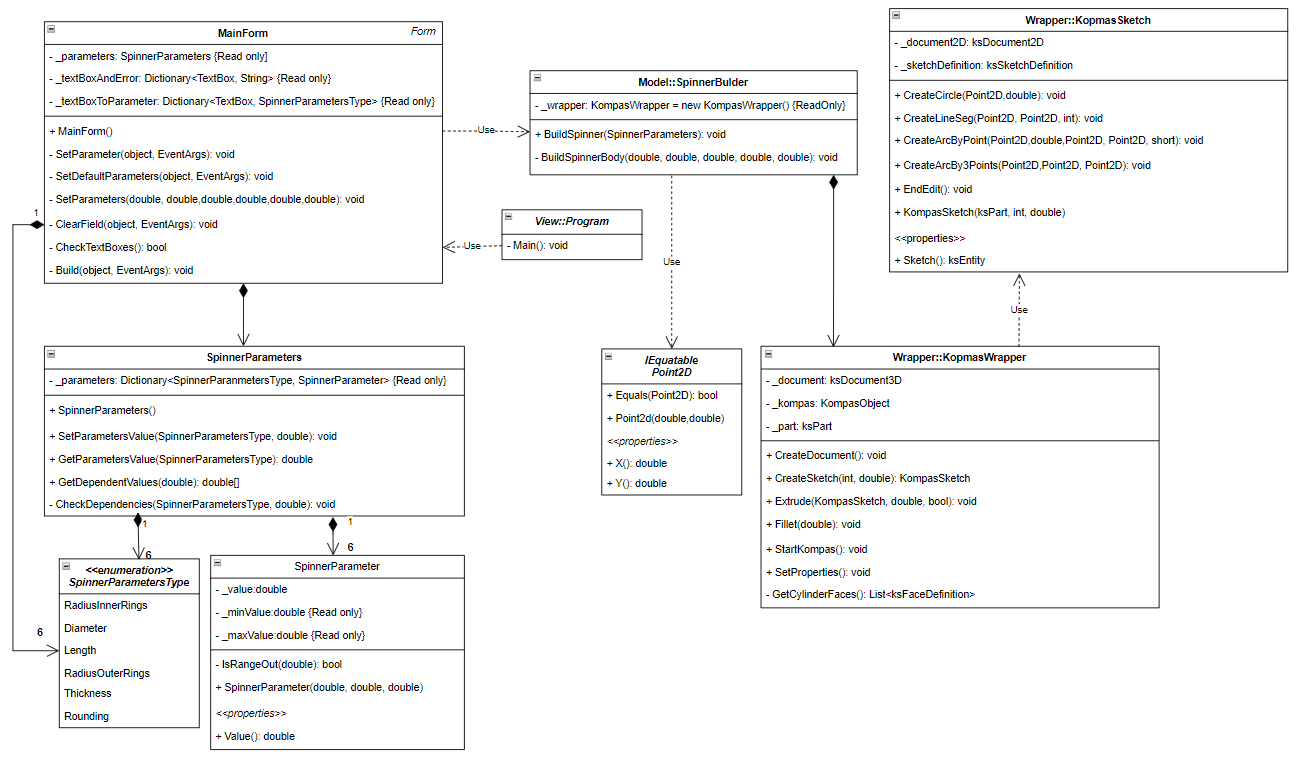


Рисунок 4.2 – Итоговая диаграмма классов

В процессе разработки была переосмысленная изначальная диаграмма классов, в результате чего:

* Была переработана логика обработки ошибок, так как изначальная оказалось недостаточно продуманной. Теперь в свойстве \_textBoxAndError сохраняется текст ошибки к определенному полю и показывается пользователю, если он ввел в него некорректные данные;
* Было решено перенести метод обработки зависимых параметров из класса MainForm в класс SpinnerParameters, так как изначальное решение подразумевало, что при изменении дизайна придется переписывать реализацию метода;
* В классе KompasWrapper были реализованы методы, отвечающие за различные операции, которые используются при построении модели корпуса спиннера, чтобы избавиться от дублирования кода;
* Был добавлен вспомогательный класс Point2D для упрощения создания 2D-эскизов;
* Был добавлен вспомогательный класс KompasSketch для упрощения взаимодействия с эскизами.

# 5 Описание программы для пользователя

Макет пользовательского интерфейса представляет собой форму для ввода параметров корпуса спиннера. Построение модели осуществляется путем нажатия на кнопку «Build».

