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 «ПОСТРОЕНИЕ ПОДВЕСНЫХ ПОЛОК» ДЛЯ САПР «КОМПАС-3D v.19»**

Пояснительная записка по лабораторному проекту по дисциплине   
«Основы разработки САПР»

Выполнил:

Студент гр. 588-1

\_\_\_\_\_\_\_\_\_\_\_\_/ Колбас Е. О.

(подпись)

«\_\_\_» \_\_\_\_\_\_\_\_20\_\_г.

Проверил:

к.т.н, доцент каф. КСУП

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

(подпись)

«\_\_\_» \_\_\_\_\_\_\_\_20\_\_г.

Томск 2022

**Реферат**

Лабораторная работа, 34 страницы, 22 рисунка, 5 таблиц, 10 источников.

Ключевые слова: САПР, КОМПАС-3D, СТЕЛЛАЖ, МОДЕЛЬ, РАЗРАБОТКА, ПЛАГИН, API.

Целью данной работы является разработка плагина для создания трехмерных моделей подвесных полок, согласно заданным параметрам для системы автоматизированного проектирования «КОМПАС-3D v.19».

Отчет по лабораторной работе выполнен в текстовом редакторе Microsoft Word.

**Оглавление**

1 Описание САПР 4

2 Описание API 6

3 Обзор аналогов 11

4 Описание предмета проектирования 13

5 Диаграмма классов 15

6 Макет пользовательского интерфейса 18

7 Тестирование плагина 21

7.1 Модульное тестирование 21

7.2 Функциональное тестирование 22

7.3 Нагрузочное тестирование 27

8 Заключение 30

Список использованных источников 31

Приложение А 32

**1 Описание САПР**

Проектирование новых видов и образцов машин, оборудования, устройств, аппаратов, приборов и других изделий представляет сложный и длительный процесс, включающий в себя разработку исходных данных, чертежей, технической документации, необходимых для изготовления опытных образцов и последующего производства, и эксплуатации объектов проектирования.

***Проектирование*** — это комплекс работ с целью получения описаний нового или модернизируемого технического объекта, достаточных для реализации или изготовления объекта в заданных условиях. В процессе проектирования возникает необходимость создания описания, необходимого для построения еще не существующего объекта. Получаемые при проектировании описания бывают окончательными или промежуточными. Окончательные описания представляют собой комплект конструкторско-технологической документации в виде чертежей, спецификаций, программ для ЭВМ и автоматизированных комплексов и так далее[1].

Основной целью автоматизации является повышение качества исполнения процесса. Автоматизированный процесс обладает более стабильными характеристиками, чем процесс, выполняемый в ручном режиме. Во многих случаях автоматизация процессов позволяет повысить производительность, сократить время выполнения процесса, снизить стоимость, увеличить точность и стабильность выполняемых операций.

***Система автоматизированного проектирования (САПР)***— это организационно-техническая система, состоящая из комплекса средств автоматизации проектирования, взаимодействующего с подразделениями проектной организации и выполняющая автоматизированное проектирование.

Однако, с приходом на отечественный рынок иностранных систем, широкое распространение получили аббревиатуры *CAD*(Computer Aided Design), которую можно перевести, как проектирование с применением компьютера, и *CAD-system*, которую можно перевести, как система для проектирования с помощью компьютера[2].

В настоящее время в среде специалистов по САПР многие термины утратили свой первоначальный смысл, а термин *САПР* теперь обозначает программу для автоматизированного проектирования.

Для реализации плагина будет использоваться программа   
«КОМПАС-3D» версии 19.

***КОМПАС-3D*** – это система трехмерного моделирования деталей и сборок, используемая для проектирования изделий в машиностроении и строительстве — от изделий народного потребления до авиа-, судостроения и продукции военного назначения[3].

Система «КОМПАС-3D» отличается проектированием изделий любой сложности, простотой освоения, бесплатной технической поддержкой, автоматизацией отраслевых задач и многим другим.

**2 Описание API**

Аббревиатура ***API*** расшифровывается как «Application Programming Interface» (интерфейс программирования приложений, программный интерфейс приложения)[4].

Сегодня встречаются задачи, решение которых не реализованы в CAD-системах. Чаще всего это очень узкоспециализированные задачи, которые встречаются на каком-то конкретном предприятии или подотрасли. Для решения подобных задач и их автоматизации используется API.

***API КОМПАС-3D*** — это ориентированные на прикладного программиста инструментальные средства разработки приложений (библиотек конструктивов, прикладных САПР) на базе системы КОМПАС. API КОМПАС-3D включает в свой состав API 5 и API 7.

Главным интерфейсом API системы КОМПАС является ***KompasObject***[5]. Методы этого интерфейса, некоторые из которых представлены в таблице 2.1, реализуют наиболее общие функции работы с документами системы, системными настройками, файлами, а также дают возможность получить указатели на другие интерфейсы (интерфейсы динамического массива, работы с математическими функциями, библиотек моделей или фрагментов и различных структур параметров определенного типа).

Тип данных ***KompasObjectPtr*** задает указатель на данный интерфейс.

Функция *public static object CreateInstance (Type type)* — создает экземпляр указанного типа, используя конструктор, который наиболее полно соответствует указанным параметрам. В нашем случае, если мы создадим указатель на интерфейс KompasObject, то с помощью вызова функции kompasPtr.CreateInstace(L«KOMPAS.Application.5») вызовем открытие программы КОМПАС-3D.

Таблица 2.1 — Некоторые методы интерфейса KompasObject

|  |  |  |  |
| --- | --- | --- | --- |
| **Метод** | **Входные параметры** | **Выходные параметры** | **Описание** |
| void Quit(); | Не используются | Не используются | Закрытие программы «КОМПАС-3D» |
| LPDISPATCH GetParamStruct(short structType); | structType | Указатель на интерфейс указанного типа из StructType2D | Возвращает указатель на интерфейс параметров объекта того или иного типа. Запрашиваемый тип объекта передается в метод в качестве значения единственного параметра. |
| LPDISPATCH ActiveDocument3D(); | Не используются | Указатель на интерфейс трехмерной модели ksDocument3D | Получает указатель на интерфейс текущего документа трехмерной модели |

Создание чертежа и фрагмента происходит в два этапа. На первом этапе подготавливаются параметры создаваемого документа. На втором – создается сам документ. Такой двухэтапный подход используется при создании практически всех объектов в системе КОМПАС.

Параметры документа описываются интерфейсом ***ksDocumentParam***. Для получения указателя на него используется метод   
GetParamStruct интерфейса KompasObject с параметром ko\_DocumentParam. Интерфейс имеет следующие свойства:

1. BSTR author — автор;
2. short type — тип документа;
3. BSTR filename — имя файла.

В таблице 2.2 приведены основные методы интерфейса.

Таблица 2.2 — Методы интерфейса ksDocumentParam

|  |  |  |
| --- | --- | --- |
| **Метод** | **Возвращаемое значение** | **Назначение** |
| LPDISPATCH GetLayoutParam(); | Указатель на интерфейс параметров оформления документа [ksSheetPar](mk:@MSITStore:C:\Program%20Files\ASCON\KOMPAS-3D%20v19%20Study\SDK\SDK.chm::/ksSheetPar.htm) | Получает указатель на интерфейс параметров оформления документа |
| bool Init(); | true — в случае удачного завершения | Инициализация параметров. Метод обнуляет все параметры чертежа или фрагмента. |

Чертеж и фрагмент описываются интерфейсом ***ksDocument2D***. Получить на него указатель можно с помощью метода Document2D  
интерфейса KompasObject.

Деталь и сборка в КОМПАС описываются интерфейсом ***ksDocument3D***. Точнее говоря, этот интерфейс описывает файл, содержащий деталь или сборку. Для получения указателя на интерфейс ksDocument3D используется метод Document3D() интерфейса KompasObject. Некоторые методы интерфейса приведены в таблице 2.3.

