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)

ПЛАГИН «ПЕПЕЛЬНИЦА»

ДЛЯ СИСТЕМЫ АВТОМАТИЗИРОВАННОГО ПРОЕКТИРОВАНИЯ

КОМПАС-3D

Пояснительная записка по дисциплине

«Основы разработки САПР»

Студент гр. 589-1

\_\_\_\_\_\_\_\_\_ Сотник Д.А.

\_\_\_\_\_\_\_\_\_

Руководитель

преподаватель каф., к.т.н.

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

(оценка) \_\_\_\_\_\_\_\_\_

Томск 2022

**Реферат**

Учебная работа 32 страниц, 12 таблиц, 18 рисунок, 16 источников.

Ключевые слова: КОМПАС-3D, МЕЧ, СИСТЕМА АВТОМАТИЗИРОВАННОГО ПРОЕКТИРОВАНИЯ, БИБЛИОТЕКА.

Целью данной работы является разработка библиотеки «Меч» для системы автоматизированного проектирования Компас-3D.

В процессе работы должны были пройдены все стадии разработки программного обеспечения: выбор темы, составление технического задания, составление проекта системы, создание первой реализации и её тестирование, создание конечной библиотеки.

Отчет по учебной работе выполнен в текстовом редакторе Microsoft Word 2016.

**Оглавление**

[Введение 4](#_Toc122070126)

[1 ПОСТАНОВКА И АНАЛИЗ ЗАДАЧИ 5](#_Toc122070127)

[1.1 Описание предмета проектирования 5](#_Toc122070128)

[1.2 Описание инструментов и средств реализации 8](#_Toc122070129)

[1.3 Назначение библиотеки 8](#_Toc122070130)

[2 ОБЗОР АНАЛОГОВ 9](#_Toc122070131)

[2.1 Программа построения 3D моделей по заданным значениям в AutoCAD «Лекало» 9](#_Toc122070132)

[2.2 Плагин «Archimesh» для Blender 10](#_Toc122070133)

[3 ОПИСАНИЕ РЕАЛИЗАЦИИ 12](#_Toc122070135)

[4 ОПИСАНИЕ ПРОГРАММЫ ДЛЯ ПОЛЬЗОВАТЕЛЯ 22](#_Toc122070136)

[5 ТЕСТИРОВАНИЕ ПРОГРАММЫ 25](#_Toc122070137)

[5.1 Функциональное тестирование 25](#_Toc122070138)

[5.2 Модульное тестирование 30](#_Toc122070139)

[5.3 Нагрузочное тестирование 31](#_Toc122070140)

[Заключение 34](#_Toc122070141)

[Список использованных источников 35](#_Toc122070142)

# Введение

Автоматизация проектирования имеет огромное значение для развития науки, техники и производства в современном обществе. В настоящее время автоматизация – основной способ повышения производительности и эффективности труда инженерно-технических работников, занимающихся проектированием сложных устройств. Использование автоматизации в проектировании позволяет создавать все более сложные технические объекты и гибко реагировать на появление новых решений и технологий в той или иной области техники.

Практическая реализация методов и идей автоматизированного проектирования происходит в рамках систем автоматизированного проектирования (САПР). В рамках современного «компьютеризированного» общества инженер любой специальности, занимающийся разработкой технических устройств, должен уметь использовать средства автоматизированного проектирования. Это позволяет повысить эффективность проектирования, улучшить его качество, снизить материальные затраты и уменьшить число разработчиков.

Таким образом, целью учебной работы является разработка библиотеки «Меч» для системы автоматизированного проектирования Компас-3D. «Компас» – это семейство систем автоматизированного проектирования, универсальная система автоматизированного проектирования, позволяющая в оперативном режиме выпускать чертежи изделий, схемы, спецификации, таблицы, инструкции, расчётно-пояснительные записки, технические условия, текстовые и прочие документы [1].

# 1 ПОСТАНОВКА И АНАЛИЗ ЗАДАЧИ

В рамках учебной дисциплины «Основы разработки САПР» требовалось разработать библиотеку в соответствии с техническим заданием. На основе заданных параметров библиотека, взаимодействуя с САПР «Компас-3D», должна строить трёхмерную модель меча [2]. Также библиотека должна позволять изменять входные параметры пепельницы.

Изменяемые параметры:

* ширина гарды;
* длина лезвия;
* толщина лезвия;
* длина рукоятки вместе с гардой;
* длина лезвия;
* диаметр рукоятки;

# 1.1 Описание предмета проектирования

Предметом проектирования является меч. На рисунке 1.1 представлен чертеж меча.