На рисунке 5.1 представлен макет пользовательского интерфейса.

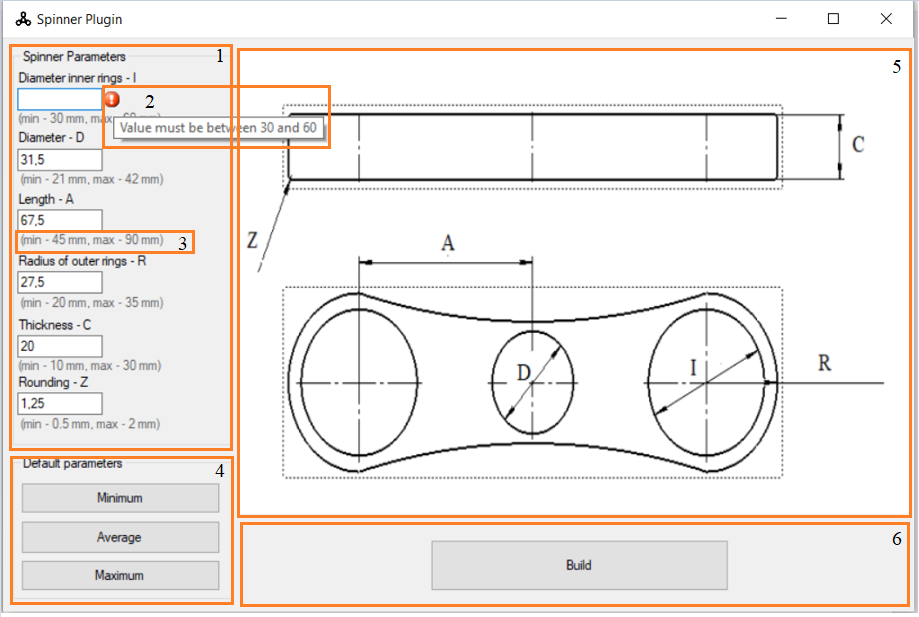


Рисунок 5.2 – Макет пользовательского интерфейса

Описание пользовательского интерфейса:

1 – поля ввода параметров корпуса спиннера;

2 – описание ошибки введенного параметра;

3 – диапазон вводимых значений;

4 – кнопки для построения модели корпуса спиннера с предустановленными параметрами;

5 – чертеж корпуса спиннера;

6 – кнопка для построения корпуса спиннера.

При вводе некорректных значений пользователь увидит сообщение об ошибке, которое отображает допустимое значение параметра.

Во время построения детали, кнопка «Build» становится неактивной.

После нажатия на кнопку «Build» при введенных некорректных значениях, появится окно, приведенное на рисунке 5.3.

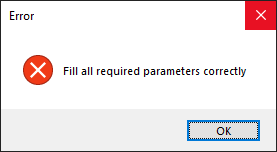


Рисунок 5.3 — Окно ошибки

После ввода необходимых параметров, построить деталь в САПР Компас 3D можно с помощью кнопки «Build». Корпус спиннера, построенный по заданным параметрам в САПР Kompas 3D, представлена на рисунке 5.4.

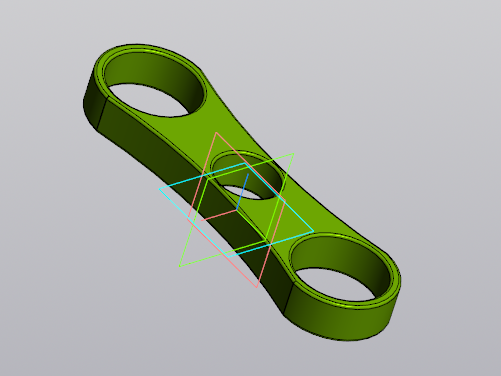


Рисунок 5.4 — Корпус спиннера, построенный по заданным параметрам в САПР Компас 3D

# 6 Тестирование программы

Тестирование позволяет убедиться в работоспособности программы, выявлять ошибки при изменении какого-либо функционала.

# 6.1 Функциональное тестирование

При функциональном тестировании проверялось корректность работы плагина «Корпус спиннера», а именно, соответствие полученного результата в виде трехмерной модели, с входными параметрами [10].

Проведено тестирование максимальных и минимальных параметров модели.

На рисунке 6.1 представлена проверка размеров модели с минимальным введенными параметрами в САПР Kompas 3D.