Таблица 2.3 — Методы интерфейса ksDocument3D

|  |  |  |  |
| --- | --- | --- | --- |
| **Метод** | **Входные параметры** | **Выходные параметры** | **Назначение** |
| bool Create(bool invisible, bool typeDoc); | invisible — признак режима редактирования документа  typeDoc — тип документа (false — сборка, true — деталь) | true — в случае успешного завершения | Создает документ-модель (деталь или сборку) |
| LPDISPATCH GetPart (long type); | type — тип компонента из перечисления | Указатель на интерфейс компонента ksPart или IPart | Получает указатель на интерфейс компонента в соответствии с заданным типом |

Продолжение таблицы 2.3

|  |  |  |  |
| --- | --- | --- | --- |
| **Метод** | **Входные параметры** | **Выходные параметры** | **Назначение** |
| long ksGetObjParam (long reference, LPDISPATCH param, long paramType); | reference — указатель на объект,  paramType — тип параметров,  param — указатель на интерфейс параметров ksUserParam. | Тип объекта — в случае удачного завершения,  0 — в случае неудачи | Получает параметры объекта |
| bool UpdateDocumentParam(); | Не используются | true — в случае успешного завершения | Активизирует измененные параметры документа |

Метод ***ksDocument3D::GetPart*** возвращает указатель на интерфейс детали или компонента сборки — ksPart. Свойства и методы этого интерфейса управляют состоянием компонентов сборки, они почти полностью дублируют команды контекстного меню и панели свойств, доступные пользователю при работе с тем или иным компонентом. Некоторые методы приведены в таблице 2.4.

Таблица 2.4 — Некоторые методы интерфейса ksPart

|  |  |  |  |
| --- | --- | --- | --- |
| **Метод** | **Входные параметры** | **Выходные параметры** | **Назначение** |
| LPDOCUMENT3D BeginEdit(); | Не используются | Указатель на интерфейс документа [ksDocument3D](mk:@MSITStore:C:\Program%20Files\ASCON\KOMPAS-3D%20v19%20Study\SDK\SDK.chm::/ksDocument3D.htm) или [IDocument3D](mk:@MSITStore:C:\Program%20Files\ASCON\KOMPAS-3D%20v19%20Study\SDK\SDK.chm::/ksDocument3D.htm). | Запустить режим редактирования на месте данного компонента. |

Продолжение таблицы 2.4

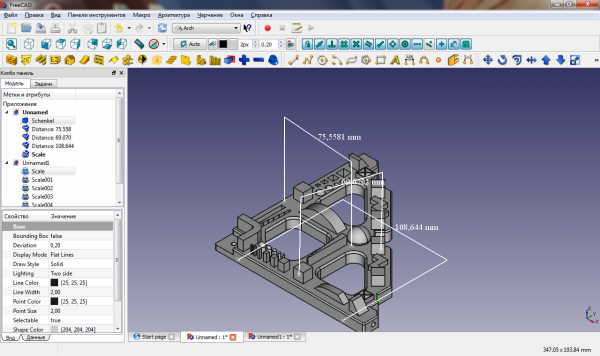
|  |  |  |  |
| --- | --- | --- | --- |
| **Метод** | **Входные параметры** | **Выходные параметры** | **Назначение** |
| bool EndEdit (bool rebuild); | rebuild — перестроить компонент | true — в случае удачного завершения | Закрыть режим редактирования на месте для данного компнента. Данный метод работает только для компонента, вставленного в сборку. |
| LPDISPATCH GetPart (long type); | type — тип компонента | Указатель на интерфейс компонента [ksPart](mk:@MSITStore:C:\Program%20Files\ASCON\KOMPAS-3D%20v19%20Study\SDK\SDK.chm::/ksPart.htm) или [IPart](mk:@MSITStore:C:\Program%20Files\ASCON\KOMPAS-3D%20v19%20Study\SDK\SDK.chm::/ksPart.htm). | Получить указатель на интерфейс компонента в соотвтествии с заданным типом. |

Это лишь небольшая часть интерфейсов для взаимодействия с системой «КОМПАС-3D». О всех остальных возможностях API программы можно прочесть в руководстве пользователя KOMPAS-Invisible, которое находится в открытом доступе в Интернете.

**3 Обзор аналогов**

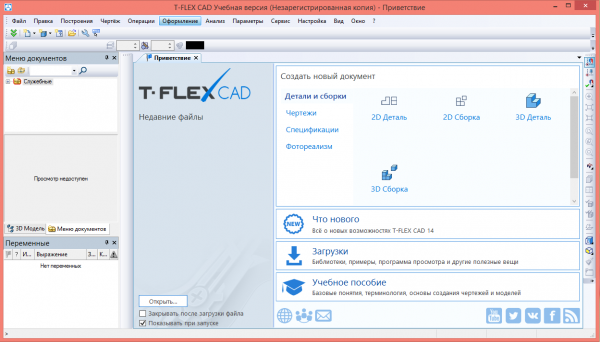
Аналогов у программы «КОМПАС-3D» достаточное количество — двенадцать и более программ. Для многих является проблемой использовать платные программы, и они ищут аналоги с таким же функционалом, только бесплатные, чтобы выполнить какую-либо поставленную задачу. Ниже приведены несколько примеров бесплатных аналогов системы «КОМПАС-3D»[6].

1. *FreeCAD* — бесплатная мультиплатформенная CAD программа для создания 3D моделей. FreeCAD может быть использована в техническом проектировании, конструировании изделий, а также в иных областях, связанных с осуществлением инженерно-технических работ. Программа хорошо подходит для создания моделей для 3D принтера, так как поддерживает STL формат. Работа программы представлена на рисунке 3.1.

Рисунок 3.1 — Снимок экрана программы FreeCAD

1. *T-FLEX CAD* — система автоматизированного проектирования, объединяет в себе 3D- и 2D-функционал, обладает обширным инструментарием для создания параметрических и непараметрических чертежей деталей и сборок, а также для оформления конструкторской документации. При этом она обеспечивает полную поддержку как ЕСКД, так и зарубежных стандартов. Программа имеет бесплатную версию, которую можно использовать в личных и учебных целях. Бесплатная версия содержит ряд ограничений, с которыми можно ознакомиться на сайте разработчика. Есть функция экспорта объектов в формат для 3D-печати.

Главное окно программы представлено на рисунке 3.2.

Рисунок 3.2 — Снимок экрана главного окна программы T-FLEX CAD

**4 Описание предмета проектирования**

Предметом проектирования являются подвесные полки.

Навесные полкиявляются полезным и функциональным предметом мебели, помогающим значительно сэкономить пространство за счет своего прямого назначения — расположения на них вещей[7].

Полки могут быть изготовлены из стекла, металла или дерева. Подвесные полки, исполняемые в проекте, изготавливаться будут из ДСП (дерево-стружечных плит), которые имеют определенную толщину от 15 до 20 миллиметров, применяющиеся при изготовлении основной массы мебели.

Под желание заказчика (покупателя) полки могут изменять параметры, приведенные ниже:

1. A — толщина всех досок: от 15 до 20 мм;
2. B — длина полок: от 500 до 700 мм;
3. C — ширина полок: от 200 до 300 мм;
4. D — высота левой стенки верхней полки: от 150 до 200 мм;
5. E — высота правой стенки нижней полки и задней стенки обеих полок: от 100 до 150 мм.

Плагин имеет следующие зависимости:

1. Параметр E должен быть строго на 50 мм меньше D;

2. Параметр F — высота стенки посередине, соединяющей обе полки, формируется самостоятельно по формуле зависимости от других параметров F = A + D + E. Его размерность варьируется от 265 до 370 мм.

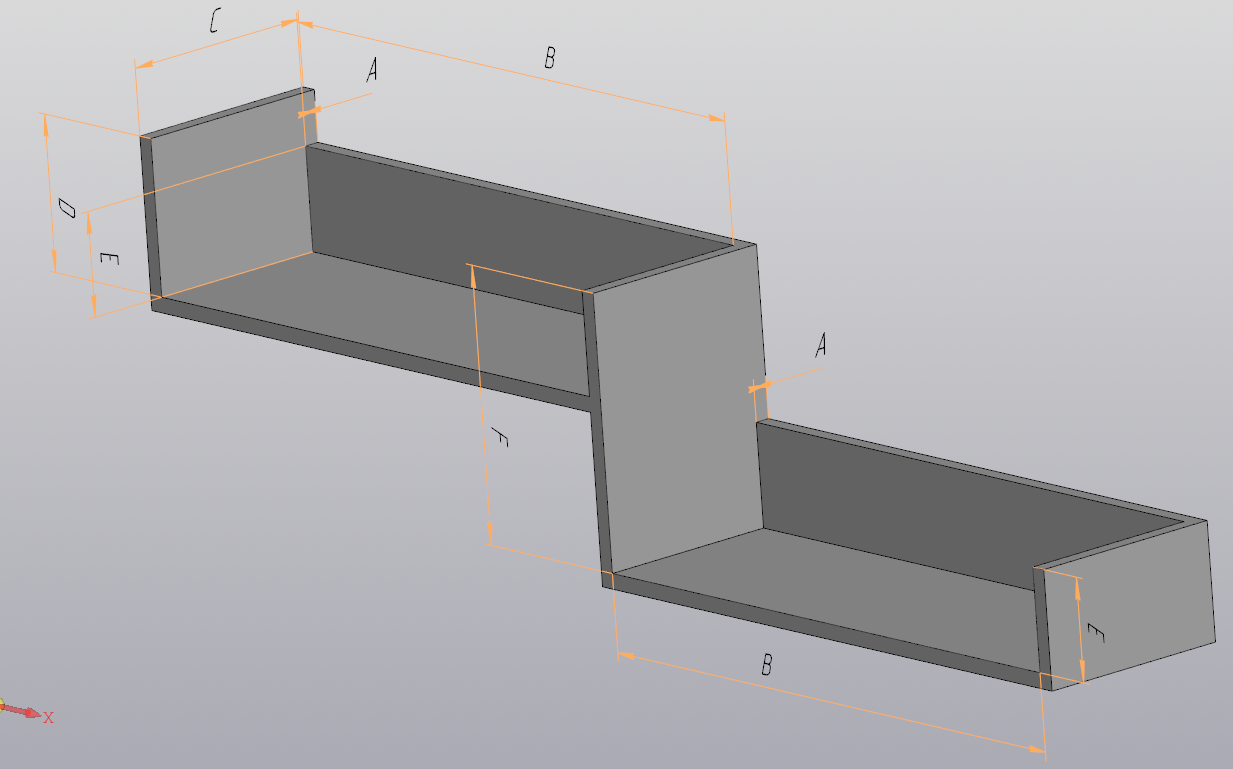
На рисунке 4.1 представлена модель полок с указанными параметрами.

