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 к лабораторному проекту

по дисциплине «Основы разработки САПР»

«Плагин моделирования самонарезающего винта

для системы Компас-3D v18.1»

Выполнил:

студент гр. 587-2

\_\_\_\_\_\_\_\_\_\_\_ Волков Л. Н.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2021 г.

Руководитель:

к.т.н., доцент каф. КСУП

\_\_\_\_\_\_\_\_ Калентьев А. А.

«\_\_\_\_» \_\_\_\_\_\_\_\_\_ 2021 г.

Томск 2021 г.

Оглавление

[1 Описание САПР 3](#_Toc70587587)

[1.1 Описание системы проектирования 3](#_Toc70587588)

[1.2 Описание API 3](#_Toc70587589)

[1.3 Обзор аналогов 7](#_Toc70587590)

[1.3.1 Плагин Механика: Пружины для КОМПАС-3D 7](#_Toc70587591)

[1.3.2 Плагин Штампы 3D для КОМПАС-3D 8](#_Toc70587592)

[2 Описание предмета проектирования 10](#_Toc70587593)

[3 Проект программы 12](#_Toc70587594)

[3.1 Диаграмма классов 12](#_Toc70587595)

[3.2 Макет пользовательского интерфейса 14](#_Toc70587596)

[4 Тестирование 17](#_Toc70587597)

[4.1 Функционально тестирование 17](#_Toc70587598)

[4.2 Модульное тестирование 24](#_Toc70587599)

[4.3 Нагрузочное тестирование 25](#_Toc70587600)

[Список литературы 26](#_Toc70587601)

[Приложение А 27](#_Toc70587602)

# 1 Описание САПР

# 1.1 Описание системы проектирования

Система автоматизированного проектирования (САПР) – сложный комплекс средств, предназначенный для автоматизации проектирования [1]. Однако, в настоящее время термин САПР утратили свой первоначальный смысл и теперь обозначает программу для автоматизированного проектирования [2]. В данный момент на рынке существуешь широкий выбор программ, которые отличаются между собой как по функциональности: AutoCAD, Bricscad, Autodesk Inventor, SolidWorks, SolidEdge, но для системы, к которой будет разработан плагин была выбрана Компас-3D v.18.1.

Компас-3D – это система параметрического моделирования деталей и сборок, используемая в областях машиностроения, приборостроения и строительства. Разработчик – компания Аскон (Россия) [3].

Система «Компас-3D» предназначена для создания трёхмерных ассоциативных моделей отдельных деталей и сборочных единиц, содержащих как оригинальные, так и стандартизованные конструктивные элементы. Многочисленные сервисные функции облегчают решение вспомогательных задач проектирования и обслуживания производства. Ключевой особенностью «Компас-3D» является использование собственного математического ядра «C3D» и параметрических технологий, разработанных специалистами компании «Аскон» [4].

# 1.2 Описание API

Application Programming Interface (API) – описание способов (набор классов, процедур, функций, структур или констант), которыми одна компьютерная программа может взаимодействовать с другой программой. API определяет функциональность, которую предоставляет программа (модуль, библиотека), при этом API позволяет абстрагироваться от того, как именно эта функциональность реализована [5].

Главным интерфейсом API системы КОМПАС-3D является KompasObject. Получить указатель на этот интерфейс 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 [6].

Основные свойства и методы интерфейса KompasObject, представлены в таблицах 1.1 – 1.4.

Таблица 1.1. – Методы интерфейса KompasObject.

|  |  |  |  |
| --- | --- | --- | --- |
| Метод | Входные параметры | Возвращаемые параметры | Описание |
| Document3D() |  | Указатель на интерфейс документа трехмерной модели ksDocument3D | Дает возможность получить указатель на интерфейс трехмерного документа (деталь или сборки) |
| ksAttachkompasLibrary |  |  | Подключает библиотеку |
| GetMathematic2D() |  | Указатель на интерфейс ksMathematic2D | Возвращает указатель на интерфейс для работы с математическими функциями в графическом документе |
| GetParamStruct | structType - тип интерфейса параметров | указатель на интерфейс указанного ти­па из StructType2D | Один из самых важных методов. Позволяет получить интерфейс структуры параметров объекта определенного типа (например, параметры прямоугольника, эллипсиса, штриховки, размеров и т.д.) |

Продолжение таблицы 1.1

|  |  |  |  |
| --- | --- | --- | --- |
| Метод | Входные параметры | Возвращаемые параметры | Описание |
| GetDynamicArray | ext – расширение имени файла,  filter – фильтр поиска (0 – фильтр формируется автоматически),  preview – признак подключения окна предварительного просмотра:  с подключением окна, без подключения окна  typeDir – стартовая папка | Строка с именем файла | Возвращает указатель на интерфейс динамического массива |

Таблица 1.2. – Методы интерфейса ksDocument3D.

|  |  |  |  |
| --- | --- | --- | --- |
| Метод | Входные параметры | Возвращаемые параметры | Описание |
| Create | invisible-признак режима редактирования документа (TRUE-невидимый режим, FALSE-видимый режим),  typeDoc-тип документа (TRUE-деталь, FALSE-сборка). | TRUE - в случае успешного завершения. | Дает возможность создать пустой документ (деталь или сборку) |
| UpdateDocumentParam |  | TRUE – в случае успешного завершения | Позволяет обновить настройки документа |
| GetPart | Type - тип компонента из перечисления | Указатель на интерфейс компонента ksPart или IPart | Возвращает указатель на интерфейс компонента (деталь или подсборки) в сборке |
| EntityСollection | objType - тип объектов, содержащихся в массиве. | Указатель на интерфейс ksEntityCollection или IEntityCollection | Дает возможность получить указатель на массив элементов, выбранных в документе (например, операций и компонентов, сборки для их копирования по массиву) |

Продолжение таблицы 1.2

|  |  |  |  |
| --- | --- | --- | --- |
| Метод | Входные параметры | Возвращаемые параметры | Описание |
| GetPart() | type - тип компонента из перечисления | Указатель на интерфейс компонента ksPart или IPart | Получить указатель на интерфейс компонента в соответствии с заданным типом |
| NewEntity() | objType - тип объекта | Указатель на интерфейс ksEntity или IEntity | Создать новый интерфейс объекта и получить указатель на него |

Таблица 1.3. – Методы интерфейса ksPart

|  |  |  |  |
| --- | --- | --- | --- |
| Метод | Входные параметры | Возвращаемые параметры | Описание |
| BeginEdit |  | SketchDoc – указатель на интерфейс IFragmentDocument | Позволяет запустить режим редактирования компонента |
| EndEdit |  | TRUE – в случае успешного завершения;  FALSE – в случае неудачи | Закрывает режим редактирования компонента «на мести» |
| EntityCollection | objType  - тип объектов | указатель на интерфейс ksEntityCollection или IEntityCollection - в случае успеха, NULL - в случае неудачи. | Формирует динамический массив трехмерных объектов и возвращает указатель на его интерфейс |
| GetDefaultEntity | objType – тип объекта | Указатель на интерфейс ksEntity или IEntity | Возвращает указатель на интерфейс объекта, создаваемого системой в трехмерном документе по умолчанию .Таких объектов всего четыре: начало координат и три ортогональных плоскости |
| GetPart | Type - тип компонента. | Указатель на интерфейс компонента ksPart или IPart | Позволяет получить указатель на интерфейс компонента |