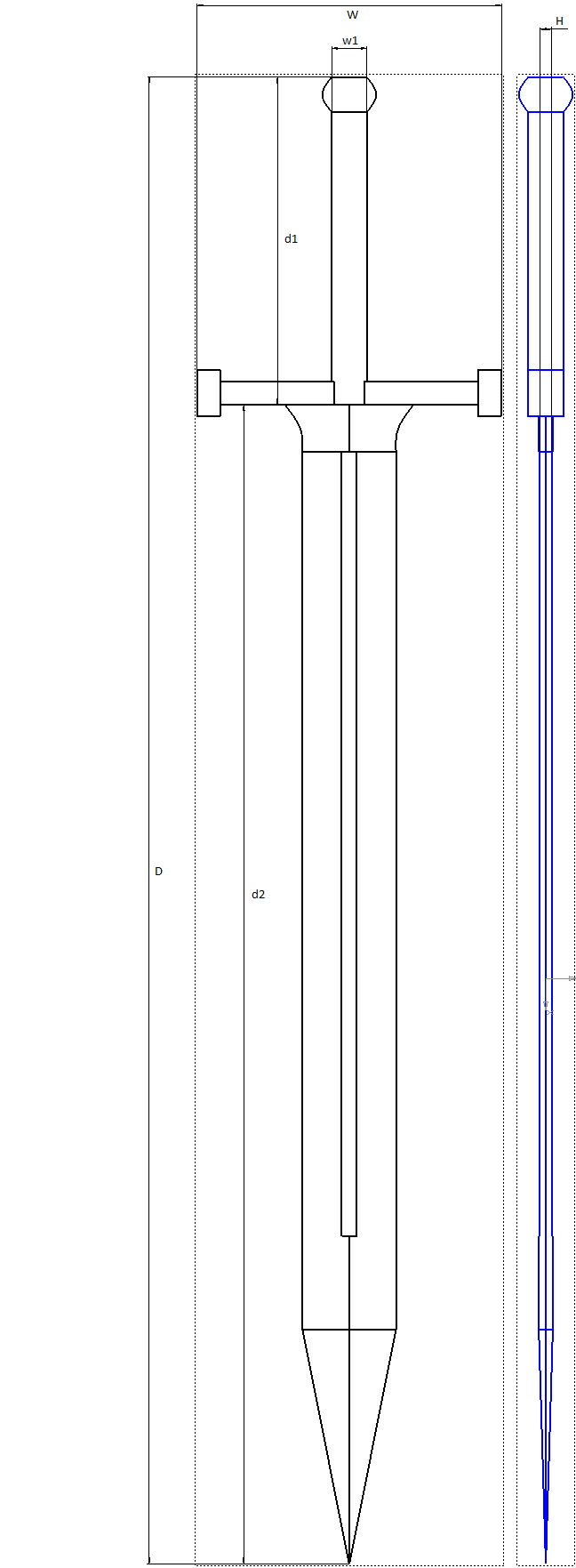


Рисунок 1.1 – чертеж меча

На рисунке 1.2 представлена 3D-модель меча

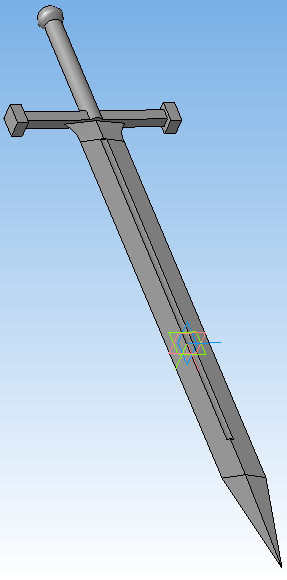
****

Рисунок 1.2 – 3D-модель меча

Параметры меча:

* W – Ширина гарды (мин. 200 мм, макс. 300 мм);
* D – Длина меча (мин. 1000 мм, макс. 1500 мм) ;
* H – Толщина лезвия (мин. 5 мм, макс. 15 мм). Толщина лезвия не может быть больше половины диаметра рукоятки;
* d1 – Длина рукоятки вместе с гардой (мин. 175 мм, макс. 250 мм). Длина рукоятки вместе с гардой не может быть меньше ¼ от длины лезвия;
* d2 – Длина лезвия (мин. 700 мм, макс. 1000 мм);
* w1 – Диаметр рукоятки (мин. 10 мм, макс. 30 мм). Высота пепельницы (B) относится к толщине дна (D) как 5 к 1;

# 1.2 Описание инструментов и средств реализации

Для создания библиотеки используется среда разработки Visual Studio 2019 [3]. Библиотека написана с пользовательским интерфейсом на WindowsForms с использованием .NET Framework 4.7.2 [4]. Инструментом для тестирования является библиотеки NUnit 3.13.3 [5] и NUnit3TestAdapter 4.2.1 [5]. В качестве системы автоматизированного проектирования выбран «Компас-3D» v.15.2 [1].

# 1.3 Назначение библиотеки

Программа предназначена для автоматизации моделирования детали «Меч».

Плагин позволяет пользователю ввести вышеперечисленные значения через графический интерфейс. В программе предусмотрена проверка корректности введенных данных и сообщение пользователю о неправильно заполненных полях с помощью цветового выделения.

При попытке запустить моделирование с некорректными введёнными данными кнопка для построения становится неактивной.

При правильно введенных значениях результатом работы программы будет созданная по ним модель меча. Размеры всех параметров, необходимых при построении, рассчитываются автоматически.

# 2 ОБЗОР АНАЛОГОВ

2.1 Модуль для Autodesk AutoCAD 3DPiping

3DPiping – подключаемый модуль для Autodesk AutoCAD, который был разработан инженерами-трубопроводчиками для своих коллег. Данный подключаемый модуль содержит библиотеку САПР из более чем 3900 3D-блоков трубопроводов для моделирования даже сложных трубопроводных систем в 3D. Эта версия плагина основана на стандартах ASME (American Society of Mechanical Engineers). Это означает, что все 3D блоки в данном плагине были разработаны в соответствии со стандартами ASME.

Библиотека 3DPiping содержит большое разнообразие различных труб, сварных и резьбовых фитингов, фланцов, прокладок, наборов болтов, крепежных элементов, клапанов с фланцами и резьбой, опор для труб, стальных профилей и других типов компонентов трубопроводов, что позволяет легко и с уверенностью создавать профессиональные чертежи трубопроводных систем [6].

