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.

ЗАДАНИЕ

по дисциплине «Основы Разработки САПР»

Выдано: студенту группы 584-2 Бабынцевой Виктории Александровны

1. Тема проекта: Разработка плагина «Компьютерная мышь» для «КОМПАС-3D V17».

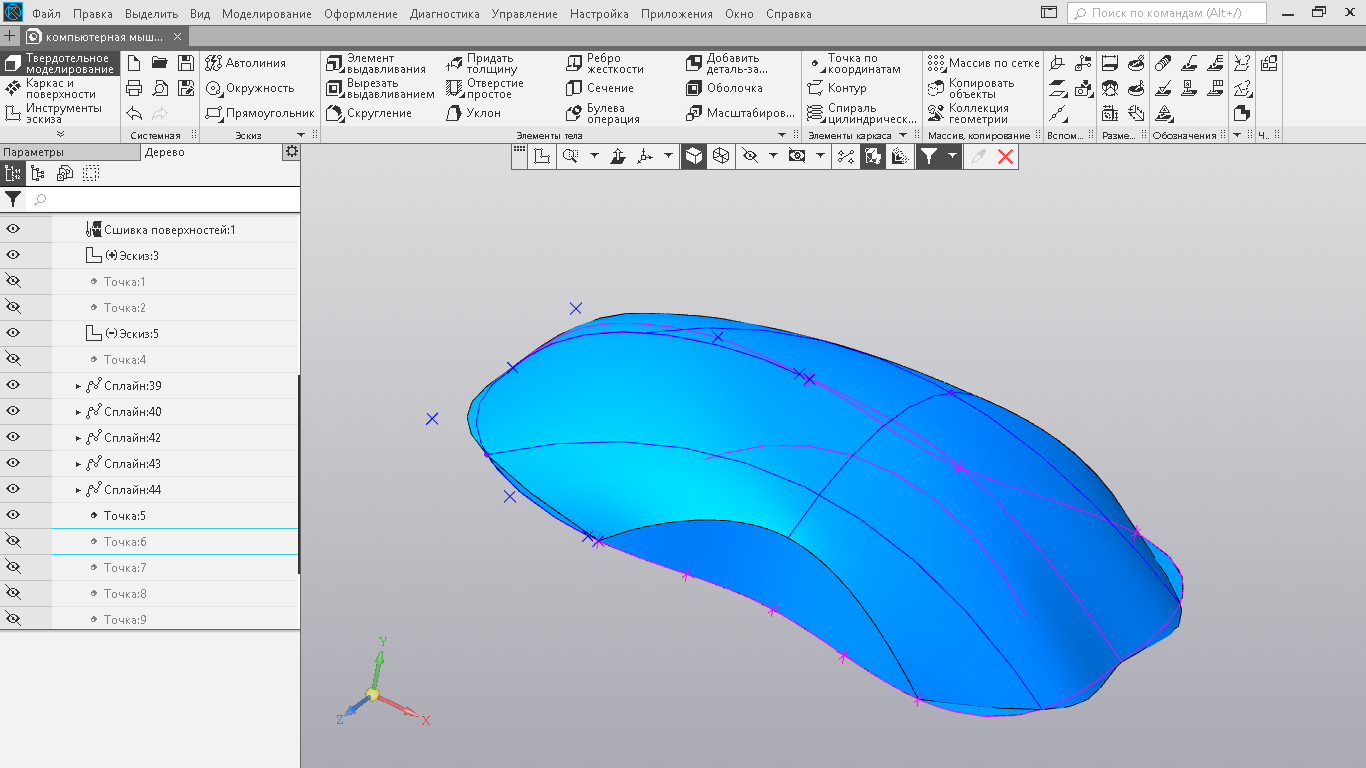


Рисунок 1.1 – Чертеж компьютерной мыши

1. Срок сдачи студентом проекта: 29.09.2017
2. Исходные данные

Разработать плагин «Компьютерная мышь» для «КОМПАС-3D V17»

1. Требования к плагину

Плагин должен обеспечивать следующую функциональность:

* выводить диалоговое окно ввода для изменения следующих параметров:
* Длина компьютерной мыши представлена на рисунке 1.2;

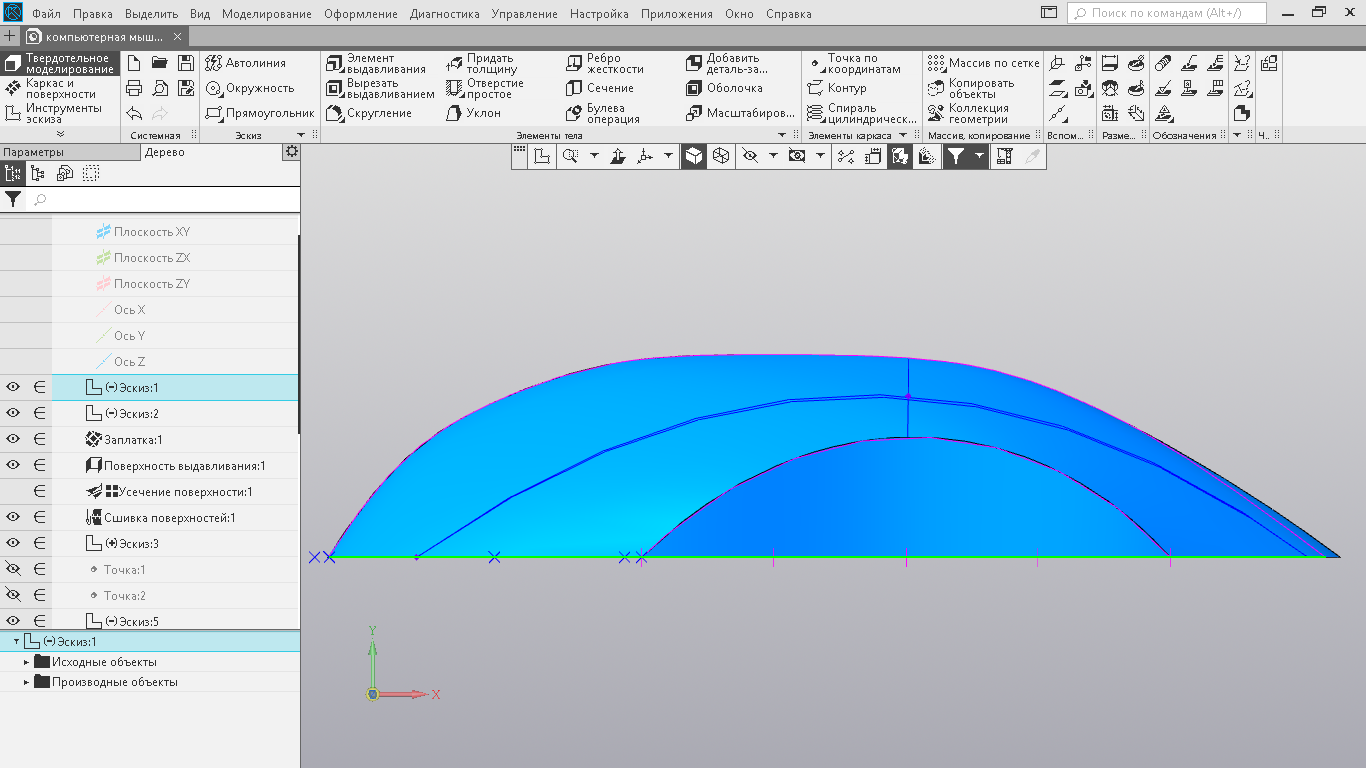


Рисунок 1.2 – Чертеж с обозначением длины компьютерной мыши

* Радиус боковой выемки (не может быть больше чем радиус бокового каркаса) представлен на рисунке 1.3;