Продолжение таблицы 1.3

|  |  |  |  |
| --- | --- | --- | --- |
| Метод | Входные параметры | Возвращаемые параметры | Описание |
| GetPlacement |  | Указатель на интерфейс компонента | Дает возможность получить указатель на интерфейс компонента |
| NewEntity | objType – тип объекта | Указатель на интерфейс ksEntity или IEntity | Создает интерфейс нового трёхмерного объекта и возвращает указатель на него |

Таблица 1.4 – Методы интерфейса ksDocument2D

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Входные параметры | Возвращаемые параметры | Описание |
| ksRectangle() | param - указатель на интерфейс параметров прямоу­гольника ksRectangleParam,  centre - признак построения обозначения центра. | указатель на прямоугольник. | Создать прямоугольник |
| ksCircle() | xc, yc - координаты центра окружности,  rad - радиус окружности,  style - стиль линии | указатель на окружность - в случае удачного завершения,  0 - в случае неудачи | Создать окружность |

# 1.3 Обзор аналогов

# 1.3.1 Плагин Механика: Пружины для КОМПАС-3D

Приложение позволяет выполнять проектные и проверочные расчеты пружин сжатия, растяжения, кручения, а также тарельчатых, конических и фасонных пружин. По результатам расчетов автоматически формируются чертежи и 3D-модели [7].

В основу приложения положены следующие методики расчета:

* Пружины сжатия и растяжения — ГОСТ 13764-86, ГОСТ 13765-86;
* Тарельчатые пружины — ГОСТ 3057-90;
* Пружины кручения — методика из книги В.И. Анурьев «Справочник конструктора-машиностроителя» том 3;
* Конические и фасонные пружины — методика из книги С. Д. Пономарёв, Л. Е. Андреева «Расчет упругих элементов машин и приборов».

В результате проектного расчета система предлагает множество решений, удовлетворяющих исходным данным, из которых конструктор может выбрать оптимальное по одному или нескольким критериям.

При создании чертежа пружины возможны выбор типа зацепов, автоматическая постановка размеров, выносных видов, диаграмм деформаций или усилий.

После вставки модели пружины в сборку можно изменять длину пружины, что позволяет выставить деталь в рабочее состояние или промежуточное.

Как показывает практика, использование приложения Пружины позволяет в 15–20 раз повысить скорость проектирования и выпуска конструкторской документации пружин.

# 1.3.2 Плагин Штампы 3D для КОМПАС-3D

Приложение Штампы 3D предназначено для автоматизации конструкторских и технологических работ при проектировании штампов для изделий из листового материала. Позволяет формировать комплект технической документации, необходимой для выпуска штампа [8].

Приложение обеспечивает:

* Разворачивание исходной детали в заготовку (развертку) и создание шагов трансформации деталь-заготовка.
* Проектирование полосы. Для обеспечения высокой производительности и экономии материала приложение автоматически сформирует оптимальный раскрой- рабочую зону штампа.
* Проектирование пуансонов. Задается шаговое размещение и конфигурация разделительных и формообразующих пуансонов.
* Проектирование пакета штампа. Центр давления штампа определяется автоматически. В зависимости от марки и толщины материала формируется зазор между пуансоном и матрицей. Уточняются конфигурация и положение пуансонов. Пресс выбирается на основе его характеристик и габаритов штампа из списка наиболее используемого на предприятиях оборудования.
* Автоматическое формирование в соответствии с ЕСКД комплекта документации, необходимой для выпуска штампа (3D-моделей, сборочных чертежей, спецификаций, деталировок).

Приложение содержит:

* Базу данных прессового оборудования, которая включает более 20 моделей прессов.
* Базу знаний конструкций штампов с возможностью ее расширения с учетом дополнительных требований пользователя.
* Параметрические библиотеки 3D деталей ГОСТ и конструктивных элементов штампов.

Приложение позволяет:

* Проектировать разделительные штампы: с жестким съемником, с верхним прижимом, штампы совмещенного действия.
* Проектировать штампы последовательного действия с совмещением операций как разделительных, так и формообразующих. В сочетании как с верхним, так и нижним прижимом.
* Проектировать гибочные штампы.

# 2 Описание предмета проектирования

Предметом проектирования является самонарезающий винт.

Самонарезающий винт, или саморез, – это крепёжное изделие в виде стержня с головкой и специальной наружной резьбой, образующей внутреннюю резьбу в отверстии соединяемого предмета. Имеет треугольную резьбу на цилиндрической поверхности, полностью или не полностью закрывающую цилиндрическую поверхность.

Саморез, построенный разработанным плагином, будет иметь шестигранную головку и острый конец.

Параметры самонарезающего винта:

1. Внутренний диаметр головки – D, от 3.0 до 18.0 мм;
2. Высота головки – K, от 0.96 до 5.0 мм;
3. Общая длина стержня – l, от 7.0 до 100.0 мм;
4. Длина части стержня с резьбой – b, от 3.0 до 97.0 мм;
5. Диаметр резьбы – d, от 1.6 до 10.0 мм;
6. Общий диаметр стержня – d1, от 1.6 до 10.0 мм;
7. Внутренний диметр резьбы – d2, от 1.1 до 7.0 мм;
8. Шаг резьбы – P, от 0.8 до 4.5 мм.

Модель будет иметь следующие зависимости:

1. b < l;
2. d2 < d < D;
3. d2 < d1 < D.

Данные параметры изображены на рисунке 2.1.



Рисунок 2.1 – Чертеж самонарезающего винта с нанесенными параметрами

# 3 Проект программы

# 3.1 Диаграмма классов

Диаграмма классов определяет типы классов системы и различного рода статические связи, которые существуют между ними. На диаграммах классов изображаются также атрибуты классов, операции классов и ограничения, которые накладываются на связи между классами [10].

На рисунке 3.1 представлена диаграмма классов проекта.