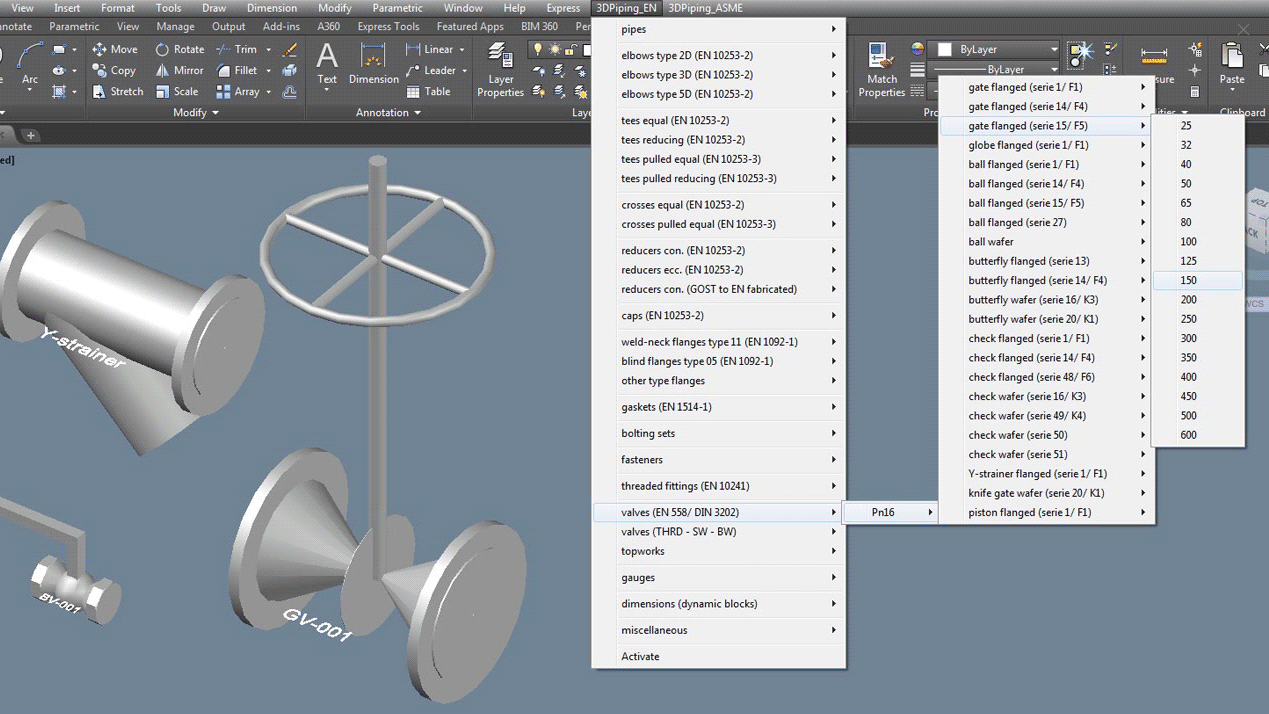


Рисунок 2.1 – Интерфейс программы 3DPiping

# 2.2 Функционал SOLIDWORKS

В SOLIDWORKS 2022 можно создать гибридное твердое тело или тело поверхности, которое включает геометрию сетки BREP и стандартную геометрию SOLIDWORKS BREP.

Во многих случаях гибридное моделирование может помочь избежать преобразования стандартного тела BREP в сетку BREP, которая часто использовалась для сравнения стандартного тела BREP с телом сетки BREP. В гибридном моделировании можно добавить стандартную геометрию BREP непосредственно в тело сетки BREP.

В прошлых версиях факт объединения сетки BREP и стандартные геометрии SOLIDWORKS BREP в одном теле отсутствовал [7].

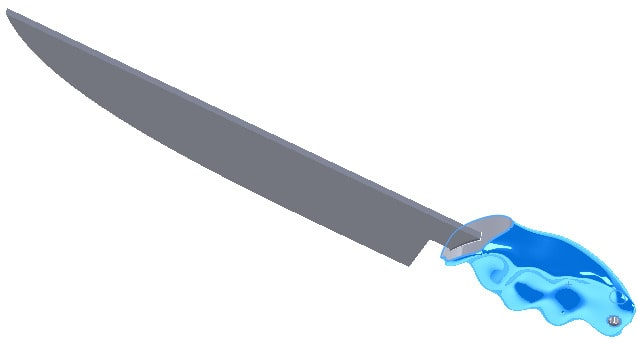


Рисунок 2.2 – Пример готовой модели

# 3 ОПИСАНИЕ РЕАЛИЗАЦИИ

**UML (Unified Modeling Language)** – это система обозначений, которую можно применять для объектно-ориентированного анализа и проектирования. Его можно использовать для визуализации, спецификации, конструирования и документирования программных систем. Словарь UML включает три вида строительных блоков: диаграммы, сущности, связи [8].

**Sparx Systems Enterprise Architect** – это инструмент визуального моделирования и проектирования, основанный на OMG UML. Платформа поддерживает: проектирование и построение программных комплексов; моделирование бизнес-процессов; и моделирование отраслевых доменов. Он используется предприятиями и организациями не только для моделирования архитектуры своих систем, но и для обработки реализации этих моделей на протяжении всего жизненного цикла разработки приложений [9].

Диаграмма классов плагина «Меч» представлена на рисунке 3.1

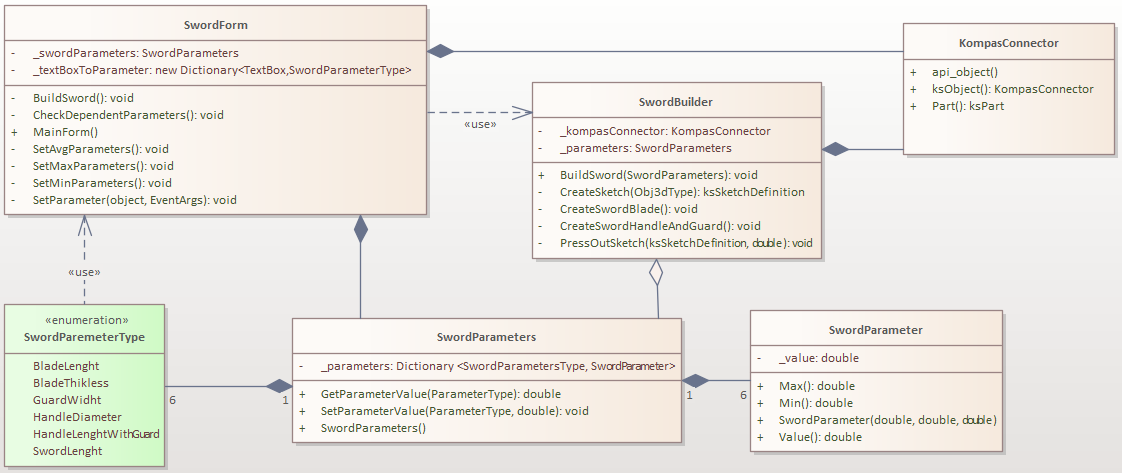


Рисунок 3.1 - UML-диаграмма классов.