Рисунок 4.1 — 3D-модель подвесных полок

**5 Диаграмма классов**

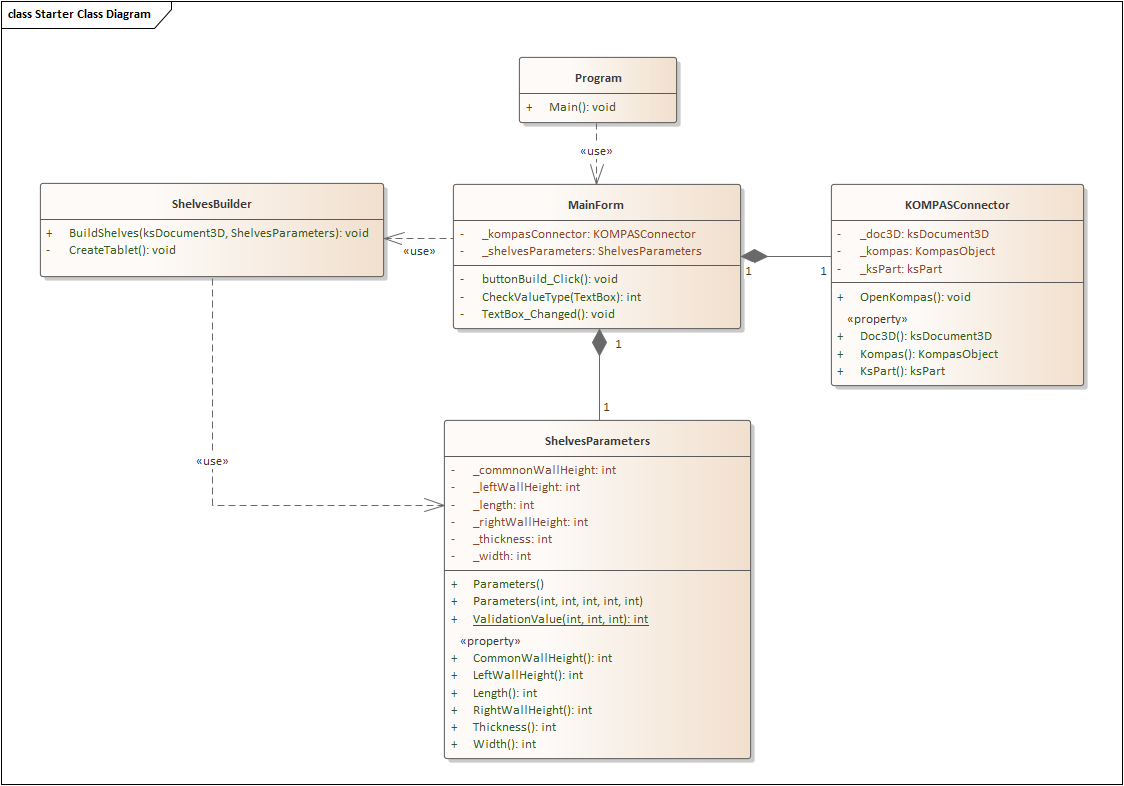
Диаграмма классов определяет типы классов системы и различного рода статические связи, которые существуют между ними. На диаграммах классов изображаются также атрибуты классов, операции классов и ограничения, которые накладываются на связи между классами[8]. Целью создания диаграммы классов является графическое представление статической структуры декларативных элементов системы. Диаграмма классов плагина представлена на рисунке 5.1.

Рисунок 5.1 — Диаграмма классов плагина «Подвесные полки»

Для реализации был выбран следующий набор классов:

1. MainForm — класс диалогового окна, который обеспечивает взаимодействие между пользователем и программой;

2. ShelvesParameters — класс, хранящий введенные параметры полок;

3. ShelvesBuilder — класс, осуществляющий вызов методов API, необходимых для постройки 3D-модели;

4. KOMPASConnector — класс для работы с API КОМПАС 3D.

На рисунке 5.2 изображена итоговая диаграмма классов приложения после завершения разработки дополнительного функционала и рефакторинга. После добавления новой функциональности в структуре приложения произошли следующие изменения:

1. Появилась связь-агрегация Builder и KompasConnector, теперь Buider хранит в себе объект KompasConnector, чтобы к нему было легче обращаться в других функциях в Builder.
2. Добавлено перечисление ParameterName для избавления от дублирования кода и упрощения работы с формой.
3. Добавлены новые функции в Builder: отдельное построение 2D-чертежа и выдавливание эскиза, а также дополнительная функциональность — скругление углов полок.
4. Добавлены новые функции в MainForm: изменение формы, реализация обработки TextBox, внесение параметров при нажатии кнопки и прочее.

