Министерство образования и науки 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 ЖУРНАЛЬНОГО СТОЛИКА» ДЛЯ САПР «КОМПАС-3D» v 15.2

Пояснительная записка к индивидуальному проекту по дисциплине «Основы разработки САПР»

Студент гр. 584-2

_____ Савындай Г. Ю.

_____

Руководитель

к. т. н., доц. каф. КСУП

_____ _____ Калентьев А. А.

_____

Томск 2018

Реферат

Индивидуальный проект, 32 с., 15 рис., 1 табл., 6 источников, 1 прилож.

КОМПАС 3D, САПР, ЖУРНАЛЬНЫЙ СТОЛИК, API, ПЛАГИН, .NET FRAMEWORK.

Объектом разработки является плагин «Построение журнального столика» для САПР «КОМПАС-3D» v 15.2.

Цель работы – создание плагина для построения трехмерной модели журнального столика по введенным параметрам в «КОМПАС-3D».

В процессе выполнения работы над индивидуальным проектом разработан плагин для построения журнального столика в рабочей плоскости САПР «КОМПАС-3D», а также проектная документация к плагину.

Министерство образования и науки 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)

Утверждаю:

Зав. кафедрой КСУП

_____Ю.А. Шурыгин

«___»_____ 2017г.

ТЕХНИЧЕСКОЕ ЗАДАНИЕ

по индивидуальному проекту по дисциплине «Основы разработки САПР»

Выдано: студенту гр. 584-2 Савындаю Гелцену Юрьевичу

1) Тема проекта: Разработка плагина «Журнальный столик» для «КОМПАС-3D» V15.2. Пример журнального столика приведен на рисунке 1.1.

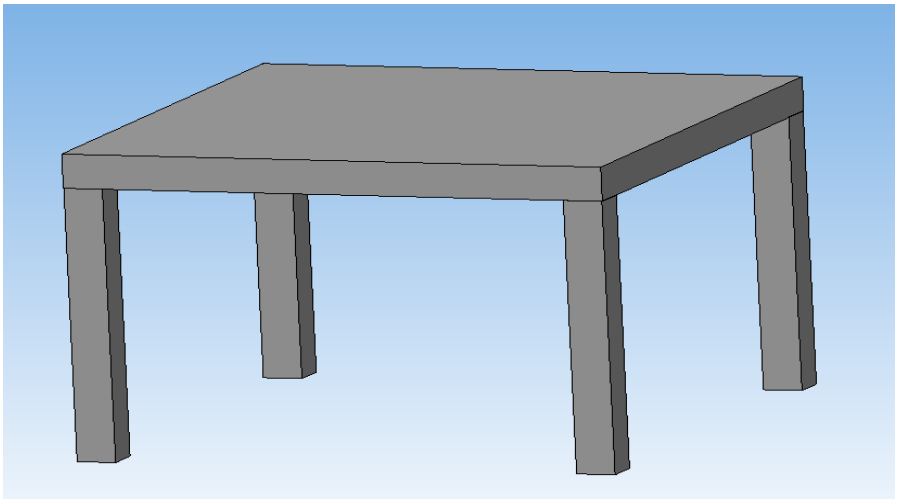

Рисунок 1.1 – Журнальный столик

2) Срок сдачи студентом проекта: 21.12.2017

3) Исходные данные

Разработать плагин «Журнальный столик» для «КОМПАС-3D V15.2 учебная версия»

4) Требования к плагину

Плагин должен обеспечивать следующую функциональность:

− выводить диалоговое окно ввода для изменения следующих параметров:

– ввод параметров журнального столика (рисунок 1.2);

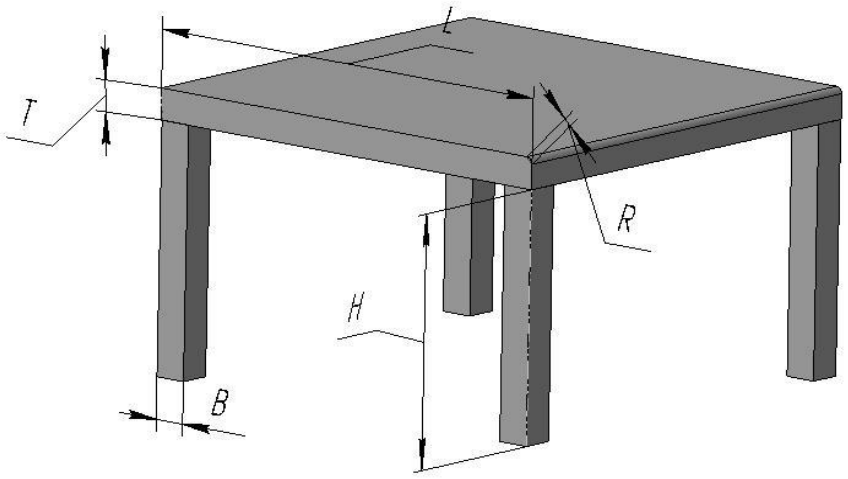

Рисунок 1.2 – Габариты столика «Лакк»

где: L — длина стороны квадратной столешницы(значение данного параметра должно быть больше или равно 50 сантиметрам и меньше или равно 90 сантиметров);

H — высота столика (значение данного параметра должно быть больше или равно 15 сантиметрам и меньше или равно 45 сантиметрам), если параметр L больше 70 см, не сможет принимать значения меньше 30 см;

T — толщина столешницы (значение данного параметра должно быть больше или равно 3 сантиметрам и меньше или

равно 10 сантиметрам, также значение этого параметра должно быть меньше параметра A);

B — длина стороны основания ножки, значение данного параметра должно быть больше или равно 3 сантиметрам и меньше или равно 6 сантиметрам;

R — скругление краев столешницы.

- обеспечивать построение трехмерной модели на графическом окне системы «КОМПАС-3D» на основе выбранной модели и введенных значений параметров;
- обеспечить корректность ввода данных и вывод информационного сообщения при вводе некорректных данных.

5) Сфера применения

Плагин применим при изготовлении журнальных столиков в сфере мебелестроения для решения задач проектирования в системе «КОМПАС-3D» V15.2.

6) Минимальные требования к программной и аппаратной частям:

- программа должна работать на операционной системе: Windows 10 (x32, x64);
- графическая карта Intel® HD Graphics 4000 либо AMD Radeon® HD 4570;
- процессор Intel® Core™ i3 CPU M 370 с тактовой частотой 2.4 ГГц;
- ОЗУ 2Гб или больше;

7) Инструменты разработки:

- язык программирования — C#, версия .NET Framework — 4.7.0;
- среда разработки Microsoft Visual Studio Community 2015;

8) Дата выдачи задания: 15.09.2017

Руководитель: к.т.н., доцент каф. КСУП

Калентьев А. А._____

Задание принял к исполнению

Савындай Г. Ю. _____

Оглавление

1 Введение

САПР – система автоматизированного проектирования, это сложные специализированные программные комплексы, которые позволяют упростить, ускорить и снизить себестоимость процесса проектирования. У большинства крупных САПР есть API (Application programming interface, программный интерфейс приложения) [1] — интерфейс, позволяющий сторонним программам расширить функциональность системы или автоматизировать рутинные операции.

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. В данном проекте стоит задача разработки плагина для построения 3D модели журнального столика в автоматизированном режиме. Плагин — независимо компилируемый программный модуль, динамически подключаемый к основной программе, предназначенный для расширения или использования ее возможностей [2].

2 Постановка и анализ задачи

Основная задача данного проекта — создания плагина, который автоматически строит журнальный столик с определенными параметрами в САПР «КОМПАС 3D» версии 15.2.

После написания ТЗ был составлен ПС [3], в котором описаны детали реализации и краткое описание API САПР, с которым плагин работает.

В ПС описаны диаграмма использования плагина, диаграмма классов плагина, макет плагина, сущности API, с помощью которых плагин строит модель в САПР «КОМПАС 3D».

В ходе разработки список требований к программе был изменён заказчиком: были добавлены два дополнительных параметра, с помощью которых строились разделители ножек у столика. Данные параметры являются опциональными. Так как в проект после утверждения ТЗ были внесены изменения, ПС не соответствует окончательной версии плагина: в диаграмме вариантов использования добавились действия по вводу двух новых параметров, также была изменена диаграмма классов и сама архитектура программы, в пользовательском интерфейсе появились поля для ввода двух новых параметров. Подробно об изменениях плагина в п. 4 данной работы.

У САПР две версии API: 5 и 7. В данной работе использовалась только 5 версия [3].

3 Описание алгоритмов построения

В данном разделе приведено описание алгоритмов, использованных в разработанном плагине для создания модели журнального столика. Описание алгоритмов является высокоуровневым, без названия конкретных сущностей. Более подробное описание в ПС [3].

3.1 Построение столешницы

Для создания столешницы нужно создать эскиз в плоскости XY. На этом эскизе рисуется контур столешницы. Контур строится по четырём точкам, отстоящих в обе стороны от начала координат и на половину стороны столешницы по оси Ox, и на половину по оси Oy. Точки соединяются, образуя квадратный контур с размером, равным длин стороны столешницы. Полученный квадрат выдавливается в сторону положительного направления ос Oz на глубину, заданную толщиной столешницы.

Округления краев столешницы делается по точкам в ребрах столешницы, то есть по сторонам квадратного контура столешницы.