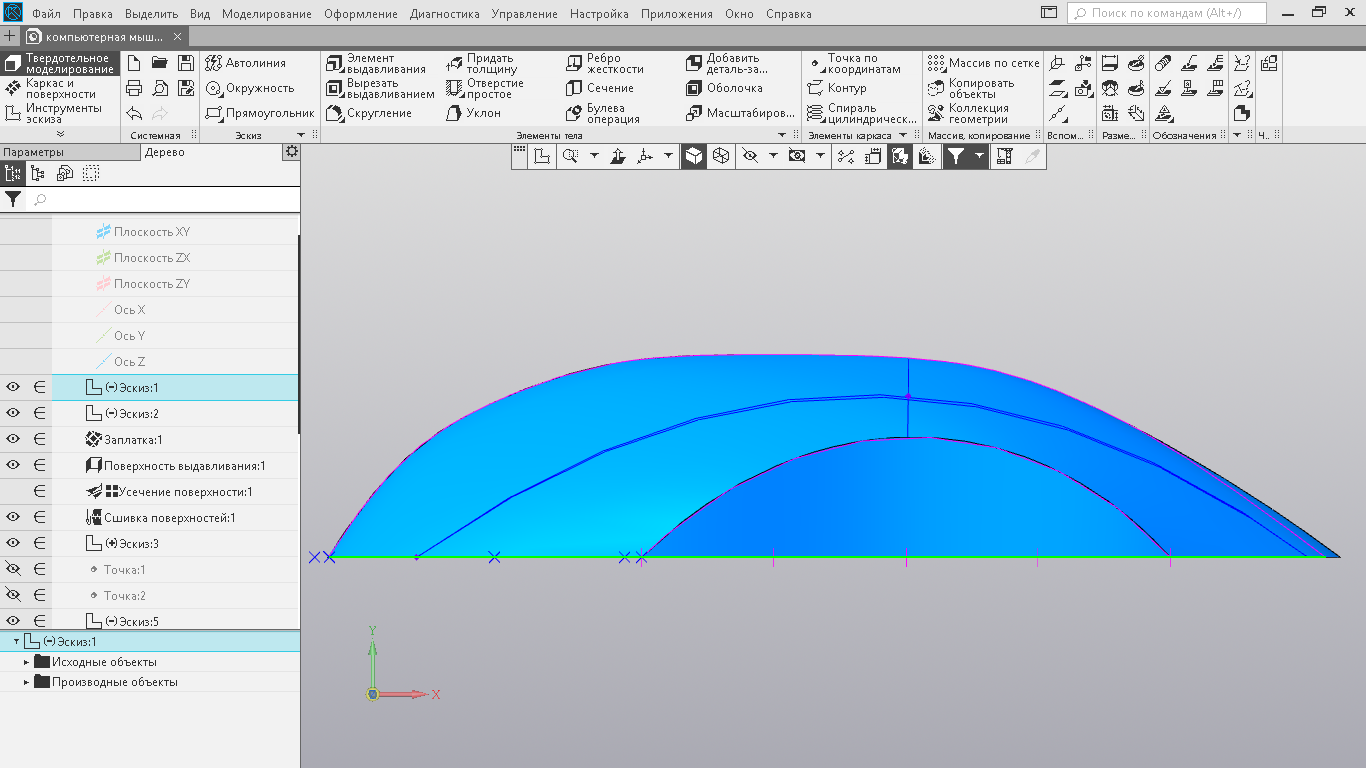


Рисунок 1.3 – Чертеж с обозначением радиуса боковой выемки

* Радиус бокового каркаса (не может быть меньше радиуса боковой выемки) представлен на рисунке 1.4;

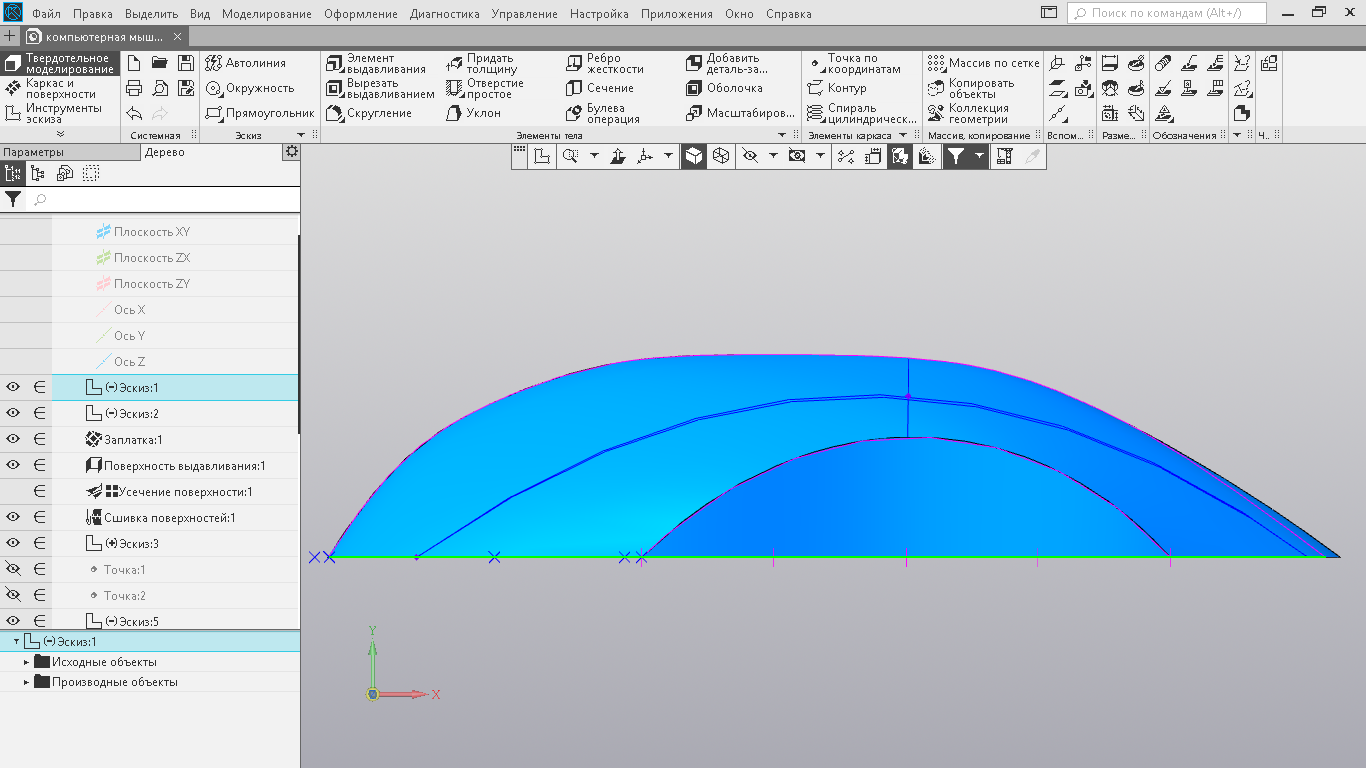


Рисунок 1.4 – Чертеж с обозначением радиуса бокового каркаса

* длина головного часть компьютерной мыши представлена на рисунке 1.5;