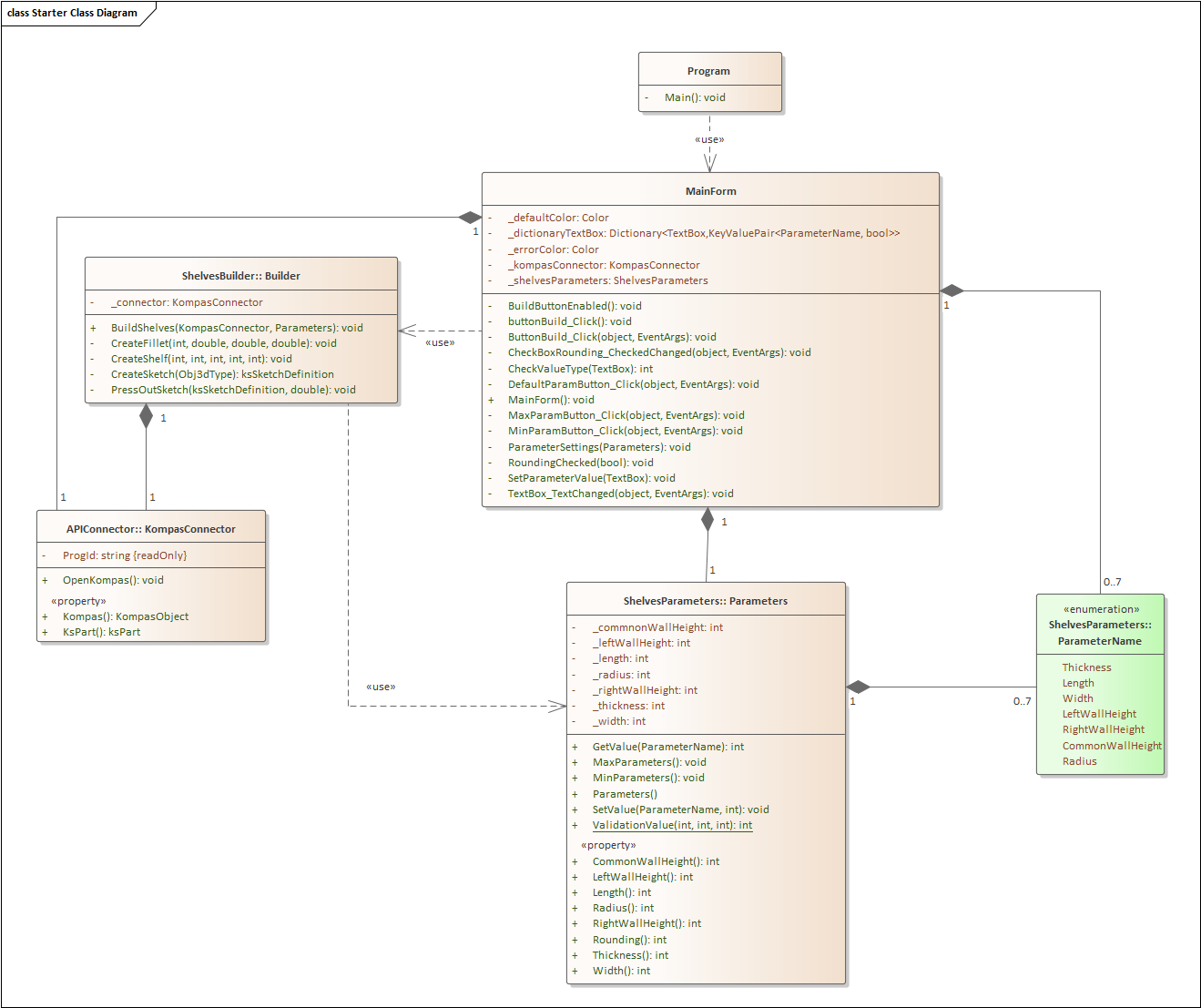


Рисунок 5.2 — Итоговая диаграмма классов

**6 Макет пользовательского интерфейса**

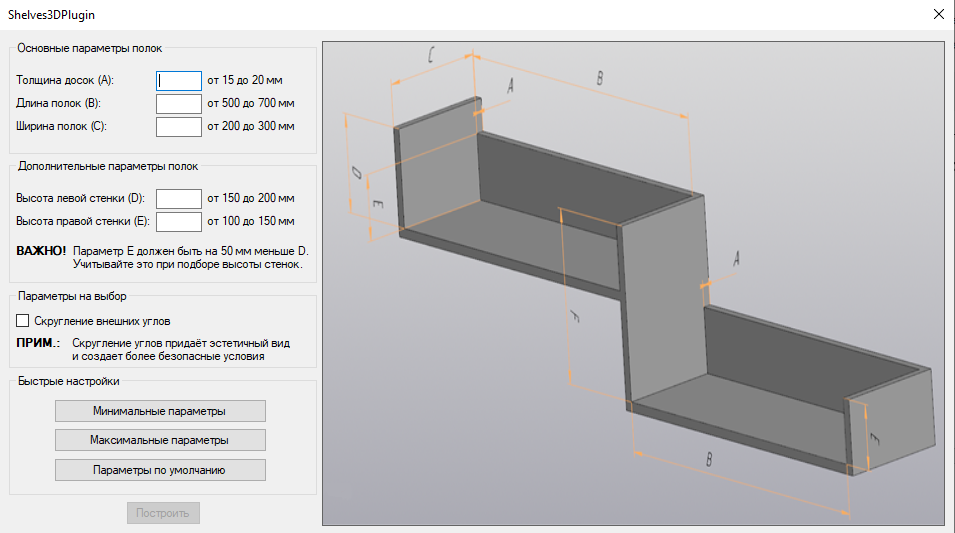
Пользовательский интерфейс представляет собой форму для ввода параметров. На форме присутствует 3D-модель с параметрами для демонстрации параметров полок справа и поля для ввода слева. Пользователь вводит значения самостоятельно, опираясь на подсказки, отображенные около полей. При нажатии на кнопку «Построить» проводится проверка зависимых параметров и, если условия соблюдены, строится 3D-модель подвесных полок. На рисунке 6.1 представлен макет пользовательского интерфейса.

Рисунок 6.1 — Макет пользовательского интерфейса

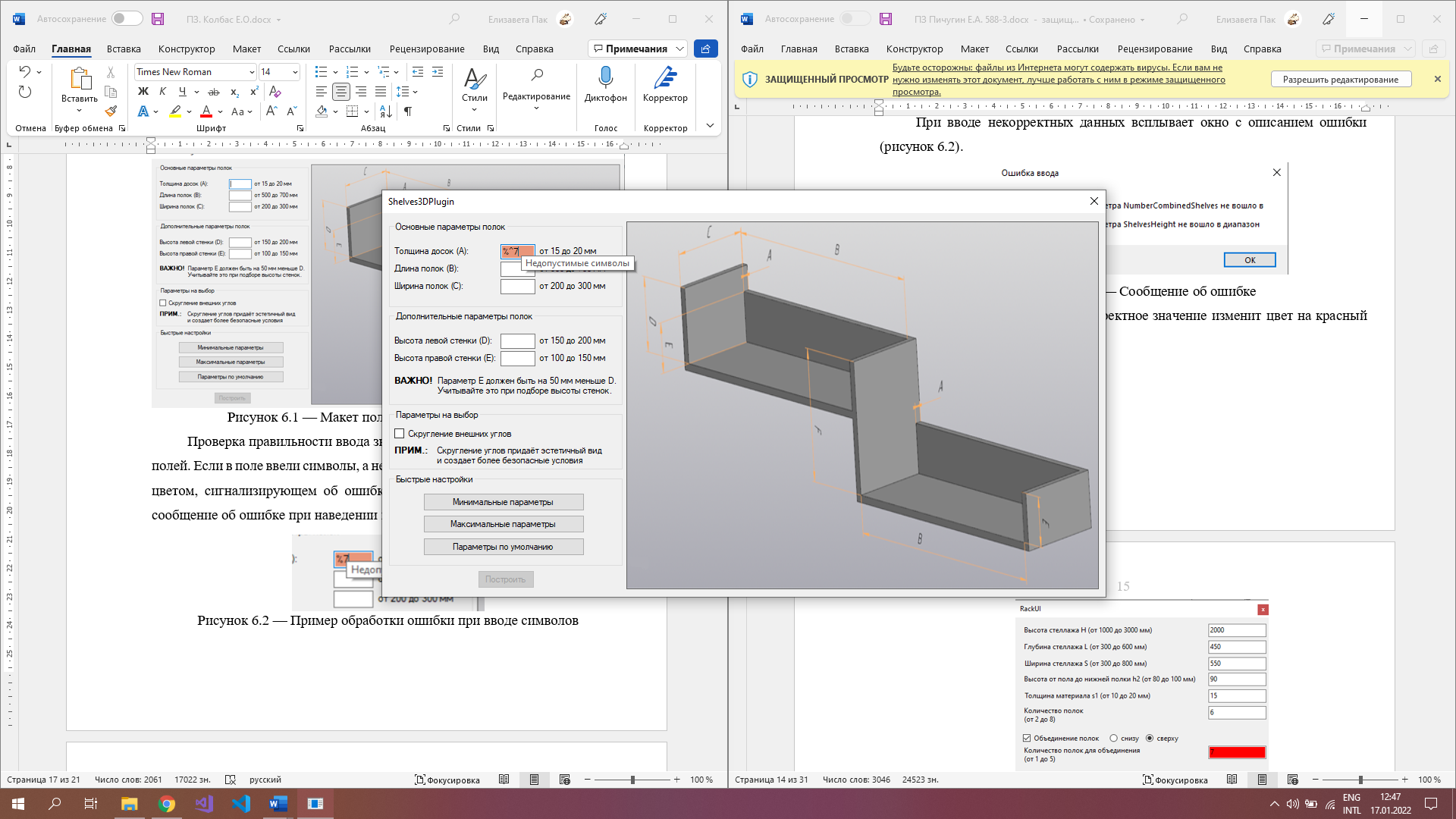
Проверка правильности ввода значений проводится по ходу заполнения полей. Если в поле ввели символы, а не цифры, то оно подсвечивается красным цветом, сигнализирующем об ошибке. Также дополнительно показывается сообщение об ошибке при наведении курсора на текст (рисунок 6.2).