3.2 Построение ножек

Для построения ножек потребуется вспомогательная плоскость, которая расположена от плоскости XY на расстоянии толщины столешницы в положительную сторону оси Oz. На этой плоскости рисуется эскиз ножек. Эскиз состоит из четырех квадратов со стороной, равной длине стороны основания ножек. Квадраты расположены по четырем сторонам контура столешницы. Для рисовки этих квадратов была написана функция, в которую нужно передать координаты верхней точки квадрата и длину стороны квадрата.

Полученные квадратики также выдавливаются в сторону положительного направления оси Oz на глубину, заданную высотой ножек.

3.3 Построение разделителей

Для построения разделителей также потребуется вспомогательная плоскость, которая находится на расстоянии, равной длине отступа разделителей от столешницы, от вспомогательной плоскости, использованной для построения ножек. В этой плоскости создается эскиз, на котором рисуются прямоугольники, длинная сторона, которой равна расстоянию между двумя смежными ножками, короткая сторона равна 20 мм. Для рисовки этих прямоугольников также была написана функция, которая принимает в параметры координаты угла прямоугольника, параметр для указания конкретного угла, так как контуры разделителей рисуются по-разному, ширина и длина прямоугольника. При рисовке прямоугольников делается отступ в 10 мм от краев контура столешницы. Полученные прямоугольники также выдавливаются в сторону положительного направления оси Oz на глубину, заданную высотой разделителей.

4 Описание реализации и анализ проекта

При разработке плагина были использованы следующие инструменты:

– язык C# 7.0, .NET Framework 4.7.02;

– Visual Studio 2015 Community;

– система контроля версий git 2.14.1.windows.1;

– KOMPAS-3D API5;

– библиотека модульного тестирования NUnit 3.10.1.

Исходный код плагина доступен в репозитории [3].

На рисунке 4.1 представлена проектная версия диаграммы вариантов использования, изначально представленная в ПС.

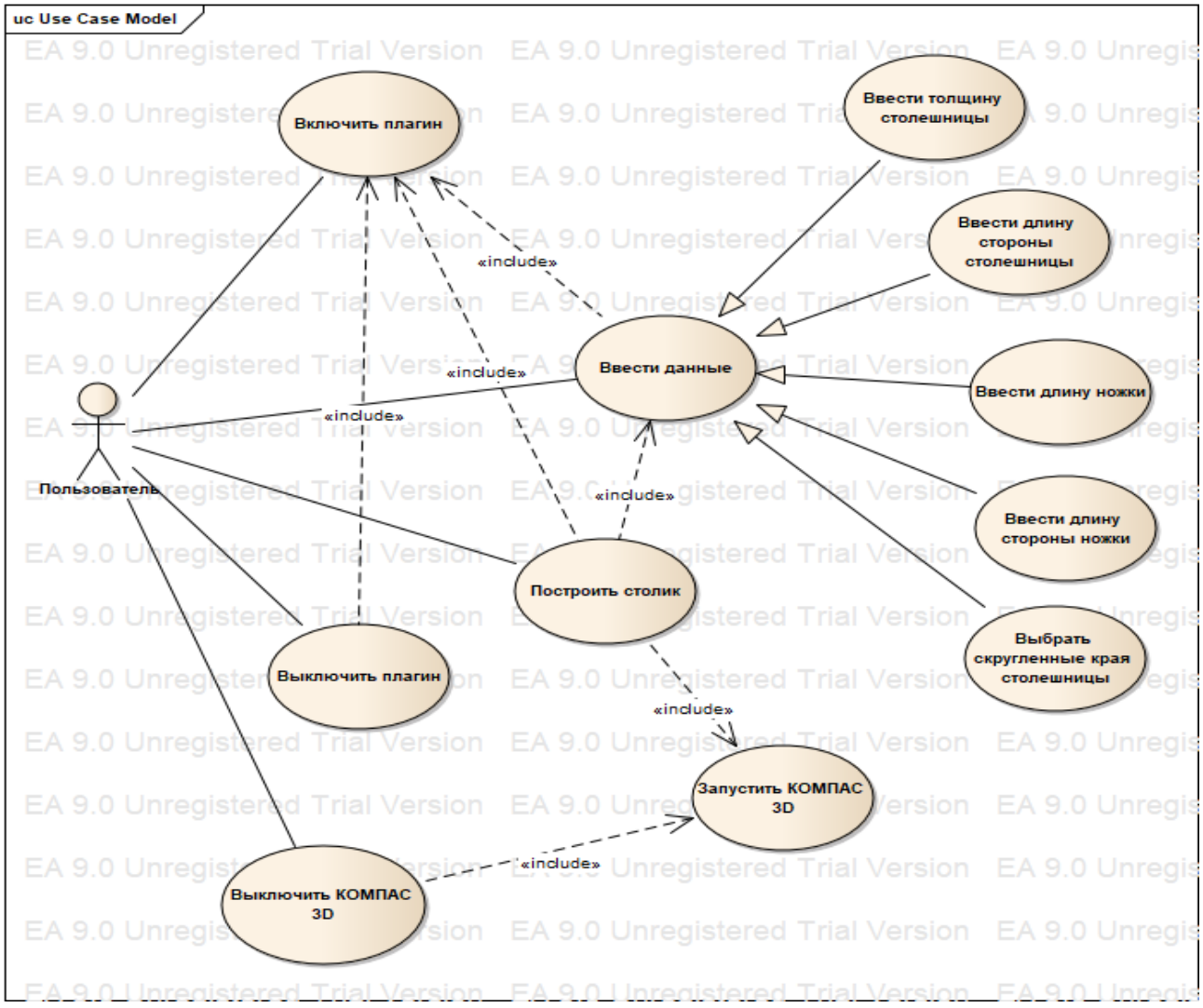

Рисунок 4.1 – Проектная версия диаграммы вариантов использования

На рисунке 4.4 представлена финальная версия диаграммы вариантов использования.

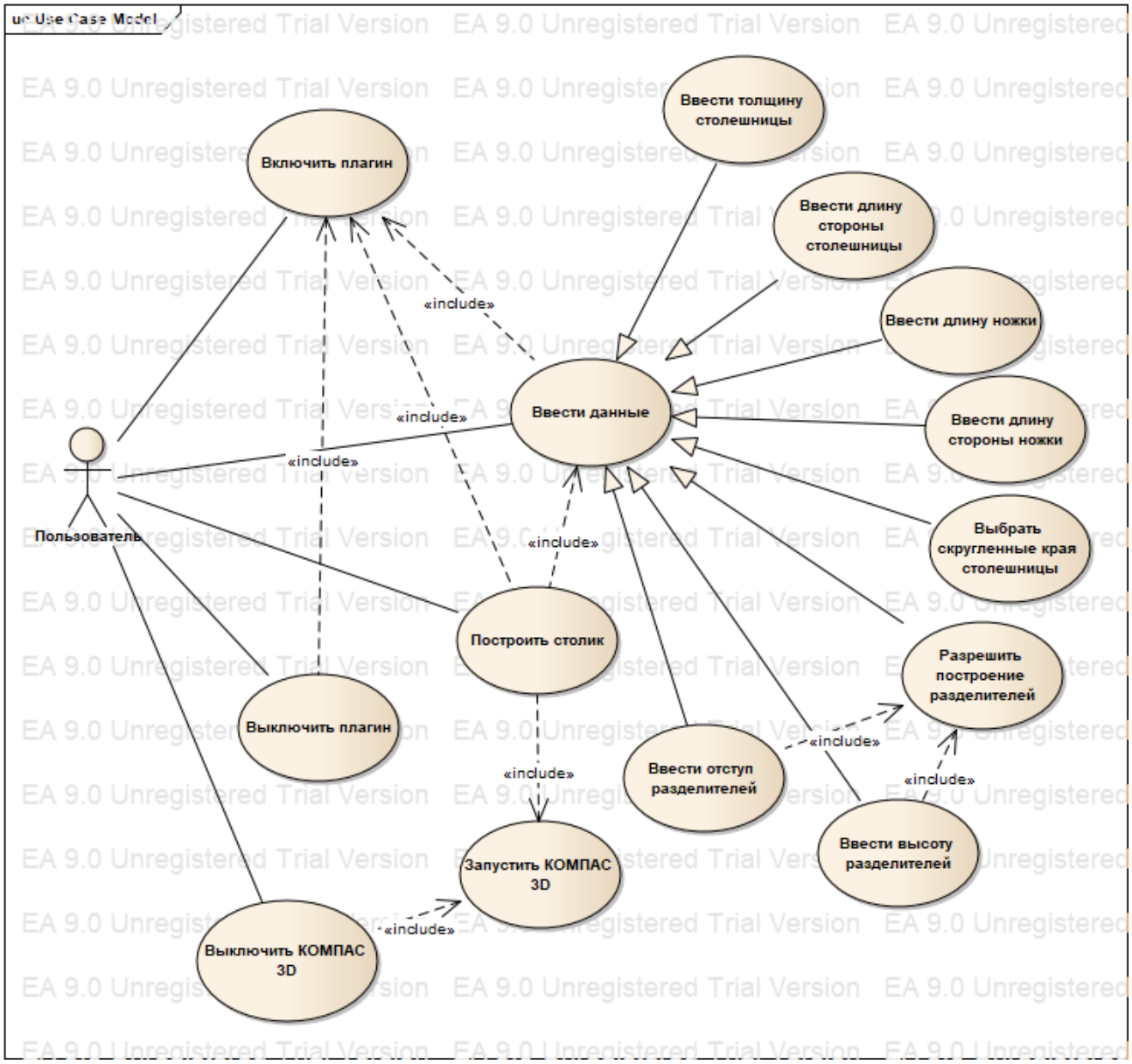

Рисунок 4.2 – Финальная версия диаграммы вариантов использования

В конечной версии диаграммы действие «Ввести данные» дополнено еще одним действием «Разрешить построение разделителей», которое состоит из двух действий: «Ввести отступ разделителя» и «Ввести высоту разделителей», которые будут доступны после выполнения действия «Построить разделители».