Далее в таблицах 3.1 – 3.4 представлено описание классов

Таблица 3.1 – Описание полей, методов, сущностей класса «SwordForm»

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| \_swordParameters |  | Хранит в себе набор методов для построения Меча |
| \_textBoxToParameter |  | Хранит в себе набор TextBox и соответствующий ему тип параметра из перечисления «SwordParametersType» |
| SetParameter(object, EventArgs) | void | Устанавливает значение параметра |
| SetMinParameters() | void | Устанавливает минимальное значение всех параметров |
| * SetMaxParameters() | void | Устанавливает максимальное значение всех параметров |
| - SetAvgParameters() | void | Устанавливает среднее значение всех параметров |
| Build() | void | Строит кружку по заданным параметрам |

Таблица 3.2 – Описание полей, методов, сущностей класса «SwordParameters»

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| \_parameters |  | Хранит данные о каждом параметре модели из перечисления «SwordParametersType» |
| SwordParameters() |  | Конструктор для создания экземпляра класса |
| SetParameterValue(ParameterType, double) | void | Устанавливает значение определённого параметра |
| GetParameterValue(ParameterType) | double | Возвращает значение определённого параметра |

Таблицы 3.2 – Описание полей, методов, сущностей класса «SwordParameter»

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| \_value |  | Хранит текущее значение |
| SwordParameter(double, double, double) |  | Конструктор для создания экземпляра класса |
| Max () | double | Возвращает максимальное допустимое значение параметра |

Продолжение таблицы 3.2– Описание полей, методов, сущностей класса «SwordParameter»

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| Min () | double | Возвращает минимальное допустимое значение параметра |
| Value() | double | Возвращает текущее значение параметра. Задает новое значение параметра |

Таблица 3.3 – Описание полей, методов, сущностей класса «SwordBuilder»

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| kompasConnector |  | Хранит в себе методы необходимые для связи с КОМПАС 3D |
| \_parameters |  | Хранит данные о каждом параметре модели из перечисления «SwordParameter» |
| BuildSword(SwordParameters) | void | Построение меча по заданным параметрам |
| CreateSwordBlade() | void | Построение лезвия меча |
| CreateSwordHandleAndGuard() |  | Построение рукоятки вместе с гардой |

В процессе реализации библиотеки и её тестирования были изменены поля классов MainForm, SwordBuilder, KompasConnector– теперь реализовано функции для базовых операций.

В KompasWrapper были добавлены новые методы, в том числе для дополнительной функциональности. В главной форме изменились методы для проверки ошибок введённых значений в режиме реального времени.

В таблицах 3.7-3.12 представлены основные классы MainForm, SwordPArameters, Parameters, SwordBuilder и KompasConnector, их поля, методы и свойства в итоговой реализации.

Итоговая диаграмма представлена на рисунке 3.6.