Минимальные параметры корпуса спиннера:

Диаметр центрального кольца 21 мм;

Длина спиннера 45 мм;

Диаметр внутренних колец спиннера 30 мм;

Толщина спиннера 10 мм;

Радиус внешних колец спиннера 20 мм;

Скругление корпуса спиннера и внутренних колец 0,5 мм.

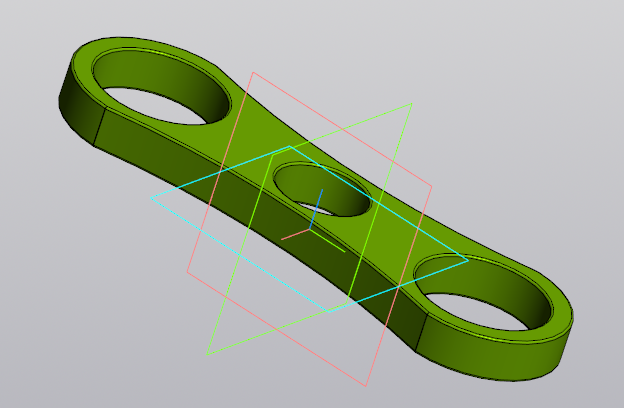


Рисунок 6.1 – Модель с минимальными введенными параметрами в Kompas 3D

Ниже на рисунке 6.2 представлена проверка размеров модели с максимальными введенными параметрами в САПР Kompas 3D

Максимальные параметры корпуса спиннера:

Диаметр центрального кольца 42 мм;

Длина спиннера 90 мм;

Диаметр внутренних колец спиннера 60 мм;

Толщина спиннера 30 мм;

Радиус внешних колец спиннера 35 мм;

Скругление корпуса спиннера и внутренних колец 2 мм.

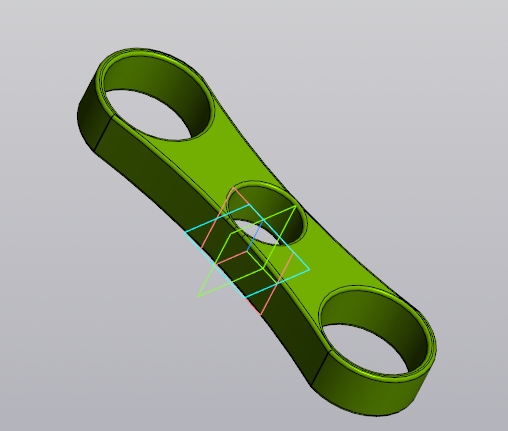
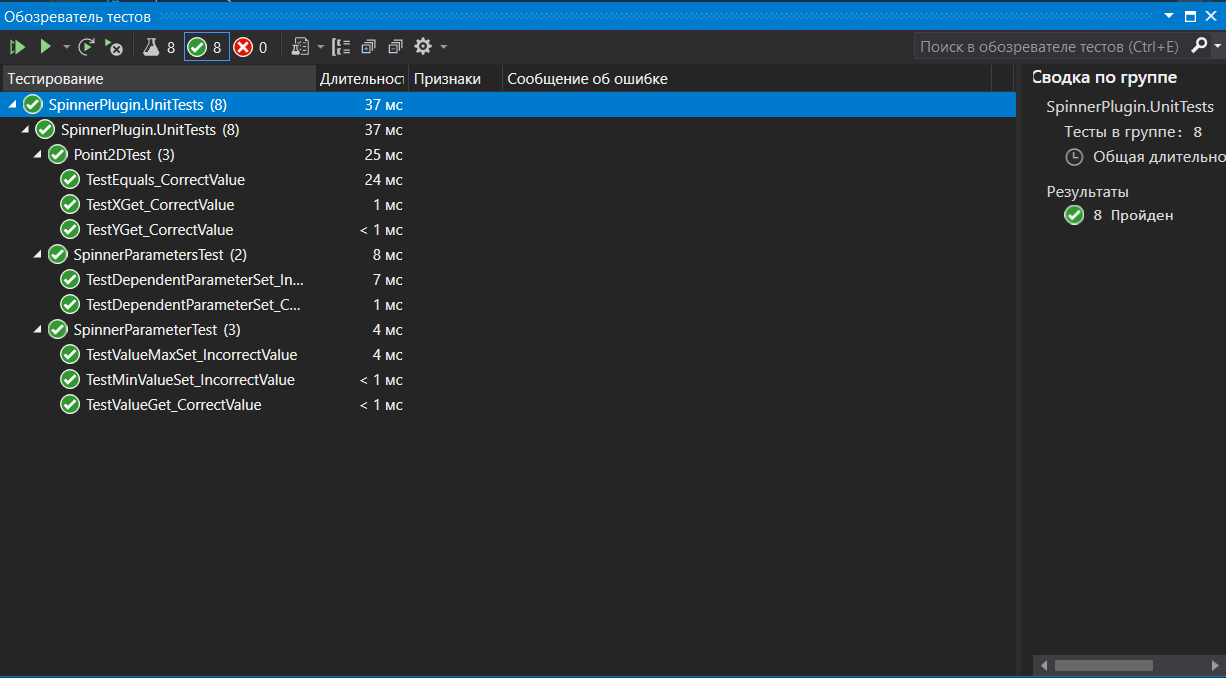