Предполагаемая структура классов программы плагина, созданная во время написания ПС до разработки программы, приведена на рисунке 4.3.

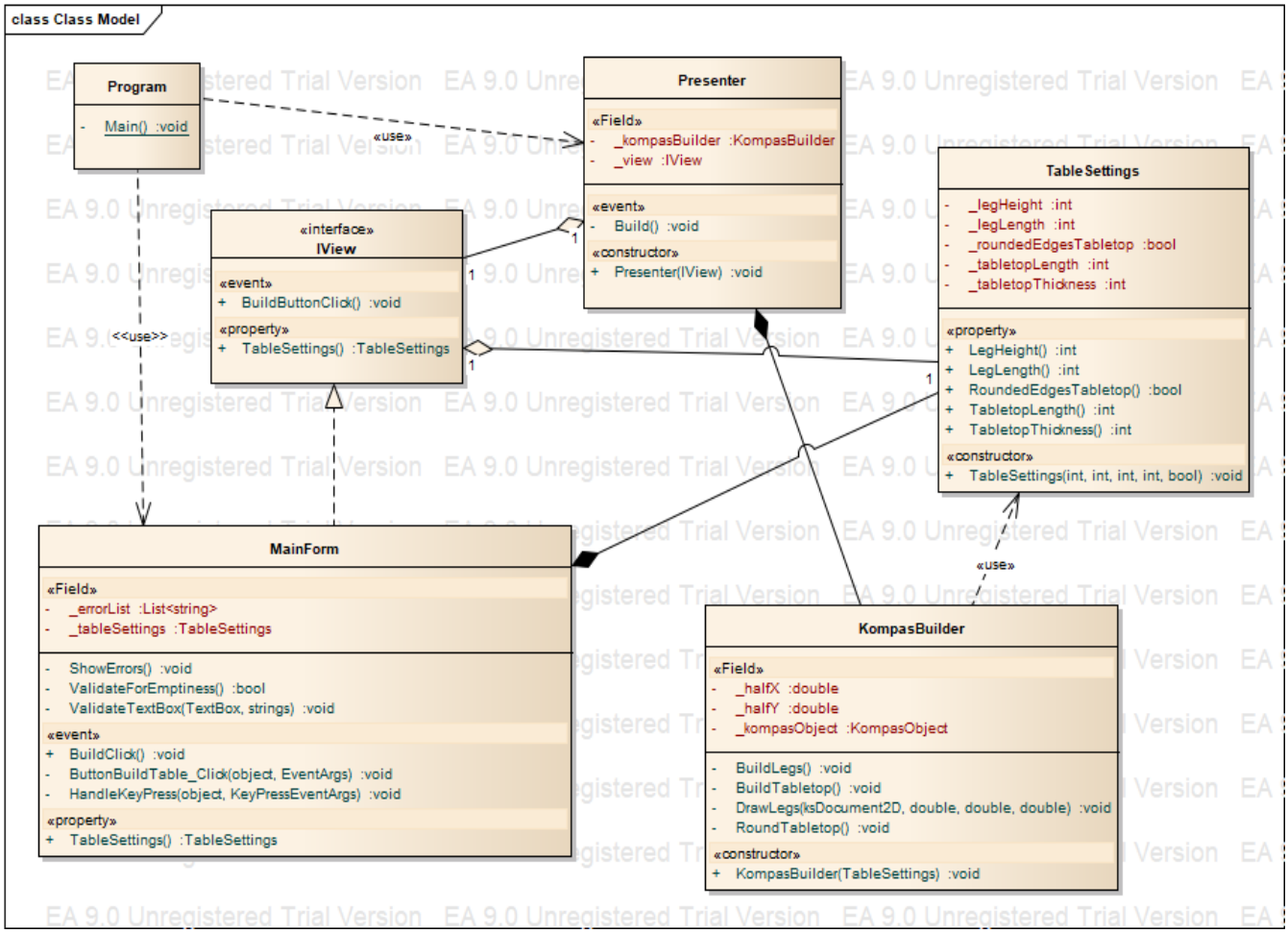

Рисунок 4.3 – Проектная версия диаграмма классов проекта

Структура классов окончательного варианта программы в виде UML-диаграммы классов уровня реализации с добавленными изменениями приведена на рисунке 4.4.

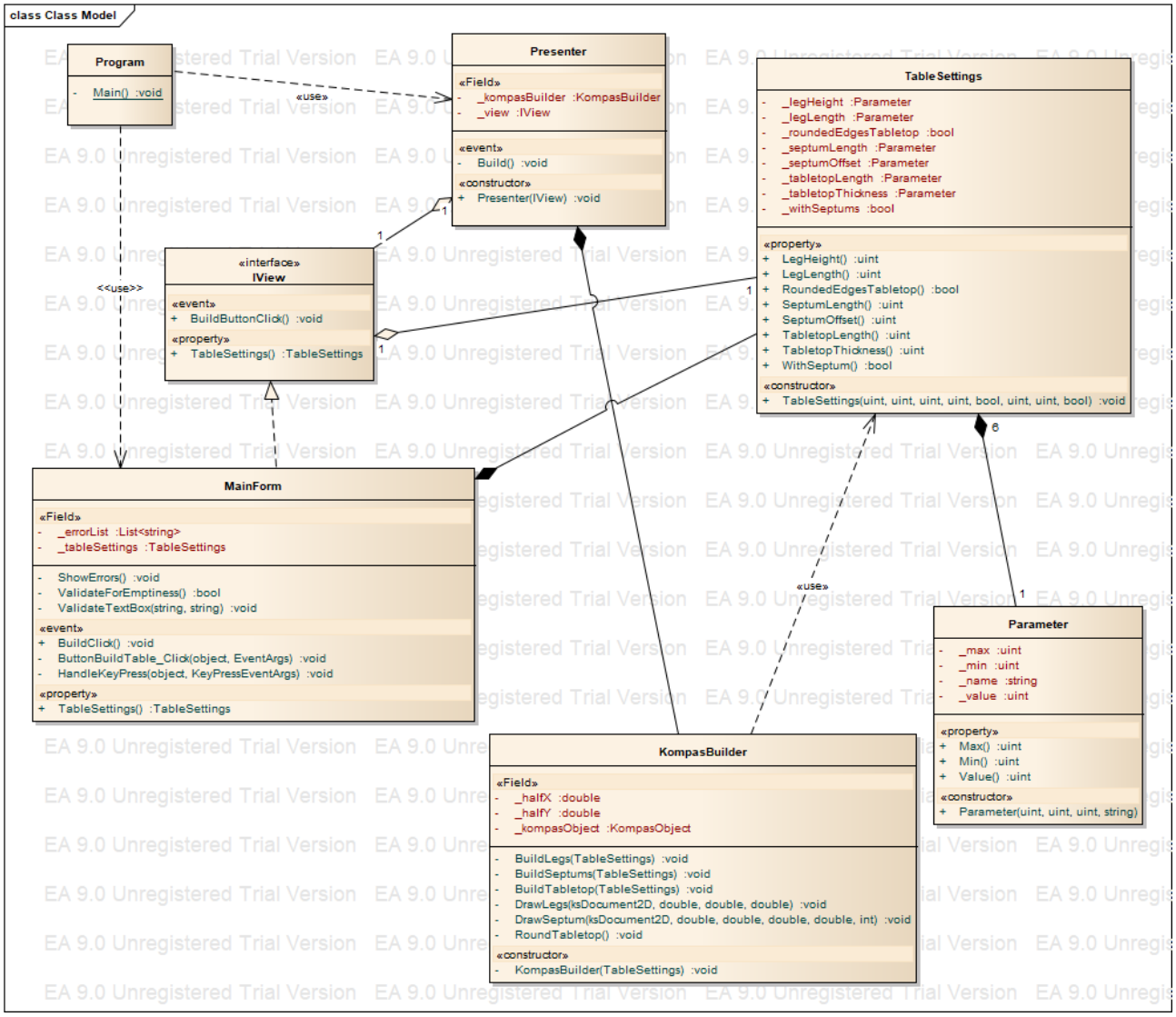

Рисунок 4.2 – Финальная версия диаграммы классов проекта

Изменения в диаграммах вызваны двум причинами. Первая причина – это изменение требований заказчика: необходимо было ввести еще два параметра для столика – высота разделителей и отступ разделителей от столешницы, то есть в класс TableSettings были добавлены поля _septumLength и _septumOffset, причем построение разделителей опционально, поэтому также добавлено поле _withSeptum для проверки на нажатие галочки построения разделителей в пользовательском интерфейсе программы. Также в класс KompasBuilder был добавлен метод BuildSeptums для построения разделителей и метод DrawSeptum для рисования разделителя. Вторая причина – слабая продуманность изначальной архитектуры программы по причине недостатка опыта в проектировании программных систем. Был создан новый класс

Parameter, характеризующий параметр класса TableSettings и служащий для валидации значения параметра. Также в классе TableSettings был закрыт публичный доступ к параметрам, то есть соблюдена инкапсуляция.