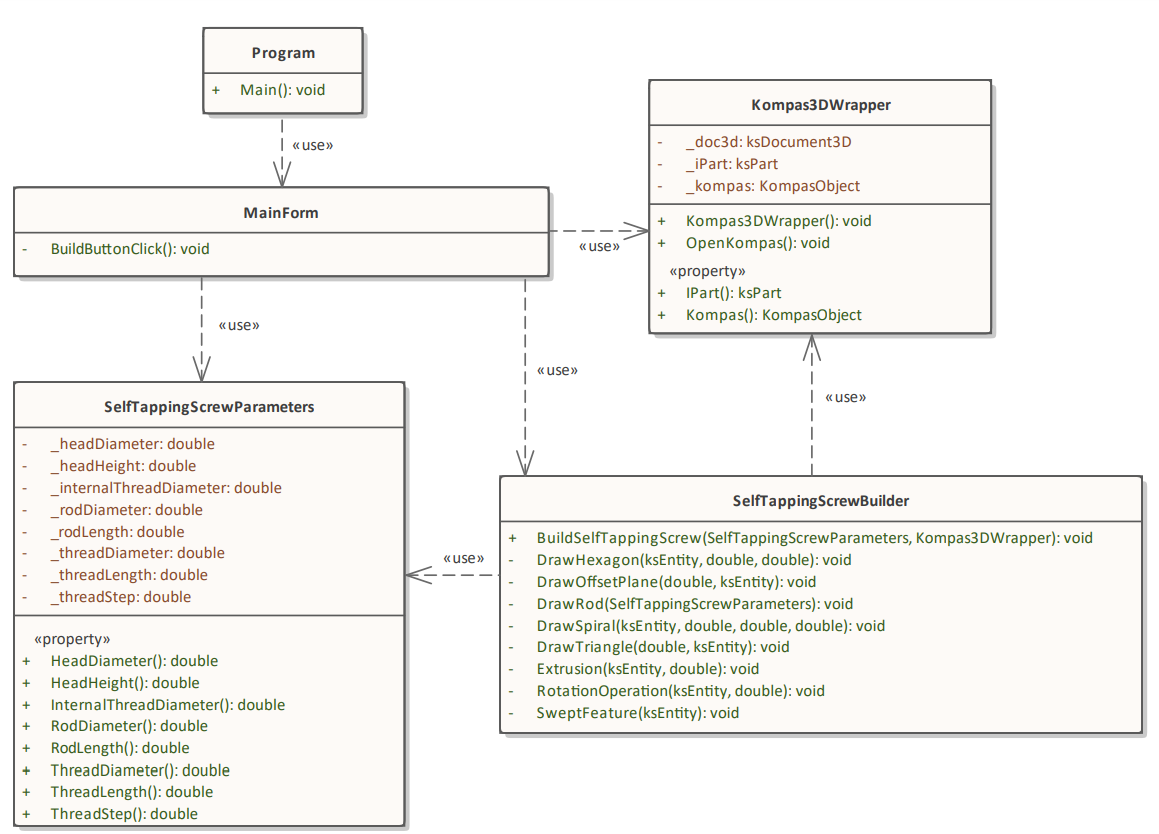


Рисунок 3.1 – Диаграмма классов

В программе будут реализованы следующие классы:

1. MainForm – класс формы, ответственный за взаимодействие с пользователем;
2. SelfTappingScrewBuilder – класс, осуществляющий вызов методов API, необходимых для постройки модели;
3. SelfTappingScrewParameters – класс для хранения и проверки параметров модели;
4. Kompas3DWrapper – класс для работы с API КОМПАС 3D.

После реализации проекта и добавления дополнительной функциональности, в диаграмму классов был внесен ряд изменений:

1. В класс MainForm добавлен метод CheckInput для проверки типа введенных значений, установки цвета текстового поля и вывода сообщения об ошибке.

Добавлены 4 обработчика событий для установки максимальных, минимальных параметров, параметров по умолчанию и очистки полей значений в текстовые поля: maxParametersRadioButton\_CheckedChanged, minParametersRadioButton\_CheckedChanged, defaultParametersRadioButton\_CheckedChanged, manualInputRadioButton\_CheckedChanged.

Добавлены обработчик событий для текстового поля TextBox\_Leave для обработки покидания текстового поля;

Добавлен метод SetParameters для установки минимальных, максимальных и значений по умолчанию, установки пустых полей.

1. На форму добавлены: 1 флаговая кнопка для реализации дополнительной функциональности, 4 радиокнопки для автоматической установки значений в текстовые поля.
2. Добавление перечисление ParametersName для хранения названий параметров.
3. В класс SelfTappingScrewParameters добавлено поле и свойство для шайбы – Washer, добавлены 3 статических словаря для хранения минимальных, максимальных параметров и значений по умолчанию.

Также добавлена функция для выбрасывания исключений с заданным текстовым сообщением.

1. В классе Kompas3DWrapper оставлены только метод OpenKompas и свойства KompasObject, KsPart. Остальные удалены за ненадобностью.
2. В классе SelfTappingScrewBuilder был убран метод DrawOffsetPlane, он был объединен в метод MakeThread. Был добавлен метод BuildWasher для отрисовки шайбы. Также некоторые методы изменили свое название.

На рисунке 3.2 представлена диаграмма классов проекта, после реализации проекта.

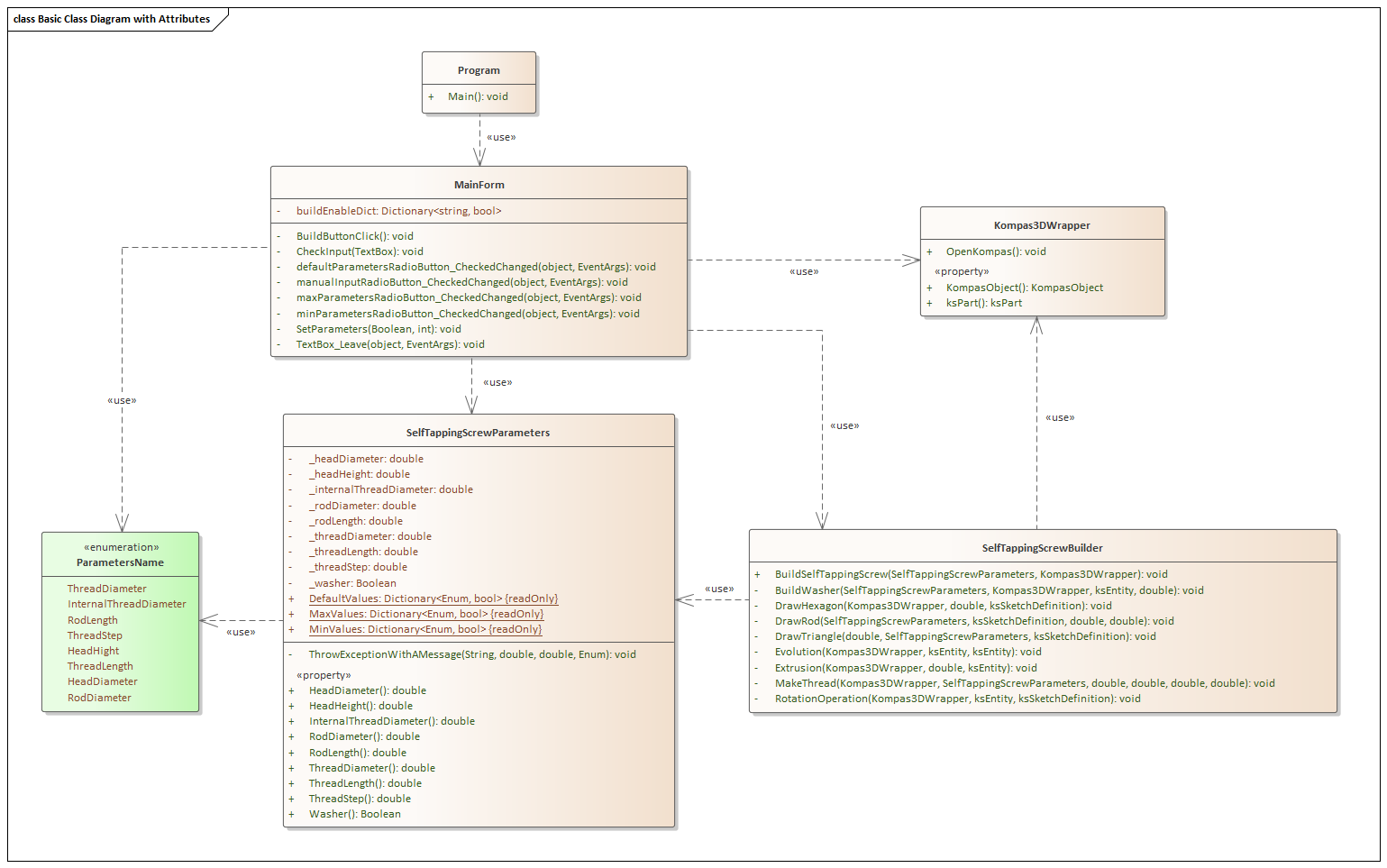


Рисунок 3.2 – Диаграмма классов после реализации и добавления дополнительной функциональности