Рисунок 6.2 — Пример обработки ошибки при вводе символов

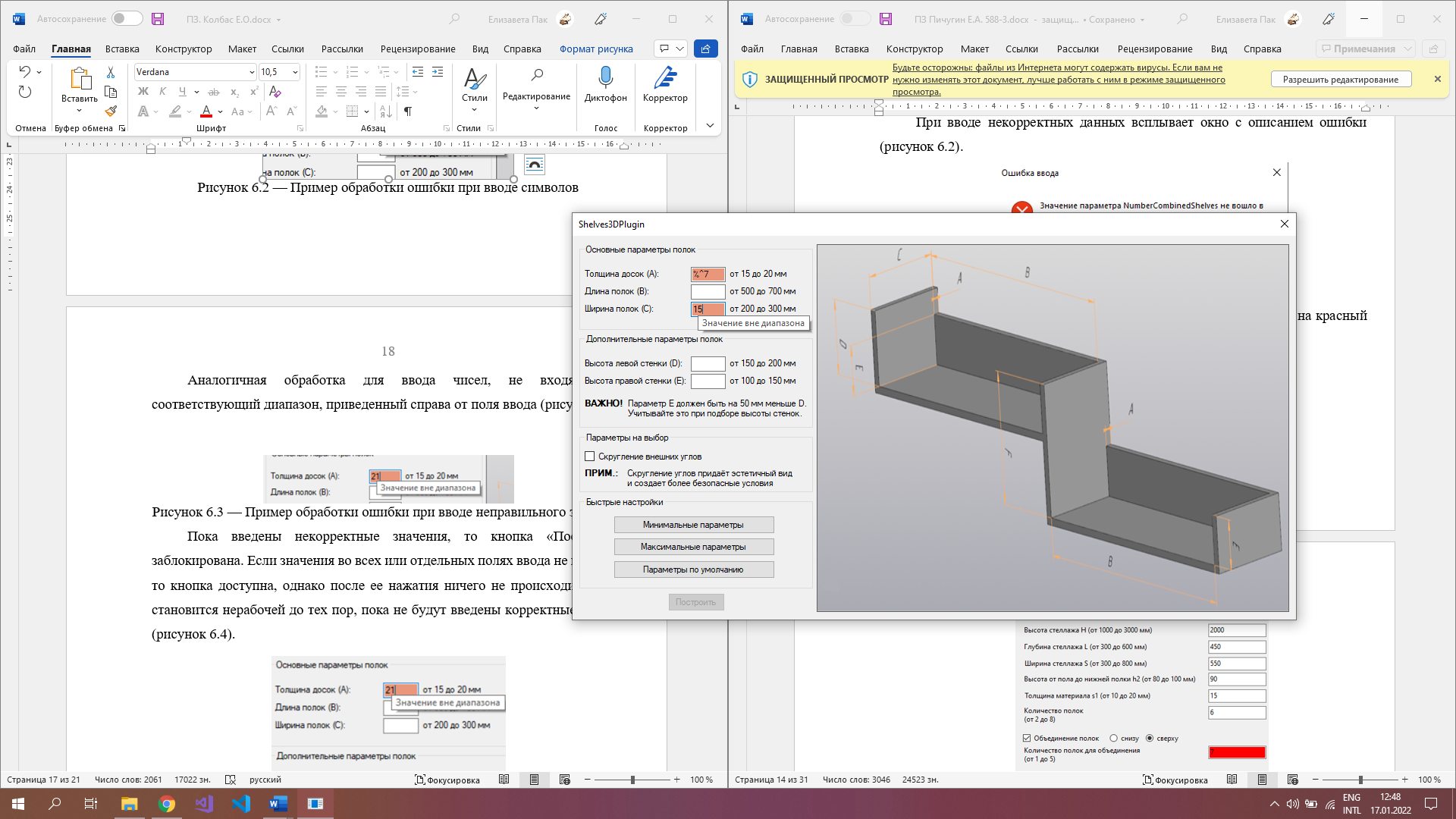
Аналогичная обработка для ввода чисел, не входящих в соответствующий диапазон, приведенный справа от поля ввода (рисунок 6.3).

Рисунок 6.3 — Пример обработки ошибки при вводе неправильного значения

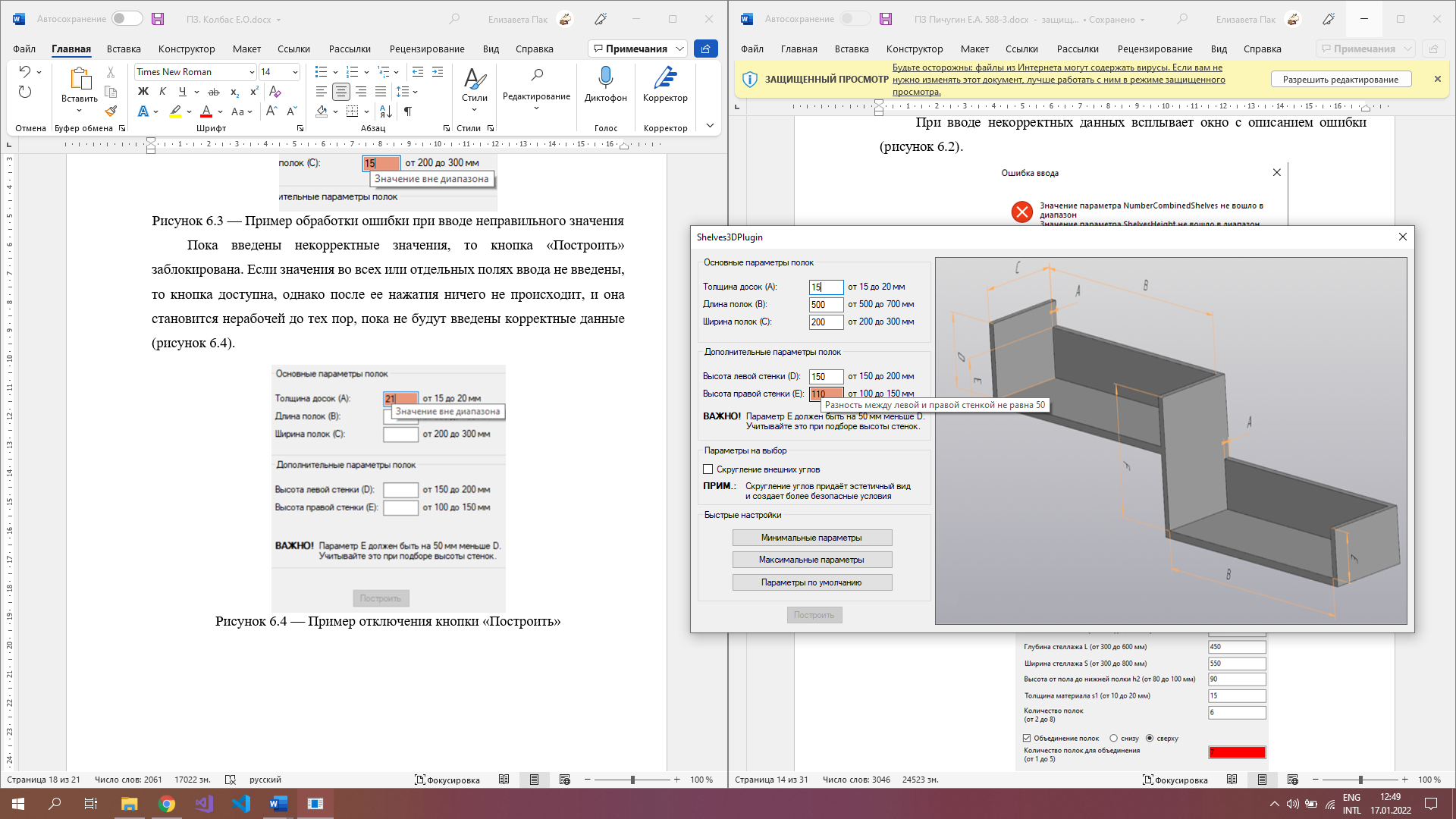
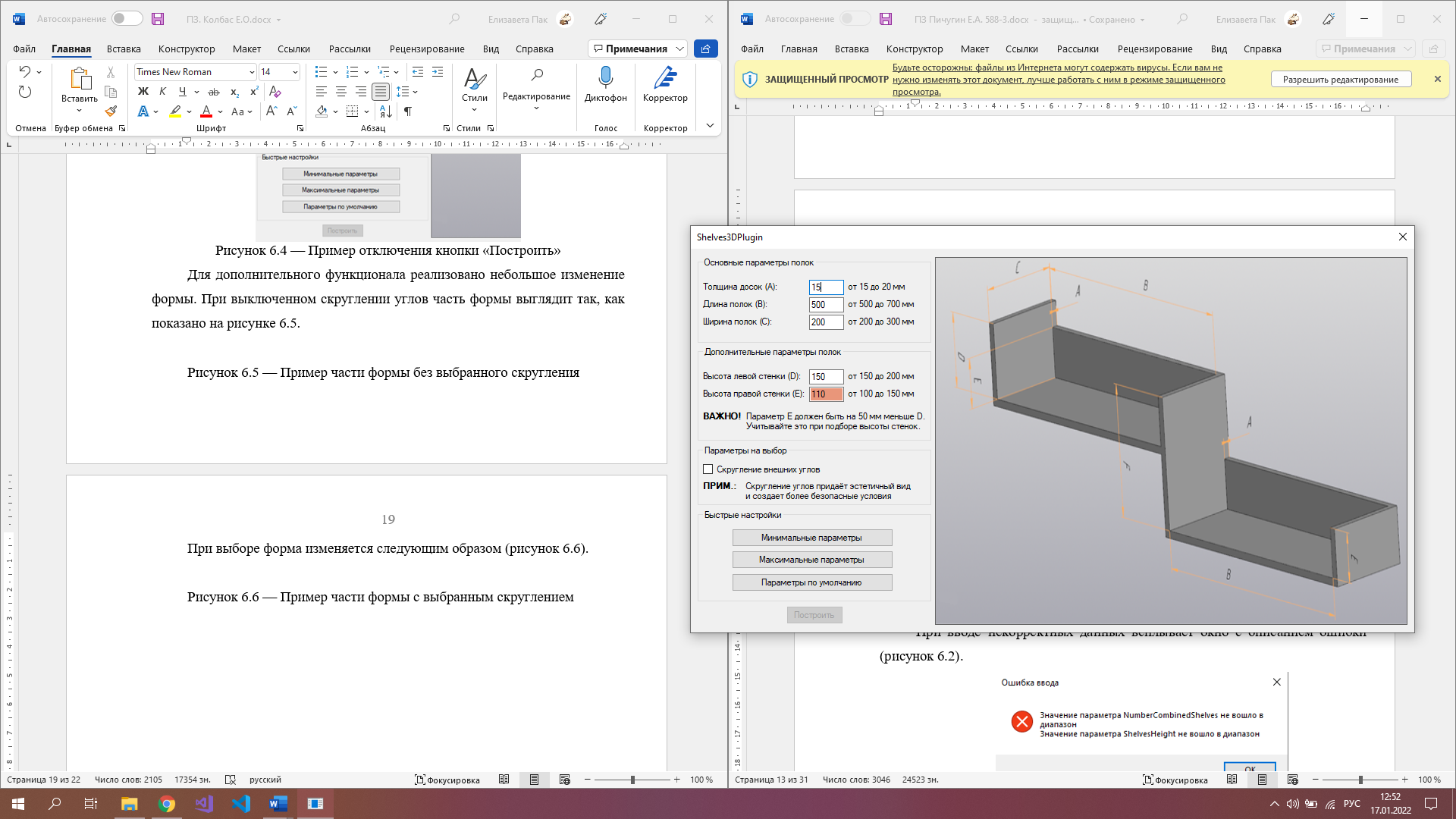
Пока введены некорректные значения, то кнопка «Построить» заблокирована. Если значения во всех или отдельных полях ввода не введены, то кнопка доступна, однако после ее нажатия ничего не происходит, и она становится нерабочей до тех пор, пока не будут введены корректные данные (рисунок 6.4).