На рисунке 4.5 приведено сравнение макета пользовательского интерфейса из ПС и интерфейса финальной версии программы при вводе корректных данных.

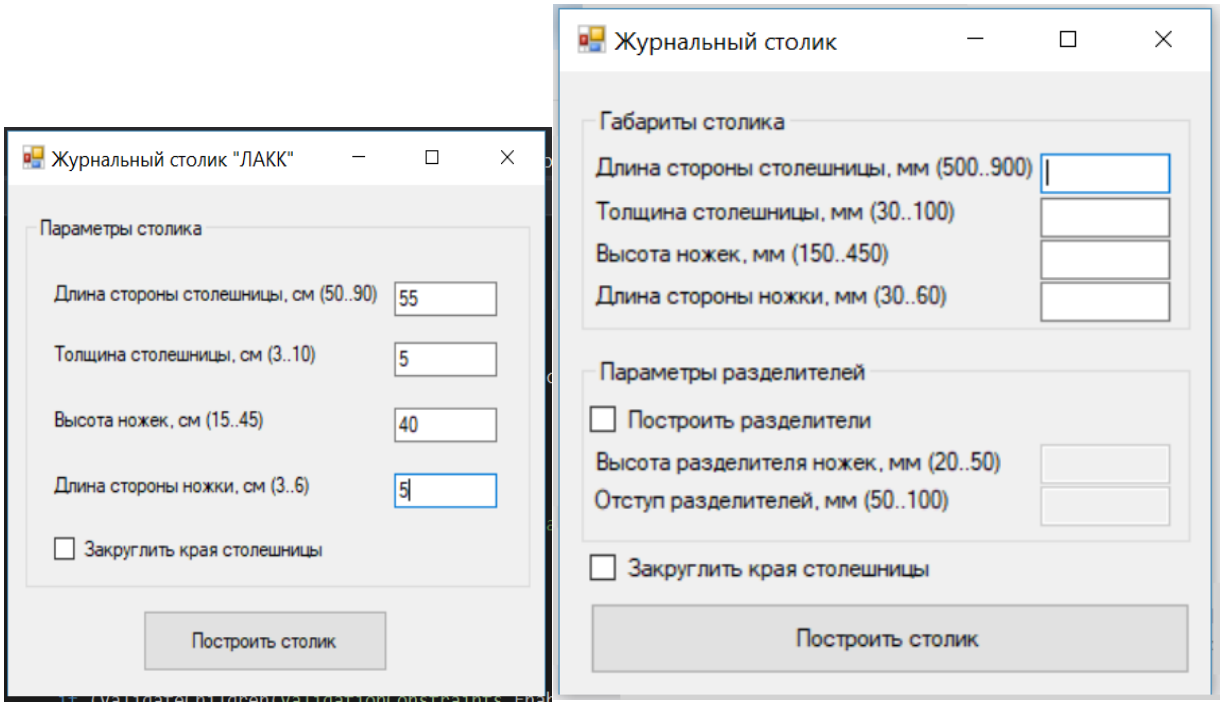

Рисунок 4.5 – Финальная версия пользовательского интерфейса (справа) и макет из ПС (слева) при вводе корректных данных

Из рисунка видно, что добавлены поля для ввода новых параметров, которые становятся доступными, если стоит галочка на переключателе «Построить разделители», изменена единицы измерения ввода (с сантиметров на миллиметры).

На рисунке 4.6 приведен старый макет программы при вводе некорректных данных.

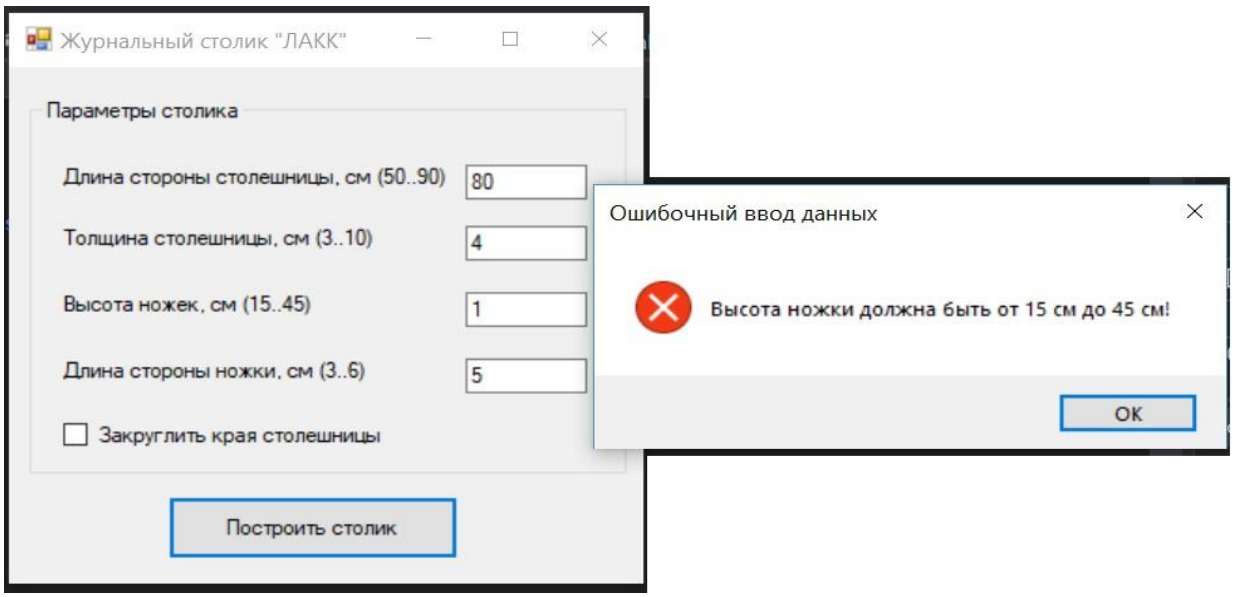

Рисунок 4.6 – Макет интерфейса программы при вводе некорректных данных

На рисунке 4.7 приведен финальный интерфейс программы при вводе некорректных данных.

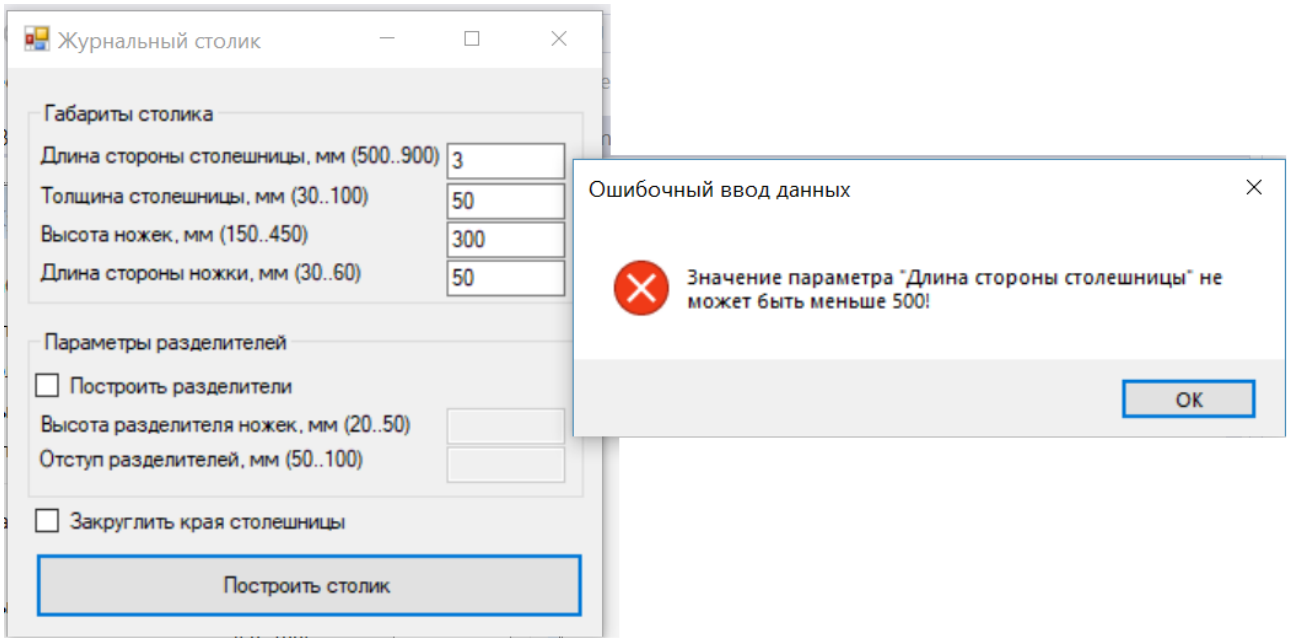

Рисунок 4.7 – Финальный интерфейс программы при вводе некорректных данных

Как видно из рисунков изменений в пользовательском интерфейсе при вводе некорректных данных нет. После нажатия на кнопку «Построить столик» выводится окно с сообщением ошибки.