# 3.2 Макет пользовательского интерфейса

Макет пользовательского интерфейса представляет собой форму для ввода параметров. Окно плагина делится на две зоны: зона с чертежом результирующей модели, на котором графически представлены параметры самонарезающего винта, и зона с текстовыми полями для ввода числовых значений параметров. Ниже этих двух зон кнопка для подтверждения ввода значений.

На рисунке 3.3 представлен макет пользовательского интерфейса.

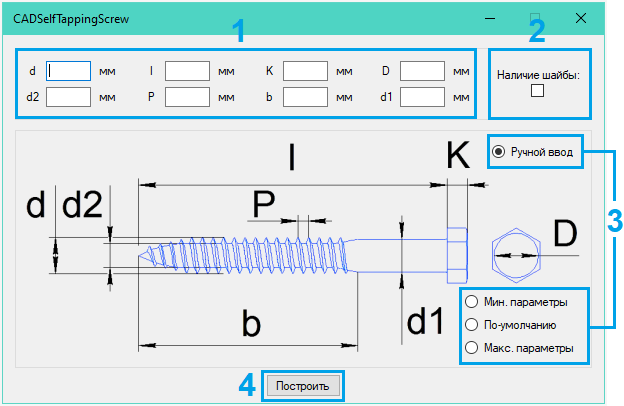


Рисунок 3.3 – Макет пользовательского интерфейса

В зоне 1 находятся текстовые поля для ввода значений параметров.

В зоне 2 находится флаговая кнопка для добавления дополнительной функциональности.

В зонах под номером 3 находятся радиокнопки для установки соответствующих параметров в текстовые поля зоны 1, либо очистки текстовых полей

В зоне 4 располагается кнопка для постройки итоговой модели.

После ввода некорректных параметров и нажатия кнопки подтверждения «Построить», тестовое поле с некорректно введенным значением выделится красным цветом. Пример ввода некорректно введенных значений представлен на рисунке 3.4.

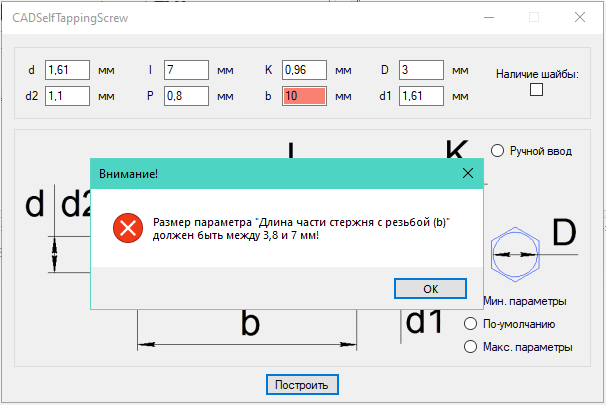


Рисунок 3.4 – Макет пользовательского интерфейса при введенном некорректном параметре

# 4 Тестирование

# 4.1 Функционально тестирование

Функциональное тестирование – это тестирование ПО в целях проверки реализуемости функциональных требований в соответствии с указанными в техническом задании.

В данном случае будет проверяться правильность построения модели при различных входных параметрах.

Построение модели с минимальными параметрами. Значение входных данных указаны в таблице 4.1.

Таблица 4.1 – Минимальные значения параметров

|  |  |
| --- | --- |
| Название параметра | Значение параметра (мм) |
| Диаметр резьбы | 1,61 |
| Внутренний диметр резьбы | 1,1 |
| Общая длина стержня | 7 |
| Шаг резьбы | 0,8 |
| Высота головки | 0,96 |
| Длина части стержня с резьбой | 3,8 |
| Внутренний диаметр головки | 3 |
| Общий диаметр стержня | 1,61 |

Итоговая модель, построенная при минимальных входных параметрах и без шайбы представлена на рисунке 4.1.

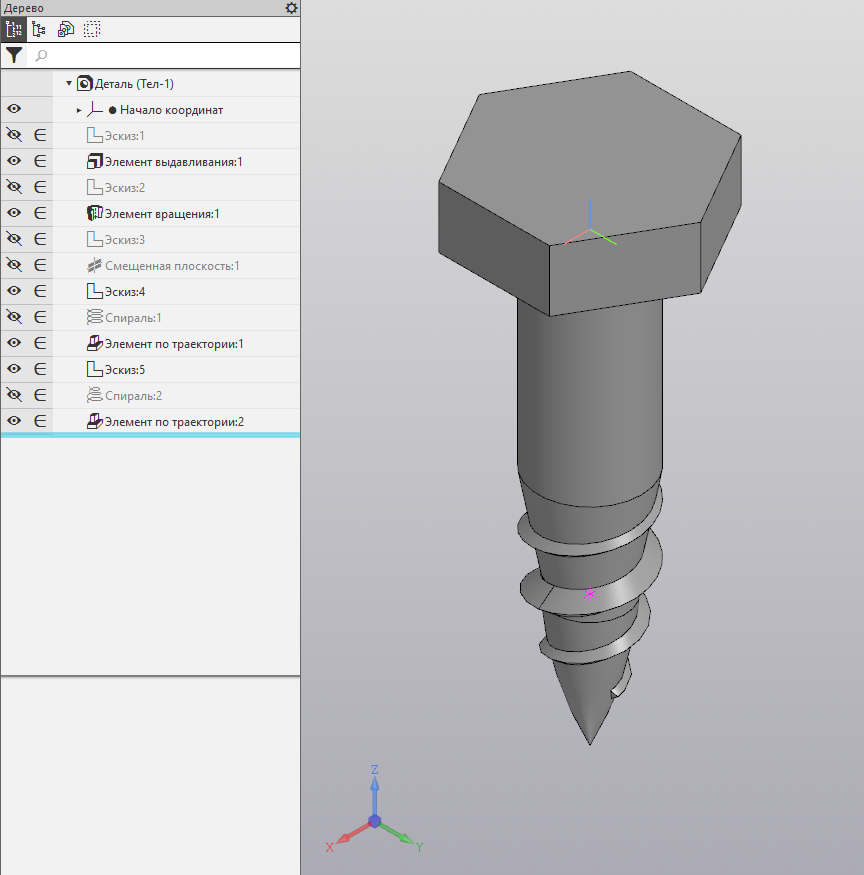


Рисунок 4.1 – Модель, построенная при минимальных входных параметрах и без шайбы

Итоговая модель, построенная при минимальных входных параметрах и с шайбой представлена на рисунке 4.2.