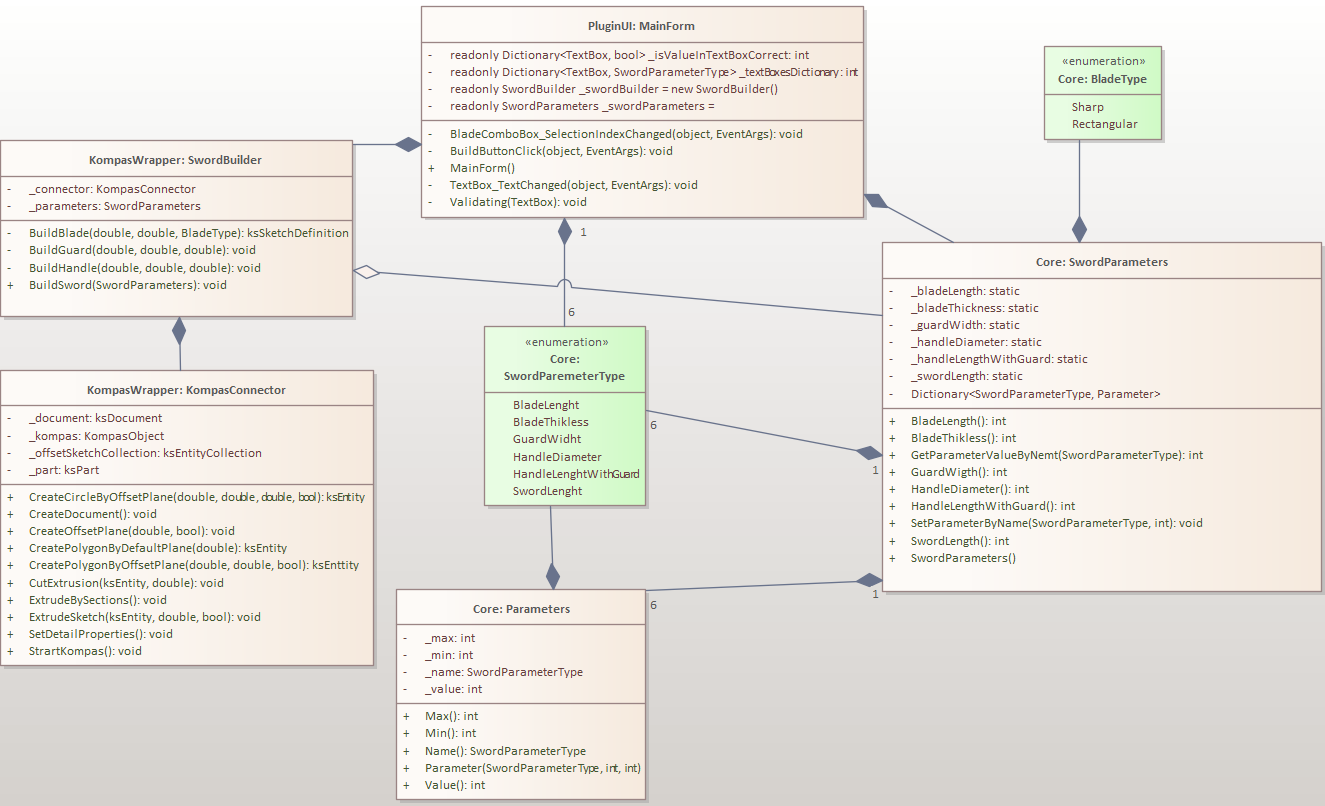


Рисунок 3.6 – Итоговая диаграмма классов

Таблица 3.1 – Описание полей, методов, сущностей класса «MainForm»

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| \_swordBuilder | SwordBuilder | Объект класса построителя |
| \_swordParameters | SwordParameters | Объект класса с параметрами |
| \_isValueInTextBoxCorrect | Dictionary<TextBox, bool> | Словарь содержащие пары (Текстбокс, корректное ли значение в нём) |
| \_textBoxesDictionary | Dictionary<TextBox, SwordParameterType> | Словарь содержащий пары (Текстбоксы, имя параметра) |
| BuildButton\_Click | void | Обработчик нажатия кнопки "Построить" |
| Validating | void | Общий метод валидации текстбокса |
| BladeComboBox\_SelectedIndexChanged | void | Метод выбора типа лезвия |

Таблица 3.2 – Описание полей, методов, сущностей класса «SwordParameters»

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| \_bladeLength |  | Хранит данные о длине лезвия |
| \_bladeThickless |  | Хранит данные о толщине лезвия |
| \_guardWigth |  | Хранит данные о ширине гарды |
| \_handleDiameter |  | Хранит данные о диаметре рукоятки |
| \_handleLengthWithGuard |  | Хранит данные о длине рукоятки вместе с гардой |
| \_swordLength |  | Хранит данные о длине меча |

Продолжение таблицы 3.2 - Описание полей, методов, сущностей класса «SwordParameters»

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| BladeLength | int | Задаёт и возвращает длину лезвия |
| BladeThickless | int | Задаёт и возвращает толщину лезвия |
| GuardWigth | int | Задаёт и возвращает ширину гарды |
| HandleDiameter | int | Задаёт и возвращает диаметр рукоятки |
| HandleLengthWithGuard | int | Задаёт и возвращает длину рукоятки вместе с гардой |
| SwordLength | int | Задаёт и возвращает длину меча |
| GetParameterValueByName | int | Возвращает значение параметра по имени |
| SetParameterByName | int | Передаёт значение в сеттер параметра по имени |
| SwordParameters |  | Конструктора класса |

Таблицы 3.3 – Описание полей, методов, сущностей класса «Parameters»

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| \_min | int | Минимальное значение параметра |
| \_max | int | Максимальное значение параметра |
| \_value | int | Присеваемое значение параметра |

Продолжение таблицы 3.3 - Описание полей, методов, сущностей класса «Parameters»

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| \_name | SwordParameterType | Название параметра для составления сообщения исключения |
| Min | int | Передаёт или задаёт минимальное значение |
| Max | int | Передаёт или задаёт максимальное значение |
| Value | int | Передаёт или задаёт значение параметра |
| Name | SwordParameterType | Передаёт или задаёт имя, которое должно быть не пустым или не является разделяющим знаком |
| Parameter |  | Конструктор шаблона параметра |

Таблица 3.4 – Описание полей, методов, сущностей класса «SwordBuilder»

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| \_connector |  | Объект класса коннектора для связи с Компас-3D |
| BuildSword | void | Построение меча по заданным параметрам |
| BuildBlade | void | Построение лезвия меча |
| BuildGuard | void | Построение гарды |
| BuildHandle | void | Построение рукоятки меча |

Таблица 3.5 - Описание полей, методов, сущностей класса «KompasConnector»

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| \_kompas | KompasObject | Обьект Компас API |
| \_part | ksPart | Деталь |
| \_document | ksDocument3D | Документ-модель |
| \_offsetScketchCollection | ksEntityCollection | Коллекция эскизов на смещённых плоскостях |
| StartKompass | void | Запуск Компас-3D |
| CreateDocument | void | Создание документа в Компас-3D |
| SetDetailProporties | void | Установка свойств детали |
| CreateCircleByOffsetPlane | ksEntity | Создание круга на смещённой плоскости |
| ExtrudeBySections | void | Выдавливание по сечениям смещённых эскизов |
| CreatePolygonByOffsetPlate | ksEntity | Создание многоугольника по смещённой плоскости |
| CreatePolygonByDefaultPlane | ksEntity | Создание многоугольника по базовой плоскости YOZ |
| CreateOffsetplane | void | Создание смещённой плоскости |
| ExtrudeSketch | void | Метод, выполняющий выдавливание по эскизу |
| CutExtrusion | void | Метод, выполняющий вырезание по эскизу |

# 4 ОПИСАНИЕ ПРОГРАММЫ ДЛЯ ПОЛЬЗОВАТЕЛЯ

Пользовательский интерфейс (UI) – интерфейс, обеспечивающий передачу информации между пользователем-человеком и программно-аппаратными компонентами компьютерной системы [10].