5 Описание программы для конечного пользователя

После запуска плагина на экране появится окно, изображенное на рисунке 5.1.

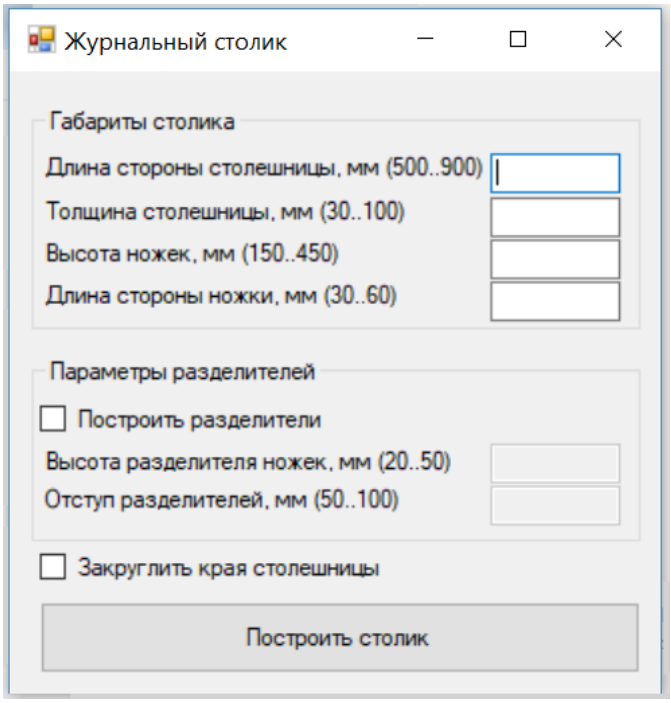

Рисунок 5.1 – Плагин сразу после запуска

Из рисунка видно, что изначально для редактирования доступны только четыре поля. Еще два поля будут доступны при нажатии на переключатель «Построить разделители».

Корректными значениями для параметров являются целые числа из диапазона, указанного в скобках справа от поля ввода. Если при нажатии на кнопку «Построить столик» в поле было введено некорректное значение или оно пусто, ты появляется окно с текстом ошибки, и модель не будет строиться (рисунок 5.2).

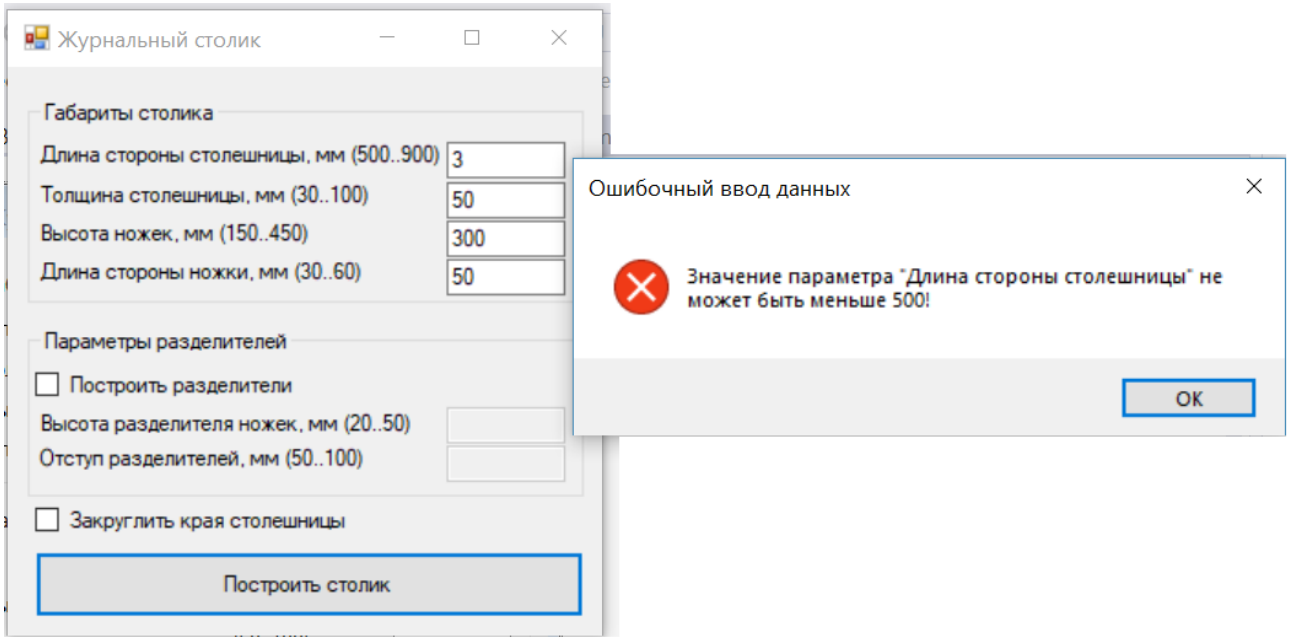

Рисунок 5.2 – Ввод некорректных данных

Если все поля заполнены корректно, то нажатие на кнопку «Построить столик» запустит САПР «КОМПАС 3D» и создаст модель столика с заданными параметрами (рисунок 5.3).

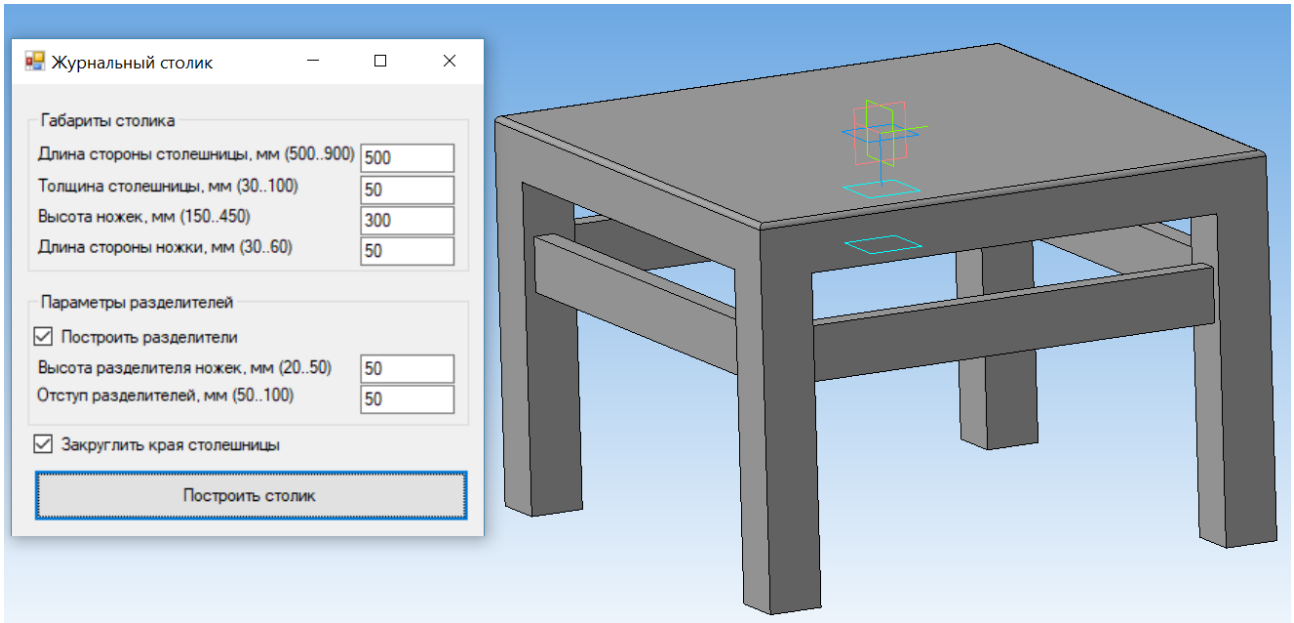

Рисунок 5.3 – Результат работы плагина

6 Тестирование программы

Целью тестирования является проверка соответствия реального поведения программы с ожидаемым, описанным в документации. В данном разделе описаны два вида тестирования, проведённые над разработанным плагином – функциональное и модульное. Все тесты выполнялись на переносном персональном компьютере (ноутбуке) со следующими характеристиками:

- центральный процессор Intel Core i7-4700MQ;
- графический процессор Nvidia GeForce GT 750M;
- 8 ГБ оперативной памяти;
- операционная система Microsoft Windows 10, 64-битная версия.

6.1 Модульное тестирование

Модульное тестирование — процесс изолированного тестирования отдельных модулей программы для подтверждения корректности их работы [4].

В данной работе тестируются публичные поля и методы классов плагина, ответственных за проверку вводимых пользователем данных на корректность и их хранение — Parameter и TableSettings. Для них с использованием библиотеки NUnit были написаны тесты: как позитивные, подтверждающие правильность работы на корректных данных, так и негативные, подтверждающие ожидаемое поведение классов [4] (вызов исключения определённого типа) при попытке использования некорректных данных.

Общее количество написанных тестовых случаев — 58, большинство тестов включают в себя несколько наборов проверочных данных для тестирования различных граничных значений для конструктора классов. В приложении А приведено описание всех тестовых случаев.

6.2 Функциональное тестирование

Функциональное тестирование предназначено для проверки реализуемости в программе функциональных требований к ней, заданных ТЗ [4].

Как уже было отмечено в п. 5, в программе присутствует проверка вводимых пользователем данных на корректность, при вводе неправильных значений появляется окно с сообщением об ошибке, что изображено на рисунке 5.2.

