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)

**РАЗРАБОТКА ПЛАГИНА «ТОРЦЕВОЙ КЛЮЧ» ДЛЯ «КОМПАС-3D v18.1»**

Пояснительная записка к лабораторному проекту

по дисциплине «Основы разработки САПР»

Выполнил:

студент гр.587-3

\_\_\_\_\_\_\_\_\_\_\_\_\_\_Андреевская О. С.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Руководитель:

к.т.н., доцент каф. КСУП

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

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Томск 2021

Оглавление

[1 Описание САПР 3](#_Toc68875997)

[1.1 Описание программы 3](#_Toc68875998)

[1.2 Описание API 4](#_Toc68875999)

[1.3 Обзор аналогов 7](#_Toc68876000)

[1.3.1 Плагин PDF для САПР КОМПАС-3D 7](#_Toc68876001)

[2 Описание предмета проектирования 8](#_Toc68876002)

[3 Проект программы 10](#_Toc68876003)

[3.1 Описание технических и функциональных аспектов проекта 10](#_Toc68876004)

[3.3 Диаграмма классов 10](#_Toc68876005)

[3.3 Макет пользовательского интерфейса 14](#_Toc68876006)

[4 Тестирование 16](#_Toc68876007)

[4.1 Функциональное тестирование 16](#_Toc68876008)

[4.2 Модульное тестирование 20](#_Toc68876009)

[4.3 Нагрузочное тестирование 22](#_Toc68876010)

[Заключение 24](#_Toc68876011)

[Список литературы 25](#_Toc68876012)

[Приложение А 26](#_Toc68876013)

# 1 Описание САПР

# Описание программы

В настоящее время проектирование в своем понимании представляет собой автоматизированный процесс и в некотором роде программно-аппаратный. Проектировщику, который занимается разработкой сложного механизма, или устройства, требующего больших расчетов, математических вычислений при построении модели и высокой точности, подходят системы автоматизации проектных решений — САПР [1].

САПР позволяют уменьшить финансовые затраты на разработку макета (модели) проекта (объекта), а также сократить время, которое тратит проектировщик на создание модели объекта и составление проектной документации.

В каждой крупной САПР есть свой средства для разработки, которые предоставляются с целью дать возможность разработчикам расширить функционал данной системы под свои конкретные нужды. Данным средством является API — программный интерфейс приложения [2]. Это набор готовых средств: классов, процедур, функций, структур, констант и т.д. API позволяет определить функциональность, которую предоставляет приложение, при этом абстрагируясь от того, как она реализована.

Расширение функционала в основном подразумевает разработку плагина или библиотеки на основе предоставленного API. Плагин — независимо компилируемый программный модуль, динамически подключаемый к основной программе, предназначенный для расширения или использования ее возможностей [3].

В качестве системы, которая предоставляет API и для которой стоит задача разработать плагин, была выбрана САПР «КОМПАС-3D» версии 18.1.

# 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 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 [4].

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

Таблица 1.1. – Методы интерфейса KompasObject.

|  |  |  |  |
| --- | --- | --- | --- |
| Метод | Входные параметры | Возвращаемое значение | Описание |
| Document3D() |  | Указатель на интерфейс документа трёхмерной модели ksDocument3D | Даёт возможность получить указатель на интерфейс трёхмерного документа (детали или сборки) |
| Visible |  |  | Свойство видимости приложения |
| GetMathematic2D() |  | Указатель на интерфейс ksMathematic2D | Метод для получения указателя на интерфейс для работы с математическими функциями |
| GetParamStruct(short structType) | structType – тип интерфейса параметров | Указатель на интерфейс указаного типа из StructType2D | Метод для получения указателя на интерфейс графического документа (чертежа или фрагмента) |

Продолжение таблицы 1.1

|  |  |  |  |
| --- | --- | --- | --- |
| Метод | Входные параметры | Возвращаемое значение | Описание |
| GetDynamicArray(long type) | 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 IPart.

|  |  |  |  |
| --- | --- | --- | --- |
| Метод | Входные параметры | Возвращаемое значение | Описание |
| EntityCollection  (short objType) | |  |  | | --- | --- | | objType | - тип объектов, содержащихся в массиве. | | Указатель на интерфейс [ksEntityCollection](mk:@MSITStore:D:\INSTAL\KOMPAS-3D%20V17.1\KOMPAS\SDK\SDK.chm::/ksEntityCollection.htm) или [IEntityCollection](mk:@MSITStore:D:\INSTAL\KOMPAS-3D%20V17.1\KOMPAS\SDK\SDK.chm::/ksEntityCollection.htm) | Формирует массив объектов и возвращает указатель на его интерфейс |
| GetDefaultEntity  (short objType) | |  |  | | --- | --- | | objType | - тип объекта. | | |  | | --- | | Указатель на интерфейс [ksEntity](mk:@MSITStore:D:\\INSTAL\\KOMPAS-3D%20V17.1\\KOMPAS\\SDK\\SDK.chm::/ksEntity.htm) или [IEntity](mk:@MSITStore:D:\\INSTAL\\KOMPAS-3D%20V17.1\\KOMPAS\\SDK\\SDK.chm::/ksEntity.htm). |   Типы объектов (objType):   |  |  |  | | --- | --- | --- | | o3d\_planeXOY | 1 | - плоскость XOY | | o3d\_planeXOZ | 2 | - плоскость XOZ | | o3d\_planeYOZ | 3 | - плоскость YOZ | | o3d\_pointCS | 4 | - точка начала системы координат | | o3d\_axisOX | 71 | - ось OX | | o3d\_axisOY | 72 | - ось OY | | o3d\_axisOZ | 73 | - ось OZ | | Получить указатель на интерфейс объекта, создаваемого системой по умолчанию |