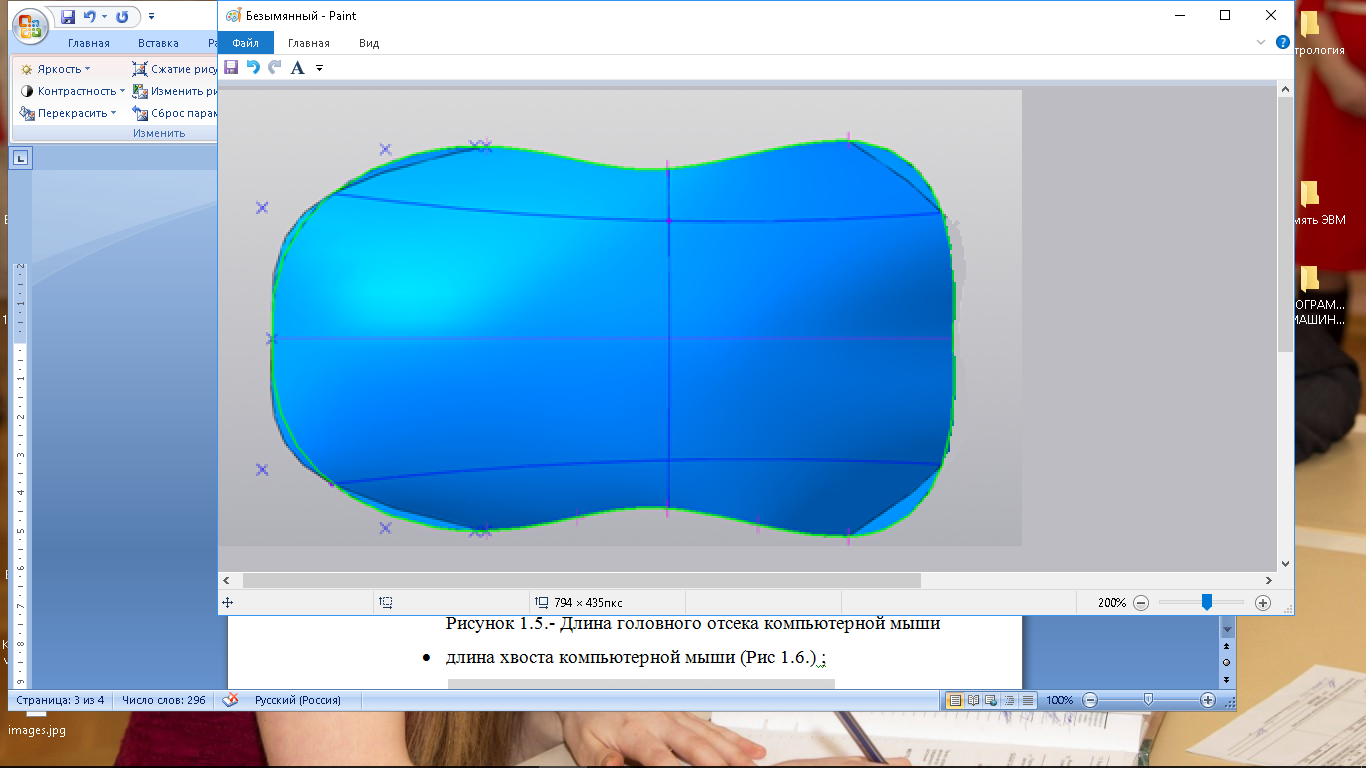


Рисунок 1.5 ­­– Чертеж с обозначением длины головной части компьютерной мыши

* длина задней части компьютерной мыши представлена на рисунке 1.6;

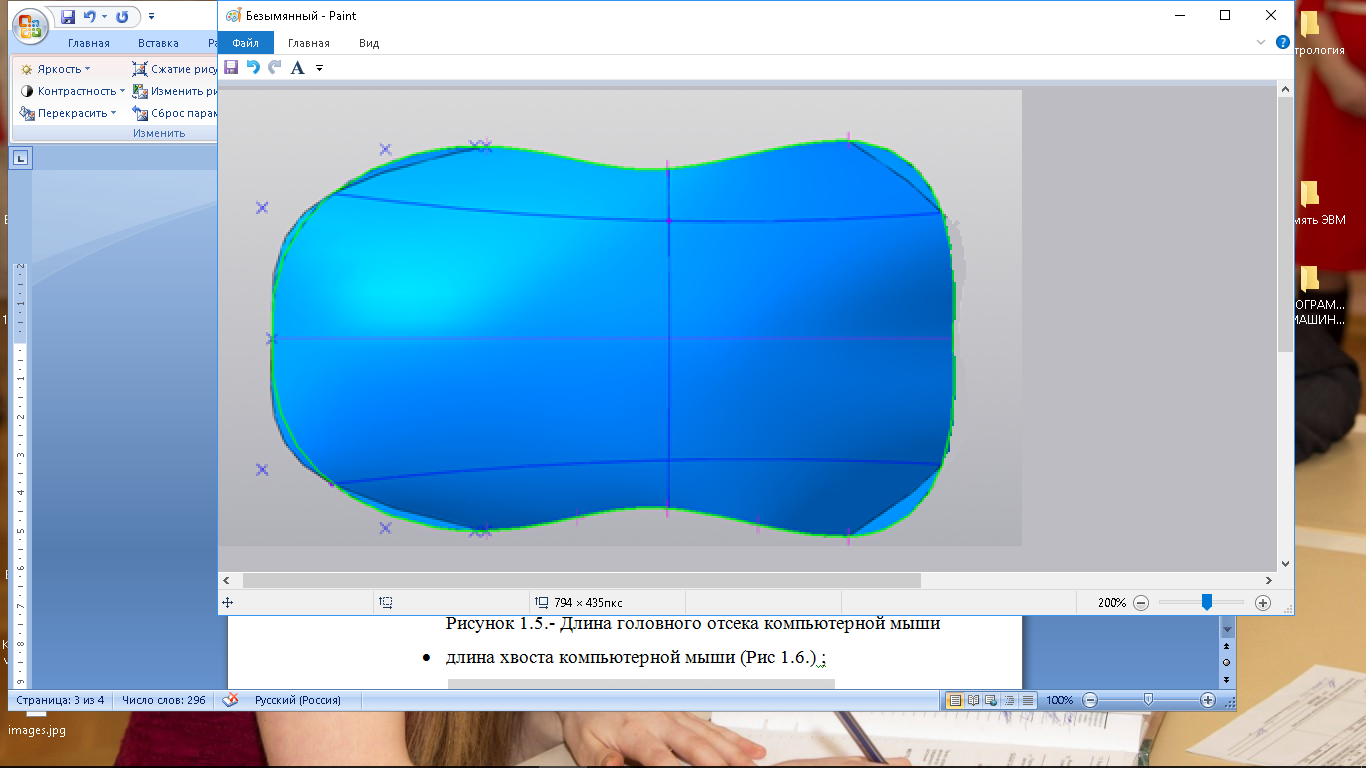


Рисунок 1.6 – Чертеж длины задней части компьютерной мыши

В сумме длина задней части и длина головной части равна длине компьютерной мыши.

* обеспечивать построение трехмерной модели на графическом окне системы «КОМПАС-3D» на основе введенных значений параметров;
* обеспечить проверку корректности ввода данных;
* вывод информационного сообщения в случае ввода некорректных данных.

1. Сфера применения

Плагин применим при проектировании комплектующих для персональных компьютеров в системе «КОМПАС-3D V17».

1. Минимальные требования к программной и аппаратной частям:

* операционная система: Windows 10 (x32/x64);
* процессор: 32-разрядный с тактовой частотой 1.75 ГГц;
* ОЗУ: 2 Гб;
* графические устройства с поддержкой DirectX 9 с драйвером WDDM 1.0;

1. Инструменты разработки:

* язык программирования C#, .NET framework 4.5;
* IDE Microsoft Visual Studio 2015;
* система контроля версий Git;

Руководитель к.т.н. доцент каф. КСУП:  
Калентьев А.А. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Задание принял к исполнению  
Бабынцева В.А. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **UML-диаграммы**

После того как плагин был сделан произошли несоответствия с заявленными UML-диаграммами (Рисунки 1.1,1.2,1.3,1.4).