Рисунок 6.2 — Модель с максимально веденными параметрами в Kompas 3D

# 6.2 Модульное тестирование

В целях проверки корректности работы методов и свойств классов при помощи тестового фреймворка NUnit версии 3.13 проведено модульное тестирование [11], проверялись открытые поля и методы. На рисунке 6.3 представлено тестирование библиотеки классов: Model.

  
Рисунок 6.3 – Тестирование классов

Перечень тестов с их описание представлен в таблице 6.1

Таблица 6.1 – тестовые случаи метода TestDependentParameterSet\_CorrectValues.

|  |  |  |
| --- | --- | --- |
| Тестируемое свойство | Параметры | Описание |
| Diameter | 21 | Проверка корректного присвоения значения свойства |
| Length | 45 | Проверка корректного присвоения значения свойства |
| RadiusInnerRings | 30 | Проверка корректного присвоения значения свойства |

Таблица 6.2 – тестовые случаи метода TestDependentParameterSet\_IncorrectValues.

|  |  |  |
| --- | --- | --- |
| Тестируемое свойство | Параметры | Описание |
| Diameter | 60 | Проверка некорректного присвоения значения свойства |
| Length | 30 | Проверка некорректного присвоения значения свойства |
| RadiusInnerRings | 20 | Проверка некорректного присвоения значения свойства |

Таблица 6.3 – тестовые случаи метода TestValueGet\_CorrectValue.

|  |  |  |
| --- | --- | --- |
| Тестируемое свойство | Параметры | Описание |
| Thickness | 20 | Проверка корректного получения значения свойства |

Таблица 6.4 – тестовые случаи метода TestMinValueSet\_IncorrectValue.

|  |  |  |
| --- | --- | --- |
| Тестируемое свойство | Параметры | Описание |
| Thickness | 5 | Проверка некорректного присвоения значения свойства |

Таблица 6.5 – тестовые случаи метода TestMaxValueSet\_IncorrectValue.

|  |  |  |
| --- | --- | --- |
| Тестируемое свойство | Параметры | Описание |
| Thickness | 40 | Проверка некорректного присвоения значения свойства |

Таблица 6.6 – тестовые случаи метода TestXGet\_CorrectValue.

|  |  |  |
| --- | --- | --- |
| Тестируемое свойство | Параметры | Описание |
| X | 10 | Проверка корректного присвоения значения свойства |

Таблица 6.7 – тестовые случаи метода TestYGet\_CorrectValue.

|  |  |  |
| --- | --- | --- |
| Тестируемое свойство | Параметры | Описание |
| Y | 10 | Проверка корректного присвоения значения свойства |

# 6.3 Нагрузочное тестирование

В целях проверки производительности работы плагина, было проведено нагрузочное тестирование [12]. Тестирование производилось на ПК со следующей конфигурацией:

* ЦП Intel(R) Core(TM) i5-8300h CPU 2.30ГГц;
* 8 ГБ ОЗУ;
* графический процессор объемом памяти 8 ГБ.

На рисунке 6.4 для проведения нагрузочного, с каждым успешным построением фигуры производилась запись результатов в текстовый файл «StressTest.txt».

