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 «Форма для льда»

ДЛЯ СИСТЕМЫ АВТОМАТИЗИРОВАННОГО ПРОЕКТИРОВАНИЯ

КОМПАС-3D

Пояснительная записка по дисциплине

«Основы разработки САПР»

Студент гр. 589-1

\_\_\_\_\_\_\_\_\_ Воротова А.А.

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

Руководитель

преподаватель каф., к.т.н.

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

(оценка) \_\_\_\_\_\_\_\_\_

Томск 2023

**Реферат**

Учебная работа 33 страницы, 9 таблиц, 19 рисунков, 16 источников.

Ключевые слова: КОМПАС-3D, ФОРМА ДЛЯ ЛЬДА, СИСТЕМА АВТОМАТИЗИРОВАННОГО ПРОЕКТИРОВАНИЯ, БИБЛИОТЕКА.

Целью данной работы является разработка библиотеки «Форма для льда» 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 Google documents.

**Оглавление**

[Введение](#_heading=h.gjdgxs)…………………………………………………………………………... 4

[1 ПОСТАНОВКА И АНАЛИЗ ЗАДАЧИ](#_heading=h.30j0zll)……………………………………... 5

[1.1 Описание предмета проектирования](#_heading=h.mglzzcwmkojc)………………………………………... 6

[1.2 Описание инструментов и средств реализации](#_heading=h.wcx64dvqbm05)…………………………….. 8

[1.3 Назначение библиотеки](#_heading=h.2et92p0)……………………………………………………… 9

[2 ОБЗОР АНАЛОГОВ](#_heading=h.kkxlon3dvauj)……………………………………………………………. 9

[2.1 Autodesk Inventor 2022](#_heading=h.2et92p0)……………………………………………………….. 9

[2.2 Solidworks 2022](#_heading=h.53oq32w8hknn)……………………………………………………………… 10

[3 ОПИСАНИЕ РЕАЛИЗАЦИИ](#_heading=h.2s8eyo1)………………………………………………… 15

[4 ОПИСАНИЕ ПРОГРАММЫ ДЛЯ ПОЛЬЗОВАТЕЛЯ](#_heading=h.17dp8vu)……………………. 15

[5 ТЕСТИРОВАНИЕ ПРОГРАММЫ](#_heading=h.3rdcrjn)………………………………………… 17

[5.1 Функциональное тестирование](#_heading=h.26in1rg)…………………………………………….. 17

[5.2 Модульное тестирование](#_heading=h.9jljrqe1zjll)…………………………………………………… 23

[5.3 Нагрузочное тестирование](#_heading=h.gab0lmiepbva)……………………………………………….. 24

[Заключение](#_heading=h.1ksv4uv)………………………………………………………………………. 27

[Список использованных источников](#_heading=h.44sinio)…………………………………………... 28

# 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 «Форма для льда» 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 формы для льда [2]. Также библиотека должна позволять изменять входные параметры формы.

Изменяемые параметры для плагина:

* Длина формы для льда;
* Ширина формы для льда;
* Глубина формы для льда;
* Толщина внешних стенок;
* Толщина внутренних перегородок;
* Толщина днища формы;
* Количество ячеек на ширину формы;
* Количество ячеек на длину формы.

# 1.1 Описание предмета проектирования

Предметом проектирования является форма для льда. На рисунке 1.1 представлен чертеж формы.