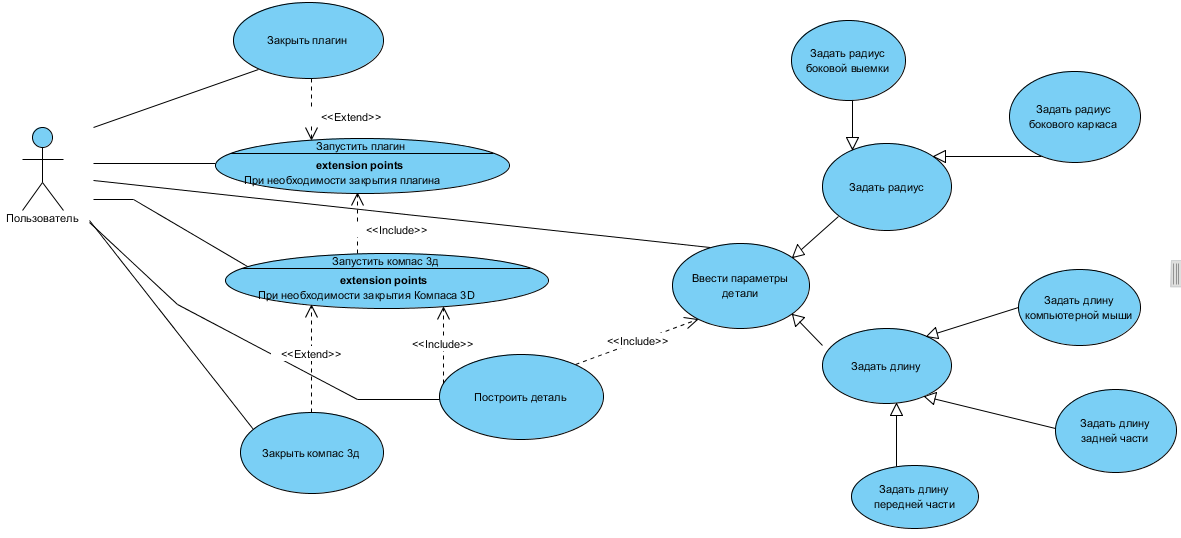


Рисунок 1.1 – Первоначально предоставленная диаграмма вариантов использования

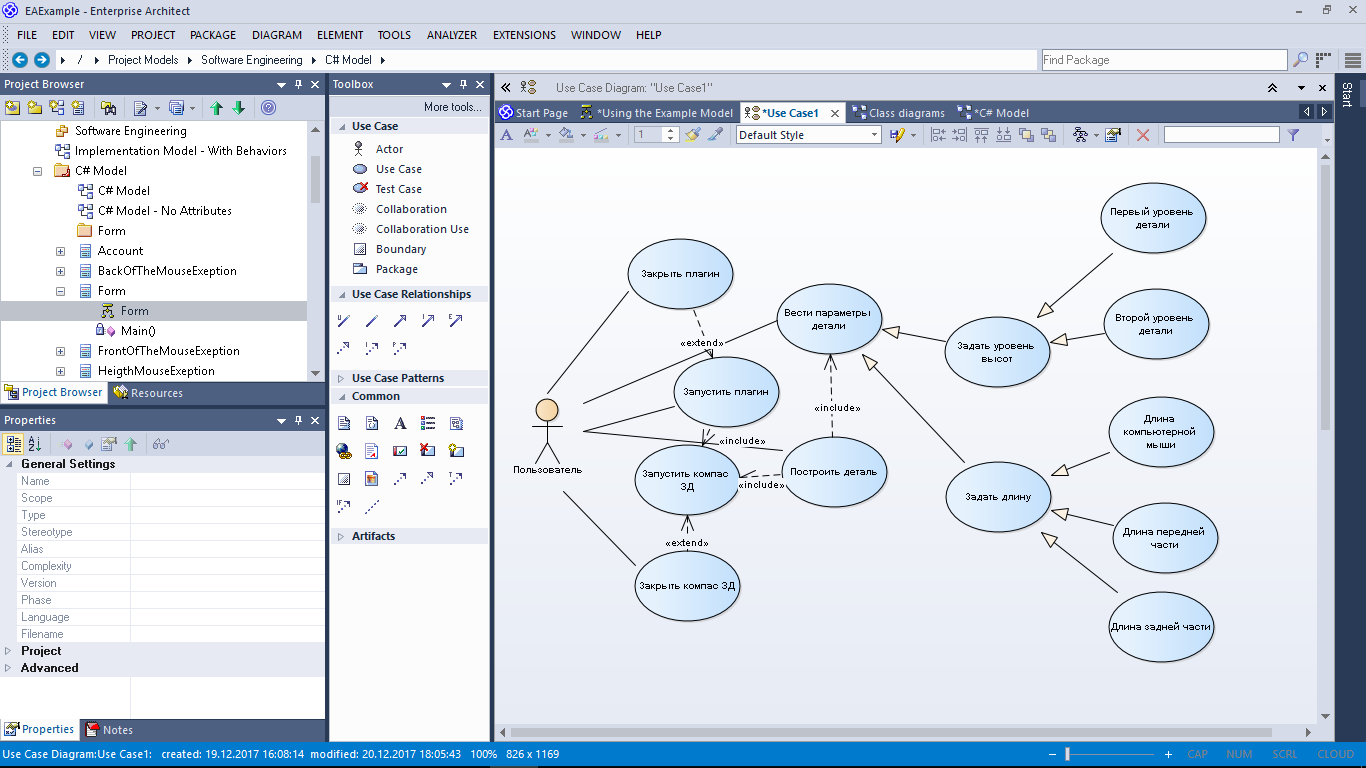


Рисунок 1.2 – Итоговая диаграмма вариантов использования

Как уже заметно диаграммы до и после вариантов использования практически не отличаются. Произошла замена названия параметров и пользователь при помощи нашего плагине не может открыть компас 3Д (так как при нажатии кнопки «Построить деталь» плагин сам открывает Компас).

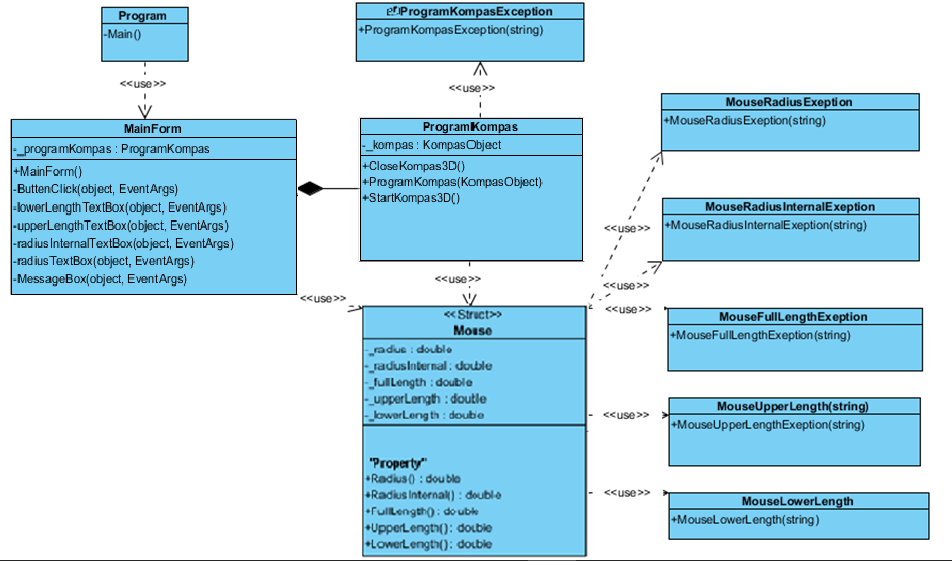


Рисунок 1.3 – Первоначальная диаграмма классов

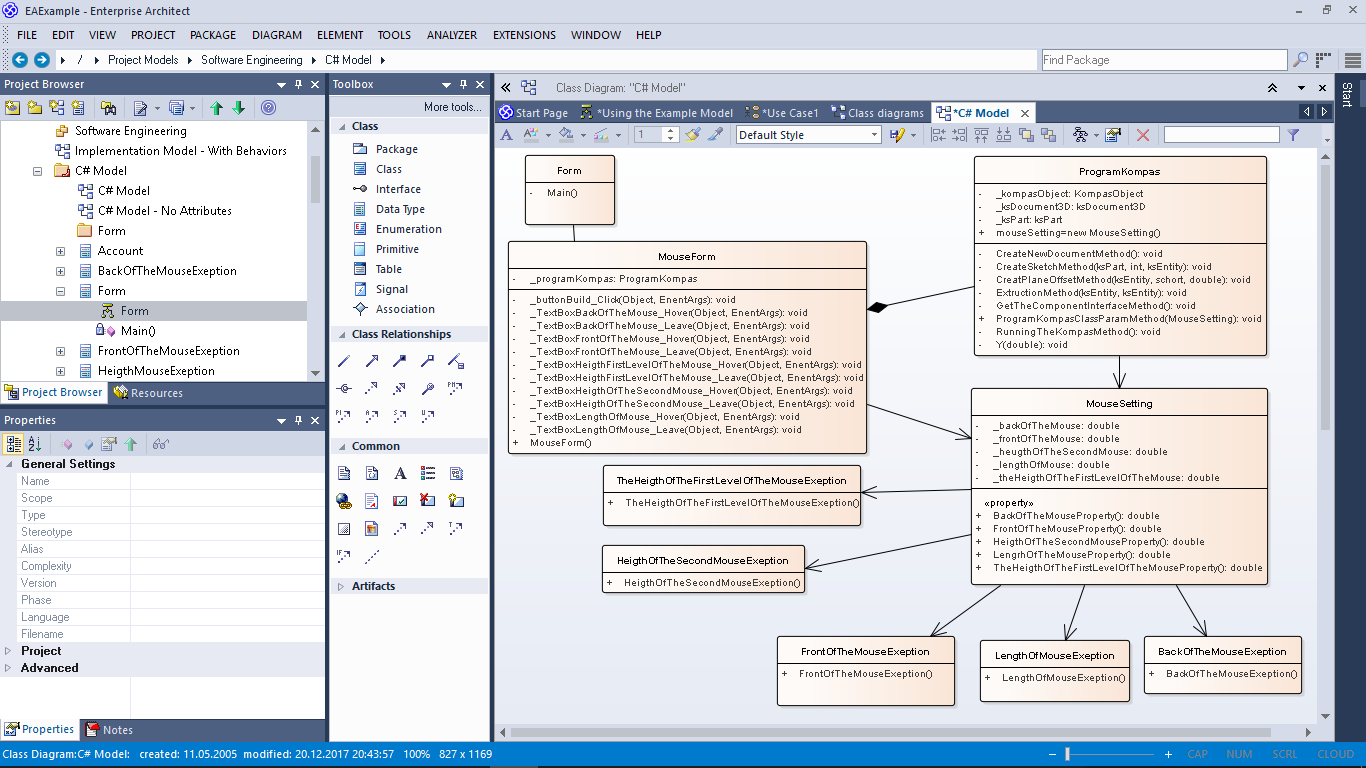


Рисунок 1.4 – Конечная диаграмма классов

Как видно из рисунков 1.3 и 1.4 сама структура не изменилась, но изменились название классов, параметров и более подробно описаны классы.