Продолжение таблицы 1.2

|  |  |  |  |
| --- | --- | --- | --- |
| Метод | Входные параметры | Возвращаемое значение | Описание |
| GetPart(int type) | |  |  | | --- | --- | | type | - тип компонента. | | указатель на интерфейс компонента [ksPart](mk:@MSITStore:D:\INSTAL\KOMPAS-3D%20V17.1\KOMPAS\SDK\SDK.chm::/ksPart.htm) или [IPart](mk:@MSITStore:D:\INSTAL\KOMPAS-3D%20V17.1\KOMPAS\SDK\SDK.chm::/ksPart.htm). | Получить указатель на интерфейс компонента в соответствии с заданным типом |
| NewEntity(short objType) | |  |  | | --- | --- | | objType | - [тип объекта](mk:@MSITStore:D:\INSTAL\KOMPAS-3D%20V17.1\KOMPAS\SDK\SDK.chm::/Obj3dType_NewEntil_Part.htm). | | указатель на интерфейс [ksEntity](mk:@MSITStore:D:\INSTAL\KOMPAS-3D%20V17.1\KOMPAS\SDK\SDK.chm::/ksEntity.htm) или [IEntity](mk:@MSITStore:D:\INSTAL\KOMPAS-3D%20V17.1\KOMPAS\SDK\SDK.chm::/ksEntity.htm). | Создать новый интерфейс объекта и получить указатель на него |

Таблица 1.3. Методы интерфейса ksDocument3D.

|  |  |  |  |
| --- | --- | --- | --- |
| Метод | Входные параметры | Возвращаемое значение | Описание |
| Create (bool invisible, bool \_typeDoc) | 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. | Активизировать измененные параметры документа |
| GetPart(int type) | type – тип компонента из перечисления Типы компонентов. |  | Получить указатель на интерфейс компонента в соответствии с заданным типом |

# 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

Предметом проектирования является торцевой ключ.

Торцевой ключ используется в быту, гараже, автосервисах и при проведении слесарных работ, когда необходимо работать с труднодоступным резьбовым соединением, где другой инструмент невозможно использовать. Он отлично подходит для крепежа, расположенного в углублениях, а также для установки и снятия колес автомобилей.

Рабочая часть, которую имеет торцевой гаечный ключ, это круглый колпак с несколькими гранями. Он схож с накидным ключом, но в отличие от него позволяет выкручивать шпильки и болты из труднодоступных мест.

Конструкция торцевого ключа представляет собой трубу, на концах которой шестигранники, а в корпусе сделаны отверстия, для стержня. Размеры шестигранников отличаются. Очень часто такие ключи идут в комплекте с автомобилем [6].

Параметры ключа:

* Длина ключа L (от 80 до 400 мм);
* Ширина зева S1 (от 4 до 75 мм);
* Ширина зева S2 (от 5 до 80 мм);
* Глубина зева t1 (от 2 до 50 мм);
* Глубина зева t2 (от 2,5 до 50 мм);
* Толщина стенки ключа w (от 2 до 14 мм);
* Диаметр отверстия d (от 2 до 50 мм);
* Ширина трубки ключа H (от 4 до 75 мм);

Плагин имеет зависимые параметры:

* Длина ключа L > (2\*(t1 + t2 + d));
* Диаметр отверстий ;
* Ширина H должна быть не больше каждой из ширин зевов ключа S1, S2.

Пример модели приведен на рисунке 2.1.



Рисунок 2.1 – Модель торцевого ключа в КОМПАС-3D

# 3 Проект программы

# 3.1 Описание технических и функциональных аспектов проекта

Для графического описания абстрактной модели проекта использован стандарт UML (Unified Modelling Language).

Для представления архитектуры программы выбирают тот или иной вид диаграмм. Язык графического описания UML позволяет представить различные аспекты функционирования программной системы с помощью определённых стандартом UML условно-графических обозначений.[7] UML был создан для определения, визуализации, проектирования и документирования, в основном, программных систем.

При использовании UML была простроена диаграмма классов.

# 3.3 Диаграмма классов

Диаграмма классов описывает типы объектов системы и различного рода статические отношения, которые существуют между ними. На диаграммах классов отображаются также свойства классов, операции классов и ограничения, которые накладываются на связи между объектами [8].

На рисунке 3.1 представлена диаграмма классов.

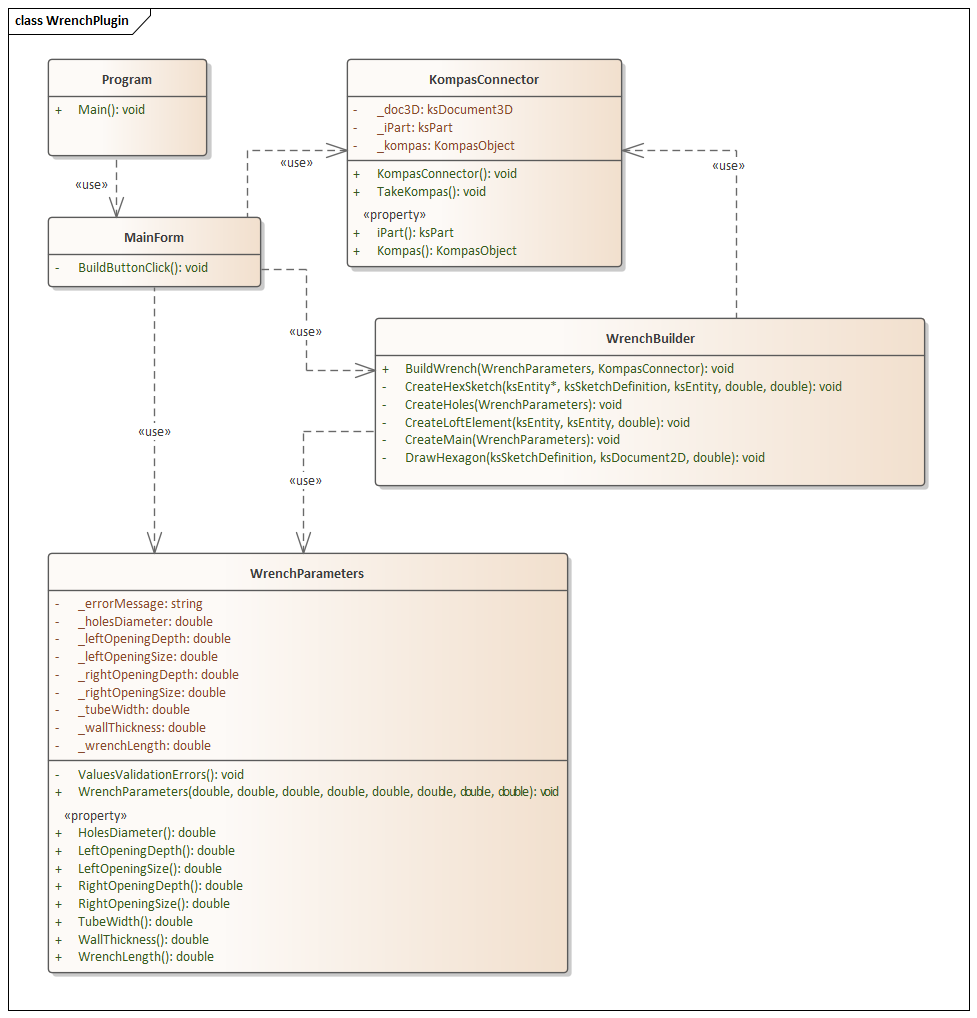


Рисунок 3.1 – Диаграмма классов

Для реализации был выбран следующий набор классов:

* MainForm – класс диалогового окна, который обеспечивает взаимодействие между пользователем и программой;
* WrenchParameters − класс, хранящий в себе все параметры проектируемой 3D-модели, осуществляет проверку зависимых параметров;
* KompasConnector – класс для работы с API КОМПАС 3D.
* WrenchBuilder – класс, осуществляющий вызов методов API, необходимых для постройки 3D-модели.

Во время разработки проекта, диаграмма классов поменяла свой вид. В

класс MainForm добавлен обработчик события defaultParamComboBox\_SelectedIndexChanged для выбора предустановленных параметров модели из списка в ComboBox.

Добавлен класс Parameter, содержащий данные отдельного параметра модели (название, минимум, максимум, значение параметра) и методы для проверки данных. После замены типа полей класса WrenchParameters с double на Parameter сократилось дублирование кода в свойствах класса, разделены зависимые и независимые параметры.

В классе WrenchParameters свойства независимых параметров заменены автосвойствами. Добавлен конструктор класса по умолчанию.

После реализации дополнительной функциональности, в класс WrenchParameters добавлено свойство RoundSection типа bool для выбора вида сечения средней части ключа (false – шестигранное сечение, true – круглое).

В классе WrenchBuilder изменено наименование методов и их количество, метод CreateHoles разделен на CutExtrusion, CutExtrusionHoles и DrawCircle. Метод CreateHexSketch переименован в CreateSketch, добавлен входной параметр roundSection для выбора фигуры, отображаемой на эскизе (шестиугольник/окружность).

Диаграмма, полученная после этапа реализации, изображена на рисунке 3.2.

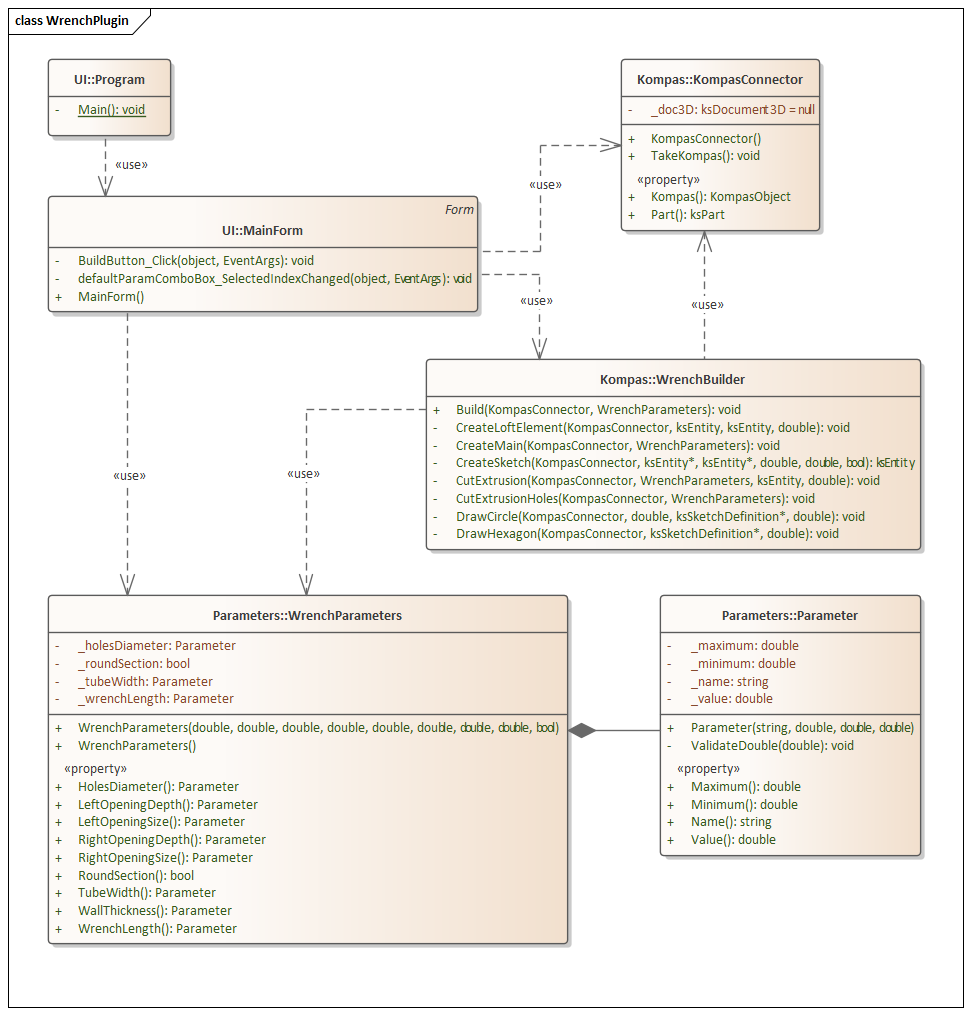
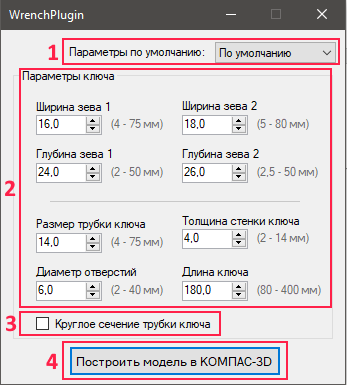


Рисунок 3.2 – Диаграмма классов после реализации и добавления дополнительной функциональности

# 3.3 Макет пользовательского интерфейса

Макет пользовательского интерфейса представляет собой форму для ввода параметров. Числовые значения параметров вводятся либо выбираются из списка в элементах NumericUpDown. Построение модели осуществляется нажатием на кнопку «Построить модель в КОМПАС-3D». На рисунке 3.3 представлен макет пользовательского интерфейса.