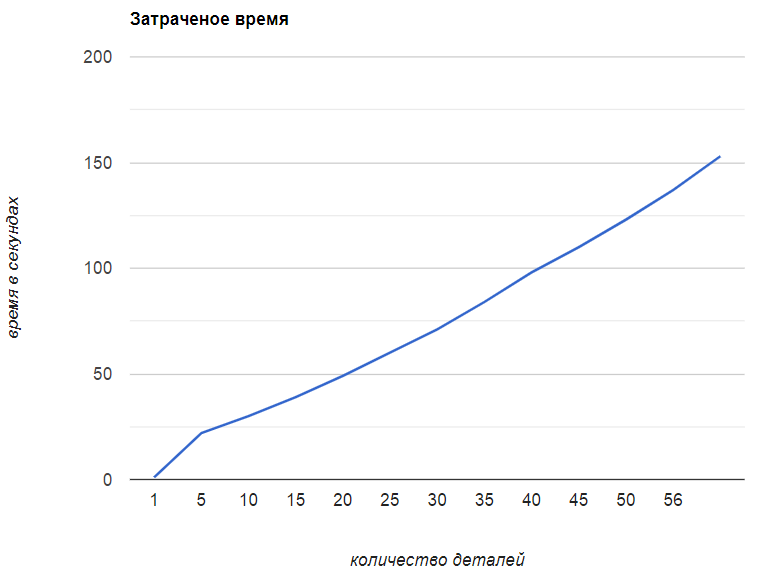


Рисунок 6.4 – Зацикливание перестроения фигуры

На графике, изображенном на рисунке 6.5 ось «Х» - количество построенных деталей, ось «Y» - количество потребляемой оперативной памяти. На графике, изображенном на рисунке 6.6 в текущей главе, ось «X» – время в минутах, ось «Y» – количество построенных деталей. На протяжении всех тестов (продолжительностью до сбоя Kompas 3D) общая загруженность процессора была в пределах 23 процентов.



Рисунок 6.5 – График зависимости загруженности памяти от количества деталей

  
Рисунок 6.6 – График зависимости времени от количества построенных деталей с параметрами по умолчанию

Исходя из графика, представленного на рисунке 6.5 можно сделать вывод, что зависимость является линейной пока оперативная память на загружена практически полностью. На графике имеются падения, 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.

Исходя из графика, представленного на рисунке 6.6 можно сделать вывод, 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.

В ходе нагрузочных тестов было замечено, что САПР Kompas 3D, экстренно прекращает свою работу по ошибке, при общем количестве построенных деталей 56 штук.

# Заключение

В ходе выполнения лабораторных работ были изучены предметная область проектирования, предмет проектирования, аналоги предмета проектирования, API, функциональное и нагрузочное тестирование и на основании полученных данных были спроектированы UML диаграммы классов, разработан плагин для создания 3D моделей «Корпус спиннера» в САПР Kompas 3D, и проведено функциональное и нагрузочное тестирование плагина.

# Список использованных источников

1. Автоматизация вычислительных процедур в прикладных задачах инженерного проектирования [Электронный ресурс]. – URL: <https://scienceforum.ru/2014/article/2014000201> (дата обращения: 24.12.2022).
2. Visual Studio [Электронный ресурс]. – URL: <https://visualstudio.microsoft.com/ru/> (дата обращения: 24.12.2022).
3. КОМПАС-3D. Официальный сайт САПР КОМПАС [Электронный ресурс]. — Режим доступа: <https://kompas.ru/> (дата обращения 24.12.2022).
4. Спиннер — Википедия. [Электронный ресурс]. — Режим доступа: <https://ru.wikipedia.org/wiki/Спиннер> (дата обращения 24.12.2022).
5. КОМПАС-3D для разработчиков [Электронный ресурс]. – URL: <https://kompas.ru/solutions/developers/> (дата обращения: 24.12.2022).
6. NUnit [Электронный ресурс]. – URL: <https://nunit.org/> (дата обращения: 24.12.2022).
7. Платформа .NET Framework 4.7.2 для Windows [Электронный ресурс]. – URL: <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: 24.12.2022).
8. Solidworks [Электронный ресурс]. – Режим доступа: <https://www.solidworks.com/> (дата обращения 24.09.2022).
9. UML. [Электронный ресурс]. – Режим доступа: http://www.uml.org/ (дата обращения 24.12.2022).
10. Функциональное тестирование [Электронный ресурс]. – URL: <https://daglab.ru/funkcionalnoe-testirovanie-programmnogo-obespechenija/> (дата обращения: 24.12.2022).
11. Юнит-тестирование для чайников [Электронный ресурс]. – URL: <https://habr.com/ru/post/169381/> (дата обращения: 24.12.2022).
12. Нагрузочное тестирование: с чего начать и куда смотреть [Электронный ресурс]. – URL: <https://habr.com/ru/company/jugru/blog/329174/> (дата обращения: 24.12.2022).