# Тестирование программы

Тестирование — это проверка соответствия объекта желаемым критериям. Несоответствие объекта желаемым критериям называется ошибкой[1].

Существует много классификаций тестов по разным признакам. Тут же стоит отметить, что данные классификации пересекаются и один тест может одновременно относиться к нескольким классификациям[1].

В данном проекте будет применятся три вида тестирования: функциональное тестирование, модульное тестирование, нагрузочное тестирование.

## Функциональные тесты

Функциональное тестирование — тестирование функциональности объекта, т. е. правильно ли объект выполняет свои функции. Фактически, выполняется проверка правильности выходных данных при соответствующих входных данных[1]. В данном случае будет проверяться правильность построения детали при различных входных параметрах (таблица 2.1):

Таблица 2.1 – Тестовые случаи и результаты работы программы на них

|  |  |  |
| --- | --- | --- |
| **Тестовый случай** | **Значения** | **Результат** |
| Параметр «Длина мыши» меньше наименьшего значения 100 мм или наибольшего значения 150 мм | (99, -, -, -,-;);  (151, -. -. -, -). | Корректно. Программа выводит MessageBox с предупреждением о том, что длина мыши может быть от 100 до 150мм. |

Окончание таблицы 2.1

|  |  |  |
| --- | --- | --- |
| **Тестовый случай** | **Значения** | **Результат** |
| Параметр «Передняя часть мыши» больше 40% или меньше 30% | (100, 41, -,-,-);  (100,29,-,-,-). | Корректно. Программа выводит MessageBox с предупреждением о том, что длина передней части мыши не должна превышать 40% или быть меньше 30% от длины мыши |
| Параметр «Передняя часть мыши» больше 70% или меньше 60% | (100,-,71,-,-);  (100,-,59,-,-). | Корректно. Программа выводит MessageBox с предупреждением о том, что длина задней части мыши не должна превышать 70% или быть меньше 60% от длины мыши |
| Параметр «Высота первого уровня» больше 30% или меньше 20% | (100,39,61,31,-)  (100,39,61,19,-) | Корректно. Программа выводит MessageBox с предупреждением о том, что высота первого уровня не должна превышать 30% или быть меньше 20% от длины мыши |
| Параметр «Высота второго уровня» больше 20% или меньше 10% | (100,39,61,20,21)  (100,39,61,20,9) | Корректно. Программа выводит MessageBox с предупреждением о том, что высота второго уровня не должна превышать 20% или быть меньше 10% от длины мыши |

Результат работы программы после ввода минимальных корректных значений представлен на рисунке 2.1.

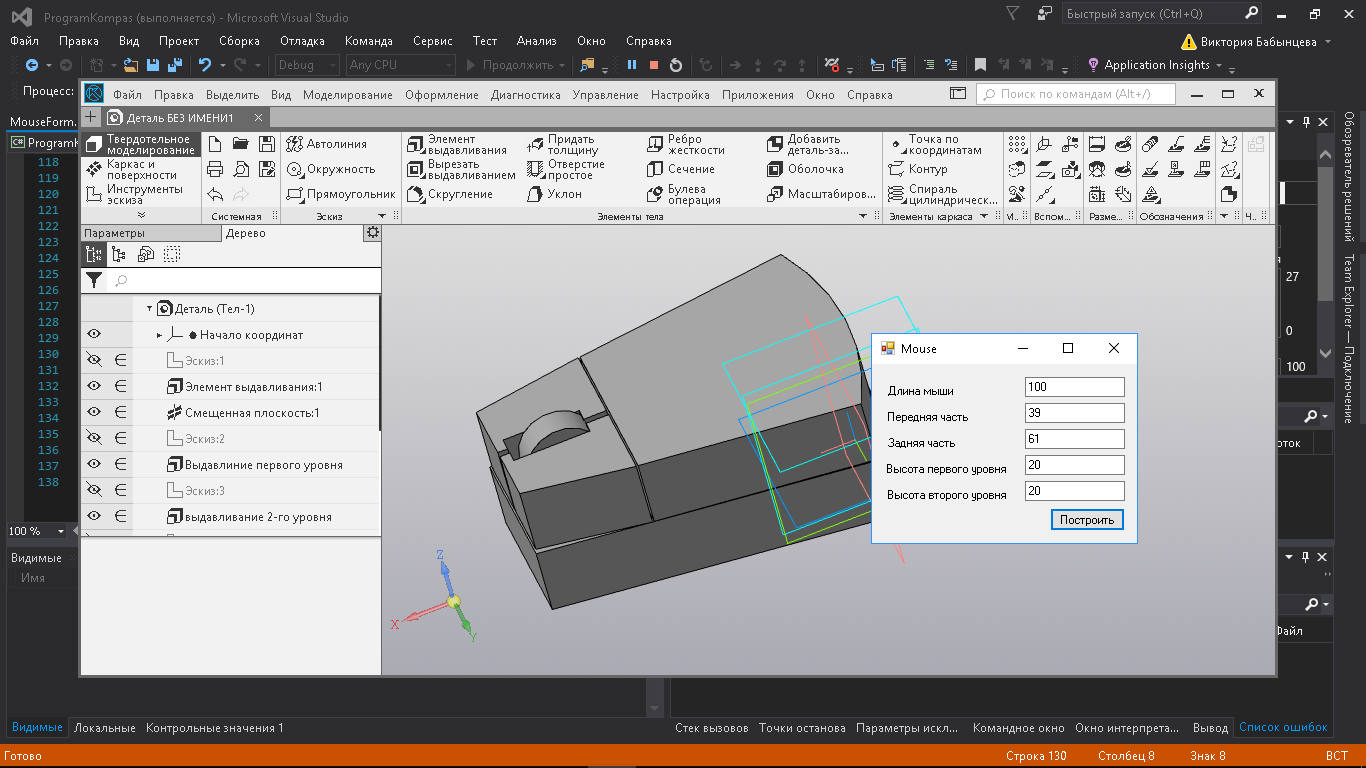
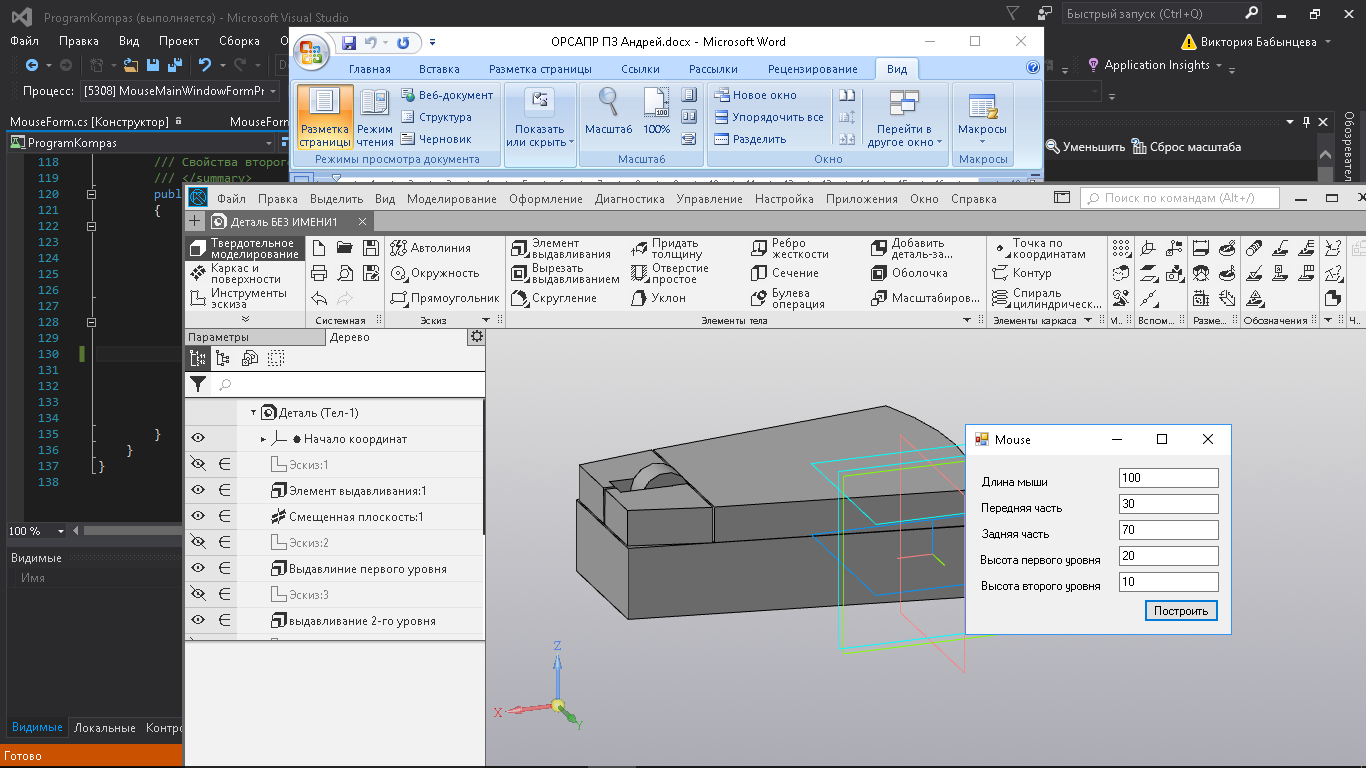