   
Рисунок 3.3 – Макет пользовательского интерфейса

В области, отмеченной цифрой 1, находится список предустановленных параметров модели: по умолчанию, минимальные и максимальные.

Область 2 содержит элементы NumericUpDown для ввода параметров 3D-модели, справа от каждого поля указан диапазон его допустимых значений. В области 4 расположена кнопка построения модели в КОМПАС 3D.

Согласно дополнительному заданию, на окно программы добавлен элемент CheckBox для выбора типа сечения при построении модели (область 3 на рисунке).

После ввода некорректных параметров и нажатия кнопки построения модели отображается окно предупреждения с описанием ошибок (рис. 3.4).

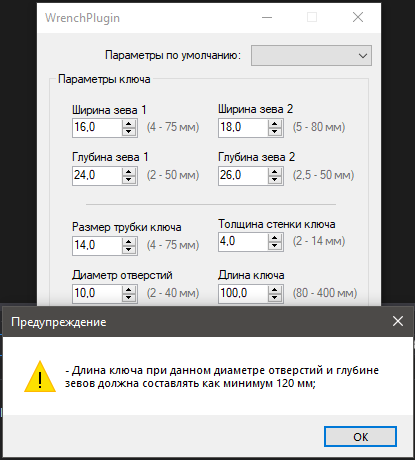


Рисунок 3.4 – Предупреждение при вводе несовместимых параметров

# 4 Тестирование

# 4.1 Функциональное тестирование

Функциональное тестирование – это тестирование функциональности и поведения программы на соответствие требованиям функциональной спецификации. Функциональная спецификация определяет, что именно делает ПО, какие задачи оно решает. В данном случае будет проверяться правильность построения детали при различных входных параметрах

Построение модели с минимальными входными данными:

* Длина ключа ­­­­– 80 мм;
* Ширина зева 1 – 4 мм;
* Ширина зева 2 – 5 мм;
* Глубина зева t1 – 2 мм;
* Глубина зева t2 – 2,5 мм;
* Толщина стенки ключа – 2 мм;
* Диаметр отверстий – 2 мм;
* Ширина трубки ключа – 4 мм.

Модели с минимальными входными параметрами показаны на рисунках 4.1 и 4.2.

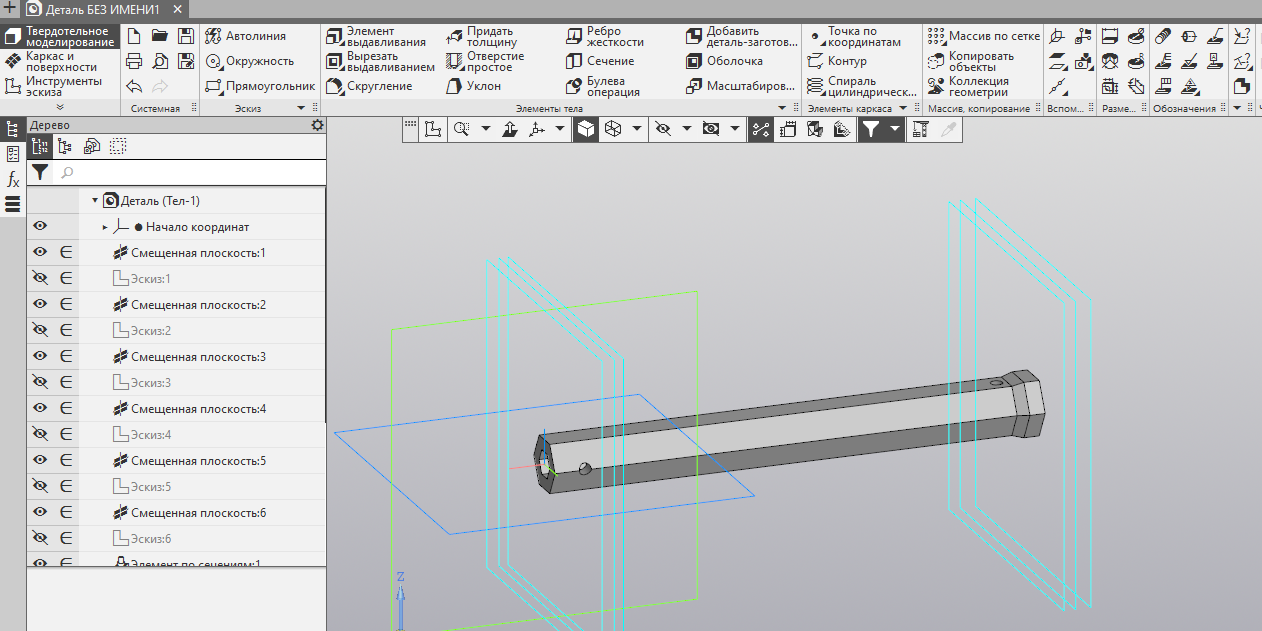


Рисунок 4.1 – Модель с шестигранным сечением, построенная с минимальными входными параметрами