Рисунок 6.4 — Пример отключения кнопки «Построить»

Для дополнительного функционала реализовано небольшое изменение формы. При выключенном скруглении углов часть формы выглядит так, как показано на рисунке 6.5.

Рисунок 6.5 — Пример части формы без выбранного скругления

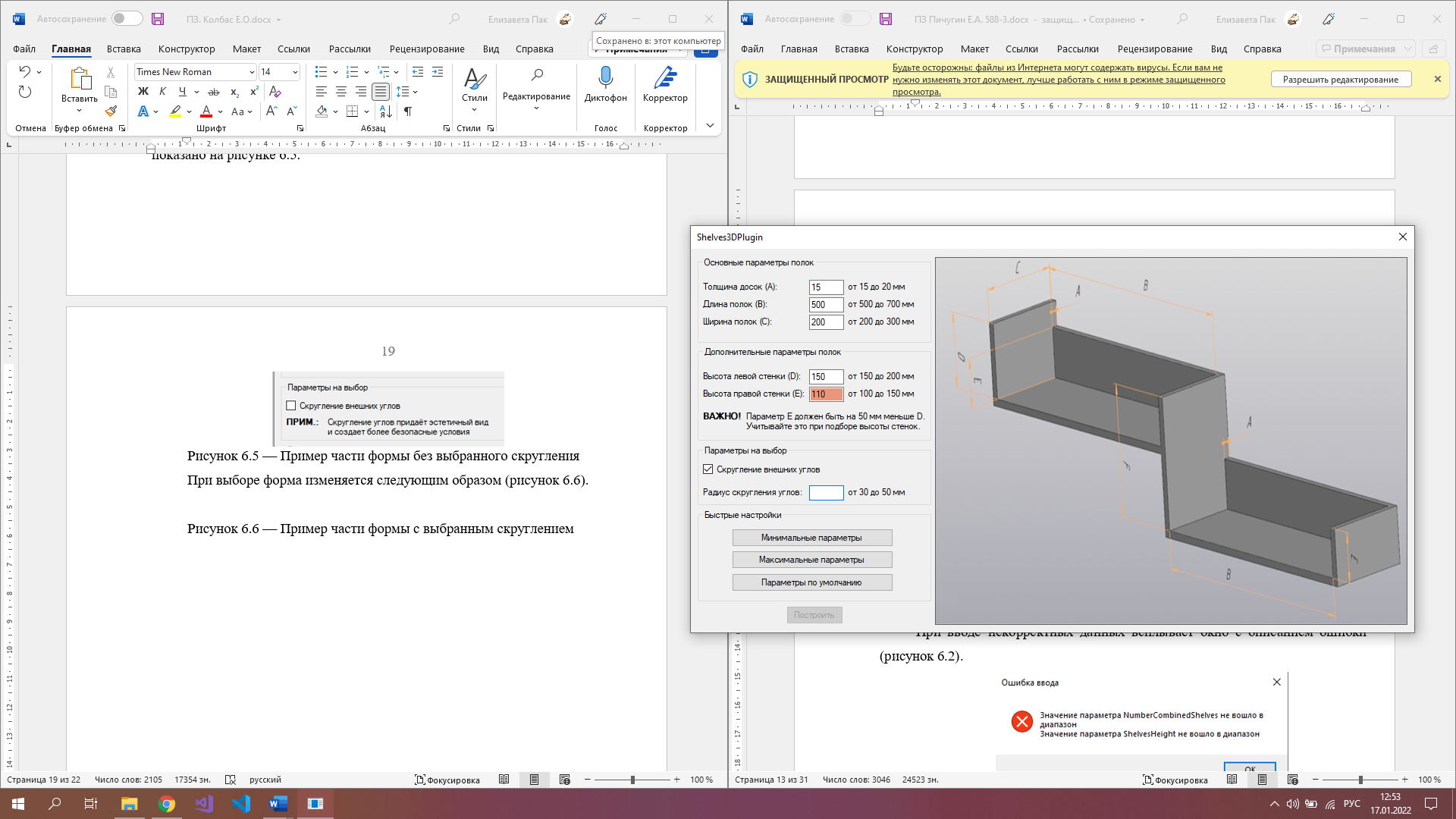
При выборе форма изменяется следующим образом (рисунок 6.6).

Рисунок 6.6 — Пример части формы с выбранным скруглением

**7 Тестирование плагина**

**7.1 Модульное тестирование**

Юнит-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 объект должен быть максимально изолирован, то есть не использовать объекты и методы других классов[9].

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 3.13.2 для языков платформы .NET.

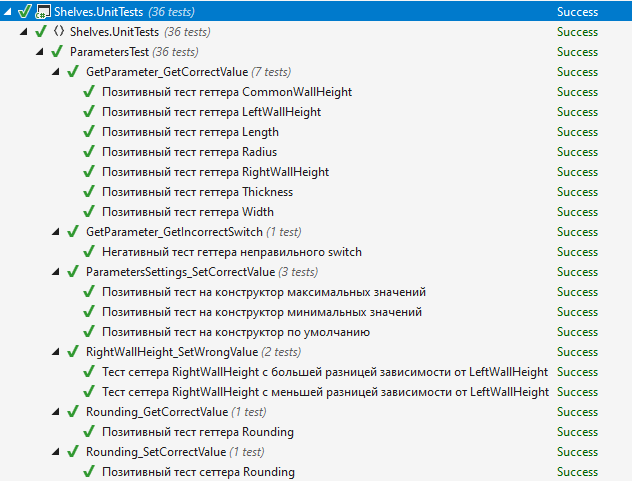
Результаты прохождения всех модульных тестов приведены на рисунках 7.1 и 7.2.