Рисунок 2.1 – Результат работы программы после ввода корректных значений корректными значениями

Результат работы программы после ввода минимально корректных значений представлен на рисунке 2.2



Результат работы программы после ввода максимально корректных значений представлен на рисунке 2.3.

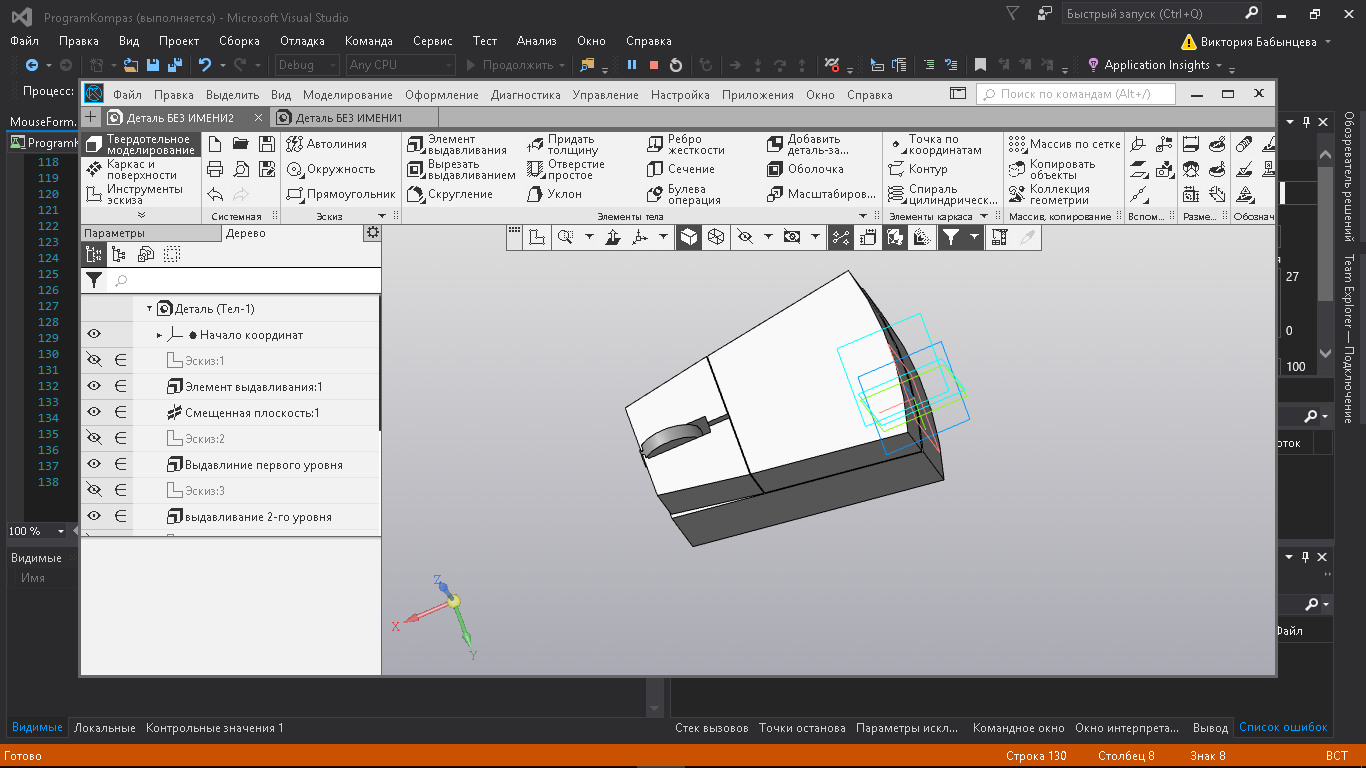


Рисунок 2.3 – Результат работы программы после ввода корректных значений с максимальными корректными значениями

1. **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[1].

На основе следующих тестовых сценариев (таблица 3.1), проводилось тестирование корректности входных параметров и правильности работы плагина с системой «Компас 3D».

Таблица 3.1 – Тестовые сценарии

|  |  |  |
| --- | --- | --- |
| **Название тестового класса** | **Название тестового метода** | **Назначение** |
| DetailParametrsTests | SetLengthMouse\_PositivTest(double LengthOfMouseCount) | Присвоение корректных данных компьютерной мыши |
| SetFrontLengthMouse\_PositivTest  (double FrontOfTheMouse, double lengthMouseCount) | Присвоение корректных данных передней компьютерной мыши |
| SetBackLengthMouse\_PositivTest  (double BackOfTheMouse, double lengthMouseCount) | Присвоение корректных данных задней компьютерной мыши |
| SetTheHeightOfTheFirstLevelOfTheMouse\_PositivTest  (double TheHeightOfTheFirstLevelOfTheMouse,  double lengthMouseCount) | Присвоение корректных данных первого уровня компьютерной мыши |
| SetHeightOfTheSecondMouse\_PositivTest  (double HeightOfTheSecondMouse, double lengthMouseCount) | Присвоение корректных данных второго уровня компьютерной мыши |
| SetLengthMouse\_NegativLessTest  (double LengthOfMouseCount) | Присвоение некорректных данных компьютерной мыши нижней грани |

Продолжение таблицы 3.1 – Тестовые сценарии

|  |  |  |
| --- | --- | --- |
| **Название тестового класса** | **Название тестового метода** | **Назначение** |
| DetailParametrsTests | SetFrontLengthMouse\_NegativLessTest  (double FrontOfTheMouse, double lengthMouseCount) | Присвоение некорректных данных передней части компьютерной мыши нижней грани |
| SetBackLengthMouse\_NegativLessTest  (double BackOfTheMouse, double lengthMouseCount) | Присвоение некорректных данных задней части компьютерной мыши нижней грани |
| SetTheHeightOfTheFirstLevelOfTheMouse\_NegativLessTest  (double TheHeightOfTheFirstLevelOfTheMouse,  double lengthMouseCount) | Присвоение некорректных данных первого уровня компьютерной мыши нижней грани |
| SetHeightOfTheSecondMouse\_NegativLessTest  (double HeightOfTheSecondMouse, double lengthMouseCount) | Присвоение некорректных данных второго уровня компьютерной мыши нижней грани |
| SetLengthMouse\_NegativMoreTest  (double LengthOfMouseCount) | Присвоение некорректных данных компьютерной мыши верхней грани |
| SetFrontLengthMouse\_NegativMoreTest  (double FrontOfTheMouse, double lengthMouseCount) | Присвоение некорректных данных передней части компьютерной мыши верхней грани |