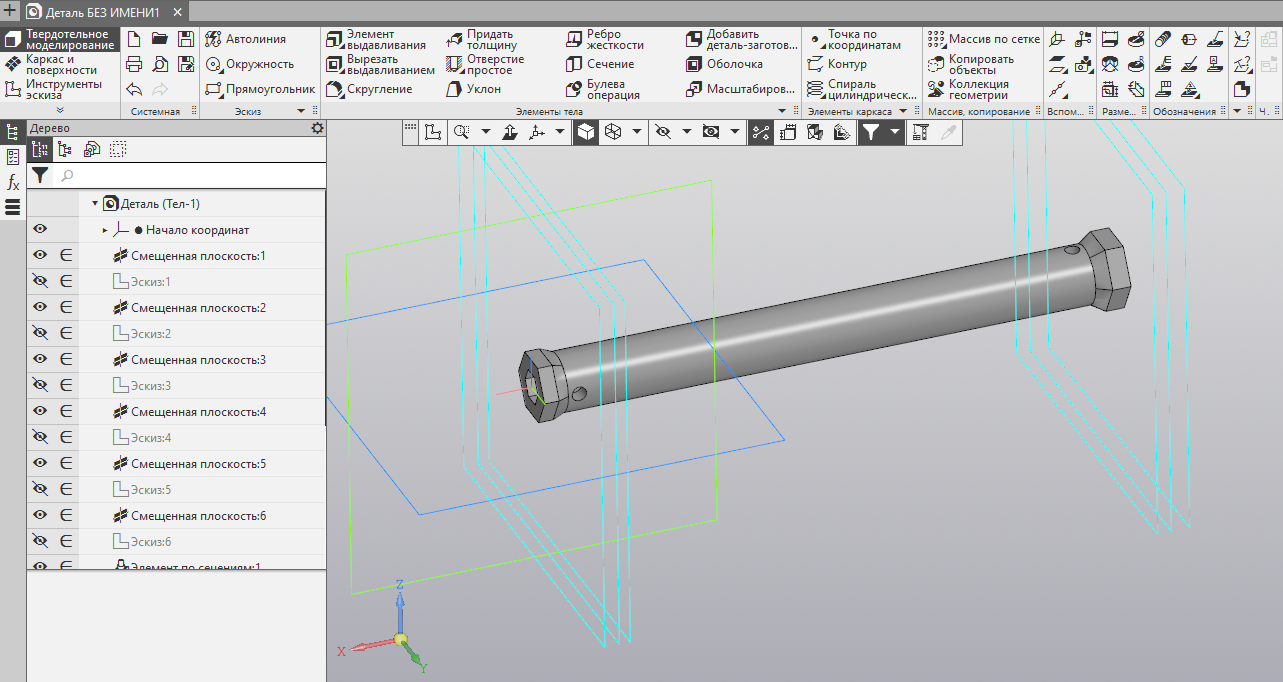


Рисунок 4.2 – Модель с круглым сечением, построенная с минимальными входными параметрами

Построение модели с минимальными входными данными:

* Длина ключа ­­­­– 400 мм;
* Ширина зева 1 – 75 мм;
* Ширина зева 2 – 80 мм;
* Глубина зева t1 – 50 мм;
* Глубина зева t2 – 50 мм;
* Толщина стенки ключа – 14 мм;
* Диаметр отверстий – 50 мм;
* Ширина трубки ключа – 75 мм.

Модели с максимальными входными параметрами показаны на рисунках 4.3 и 4.2.

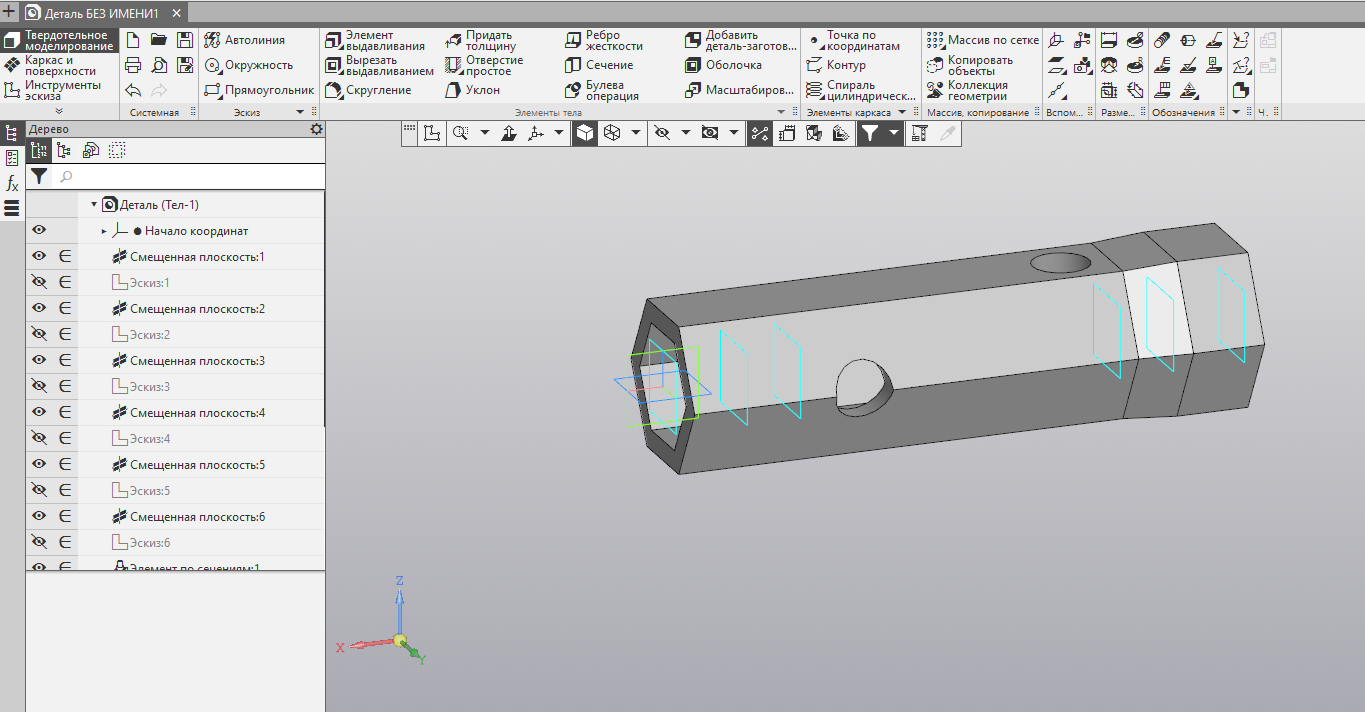


Рисунок 4.3 – Модель с шестигранным сечением, построенная с минимальными входными параметрами