Так же работа плагина была проверена на минимальных, средних и максимальных возможных корректных входных данных. На рисунке 6.2 приведён результат запуска построения модели столика с минимально возможными параметрами.

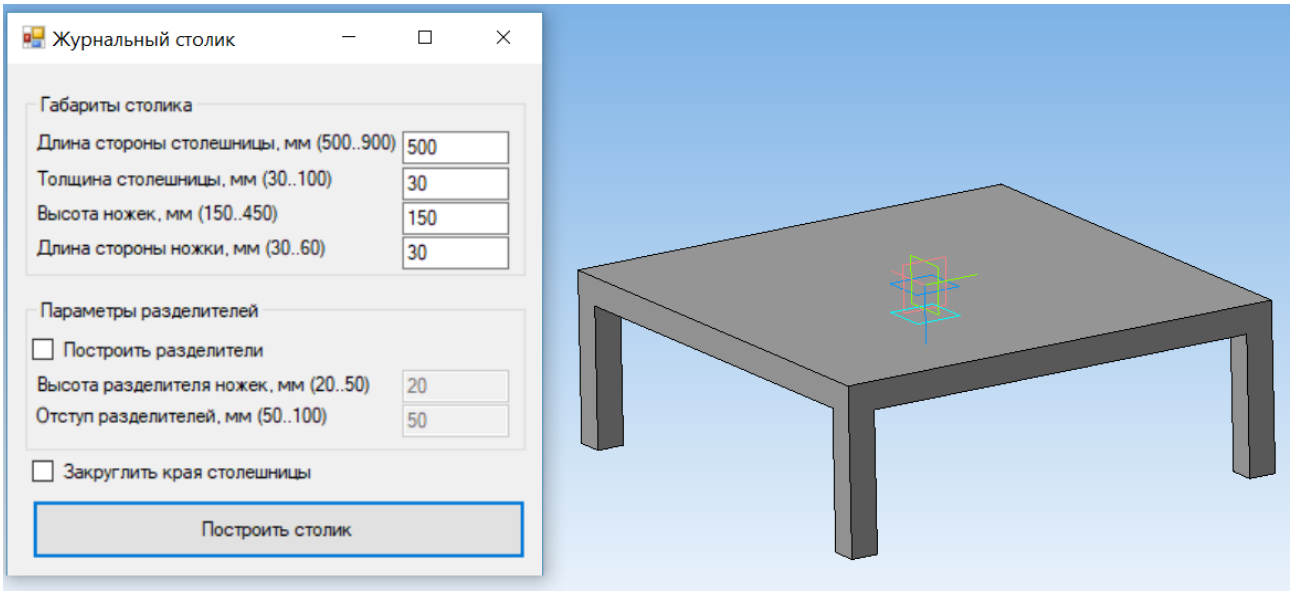

Рисунок 6.2 – Модель столика с наименьшими параметрами

На рисунке 6.3 приведён результат запуска построения модели столика со средними значениями параметров.

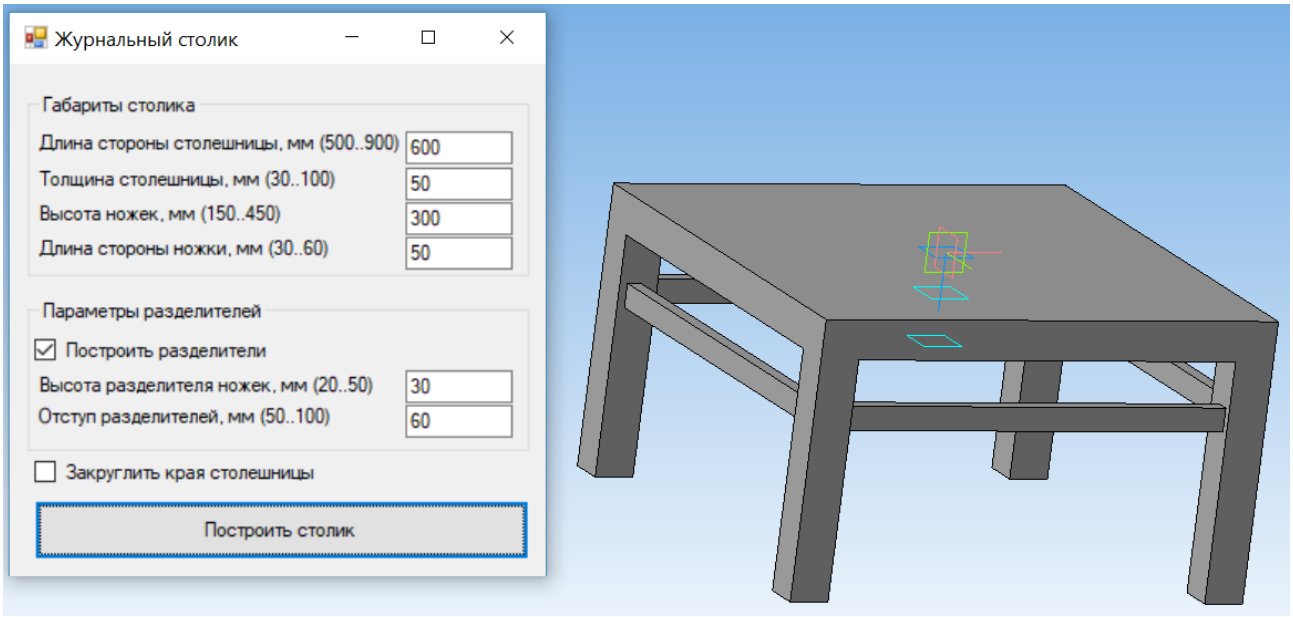

Рисунок 6.3 – Модель со средними параметрами

На рисунке 6.4 приведён результат запуска построения модели столика с максимально возможными параметрами.

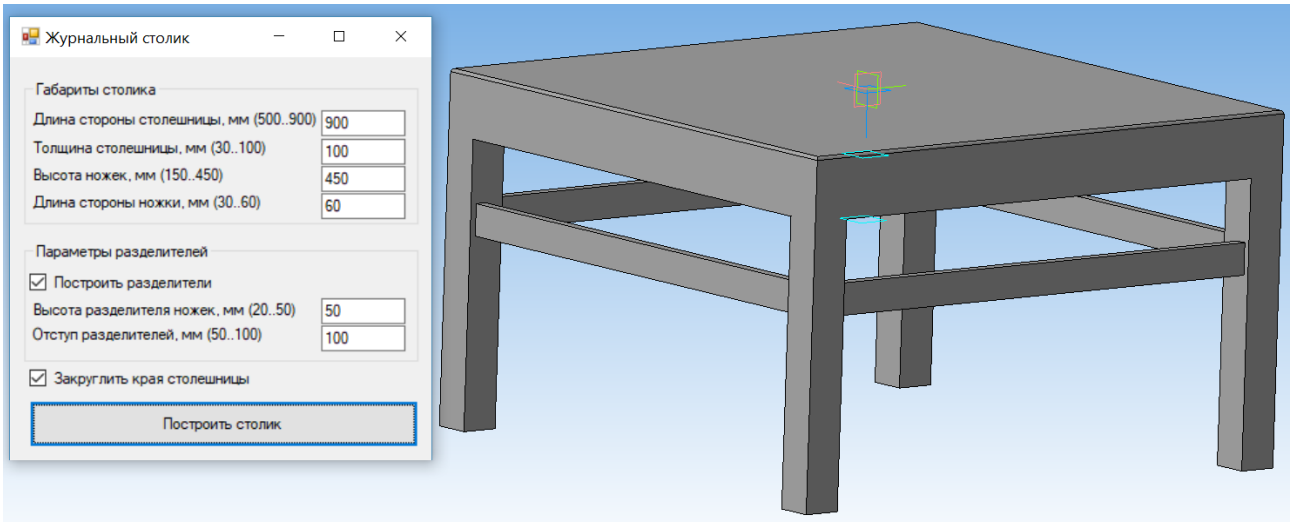

Рисунок 6.4 – Модель с максимальными параметрами

Из рисунков 6.2, 6.3 и 6.4 видно, что плагин функционирует корректно и при минимальных, и при максимальных граничных значениях параметров.

6.3 Нагрузочное тестирование

Цель нагрузочного тестирования – проверка работоспособности программы под большой вычислительной нагрузкой. В данной работе для

тестирования плагин создал в одном экземпляре САПР «KOMPAS 3D» v 15.2 сто моделей журнального столика.

Для осуществления такого тестирования на языке Python был написан скрипт, симулирующий сто раз нажатие на кнопку построения модели каждую секунду. Для замера используемых плагином ресурсов был использована стандартная утилита операционной системы Windows 10 «Системный монитор» [5], где была создана группа сборщиков данных, настроенная на запись системных ресурсов, используемых процессом «KOMPAS.exe».

Результат сборки данных по использованию процессорного времени приведён на рисунке 6.5. Из рисунка видно, что при построении модели столика процессору требуется около десяти процентов ресурсов. При тестировании использовался четырёхъядерный процессор с технологией Hyper-Threading [6], благодаря которому четыре ядра способны одновременно обрабатывать до восьми потоков данных, что позволяет лучше использовать ресурсы процессора.