Окончание таблицы 3.1 – Тестовые сценарии

|  |  |  |
| --- | --- | --- |
| **Название тестового класса** | **Название тестового метода** | **Назначение** |
| DetailParametrsTests | SetBackLengthMouse\_NegativMoreTest  (double BackOfTheMouse, double lengthMouseCount) | Присвоение некорректных данных задней части компьютерной мыши верхней грани |
| SetTheHeightOfTheFirstLevelOfTheMouse\_NegativMoreTest  (double TheHeightOfTheFirstLevelOfTheMouse,  double lengthMouseCount) | Присвоение некорректных данных первого уровня компьютерной мыши верхней грани |
| SetHeightOfTheSecondMouse\_NegativMoreTest  (double HeightOfTheSecondMouse, double lengthMouseCount) | Присвоение некорректных данных второго уровня компьютерной мыши верхней грани |

Тестирование проводилось с помощью фреймворка модульного тестирования NUnit 4.6.1 для языков платформы .NET.

Ниже приведен рисунок успешного прохождения всех описанных модульных тестов (см. рисунок 3.1).

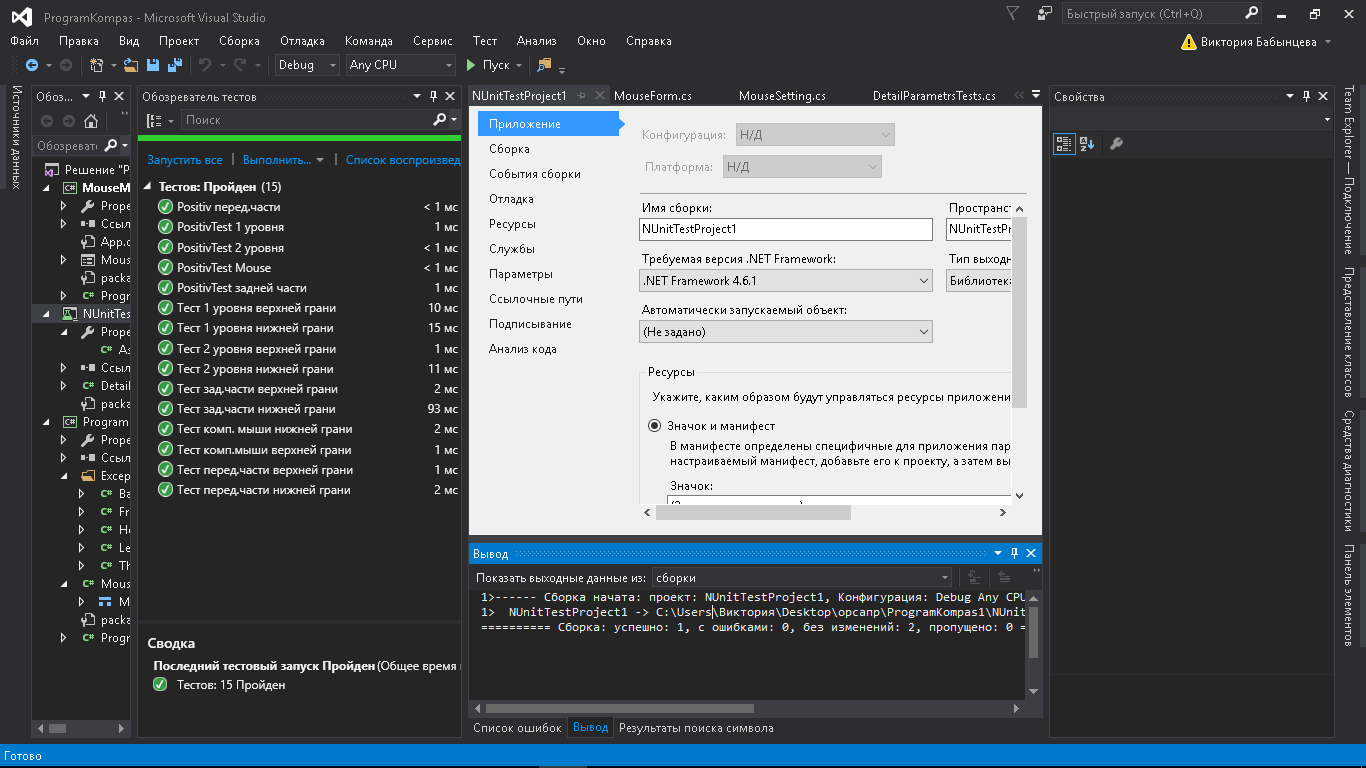


Рисунок 3.1 – Прохождение модульного тестирования приложения

1. **Нагрузочное тестирование**

Нагрузочное тестирование — подвид тестирования производительности, сбор показателей и определение производительности и времени отклика программно-технической системы или устройства в ответ на внешний запрос с целью установления соответствия требованиям, предъявляемым к данной системе (устройству).

Максимальное загрузок моделей одновременно запускает 22 раза, при большем загрузок «КОМПАС-3D» в автономном режиме закрывается и вылезает Exception об непредвиденной ошибке.

На рисунке 4.1 нарисованы график зависимости от количества непрерывно-строящихся деталей (ось х) объема памяти в MB.

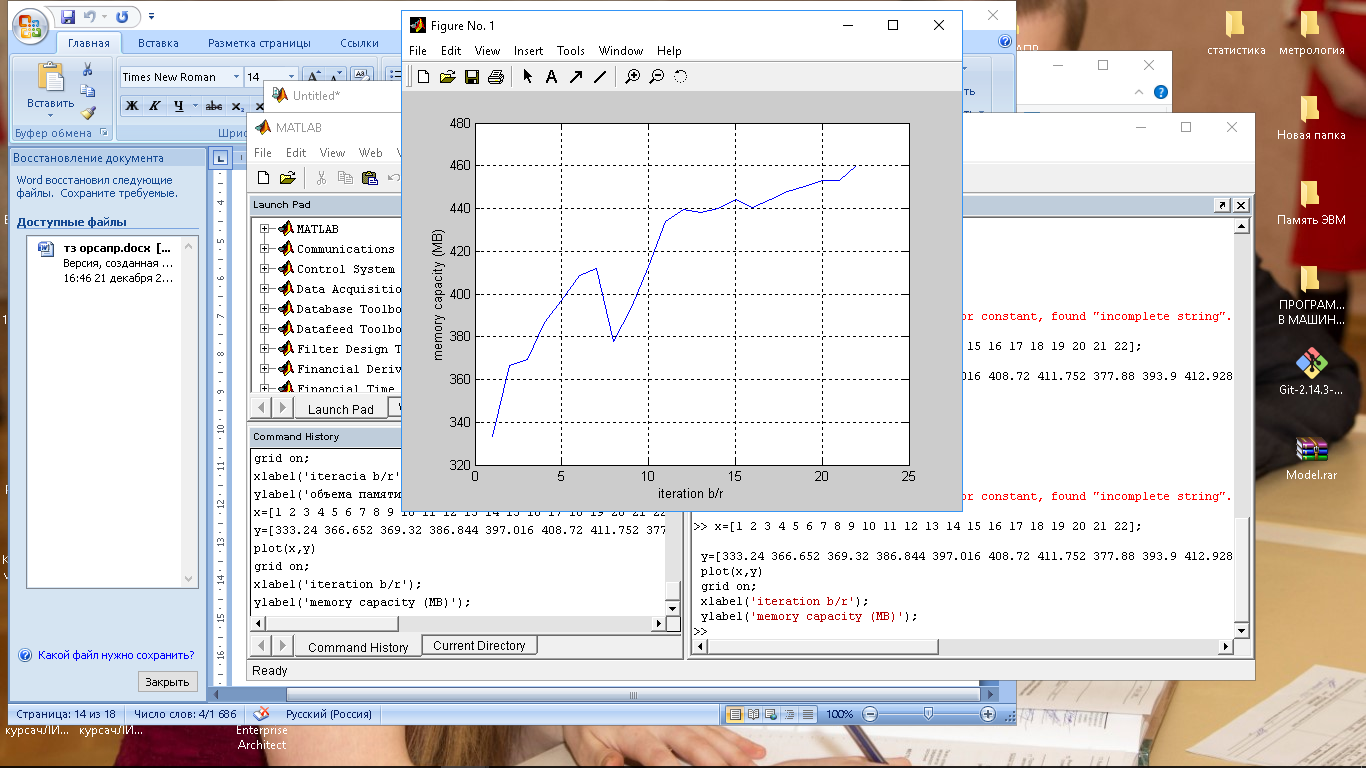


Рисунок 4.1 – Результат нагрузочного теста на затрачиваемые ресурсы

На рисунке 4.1 нарисованы график зависимости от количества непрерывно-строящихся деталей (ось х) и время построения в секундах.