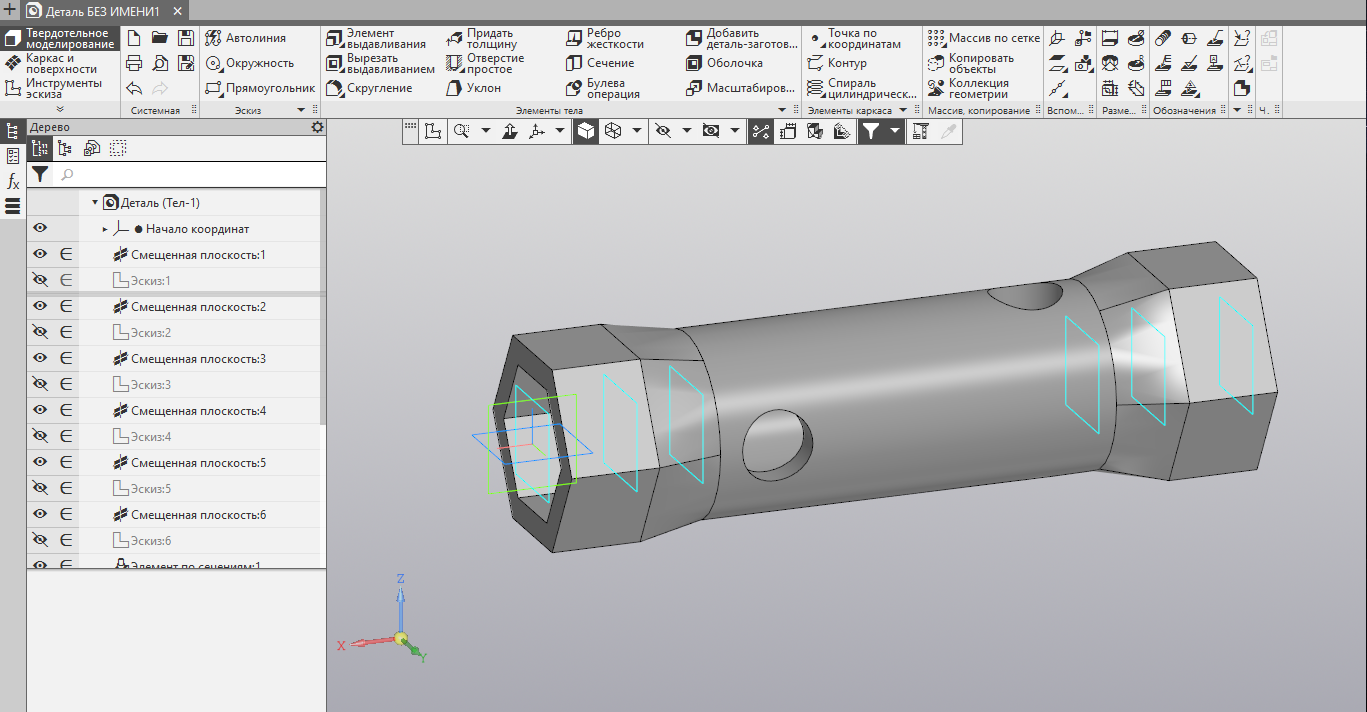


Рисунок 4.4 – Модель с круглым сечением, построенная с минимальными входными параметрами

При вводе заведомо некорректных данных программа сообщает об ошибках (рис. 4.5 – 4.10).

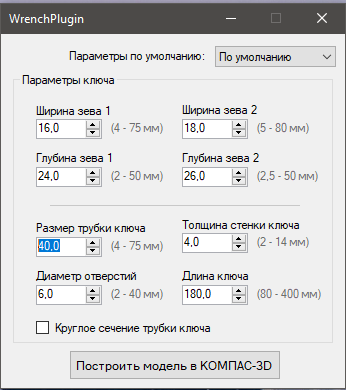


Рисунок 4.5 – Ввод некорректного размера трубки ключа

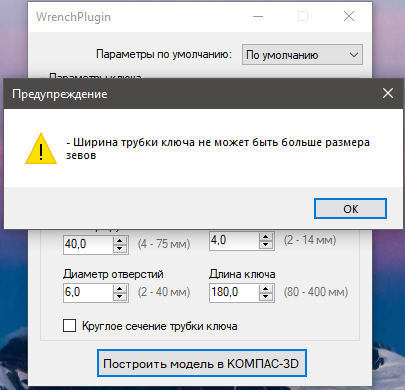


Рисунок 4.6 – Вывод MessageBox с сообщением об ошибке

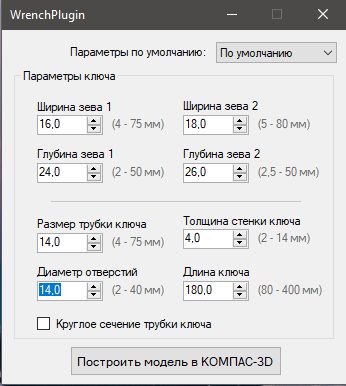


Рисунок 4.7 – Ввод некорректного диаметра отверстий

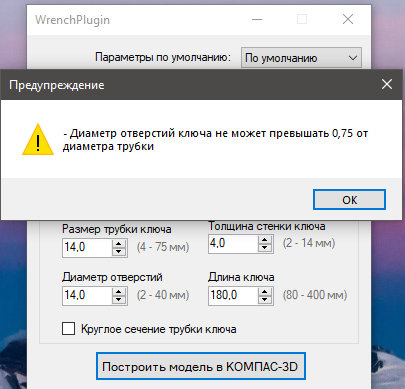


Рисунок 4.8 – Вывод MessageBox с сообщением об ошибке

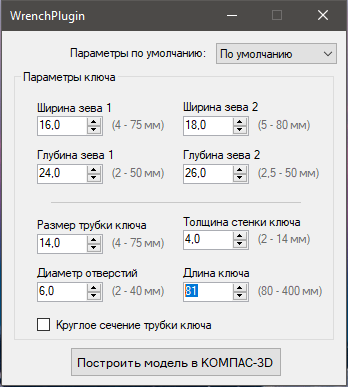


Рисунок 4.9 – Ввод некорректной длины ключа

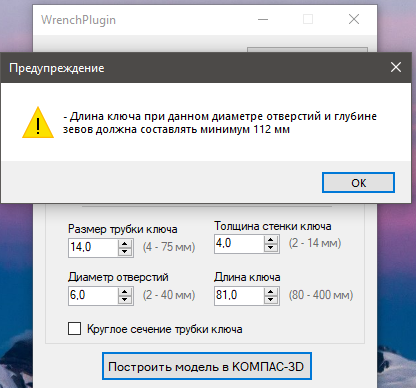


Рисунок 4.10 – Вывод MessageBox с сообщением об ошибке

# 4.2 Модульное тестирование

Юнит-тестирование (англ. «*unit*-*testing*», или модульное тестирование) — тестирование отдельного элемента изолированно от остальной системы. Относительно парадигмы объектно-ориентированного программирования системой является вся программа, а отдельным элементом — класс или его метод. Юнит-тестирование предназначено для проверки правильности работы отдельно взятого элемента. Чтобы исключить из результатов тестирования влияние потенциальных ошибок других элементов, тестируемый элемент должен быть максимально изолирован, то есть не использовать объекты и методы других классов [7].