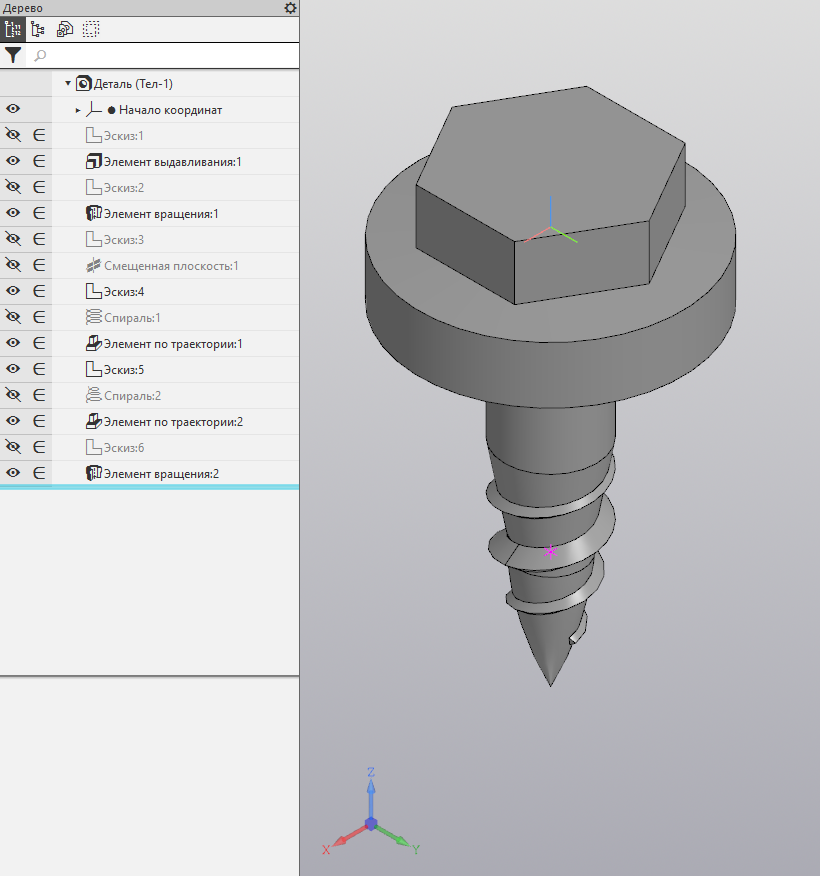


Рисунок 4.2 – Модель, построенная при минимальных входных параметрах и с шайбой

Построение модели с параметрами по умолчанию. Значение входных данных указаны в таблице 4.2.

Таблица 4.2 – Значения параметров по умолчанию

|  |  |
| --- | --- |
| Название параметра | Значение параметра (мм) |
| Диаметр резьбы | 7,5 |
| Внутренний диметр резьбы | 5 |
| Общая длина стержня | 70 |
| Шаг резьбы | 2,5 |
| Высота головки | 4 |
| Длина части стержня с резьбой | 50 |
| Внутренний диаметр головки | 12 |
| Общий диаметр стержня | 7 |

Итоговая модель, построенная при входных параметрах по умолчанию и без шайбы представлена на рисунке 4.3.

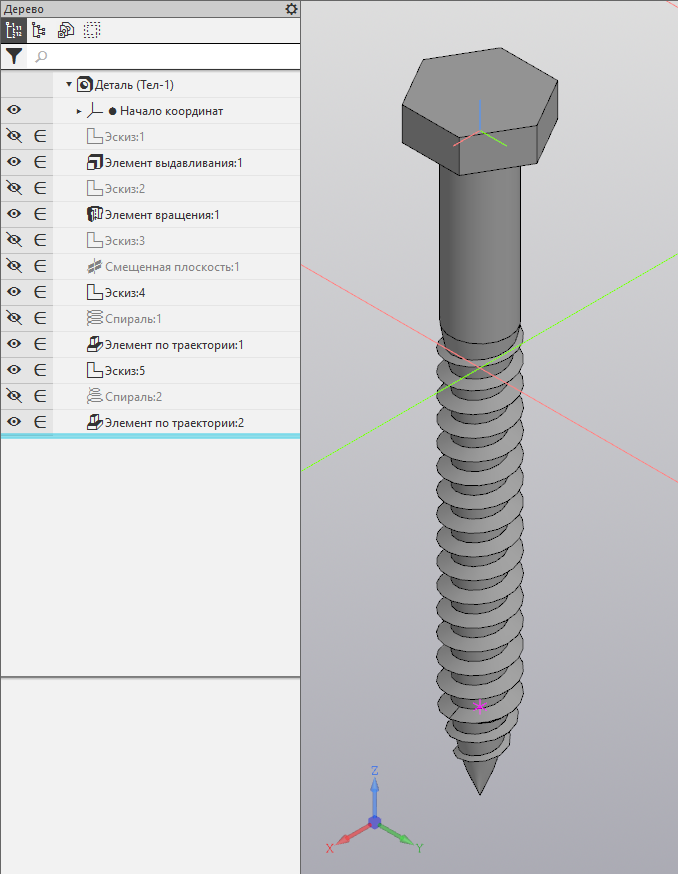


Рисунок 4.3 – Модель, построенная при входных параметрах по умолчанию и без шайбы

Итоговая модель, построенная при входных параметрах по умолчанию и с шайбой представлена на рисунке 4.4.

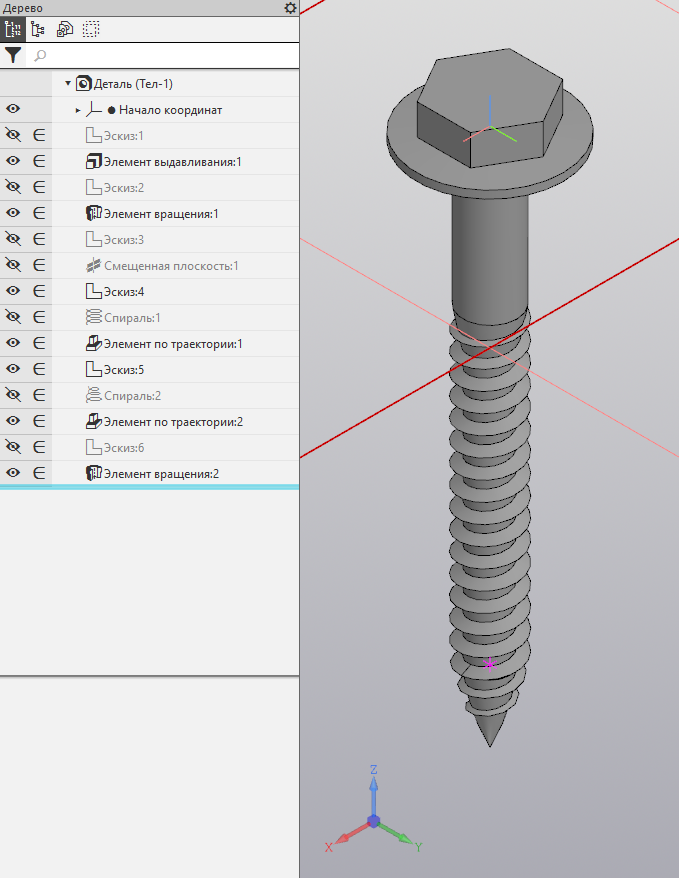


Рисунок 4.4 – Модель, построенная при входных параметрах по умолчанию и с шайбой

Построение модели с максимальными параметрами. Значение входных данных указаны в таблице 4.3.