Плагин представляет собой пользовательскую форму с полями для ввода соответствующих параметров, форму можно увидеть на рисунке 4.1. Посредством кнопки «BUILD» осуществляется запуск САПР «Компас-3D», на рабочей области которой построится трехмерная модель детали по заданным параметрам. Если построение детали выполняется несколько раз, то плагин не запускает несколько копий программы «Компас-3D», а создает в ней новый документ. В случае ввода значений параметров, не входящих в допустимый диапазон, поле для ввода окрашивается в светло-оранжевый цвет. Доступен выбор варианта лезвия меча. Пример неправильного ввода представлен на рисунке 4.2.

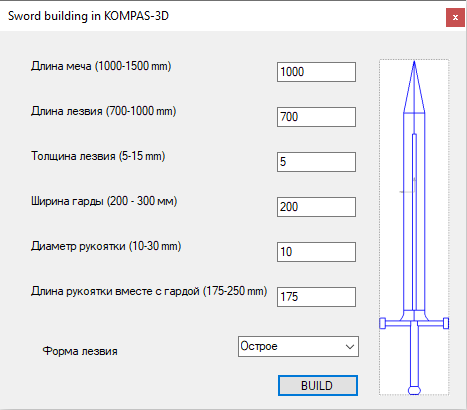


Рисунок 4.1 – Макет пользовательского интерфейса

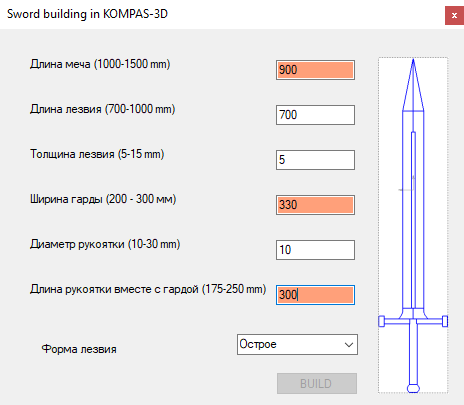


Рисунок 4.2 – Реакция приложения на ввод некорректных значений

Если пользователь ввёл правильные значения, при нажатии кнопки «BUILD» происходит построение детали. Трёхмерная модель меча, построенная в системе автоматизированного проектирования Компас-3D с помощью разработанной библиотеки представлена на рисунке 4.3

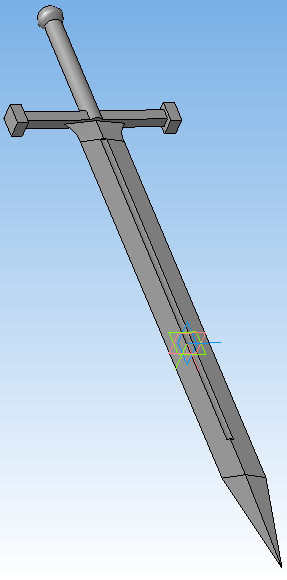
****

Рисунок 4.3 – Трёхмерная модель меча

# 5 ТЕСТИРОВАНИЕ ПРОГРАММЫ

# 5.1 Функциональное тестирование

Функциональное тестирование – это тип тестирования программного обеспечения, при котором тестовые сценарии выполняются тестировщиком вручную без использования автоматизированных инструментов. Целью ручного тестирования является выявление ошибок, проблем и дефектов в программном приложении. Функциональное тестирование программного обеспечения – это самый примитивный метод из всех видов тестирования. Концепции ручного тестирования не требуют знания какого-либо инструмента тестирования [11].

В рамках функционального тестирования будет проверка библиотеки при введённых минимальных, средних и максимальных параметрах.

Тестирование при минимальных параметрах:

* ширина гарды равна 200 мм;
* длина меча равна 1000 мм;
* толщина лезвия равна 5 мм;
* длина рукоятки вместе с гардой равна 175 мм;
* длина лезвия равна 700 мм;
* диаметр рукоятки равен 10 мм

Результат построения при заданных параметрах представлен на рисунках 5.1-5.2.

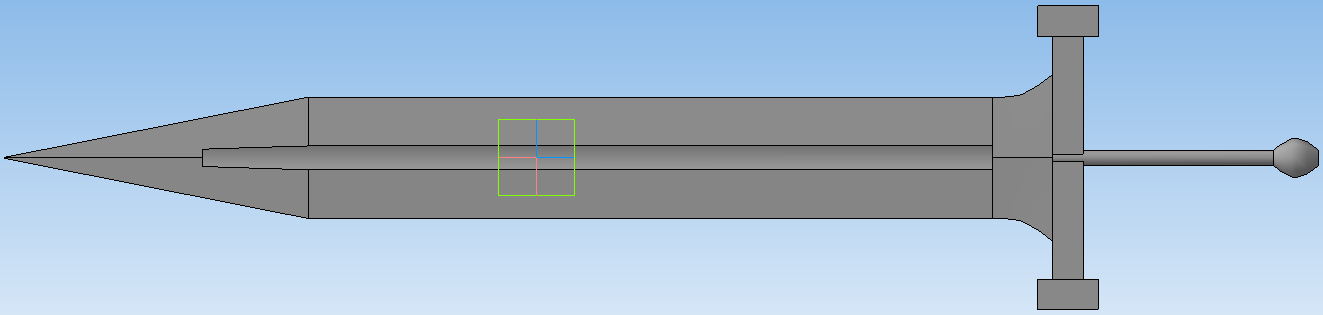


Рисунок 5.1 – Вид на модель спереди



Рисунок 5.2 – Вид на модель сбоку

Тестирование при средних параметрах:

* ширина гарды равна 250 мм;
* длина меча равна 1250 мм;
* толщина лезвия равна 10 мм;
* длина рукоятки вместе с гардой равна 210 мм;
* длина лезвия равна 850 мм;
* диаметр рукоятки равен 20 мм

Результат построения при заданных параметрах представлен на рисунках 5.3-5.4.