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

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

Результаты прохождения всех модульных тестов приведены на рисунке 4.11.

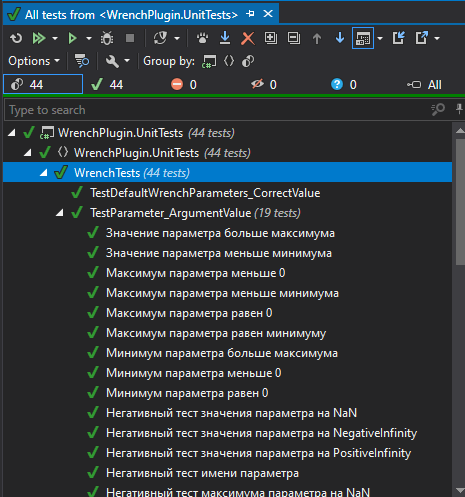


Рисунок 4.11 – Список пройденных юнит-тестов

Результат покрытия моделей Parameters и WrenchParameters тестами приведен на рисунке 4.12. Цикломатическая сложность равна 105.

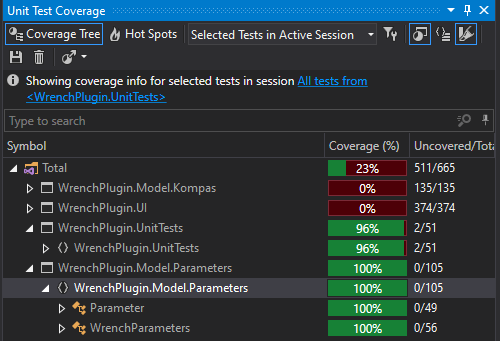


Рисунок 4.11 – Результат покрытия тестами

# 4.3 Нагрузочное тестирование

Нагрузочное тестирование – тип тестирования, который позволяет оценить поведение системы при возрастающей нагрузке, целью нагрузочного тестирования является также определение максимальной нагрузки, которую может выдержать система [9].

Нагрузочное тестирование проводилось на компьютере со следующими характеристиками:

* Процессор: Intel Pentium N3700 1.60-2.40 GHz;
* Оперативная память: 8192MB DDR3L 1600 MHz;
* Графический процессор: NVIDIA GeForce 920M;
* Операционная система: Windows 10 Home 64-bit.

При задании максимального количества последовательно строящихся моделей равным 200, САПР «КОМПАС-3D» остановила дальнейшие построения после 179 модели, окно программы перестало реагировать на ввод. В диспетчере задач у процесса запущенной САПР отображался статус «Не работает», также произошел сбой и перезагрузка Проводника Windows. Исходя из этого, для выполнения нагрузочного тестирования, количество построенных деталей было принято равным 179.

Зависимость потребления оперативной памяти САПР «КОМПАС-3D» при построенных 179 моделях в представлена на графике (рис. 4.12).



Рисунок 4.12 – Зависимость потребление ОЗУ от количества построенных моделей

Потребление памяти резко возросло после построения 23 моделей. Пик потребления был достигнут на 70 построении и составил 1775 МБ. При дальнейших построениях количество используемой ОЗУ уменьшается, на 118 построении падает с 1204 МБ до 554 MB, затем снова постепенно возрастает, не превышая 1200 МБ. Снижение потребляемой ОЗУ связано с использованием системой виртуальной памяти (в Windows эту функцию выполняет файл подкачки pagefile.sys) на жестком диске.

График на рисунке 4.13 отображает зависимость застрачиваемого времени на каждое новое построение от количества уже построенных моделей. Время на построение резко возрастает на построении 103 модели, составляя 42.5 секунды. Далее наблюдаются резкие колебания во времени построения, каждая новая модель строится не менее 15 секунд. Увеличение времени построения происходит по той причине, что скорость чтения и загрузки данных с использованием виртуальной памяти значительно ниже из-за ее меньшей пропускной способности.



Рисунок 4.13 – Зависимость затрат времени на построение от количества построенных 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.

# Список литературы

Норенков И.П. «Основы автоматизированного проектирования». Издательство: МГТУ; Москва:, 2002 – 336 с.

API – Википедия. [Электронный ресурс]. — Режим доступа: <https://ru.wikipedia.org/wiki/API> (дата обращения 04.04.2020)

Плагин – Википедия. [Электронный ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/Плагин> (дата обращения 04.04.2021)

Кидрук Максим. КОМПАС-3D V10 на 100% / М. Кидрук. – СПб.: Питер, 2009 – 560 с.

Экспорт в формате 3D PDF из КОМПАС-3D. [Электронный ресурс]. – Режим доступа: <http://isicad.ru/ru/news.php?news=16278> (дата обращения 04.04.2021)

Торцевые и гаечные ключи: характеристики, условия эксплуатации, фото. [Электронный ресурс]. – <https://stanok.guru/oborudovanie/klyuchi/gaechnye-i-torcevye-klyuchi-foto-harakteristiki.html#i-6> (дата обращения 04.04.2021)

Новые технологии в программировании : учебное пособие / А.А.Калентьев, Д.В.Гарайс, А.Е.Горяинов – Томск : Эль Контент, 2014.—176 с.

М. Фаулер. UML. Основы, 3-е издание. Книга по UML для начинающих – Москва : Символ-Плюс, 2018 – 192 с.