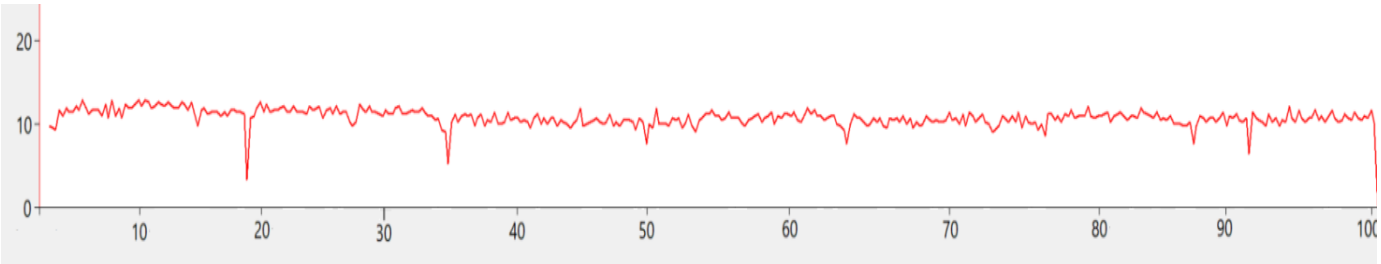

Рисунок 6.5 – График загруженности процессора (по оси X – количество построений, по оси Y – процент загруженности процессора)

На рисунке 6.6 приведена зависимость процента потребляемой процессом САПР оперативной памяти от количества построенных в ней моделей.

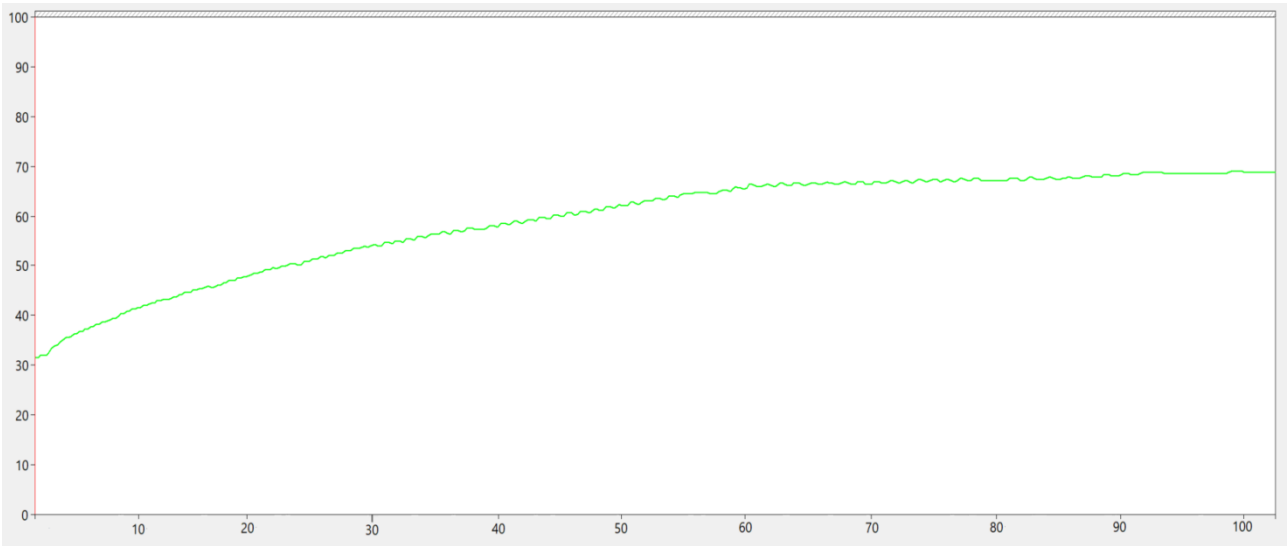

Рисунок 6.6 – График использования памяти (по оси X – количество построений, по оси Y – процент расхода оперативной памяти)

Из рисунка 6.6 можно сделать вывод, что потребление оперативной памяти САПР с ростом числа построений возрастает относительно линейно, что свидетельствует об отсутствии ошибок при работе с памятью.

На рисунке 6.7 приведена зависимость времени построения модели от номера строящейся модели.

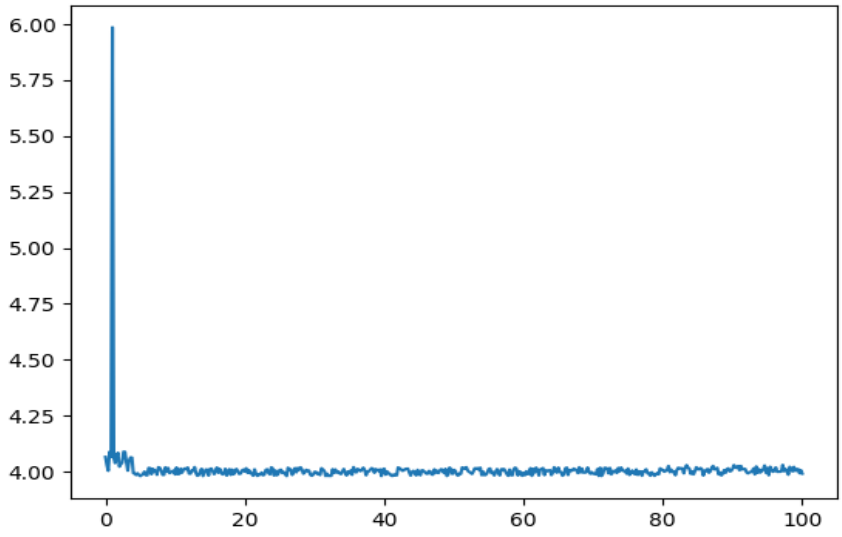

Рисунок 6.7 – Зависимость времени построения от номера модели (по оси X – номер модели, по оси Y – время построения в секундах)

Как видно из рисунка 6.7 для построения первой модели требуется значительно больше времени по сравнению с остальными номерам, это связано

с тем, что при первом запуске САПР долго подгружается необходимые ресурсы для построения.

После проведения нагрузочного тестирования не было обнаружено проблем у плагина при работе с оперативной памятью и ресурсами процессора.

7 Заключение

В ходе выполнения данного проекта был сделан плагин для САПР «КОМПАС 3D» v 15.3, который автоматизирует построение модели журнального столика по набору параметров.

Были изучены основные этапы проектирования программного продукта: написание ТЗ, составление ПС. Во время выполнения проекта были получены навыки проектирования архитектуры программы, навыки работы с UML-диаграммами, навыки написания и использования модульных тестов, использования системы контроля версий. Также были приобретены знания по построению таких программных архитектур, которые позволяют адаптироваться к меняющимся во время процесса разработки требованиям заказчиков. Были изучены принципы использования API у крупных программных комплексов.

Разработанный плагин был протестирован модульными и функциональными тестами, плагин успешно прошел тесты, что доказывает работоспособность плагина и соответствие функциональности плагина требования, заявленным в ТЗ.

Список использованных источников

1. Application programming interface – Wikipedia. [Электронный ресурс]. URL: https://en.wikipedia.org/wiki/Application_programming_interface (дата обращения 11.01.18)

2. Плагин – Википедия. [Электронный ресурс]. URL: https://ru.wikipedia.org/wiki/Плагин (дата обращения 21.12.17)

3. Разработка плагина «Построение журнального столика» для САПР «КОМПАС 3D» v 15.2. Проект системы. [Электронный ресурс]. URL: https://github.com/gelcen/coffee_table/blob/master/PS.pdf (дата обращения 21.04.18)

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

5. How To: Use Perfmon in Windows 7 – Security Tolls. [Электронный ресурс]. URL: https://blogs.msdn.microsoft.com/securitytools/2009/11/03/how-to-use-perfmonin-windows-7/ (Дата обращения 29.04.18)

6. NotebookCheck. Intel Core i7-4700MQ. [Электронный ресурс]. URL: https://www.notebookcheck-ru.com/Intel-Core-i7-4700MQ-Processor.103149.0.html (Дата обращения 04.05.18)

Приложение А
(справочное)

Описание тестовых случаев

В таблице А.1 приведено описание тестовых случаев для модульного тестирования классов TableSettings и Parameter.

Таблица А.1 – Модульные тесты

Входные данные	Описание тестового случая
TableSettingsConstructPosTest – позитивный тест конструктора класса TableSettings. Входные данные: uint tabletopLength – длина стороны столешницы; uint tabletopThickness – толщина столешницы; uint legHeight – высота ножек; uint legLength – длина стороны основания ножек; bool withSeptum – построение с разделителями; uint septumLength – высота разделителей; uint septumOffset – отступ разделителей от столешницы.	
tabletopLength = 600, tabletopThickness = 50, legHeigth = 300, legLength = 50, withSeptum = true, septumLength = 30, septumOffset = 60	Проверка на различных значениях
tabletopLength = 900, tabletopThickness = 100, legHeigth = 450, legLength = 60, withSeptum = true, septumLength = 50, septumOffset = 100	Проверка на максимальные значения
tabletopLength = 500, tabletopThickness 30, legHeigth = 150, legLength = 30, withSeptum = false, septumLength = 30, septumOffset = 60	Проверка без построения разделителей
TableSettingsConstructNegTest – негативный тест конструктора класса. Входные данные: uint tabletopLength – длина стороны столешницы; uint tabletopThickness – толщина столешницы; uint legHeight – высота ножек; uint legLength – длина стороны основания ножек; bool withSeptum – построение с разделителями; uint septumLength – высота разделителей; uint septumOffset – отступ разделителей от столешницы.	