Таблица 4.3 – Максимальные значения параметров

|  |  |
| --- | --- |
| Название параметра | Значение параметра (мм) |
| Диаметр резьбы | 9,9 |
| Внутренний диметр резьбы | 7 |
| Общая длина стержня | 100 |
| Шаг резьбы | 4,5 |
| Высота головки | 5 |
| Длина части стержня с резьбой | 97 |
| Внутренний диаметр головки | 18 |
| Общий диаметр стержня | 9,9 |

Итоговая модель, построенная при максимальных входных параметрах и без шайбы представлена на рисунке 4.5.

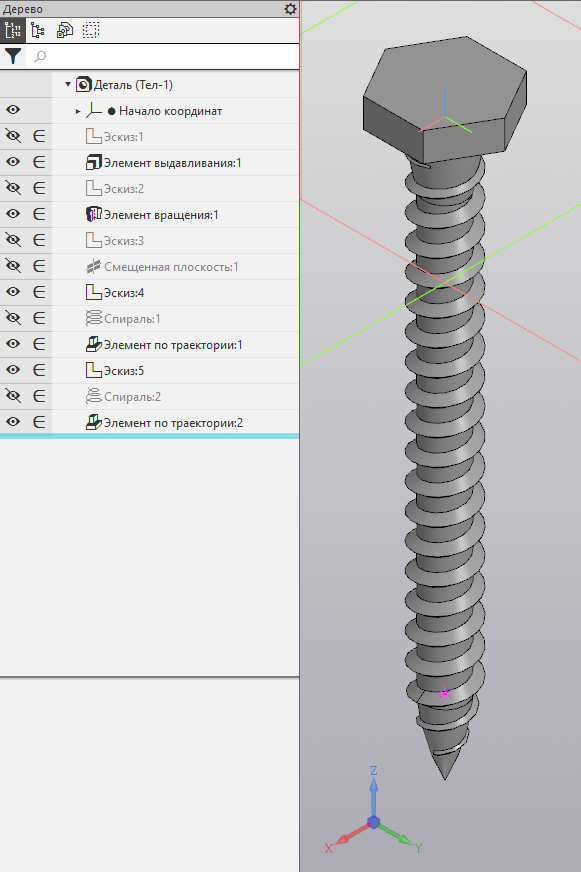


Рисунок 4.5 – Модель, построенная при максимальных входных параметрах и без шайбы

Итоговая модель, построенная при максимальных входных параметрах и с шайбой представлена на рисунке 4.6.

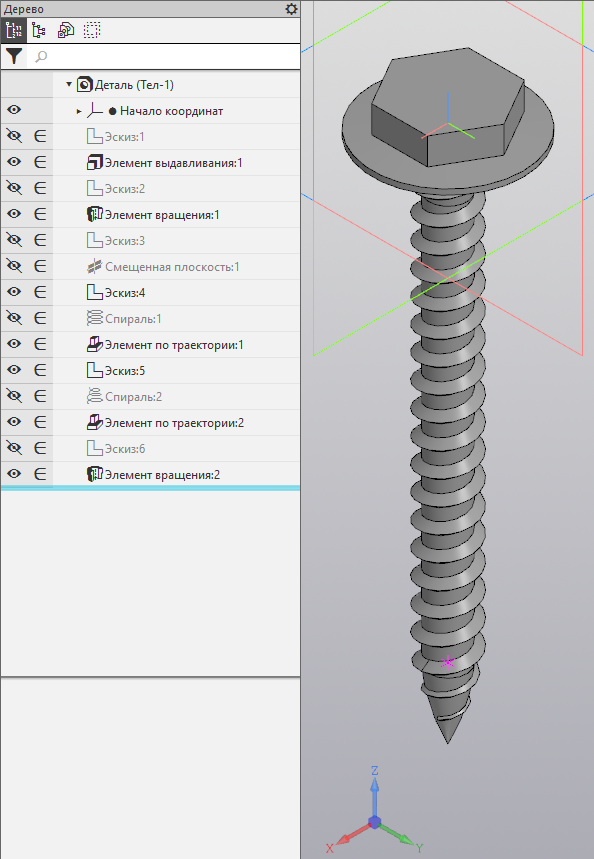


Рисунок 4.6 – Модель, построенная при максимальных входных параметрах и с шайбой

Если пользователь введет в текстовое поле значение, которое невозможно будет преобразовать в значение double, то, как только он покинет текстовое поле, то ему высветится сообщение о некорректно введенных данных, а текстовое поле выделится красным цветом. Если пользователь сотрет некорректные данные, то текстовое поле примет изначальны вид. Сообщение о некорректных данных представлено на рисунке 4.7.

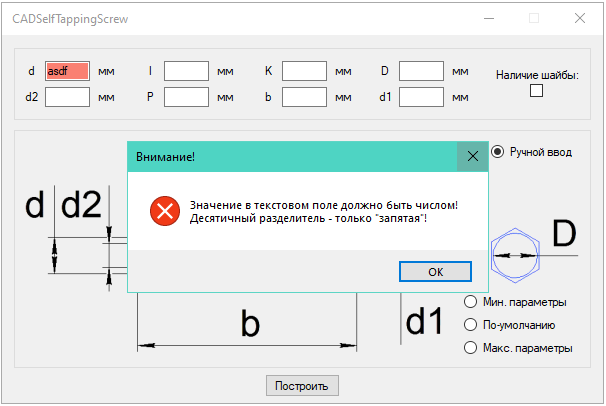


Рисунок 4.7 – Сообщение при некорректных данных

Если пользователь оставит поля незаполненными, то при нажатии на кнопку выведется сообщение о необходимости заполнить пустые поля. Данное сообщение представлено на рисунке 4.8.

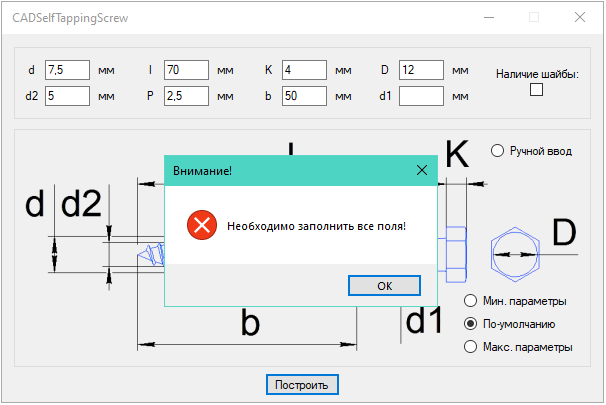


Рисунок 4.8 – Сообщение при незаполненных полях

Если пользователь введет в поле данные, непопадающие в диапазон, то ему высветится сообщение с текстом о том, в каком диапазоне должно быть значение, а текстовое поле с ошибочно введенными значениями окрасится в красный цвет. Сообщение о некорректных данных представлено на рисунке 4.9.