Рисунок 7.1 — Список пройденных юнит-тестов для класса Parameters

Изображение выглядит как стол

Автоматически созданное описаниеРисунок 7.2 — Список пройденных юнит-тестов для класса Parameters

Изображение выглядит как стол

Автоматически созданное описаниеРезультат покрытия модели Parameters тестами приведен на рисунке 7.3. Цикломатическая сложность для класса Parameters равна …

Рисунок 7.3 — Результат покрытия тестами

**7.2 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 параметрами (рисунки 7.4 и 7.5).

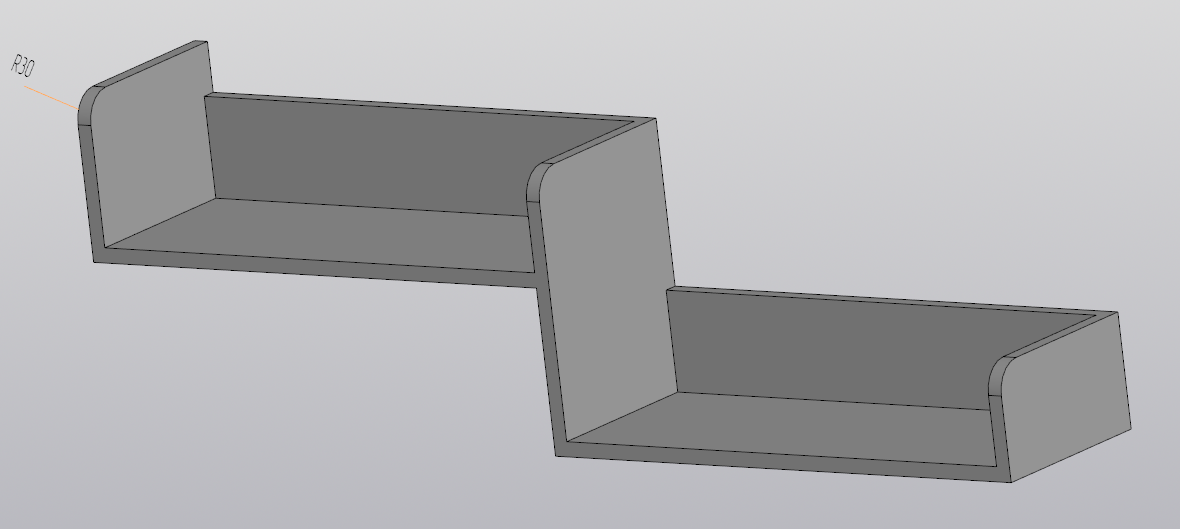
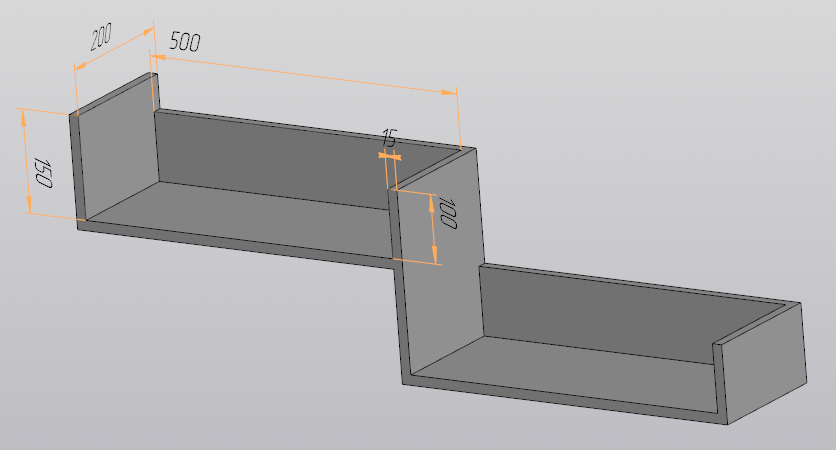
1. Толщина досок: 15 мм.
2. Длина полок: 500 мм.
3. Ширина полок: 200 мм.
4. Высота левой стенки: 150 мм.
5. Высота правой стенки: 100 мм.
6. Скругление внешних углов: да.
7. Радиус скругления: 30 мм.

Рисунок 7.4 — Модель подвесных полок с минимальными параметрами со скруглением углов

Рисунок 7.5 — Модель подвесных полок с минимальными параметрами без скругления углов

Построение модели с максимальными входными параметрами (рисунки 7.6 и 7.7).

1. Толщина досок: 20 мм.
2. Длина полок: 700 мм.
3. Ширина полок: 300 мм.
4. Высота левой стенки: 200 мм.
5. Высота правой стенки: 150 мм.
6. Скругление внешних углов: да.
7. Радиус скругления: 50 мм.

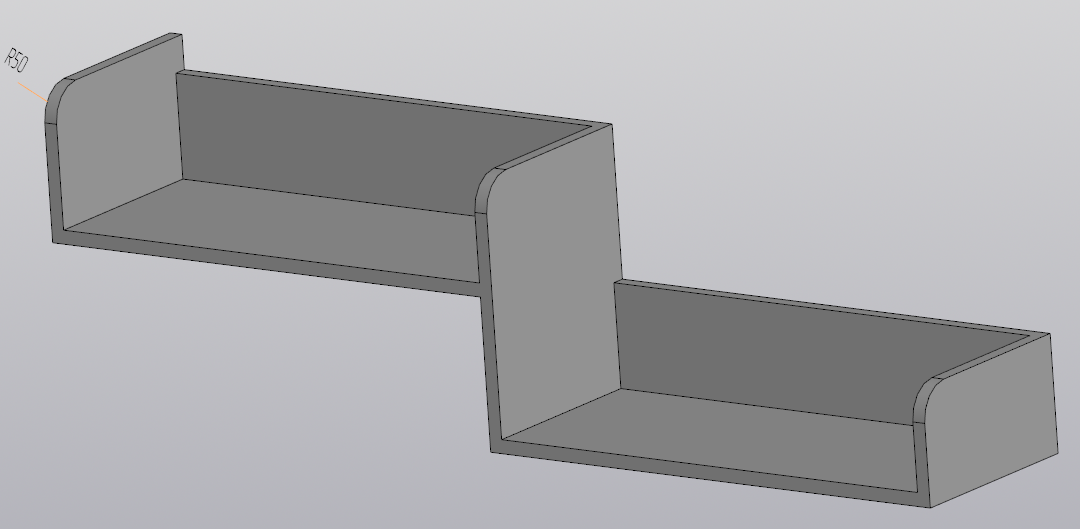
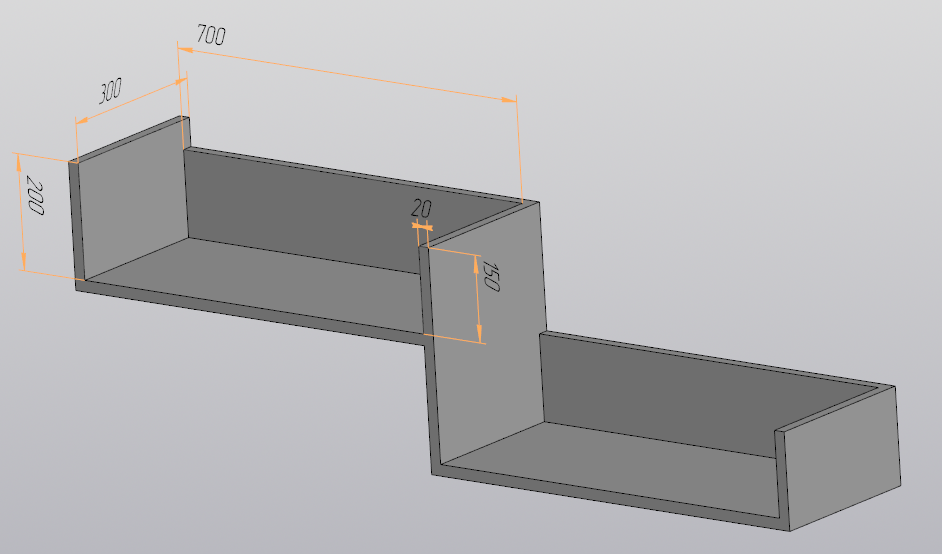
Рисунок 7.7 — Модель подвесных полок с максимальными параметрами со скруглением углов

Рисунок 7.8 — Модель подвесных полок с максимальными параметрами без скругления углов

Построение модели с входными параметрами по умолчанию (рисунки 7.9 и 7.10).

1. Толщина досок: 17 мм.
2. Длина полок: 600 мм.
3. Ширина полок: 250 мм.
4. Высота левой стенки: 175 мм.
5. Высота правой стенки: 125 мм.
6. Скругление внешних углов: да.
7. Изображение выглядит как текст, легкий, футляр

   Автоматически созданное описаниеРадиус скругления: 40 мм.