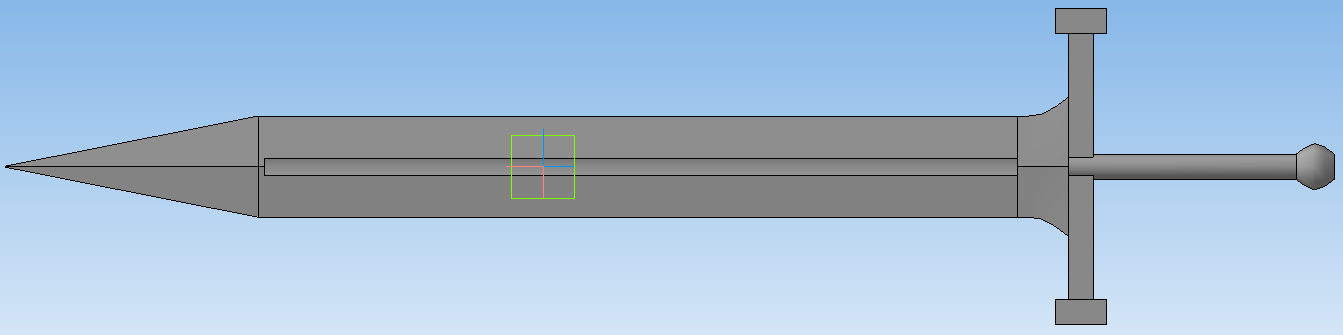


Рисунок 5.3 – Вид на модель спереди

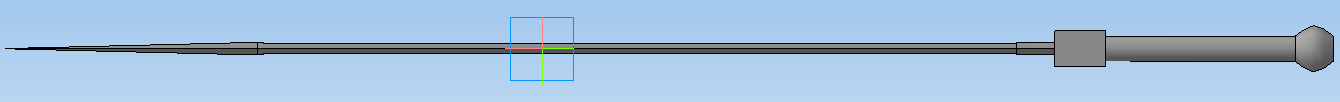


Рисунок 5.4 – Вид на модель сбоку

Тестирование при максимальных параметрах:

* ширина гарды равна 300 мм;
* длина меча равна 1500 мм;
* толщина лезвия равна 15 мм;
* длина рукоятки вместе с гардой равна 250 мм;
* длина лезвия равна 1000 мм;
* диаметр рукоятки равен 30 мм

Результат построения при заданных параметрах представлен на рисунках 5.5-5.6.

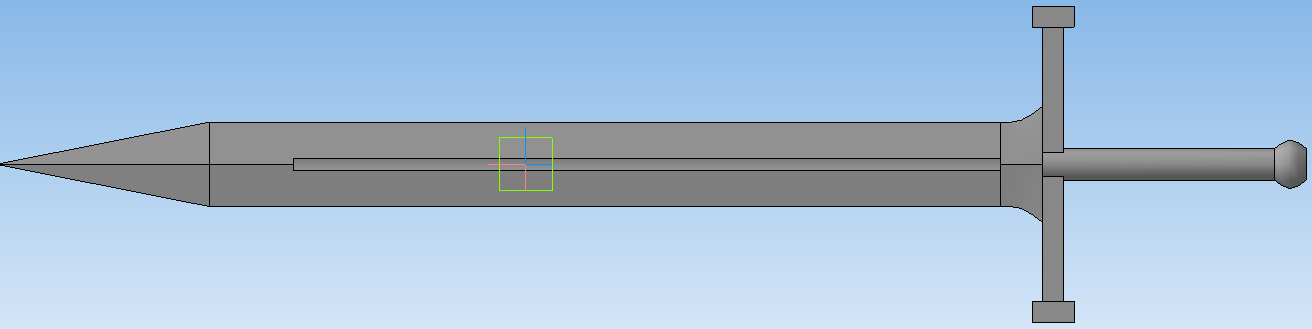


Рисунок 5.5 – Вид на модель в спереди

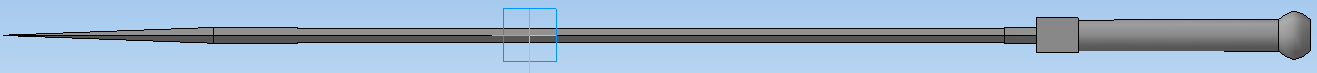


Рисунок 5.6 – Вид на модель снизу

# 5.2 Модульное тестирование

**Модульное тестирование (Unit Testing)**– это тип тестирования программного обеспечения, при котором тестируются отдельные модули или компоненты программного обеспечения. Его цель заключается в том, чтобы проверить, что каждая единица программного кода работает должным образом. Данный вид тестирование выполняется разработчиками на этапе кодирования приложения. Модульные тесты изолируют часть кода и проверяют его работоспособность. Единицей для измерения может служить отдельная функция, метод, процедура, модуль или объект [13].

Для тестирования используется библиотеки NUnit 3.13.3 [5] и NUnit3TestAdapter 4.2.1 [5]. Тестами покрыта вся логика программы. Всего было написано 14 тестов. На рисунке 5.7 представлено тестирование логики.