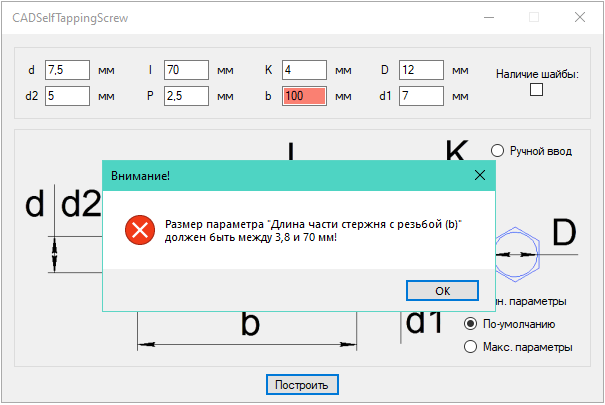


Рисунок 4.9 – Сообщение при некорректных данных

# 4.2 Модульное тестирование

Юнит-тестирование (англ. «unit-testing», или модульное тестирование) — тестирование программы в виде отдельных, изолированных друг от друга минимальных модулей. Условие изоляции тестируемого модуля от других необходимо для того, чтобы в случае обнаружения ошибки быть уверенным, что ошибка возникла именно в тестируемом модуле [9].

На основе таблицы приведенных в приложении А тестовых сценариев (таблица А.1), проводилось тестирование корректности входных параметров 3D-модели.

Тестирование проводилось с помощью программной платформы модульного тестирования NUnit 3.13.1 для языков платформы .NET.

Результаты прохождения всех модульных тестов приведены на рисунке 4.10. Все 25 тестов корректно пройдены.

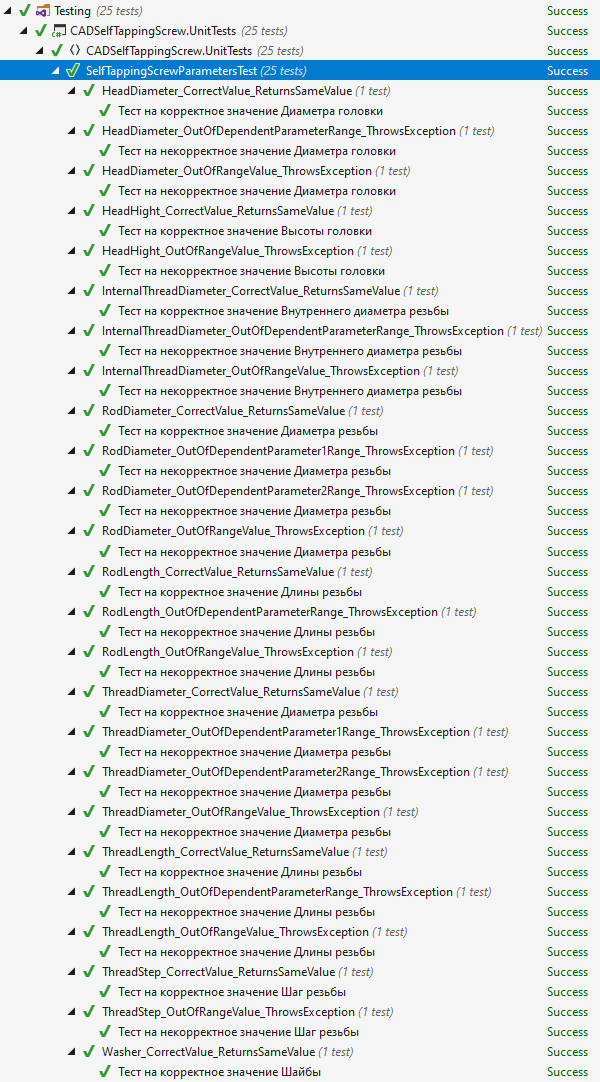


Рисунок 4.10 – Список пройденных юнит-тестов

Результат покрытия модуля SelfTappingScrewParameters тестами приведен на рисунке 4.11.

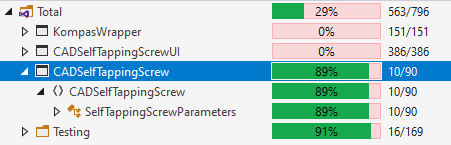


Рисунок 4.11 – Результат покрытия тестами

# 4.3 Нагрузочное тестирование

В целях проверки производительности работы плагина, было проведено нагрузочное тестирование [11]. Нагрузочное тестирование – это тестирование производительности, определение производительности и времени отклика устройства в ответ на внешний запрос с целью установления соответствия требованиям, которые были предъявлены к данному устройству.

Нагрузочное тестирование проводилось на компьютере со следующими характеристиками:

* Процессор: Intel Core i5-4670 3.40 ГГц;
* Оперативная память: 16GB DDR3 1333 МГц;
* Графический процессор: NVIDIA GeForce GTX 1050 Ti
* Операционная система: Windows 10 Pro x64

Нагрузочное тестирование проводилось с параметрами по умолчанию (Таблица 4.2). При задании максимального количества строящихся моделей равным 200, программа «КОМПАС-3D» после построения 137 модели программа завершилась со сбоем из-за нехватки памяти.

График зависимости потребления оперативной памяти от количества построенных моделей представлен на рисунке 4.12.

Рисунок 4.12 – Зависимость потребление оперативной памяти от количества построенных моделей

Как видно из графика на рисунке 4.12 зависимость росла в линейно до 24 модели, далее рост потребления оперативной памяти резко увеличился и зависимость продолжала расти и в пике достигла 7938 Мб, далее, после построения 137 модели, программа преждевременно завершилась со сбоем из-за нехватки памяти.

График зависимости времени построения от количества построенных моделей представлен на рисунке 4.13.

Рисунок 4.13 – Зависимость времени построения от количества построенных моделей

Как видно из графика на рисунке 4.13 время построения модели резко возрастает на построении 50 модели, но в целом было стабильным и в среднем составляло 3.77 секунд. В пике время построение достигало 6,05 секунд на 113 модели.

Исходя из результатов тестирования, можно сделать вывод, что программа выполняет заявленную функциональность в полном объеме. Результаты нагрузочного тестирования показали, что среда «КОМПАС-3D» не предназначена для построения подряд количества моделей более 50. Исходя из этого, рекомендуется строить и удерживать в памяти не более 20 моделей.

# Заключение

При выполнения лабораторных работ были изучены основные этапы проектирования программного продукта и его реализации, предметная область объекта проектирования, SDK «КОМПАС-3D». Было составлено техническое задание, разработан проект системы, составлена UML диаграмма классов, разработан макет пользовательского интерфейса.

В результате работы был разработан и реализован плагин для САПР «КОМПАС-3D», выполняющий построение 3D-модели торцевого ключа по задаваемым параметрам.

Над реализованным плагином были проведены функциональное, модульное и нагрузочное тестирование на платформе Windows 10.