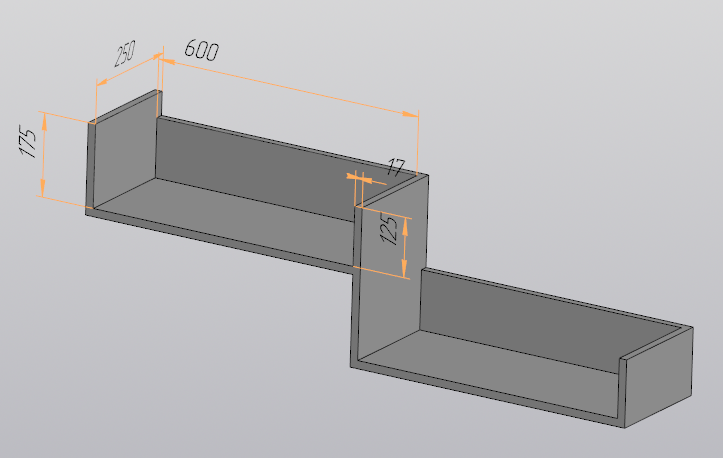
Рисунок 7.9 — Модель подвесных полок с параметрами по умолчанию со скруглением углов

Рисунок 7.10 — Модель подвесных полок с параметрами по умолчанию без скругления углов

**7.3 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[10].

Нагрузочное тестирование проводилось на компьютере со следующими характеристиками:

— Процессор: Intel(R) Core(TM) i5-7200U CPU;

— Оперативная память: 4.00Gb;

— Графический профессор: NVIDIA GeForce 940MX;

— Операционная система: Windows 10 Home 64-bit.

На рисунке 7.11 представлено затраченное время при построении 94 полок, а на рисунке 7.12 количество оперативной памяти, потребляемой при построении.

Рисунок 7.11 — График зависимости количества построенных полок от времени построения

Рисунок 7.12 — График зависимости количества построенных полок от потребляемой памяти

На построение 94 полок примерно было затрачено чуть менее 5 минут. Среднестатистическое время построения слова составляет 2,6 секунды. Высокая скорость построения полок объясняется тем, что модель имеет не много деталей и содержит легкие операции.

Из рисунка 7.12 можно сделать выводы, что память расходуется неравномерно. Самое большое количество занимаемой памяти приходится на моменты из рисунка 7.11, когда время построения резко возрастает.

Первый скачок времени и памяти предположительно обуславливается работой сборщика мусора.

Так же можно предположить, что в этот момент произошел переход в виртуальную память, то есть использование файла подкачки. Виртуальная память — метод управления памятью компьютера, позволяющий выполнять программы, требующие больше оперативной памяти, чем имеется в компьютере, путём автоматического перемещения частей программы между основной памятью и вторичным хранилищем (например, твердотельным накопителем).

Второй скачок обуславливается одновременной загрузкой обновлений самой системы. В этот момент на экране было показано уведомление о возможности перезагрузки компьютера для их установки.

**8 Заключение**

При выполнении лабораторных работ были изучены основные этапы проектирования программного продукта и составления проекта системы плагина, предметная область объекта построения, API «КОМПАС-3D». Были составлены такие документы как техническое задание, разработан проект системы, составлены UML-диаграммы классов, а также разработан макет пользовательского интерфейса и в заключении пояснительная записка к готовому продукту.

Результатом работы является плагин для САПР «Компас-3D», который выполняет построения подвесных полок с различными переменными параметрами.

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

**Список использованных источников**

1. Общие сведения о САПР [Электронный ресурс]. – Режим доступа: http://www.hi-edu.ru/e-books/xbook116/01/part-002.htm, свободный (дата обращения: 16.10.2021).

2. Обзор популярных систем автоматического проектирования (CAD) [Электронный ресурс]. – Режим доступа: https://www.pointcad.ru/novosti/obzor-sistem-avtomatizirovannogo-proektirovaniya, свободный (дата обращения: 16.10.2021).

3. КОМПАС-3D: О программе [Электронный ресурс]. – Режим доступа: https://kompas.ru/kompas-3d/about/, свободный (дата обращения: 18.10.2021).

4. Что такое API? [Электронный ресурс]. – Режим доступа: https://dev.by/news/chto-takoe-api-prostym-yazykom, свободный (дата обращения: 18.10.2021).

5. Работа с API КОМПАС-3D [Электронный ресурс]. – Режим доступа: https://habr.com/ru/company/ascon/blog/328088/, свободный (дата обращения: 21.10.2021).

6. Бесплатные аналоги КОМПАС-3D [Электронный ресурс]. – Режим доступа: https://freeanalogs.ru/Kompas3D, свободный (дата обращения: 22.10.2021).

7. Навесные полки [Электронный ресурс]. – Режим доступа: http://mebelsuite.ru/navesnye-polki/, свободный (дата обращения: 23.10.2021).

8. Фаулер М. UML. Основы. 3-е издание / М.Фаулер. – 3-е изд., пер. с англ. – СПб.: Символ-Плюс, 2019. – 192 с.

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

10. Нагрузочное тестирование vs Тестирование производительности. [Электронный ресурс]. – https://performance-lab.ru/blog/load-testing/testirovanie-proizvoditelnosti (дата обращения 29.12.2021).

**Приложение А**

(справочное)

Таблица А.1 — Тестовые сценарии

|  |  |  |
| --- | --- | --- |
| **Тестовый метод** | **Входные параметры** | **Описание тестового случая** |
| GetParameter\_GetCorrectValue  (ParameterName parameterName,  int expected) | ParameterName.Thickness, 17 | Позитивный тест геттеров по значениям по умолчанию |
| ParameterName.Length, 600 |
| ParameterName.Width, 250 |
| ParameterName.LeftWallHeight, 175 |
| ParameterName.RightWallHeight, 125 |
| ParameterName.CommonWallHeight, 317 |
| ParameterName.Radius, 40 |
| Rounding\_GetCorrectValue  (bool expected) | true | Позитивный тест геттера наличия скругления |
| GetParameter\_GetIncorrectSwitch  (ParameterName parameterName) | 10 | Негативный тест со значением, не входящим в выбор |
| SetParameter\_SetCorrectValue  (ParameterName parameterName,  int value) | ParameterName.Thickness, 17 | Позитивный тест на внесение корректных данных |
| ParameterName.Length, 600 |
| ParameterName.Width, 250 |
| ParameterName.LeftWallHeight, 175 |
| ParameterName.RightWallHeight, 125 |
| ParameterName.CommonWallHeight, 0 |
| ParameterName.Radius, 40 |

Продолжение таблицы А.1

|  |  |  |
| --- | --- | --- |
| **Тестовый метод** | **Входные параметры** | **Описание тестового случая** |
| Rounding\_SetCorrectValue  (bool value) | true | Позитивный тест на внесение корректного значение скругления |
| SetParameter\_SetIncorrectSwitch  (ParameterName parameterName,  int value) | 10, 100 | Негативный тест на внесение данных с выходящим из диапазона выбора значений |
| SetParameter\_SetOutOfRangeValue  (ParameterName parameterName,  int value) | ParameterName.Thickness, 10 | Негативный тест на внесение некорректных данных: выходящих из диапазона мин-макс |
| ParameterName.Thickness, 30 |
| ParameterName.Length, 400 |
| ParameterName.Length, 800 |
| ParameterName.Width, 100 |
| ParameterName.Width, 400 |
| ParameterName.LeftWallHeight, 100 |
| ParameterName.LeftWallHeight, 300 |
| ParameterName.RightWallHeight, 50 |
| ParameterName.RightWallHeight, 200 |
| ParameterName.CommonWallHeight, 100 |
| ParameterName.Radius, 20 |
| ParameterName.Radius, 60 |

Продолжение таблицы А.1

|  |  |  |
| --- | --- | --- |
| **Тестовый метод** | **Входные параметры** | **Описание тестового случая** |
| ParametersSettings\_SetCorrectValue  (int thickness, int length, int width, int leftwall, int rightwall,  int commonwall, bool rounding,  int radius, string name, int count) | 17, 600, 250, 175, 125,  0, true, 40, "по умолчанию", -1 | Тест на корректность внесения статичных параметров: по умолчанию, минимум, максимум |
| 15, 500, 200, 150, 100,  0, true, 30, "минимума", 0 |
| 20, 700, 300, 200, 150,  0, true, 50, "максимума", 1 |
| RightWallHeight\_SetWrongValue  (int wrongValue, string message) | 130 | Негативный тест зависимого параметра: больше разницы и меньше разницы |
| 120 |