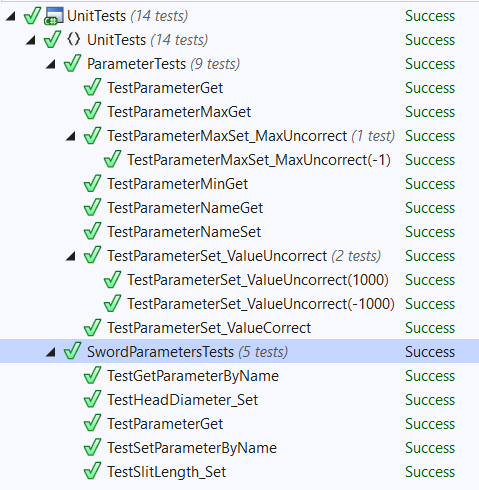


Рисунок 5.7 – Тестирование логики программы

# 5.3 Нагрузочное тестирование

Нагрузочное тестирование – это 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) [14].

Нагрузочное тестирование проводилось на персональном компьютере со следующей конфигурацией:

* процессор Intel(R) Core(TM) i5-8250U CPU @ 1.60GHz;
* оперативная память 8,0 ГБ;
* видеокарта NVIDIA GeForce MX130.

Для тестирования будет зациклено построение модели со следующими параметрами:

* ширина гарды равна 200 мм;
* длина меча равна 1000 мм;
* толщина лезвия равна 5 мм;
* длина рукоятки вместе с гардой равна 175 мм;
* длина лезвия равна 700 мм;
* диаметр рукоятки равен 10 мм

В результате тестирования было построено 36 деталей. По полученным измерениям был рассчитан средний результат, по которому были построены графики. На рисунке 5.13 показана зависимость количества используемой оперативной памяти в гигабайтах от количества построенных деталей.



Рисунок 5.13 – Зависимость оперативной памяти от количества деталей

Из графика видно, что зависимость является линейной пока оперативная память на загружена практически полностью. На графики имеются скачки вниз, то есть происходит уменьшение нагрузки на оперативную память. Это обусловлено устройством оперативной памяти, что для её регенерации периодически приостанавливается обращение, это снижает среднюю скорость обмена и понижает нагрузку [12]. Также в середине графика видно, что нагрузка на память не увеличивается. Это связано с тем, что операционная система имеет файл подкачки, в который выгружаются неактивные и неиспользуемые данные, снимая нагрузку с оперативной памяти [13].

На рисунке 5.14 показана зависимость потраченного времени от количества построенных деталей.



Рисунок 5.14 – Зависимость времени от количества построенных деталей

Из графика видно, что зависимость количества необходимого времени от числа деталей является линейной с изменением угла наклона в течение теста. Это связано с тем, что при большой загрузке оперативной памяти и центрального процессора падает скорость построения одной детали. Поэтому увеличивается коэффициент наклона прямой.

# Заключение

В результате выполнения данной учебной работы были пройдены все стадии разработки программного обеспечения: выбор темы, составление технического задания, составление проекта системы, создание первой реализации и её тестирование, создание конечной библиотеки. Был изучен API приложения Компас-3D. Также были найдены аналоги разрабатываемого плагина. Были спроектированы UML диаграммы классов и было проведено функциональное, модульное и нагрузочное тестирования созданной библиотеки.

# Список использованных источников

1. Официальный сайт системы автоматизированного проектирования «Компас-3D» [Электронный ресурс] – Режим доступа: https://kompas.ru/kompas-3d/about/ (дата обращения: 07.10.2022)
2. Компас (САПР) [Электронный ресурс] – Режим доступа: https://ru.wikipedia.org/wiki/Компас\_(САПР) (дата обращения: 17.10.2022)
3. Заметки о выпуске Visual Studio 2019 версии 16.10 2019 [Электронный ресурс] – Режим доступа: https://learn.microsoft.com/ru-ru/visualstudio/releases/2019/release-notes-v16.10 (дата обращения: 14.01.22)
4. Платформа .NET Framework 4.7.2 для Windows [Электронный ресурс] – Режим доступа: https: //support.microsoft.com/ru-ru/topic/автономный-установщик-microsoft-платформа-net-framework-4-7-2-для-windows-05a72734-2127-a15d-50cf-daf56d5faec2 (дата обращения: 14.01.22)
5. Фреймворк NUnit [Электронный ресурс] – Режим доступа: https://nunit.org/ (дата обращения: 14.01.22)
6. Программа автоматического построения 3D моделей труб в AutoCAD [Электронный ресурс] – Режим доступа: https://apps.autodesk.com/ACD/ru/Detail/Index?id=6989112356543811151&appLang=en&os=Win32\_64 (дата обращения: 17.10.2022)
7. Обзор функционала SOLIDWORKS 2022 [Электронный ресурс] – Режим доступа: https://kb20.ru/articles/chto-novogo-v-solidworks-2022-beta/ (дата обращения: 17.10.2022)
8. UML-диаграммы классов [Электронный ресурс] – Режим доступа: http://www.uml.org/ (дата обращения: 17.10.2022)
9. Enterprise Architect (software) [Электронный ресурс] – Режим доступа: https://sparxsystems.com/ (дата обращения: 17.10.2022)
10. Интерфейс пользователя [Электронный ресурс] – Режим доступа: https://ru.wikipedia.org/wiki/Интерфейс\_пользователя (дата обращения: 17.10.2022)
11. Куликов С. Тестирование программного обеспечения. Базовый курс. 3-е издание, 301 с
12. Оперативная память [Электронный ресурс] – Режим доступа: https://ru.wikipedia.org/wiki/Оперативная\_память (дата обращения: 14.01.22)
13. Файл подкачки [Электронный ресурс] – Режим доступа: https://www.zeluslugi.ru/info-czentr/it-glossary/term-file-podkachki (дата обращения: 14.01.22)
14. Нагрузочное тестирование [Электронный курс] – Режим доступа: https://ru.wikipedia.org/wiki/Нагрузочное\_тестирование (Дата обращения: 14.01.22)