В ходе выполнения данного проекта были изучены основные этапы проектирования и реализации ПО на примере создания библиотеки для построения модели самонарезающего винта для САПР «КОМПАС-3D v.18.1». Было составлено техническое задание, разработан проект системы, составлена UML диаграмма классов, разработан макет пользовательского интерфейса.

Также проведены модульные, функциональные и нагрузочные тесты разработанной библиотеки на платформе Windows 10.

# Список литературы

1. Система автоматизированного проектирования [Электронный ресурс] Википедия – Режим доступа: https://ru.wikipedia.org/wiki/Система\_автоматизированного\_проектирования (дата обращения: 23.03.2021)
2. Обзор популярных систем автоматизированного проектирования (CAD) [Электронный ресурс] «ПОИНТ» – Режим доступа: https://www.pointcad.ru/novosti/obzor-sistem-avtomatizirovannogo-proektirovaniya (дата обращения: 23.03.2021)
3. КОМПАС-3D. Сертифицировано ФСТЭК [Электронный ресурс] Официальный сайт «АСКОН» – Режим доступа: https://ascon.ru/products/1178/review/ (дата обращения: 23.03.2021)
4. КОМПАС-3D [Электронный ресурс] Официальный сайт «АСКОН» – Режим доступа: https://ascon.ru/products/7/review/ (дата обращения: 23.03.2021)
5. API [Электронный ресурс] Википедия – Режим доступа: https://ru.wikipedia.org/wiki/API (дата обращения: 23.03.2021)
6. Кидрук, М. И. КОМПАС-3D V10 на 100 – СПб., ООО «Питер», 2009. – 980 с.
7. Механика: Пружины [Электронный ресурс] Официальный сайт КОМПАС-3D – Режим доступа: https://kompas.ru/kompas-3d/application/machinery/springs/ (дата обращения: 23.03.2021)
8. Штампы 3D [Электронный ресурс] Официальный сайт КОМПАС-3D – Режим доступа: https://kompas.ru/kompas-3d/application/machinery/stamp-3d/ (дата обращения: 23.03.2021)
9. Фаулер, М. UML. Основы, 3-е издание / М. Фаулер; пер. с англ. А. Петухов – СПб: Издательство Символ-Плюс, 2018. – 192 с.
10. Горяинов А.Е. Новые технологии в программировании: учебное пособие / А.А.Калентьев, Д.В.Гарайс, А.Е.Горяинов — Томск: Эль Контент, 2014. —176 с.
11. Нагрузочное тестирование [Электронный ресурс]. – URL: https://daglab.ru/nagruzochnoe-testirovanie-programmnogo-obespechenija/ (дата обращения: 29.04.2021);

# Приложение А

Таблица А.1 – Тестовые сценарии

|  |  |  |
| --- | --- | --- |
| Тестовый метод | Входные параметры | Описание тестового случая |
| HeadDiameter\_CorrectValue\_  ReturnsSameValue() | HeadDiameter = 5 | Тест на корректное значение Диаметра головки |
| HeadDiameter\_OutOfRangeValue\_  ThrowsException() | HeadDiameter = 1000 | Тест на некорректное значение Диаметра головки |
| HeadDiameter\_  OutOfDependentParameterRange\_  ThrowsException() | ThreadDiameter = 5,  HeadDiameter = 5 | Тест на некорректное значение Диаметра головки |
| HeadHight\_CorrectValue\_  ReturnsSameValue() | HeadHight = 3 | Тест на корректное значение Высоты головки |
| HeadHight\_OutOfRangeValue\_  ThrowsException() | HeadHight = 1000 | Тест на некорректное значение Высоты головки |
| InternalThreadDiameter\_CorrectValue\_  ReturnsSameValue() | InternalThreadDiameter = 5 | Тест на корректное значение Внутреннего диаметра резьбы |
| InternalThreadDiameter\_  OutOfRangeValue\_  ThrowsException() | InternalThreadDiameter = 1000 | Тест на некорректное значение Внутреннего диаметра резьбы |
| InternalThreadDiameter\_  OutOfDependentParameterRange\_  ThrowsException() | ThreadDiameter = 5,  HeadDiameter = 5.1,  RodDiameter = 5,  InternalThreadDiameter = 5 | Тест на некорректное значение Внутреннего диаметра резьбы |
| RodDiameter\_CorrectValue\_  ReturnsSameValue() | RodDiameter = 5 | Тест на корректное значение Диаметра резьбы |
| RodDiameter\_OutOfRangeValue\_  ThrowsException() | RodDiameter = 1000 | Тест на некорректное значение Диаметра резьбы |

Продолжение таблицы А.1

|  |  |  |
| --- | --- | --- |
| RodDiameter\_  OutOfDependentParameter1Range\_  ThrowsException() | InternalThreadDiameter = 5,  RodDiameter = 5 | Тест на некорректное значение Диаметра резьбы |
| RodDiameter\_  OutOfDependentParameter2Range\_  ThrowsException() | HeadDiameter = 5,  RodDiameter = 5 | Тест на некорректное значение Диаметра резьбы |
| RodLength\_CorrectValue\_  ReturnsSameValue() | RodLength = 9 | Тест на корректное значение Длины резьбы |
| RodLength\_OutOfRangeValue\_  ThrowsException() | RodLength = 1000 | Тест на некорректное значение Длины резьбы |
| RodLength\_  OutOfDependentParameterRange\_  ThrowsException() | ThreadDiameter = 9,  RodLength = 9 | Тест на некорректное значение Длины резьбы |
| ThreadDiameter\_CorrectValue\_  ReturnsSameValue() | ThreadDiameter = 5 | Тест на корректное значение Диаметра резьбы |
| ThreadDiameter\_OutOfRangeValue\_  ThrowsException() | ThreadDiameter = 1000 | Тест на некорректное значение Диаметра резьбы |
| ThreadDiameter\_  OutOfDependentParameter1Range\_  ThrowsException() | InternalThreadDiameter = 5,  ThreadDiameter = 5 | Тест на некорректное значение Диаметра резьбы |
| ThreadDiameter\_  OutOfDependentParameter2Range\_  ThrowsException() | HeadDiameter = 5,  ThreadDiameter = 5 | Тест на некорректное значение Диаметра резьбы |
| ThreadLength\_CorrectValue\_  ReturnsSameValue() | ThreadLength = 50 | Тест на корректное значение Длины резьбы |
| ThreadLength\_OutOfRangeValue\_  ThrowsException() | ThreadLength = 1000 | Тест на некорректное значение Длины резьбы |
| ThreadLength\_  OutOfDependentParameterRange\_  ThrowsException() | RodLength = 50,  ThreadLength = 50 | Тест на некорректное значение Длины резьбы |
| ThreadStep\_CorrectValue\_  ReturnsSameValue() | ThreadStep = 3 | Тест на корректное значение Шаг резьбы |

Продолжение таблицы А.1

|  |  |  |
| --- | --- | --- |
| ThreadStep\_OutOfRangeValue\_  ThrowsException() | ThreadStep = 1000 | Тест на некорректное значение Шаг резьбы |
| Washer\_CorrectValue\_  ReturnsSameValue() | Washer = true | Тест на корректное значение Шайбы |