Продолжение таблицы А.1

tabletopLength = 6000, tabletopThickness = 50, legHeigth = 300, legLength = 50, withSeptum = true, septumLength = 30, septumOffset = 60	Проверка TabletopLegnth на слишком большое значения
tabletopLength = 300, tabletopThickness = 50, legHeigth = 300, legLength = 50, withSeptum = true, septumLength = 30, septumOffset = 60	Проверка TabletopLegnth на значения меньше допустимого
tabletopLength = 600.1, tabletopThickness = 50, legHeigth = 300, legLength = 50, withSeptum = true, septumLength = 30, septumOffset = 60	Проверка TabletopLegnth на вещественное значения
tabletopLength = 0, tabletopThickness = 50, legHeigth = 300, legLength = 50, withSeptum = true, septumLength = 30, septumOffset = 60	Проверка TabletopLegnth на слишком малое значения
tabletopLength = -1, tabletopThickness = 50, legHeigth = 300, legLength = 50, withSeptum = true, septumLength = 30, septumOffset = 60	Проверка TabletopLegnth на отрицательное значения
tabletopLength = 600, tabletopThickness = 500, legHeigth = 300, legLength = 50, withSeptum = true, septumLength = 30, septumOffset = 60	Проверка TabletopThickness на слишком большое значения
tabletopLength = 600, tabletopThickness = 0, legHeigth = 300, legLength = 50, withSeptum = true, septumLength = 30, septumOffset = 60	Проверка TabletopThickness равном нулю
tabletopLength = 600, tabletopThickness = -500, legHeigth = 300, legLength = 50, withSeptum = true, septumLength = 30, septumOffset = 60	Проверка TabletopThickness на отрицательное значения
tabletopLength = 600, tabletopThickness = 1, legHeigth = 300, legLength = 50, withSeptum = true, septumLength = 30, septumOffset = 60	Проверка TabletopThickness на слишком малое значения
tabletopLength = 600, tabletopThickness = 50.2, legHeigth = 300, legLength = 50, withSeptum = true, septumLength = 30, septumOffset = 60	Проверка TabletopThickness на вещественное значения

Продолжение таблицы А.1

tabletopLength = 600, tabletopThickness = 50, legHeigth = 3000, legLength = 50, withSeptum = true, septumLength = 30, septumOffset = 60	Проверка LegHeight на слишком большое значения
tabletopLength = 600, tabletopThickness = 50, legHeigth = 3, legLength = 50, withSeptum = true, septumLength = 30, septumOffset = 60	Проверка LegHeight на слишком малое значения
tabletopLength = 600, tabletopThickness = 50, legHeigth = 0, legLength = 50, withSeptum = true, septumLength = 30, septumOffset = 60	Проверка LegHeight на ноль
tabletopLength = 600, tabletopThickness = 50, legHeigth = -30, legLength = 50, withSeptum = true, septumLength = 30, septumOffset = 60	Проверка LegHeight на отрицательное значения
tabletopLength = 600, tabletopThickness = 50, legHeigth = 303.3, legLength = 50, withSeptum = true, septumLength = 30, septumOffset = 60	Проверка LegHeight на вещественное значения
tabletopLength = 600, tabletopThickness = 50, legHeigth = 300, legLength = 500, withSeptum = true, septumLength = 30, septumOffset = 60	Проверка LegLength на слишком большое значения
tabletopLength = 600, tabletopThickness = 50, legHeigth = 300, legLength = 0, withSeptum = true, septumLength = 30, septumOffset = 60	Проверка LegLength на ноль
tabletopLength = 600, tabletopThickness = 50, legHeigth = 300, legLength = 1, withSeptum = true, septumLength = 30, septumOffset = 60	Проверка LegLength на слишком малое значения
tabletopLength = 600, tabletopThickness = 50, legHeigth = 300, legLength = -50, withSeptum = true, septumLength = 30, septumOffset = 60	Проверка LegLength на отрицательное значения
tabletopLength = 600, tabletopThickness = 50, legHeigth = 300, legLength = 55.5, withSeptum = true, septumLength = 30, septumOffset = 60	Проверка LegLength на слишком вещественное значения

Прододжение таблицы А.1

tabletopLength = 600, tabletopThickness = 50, legHeigth = 300, legLength = 50, withSeptum = true, septumLength = 3000, septumOffset = 60	Проверка septumLength на слишком большое значения
tabletopLength = 600, tabletopThickness = 50, legHeigth = 300, legLength = 50, withSeptum = true, septumLength = 0, septumOffset = 60	Проверка septumLength на ноль
tabletopLength = 600, tabletopThickness = 50, legHeigth = 300, legLength = 50, withSeptum = true, septumLength = 1, septumOffset = 60	Проверка septumLength на слишком малое значения
tabletopLength = 600, tabletopThickness = 50, legHeigth = 300, legLength = 50, withSeptum = true, septumLength = -30, septumOffset = 60	Проверка septumLength на слишком отрицательное значения
tabletopLength = 600, tabletopThickness = 50, legHeigth = 300, legLength = 50, withSeptum = true, septumLength = 30.2, septumOffset = 60	Проверка septumLength на вещественное значения
tabletopLength = 800, tabletopThickness = 50, legHeigth = 200, legLength = 50, withSeptum = true, septumLength = 30, septumOffset = 60	legHeight не может принимать значение меньше 300, если tabletopLength больше 700
tabletopLength = 600, tabletopThickness = 50, legHeigth = 300, legLength = 50, withSeptum = true, septumLength = 30, septumOffset = 6000	Проверка septumOffset на слишком большое значения
tabletopLength = 600, tabletopThickness = 50, legHeigth = 300, legLength = 50, withSeptum = true, septumLength = 30, septumOffset = 0	Проверка septumOffset на ноль
tabletopLength = 600, tabletopThickness = 50, legHeigth = 300, legLength = 50, withSeptum = true, septumLength = 30, septumOffset = 1.6	Проверка septumOffset на вещественное значения
tabletopLength = 600, tabletopThickness = 50, legHeigth = 300, legLength = 50, withSeptum = true, septumLength = 30, septumOffset = 1	Проверка septumOffset на слишком малое значения

Продолжение таблицы А.1

tabletopLength = 600, tabletopThickness = 50, legHeigth = 300, legLength = 50, withSeptum = true, septumLength = 30, septumOffset = -6000	Проверка septumOffset на отрицательное значения
tabletopLength = -600, tabletopThickness = -50, legHeigth = -300, legLength = -50, withSeptum = true, septumLength = -30, septumOffset = -6000	Проверка всех параметров на отрицательное значения
tabletopLength = 6000, tabletopThickness = 5000, legHeigth = 3000, legLength = 5000, withSeptum = true, septumLength = 3000, septumOffset = 6000	Проверка всех параметров на слишком больше значения
tabletopLength = 600.8, tabletopThickness = 50.8, legHeigth = 300.8, legLength = 50.8, withSeptum = true, septumLength = 30.8, septumOffset = 60.8	Проверка всех параметров на вещественное значения
tabletopLength = 0, tabletopThickness = 0, legHeigth = 0, legLength = 0, withSeptum = true, septumLength = 0, septumOffset = 0	Проверка всех параметров на ноль
tabletopLength = 1, tabletopThickness = 1, legHeigth = 1, legLength = 1, withSeptum = true, septumLength = 1, septumOffset = 1	Проверка всех параметров на слишком малое значения
ParameterConstructorPositiveTest – позитивный тест конструктора класса Parameter. Входные данные: uint min – минимальное значение параметра; uint max – максимальное значение параметра; uint value – значение параметра; string name – имя параметра.	
min = 50, max = 100, value = 60, name=”normal”	Проверка на различных значениях
min = 2, max = 999, value = 60, name=”normal”	Проверка на предельных значениях
ParameterConstructorNegativeTest – негативный тест конструктора класса Parameter. Входные данные: uint min – минимальное значение параметра; uint max – максимальное значение параметра; uint value – значение параметра; string name – имя параметра.	
min = 0, max = 10000, value = 60, name=”normal”	Проверка min и max при значениях вне диапазона

Окончание таблицы А.1

min = -50, max = -100, value = 60, name=”normal”	Проверка min и max на отрицательные значения
min = 6.6, max = 106.6, value = 60, name=”normal”	Проверка min и max на вещественные значения
min = 500, max = 100, value = 60, name=”normal”	Проверка при min больше max
min = 0, max = 100, value = 60, name=”normal”	Проверка min на значение вне диапазона
min = 50, max = 1100, value = 60, name=”normal”	Проверка max на значения вне диапазона
min = 50, max = 100, value = 40, name=”normal”	Проверка value на значения вне диапазона
min = 50, max = 100, value = 140, name=”normal”	Проверка value на значения вне диапазона
min = 50, max = 100, value = -60, name=”normal”	Проверка value на отрицательное значения
min = 50, max = 100, value = 60.6, name=”normal”	Проверка value на вещественное значения
min = 50, max = 100, value = 60, name=””	Проверка name на пустую строку
min = 50, max = 100, value = 40, name = null	Проверка name на значения null
min = 50, max = 100, value = 40, name=5	Проверка name на численное значения