Нагрузочное тестирование vs Тестирование производительности. [Электронный ресурс]. – <https://performance-lab.ru/blog/load-testing/testirovanie-proizvoditelnosti> (дата обращения 09.04.2021)

# Приложение А

(справочное)

Таблица А.1 – Тестовые сценарии

|  |  |  |
| --- | --- | --- |
| Тестовый метод | Входные параметры | Описание тестового случая |
| TestWrenchParameters\_CorrectValue() | 18, 18, 14, 14, 2, 10, 4, 180, false | Позитивный тест конструктора класса WrenchParameters |
| TestDefaultWrenchParameters\_CorrectValue() | 16, 24, 18, 26, 4, 14, 6, 180, false | Позитивный тест конструктора по умолчанию класса WrenchParameters |
| TestParameter\_CorrectValue() | “Имя параметра”, 0.5, 10.5, 5 | Позитивный тест конструктора класса Parameter |
| TestWrenchParameters\_ArgumentValue (double rightOpeningSize, double rightOpeningDepth, double leftOpeningSize, double leftOpeningDepth, double wallThickness, double tubeWidth, double holesDiameter, double wrenchLength, bool roundSection, string attribute) | 3, 18, 14, 14, 2, 10, 4, 80, false, nameof(WrenchParameters.  LeftOpeningSize) | Присвоение полю LeftOpeningSize значения меньше минимума |
| 76, 18, 14, 14, 2, 10, 4, 80, false, nameof(WrenchParameters.  LeftOpeningSize) | Присвоение полю LeftOpeningSize значения больше максимума |
| 18, 1, 14, 14, 2, 10, 4, 80, false, nameof(WrenchParameters.  LeftOpeningDepth) | Присвоение полю LeftOpeningDepth значения меньше минимума |
| 18, 51, 14, 14, 2, 10, 4, 80, false, nameof(WrenchParameters.  LeftOpeningDepth) | Присвоение полю LeftOpeningDepth значения больше максимума |
| 18, 18, 4, 14, 2, 10, 4, 80, false, nameof(WrenchParameters.  RightOpeningSize) | Присвоение полю RightOpeningSize значения меньше минимума |
| 18, 18, 4, 14, 2, 10, 4, 80, false, nameof(WrenchParameters.  RightOpeningSize) | Присвоение полю RightOpeningSize значения больше максимума |
| 18, 18, 14, 2, 2, 10, 4, 80, false, nameof(WrenchParameters.  RightOpeningDepth) | Присвоение полю RightOpeningDepth значения меньше минимума |
| 18, 18, 14, 51, 2, 10, 4, 80, false, nameof(WrenchParameters.  RightOpeningDepth) | Присвоение полю RightOpeningDepth значения больше максимума |

Продолжение таблицы А.1

|  |  |  |
| --- | --- | --- |
| Тестовый метод | Входные параметры | Описание тестового случая |
|  | 18, 18, 14, 14, 1, 10, 4, 80, false, nameof(WrenchParameters.WallThickness) | Присвоение полю WallThickess значения меньше минимума |
| 18, 18, 14, 14, 15, 10, 4, 80, false, nameof(WrenchParameters.WallThickness) | Присвоение полю WallThickess значения больше максимума |
| 18, 18, 14, 14, 2, 19, 4, 80, false, nameof(WrenchParameters.  TubeWidth) | Негативный тест при значении TubeWidth > LeftOpeningSize |
| 18, 18, 14, 14, 2, 20, 4, 80, false, nameof(WrenchParameters.  TubeWidth) | Негативный тест при значении TubeWidth > RightOpeningSize |
| 18, 18, 14, 14, 2, 3, 4, 80, false, nameof(WrenchParameters.  TubeWidth) | Присвоение полю TubeWidth значения меньше минимума |
| 75, 18, 80, 14, 2, 76, 4, 80, false, nameof(WrenchParameters.  TubeWidth) | Присвоение полю TubeWidth значения больше максимума |
| 18, 18, 14, 14, 2, 10, 1, 80, false, nameof(WrenchParameters.  HolesDiameter) | Присвоение полю HolesDiameter значения меньше минимума |
| 75, 18, 80, 14, 2, 70, 41, 100, false, nameof(WrenchParameters.  HolesDiameter) | Присвоение полю HolesDiameter значения больше максимума |
| 18, 18, 14, 14, 2, 10, 9, 80, false, nameof(WrenchParameters.  HolesDiameter) | Негативный тест при значении HolesDiameter > 0,75\*TubeWidth |
| 18, 18, 14, 14, 2, 10, 1, 79, false, nameof(WrenchParameters.WrenchLength) | Присвоение полю WrenchLength значения меньше минимума |
| 75, 18, 80, 14, 2, 70, 41, 401, false, nameof(WrenchParameters.WrenchLength) | Присвоение полю WrenchLength значения больше максимума |

Продолжение таблицы А.1

|  |  |  |
| --- | --- | --- |
| Тестовый метод | Входные параметры | Описание тестового случая |
|  | 18, 30, 14, 20, 2, 10, 4, 107, false, nameof(WrenchParameters.WrenchLength) | Негативный тест при значении поля WrenchLength > ((LeftOpeningDepth + RightOpeningDepth + HolesDiameter) \* 2) |
| TestParameter\_ArgumentValue(string name, double minimum, double maximum, double value, string attribute) | "Имя параметра", 0, 10, 5, nameof(Parameter.Minimum) | Минимум параметра равен 0 |
| "Имя параметра", -1, 10, 5, nameof(Parameter.Minimum) | Минимум параметра меньше 0 |
| "Имя параметра", 5, 4, 5, nameof(Parameter.Minimum) | Минимум параметра больше максимума |
| "Имя параметра", double.PositiveInfinity, 10, 2, nameof(Parameter.Minimum) | Негативный тест минимума параметра на PositiveInfinity |
| "Имя параметра", double.NegativeInfinity, 10, 2, nameof(Parameter.Minimum) | Негативный тест минимума параметра на NegativeInfinity |
| "Имя параметра", double.NaN, 10, 2, nameof(Parameter.Minimum) | Негативный тест минимума параметра на NaN |
| "Имя параметра", 1, 5, 6, nameof(Parameter.Value) | Значение параметра больше максимума |
| "Имя параметра", 1, 5, 0.5, nameof(Parameter.Value) | Значение параметра меньше минимума |
| "Имя параметра", 1, 10, double.PositiveInfinity, nameof(Parameter.Value) | Негативный тест значения параметра на PositiveInfinity |
| "Имя параметра", 1, 10, double.NegativeInfinity, nameof(Parameter.Value) | Негативный тест значения параметра на NegativeInfinity |
| "Имя параметра", 1, 10, double.NaN, nameof(Parameter.Value) | Негативный тест значения параметра на NaN |
| "", 1, 10, 6, nameof(Parameter.Name) | Негативный тест имени параметра |