По графикам можно определить, что зависимость прямая. То есть используемая памяти и загрузка на ЦП, при увеличении количества деталей, увеличивается линейно.

**СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ**

1. Новые технологии в программировании: учебное пособие / А.А. Калентьев, Д. В. Гарайс, А. Е. Горяинов. – Томск, 2014. − ­ 176 стр.

Приложение А

U-nit тестирование

{

[TestFixture]

public class DetailParametrsTests

{

[Test]

[TestCase(99, TestName = "Тест комп. мыши нижней грани")]

public void SetLengthMouse\_NegativLessTest(double LengthOfMouseCount)

{

var programKompas = new MouseSetting();

Assert.Throws<ProgramKompas.ExceptionFolder.LengthOfMouseException>(() =>

programKompas.LengthOfMouseProperty = LengthOfMouseCount);

}

[TestCase(151, TestName = "Тест комп.мыши верхней грани")]

public void SetLengthMouse\_NegativMoreTest(double LengthOfMouseCount)

{

var programKompas = new MouseSetting();

Assert.Throws<ProgramKompas.ExceptionFolder.LengthOfMouseException>(() =>

programKompas.LengthOfMouseProperty = LengthOfMouseCount);

}

[TestCase(120, TestName = "PositivTest Mouse")]

public void SetLengthMouse\_PositivTest(double LengthOfMouseCount)

{

var programKompas = new MouseSetting();

programKompas.LengthOfMouseProperty = LengthOfMouseCount;

}

//Передняя часть компьютерной мыши

[TestCase(29, 100, TestName = "Тест перед.части нижней грани")]

public void SetFrontLengthMouse\_NegativLessTest(double FrontOfTheMouse, double lengthMouseCount)

{

var programKompas = new MouseSetting();

programKompas.LengthOfMouseProperty = lengthMouseCount;

Assert.Throws<ProgramKompas.ExceptionFolder.FrontOfTheMouseException>(() =>

programKompas.FrontOfTheMouseProperty = FrontOfTheMouse);

}

[TestCase(41, 100, TestName = "Тест перед.части верхней грани")]

public void SetFrontLengthMouse\_NegativMoreTest(double FrontOfTheMouse, double lengthMouseCount)

{

var programKompas = new MouseSetting();

programKompas.LengthOfMouseProperty = lengthMouseCount;

Assert.Throws<ProgramKompas.ExceptionFolder.FrontOfTheMouseException>(() =>

programKompas.FrontOfTheMouseProperty = FrontOfTheMouse);

}

[TestCase(35, 100, TestName = "Positiv перед.части")]

public void SetFrontLengthMouse\_PositivTest(double FrontOfTheMouse, double lengthMouseCount)

{

var programKompas = new MouseSetting();

programKompas.LengthOfMouseProperty = lengthMouseCount;

programKompas.FrontOfTheMouseProperty = FrontOfTheMouse;

}

//Задняя часть компьютерной мыши

[TestCase(59, 100, TestName = "Тест зад.части нижней грани")]

public void SetBackLengthMouse\_NegativLessTest(double BackOfTheMouse, double lengthMouseCount)

{

var programKompas = new MouseSetting();

programKompas.LengthOfMouseProperty = lengthMouseCount;

Assert.Throws<ProgramKompas.ExceptionFolder.BackOfTheMouseException>(() =>

programKompas.BackOfTheMouseProperty = BackOfTheMouse);

}

[TestCase(71, 100, TestName = "Тест зад.части верхней грани")]

public void SetBackLengthMouse\_NegativMoreTest(double BackOfTheMouse, double lengthMouseCount)

{

var programKompas = new MouseSetting();

programKompas.LengthOfMouseProperty = lengthMouseCount;

Assert.Throws<ProgramKompas.ExceptionFolder.BackOfTheMouseException>(() =>

programKompas.BackOfTheMouseProperty = BackOfTheMouse);

}

[TestCase(65, 100, TestName = "PositivTest задней части")]

public void SetBackLengthMouse\_PositivTest(double BackOfTheMouse, double lengthMouseCount)

{

var programKompas = new MouseSetting();

programKompas.LengthOfMouseProperty = lengthMouseCount;

programKompas.BackOfTheMouseProperty = BackOfTheMouse;

}

//Первый ярус мыши

[TestCase(19, 100, TestName = "Тест 1 уровня нижней грани")]

public void SetTheHeightOfTheFirstLevelOfTheMouse\_NegativLessTest(double TheHeightOfTheFirstLevelOfTheMouse, double lengthMouseCount)

{

var programKompas = new MouseSetting();

programKompas.LengthOfMouseProperty = lengthMouseCount;

Assert.Throws<ProgramKompas.ExceptionFolder.TheHeightOfTheFirstLevelOfTheMouseException> (() =>

programKompas.TheHeightOfTheFirstLevelOfTheMouseProperty = TheHeightOfTheFirstLevelOfTheMouse);

}

[TestCase(31, 100, TestName = "Тест 1 уровня верхней грани")]

public void SetTheHeightOfTheFirstLevelOfTheMouse\_NegativMoreTest(double TheHeightOfTheFirstLevelOfTheMouse, double lengthMouseCount)

{

var programKompas = new MouseSetting();

programKompas.LengthOfMouseProperty = lengthMouseCount;

Assert.Throws<ProgramKompas.ExceptionFolder.TheHeightOfTheFirstLevelOfTheMouseException>(() =>

programKompas.TheHeightOfTheFirstLevelOfTheMouseProperty = TheHeightOfTheFirstLevelOfTheMouse);

}

[TestCase(25, 100, TestName = "PositivTest 1 уровня")]

public void SetTheHeightOfTheFirstLevelOfTheMouse\_PositivTest(double TheHeightOfTheFirstLevelOfTheMouse, double lengthMouseCount)

{

var programKompas = new MouseSetting();

programKompas.LengthOfMouseProperty = lengthMouseCount;

programKompas.TheHeightOfTheFirstLevelOfTheMouseProperty= TheHeightOfTheFirstLevelOfTheMouse ;

}

//Второй уровень компьютерной мыши

[TestCase(15, 100, TestName = "PositivTest 2 уровня")]

public void SetHeightOfTheSecondMouse\_PositivTest(double HeightOfTheSecondMouse, double lengthMouseCount)

{

var programKompas = new MouseSetting();

programKompas.LengthOfMouseProperty = lengthMouseCount;

programKompas.HeightOfTheSecondMouseProperty = HeightOfTheSecondMouse;

}

[TestCase(9, 100, TestName = "Тест 2 уровня нижней грани")]

public void SetHeightOfTheSecondMouse\_NegativLessTest(double HeightOfTheSecondMouse, double lengthMouseCount)

{

var programKompas = new MouseSetting();

programKompas.LengthOfMouseProperty = lengthMouseCount;

Assert.Throws<ProgramKompas.ExceptionFolder.ExceptionHeightOfTheSecondMouse>(() =>

programKompas.HeightOfTheSecondMouseProperty = HeightOfTheSecondMouse);

}

[TestCase(21, 100, TestName = "Тест 2 уровня верхней грани")]

public void SetHeightOfTheSecondMouse\_NegativMoreTest(double HeightOfTheSecondMouse, double lengthMouseCount)

{

var programKompas = new MouseSetting();

programKompas.LengthOfMouseProperty = lengthMouseCount;

Assert.Throws<ProgramKompas.ExceptionFolder.ExceptionHeightOfTheSecondMouse>(() =>

programKompas.HeightOfTheSecondMouseProperty = HeightOfTheSecondMouse);

}

}

}

ПРИЛОЖЕНИЕ Б

ОПИСАНИЕ КЛАССОВ