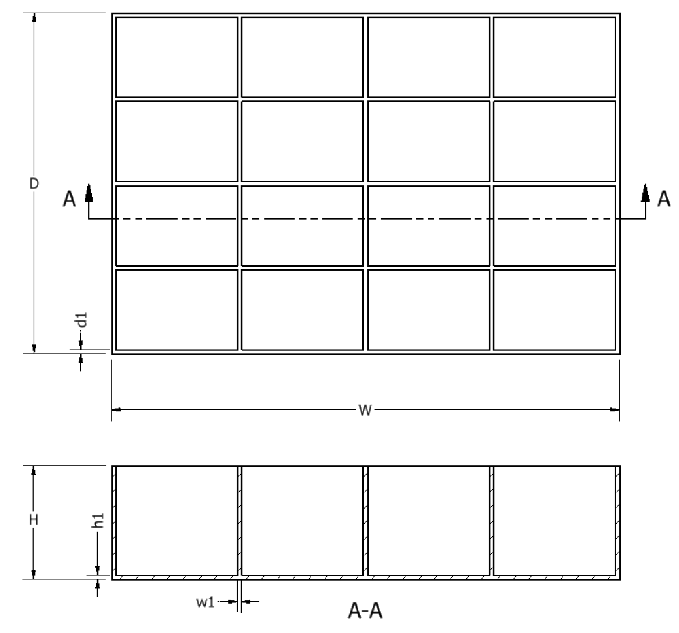


Рисунок 1.1 – чертеж формы для льда

Ниже на рисунке 1.2 представлена 3D-модель формы для льда

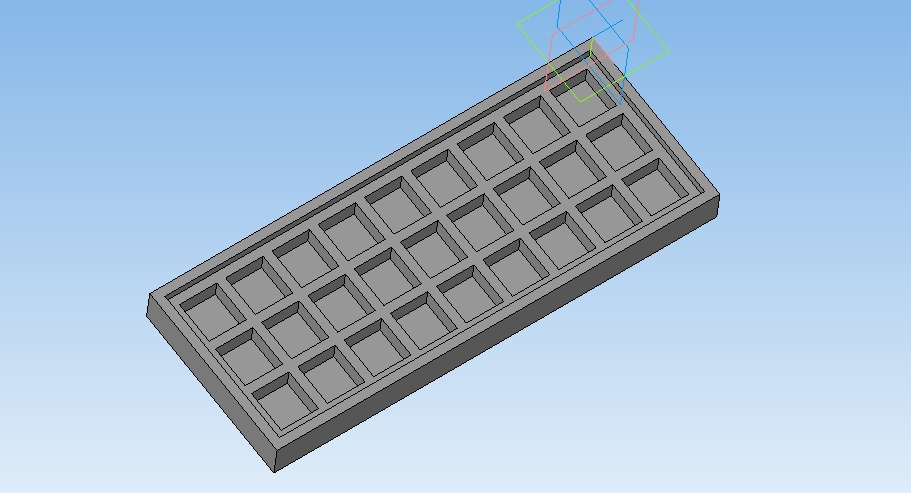


Рисунок 1.2 – 3D-модель кровати

Параметры формы:

* W – длинна формы (100 - 150 мм);
* D – ширина формы (250 - 350 мм);
* H – глубина формы (20 - 40 мм);
* d1 – толщина внешних стенок (5 - 10 мм);
* w1 – толщина внутренних перегородок (5 - 10 мм);
* h1 – толщина днища формы (5 - 10 мм);
* N1 – количество ячеек на ширину формы (2-4 шт);
* N2 – количество ячеек на длину формы (7-13 шт).

# 1.2 Описание инструментов и средств реализации

Для создания библиотеки используется среда разработки Visual Studio 2022 [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 «Форма для льда».

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 запуске моделирования с некорректными значениями программа выделяет цветом поле с недопустимым размером и не запускает построение модели.

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

# 2 ОБЗОР АНАЛОГОВ

# 2.1 Autodesk Inventor 2022

Данная программа предоставляет указанные ниже возможности 3D модели в Autodesk Inventor 2022:

* Состояния модели;
* Свойства экземпляра: назначение свойств отдельным экземплярам компонентов;
* Усовершенствования тонированного вида: использование стиля освещения модели на чертеже;
* Параметры вида чертежа: извлечение вида камеры и 3D-аннотаций;
* Форматирование текста: объединены элементы управления диалогового окна.

На рисунке 1.1 представлен пользовательский интерфейс программы «Inventor» для построения детали.

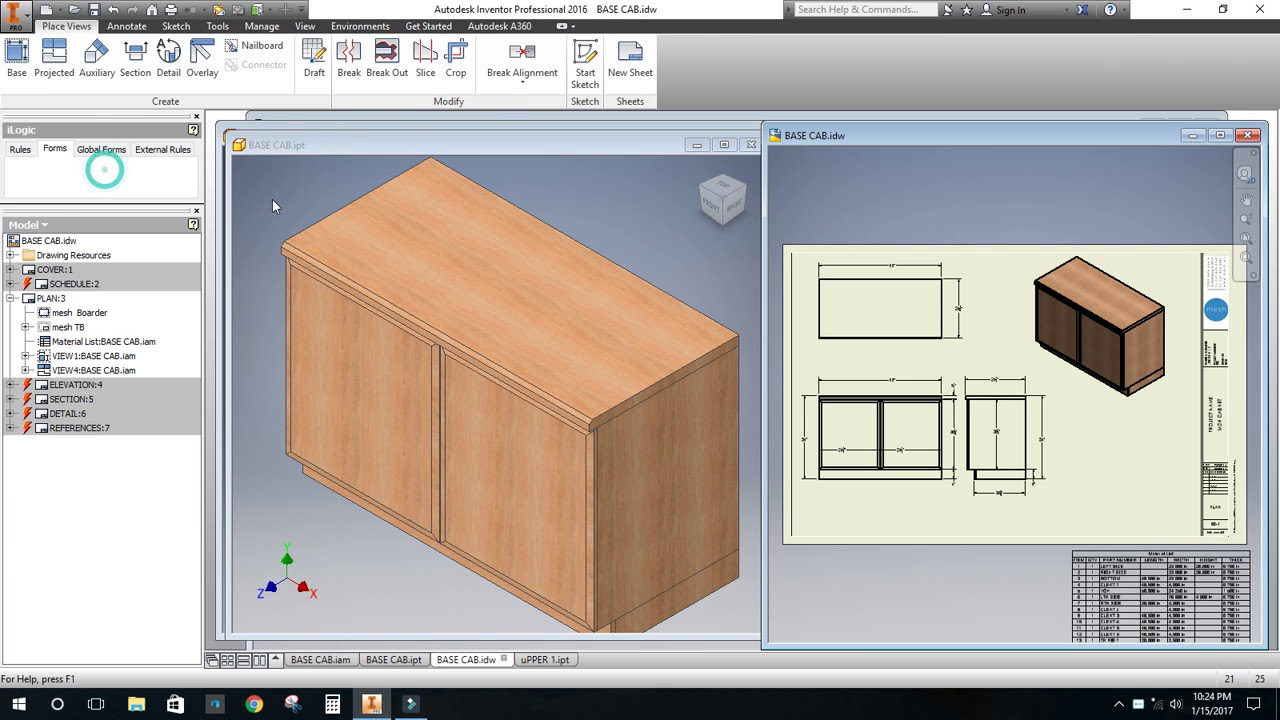


Рисунок 2.1 – Пользовательский интерфейс программы «Inventor» для построения ящика

# 2.2 Solidworks 2022

[SOLIDWORKS](https://my.solidworks.com/try-solidworks?lang=ru&utm_campaign=202112_rus_swks_TrySW_ru_terr&utm_medium=socialnetwork&utm_source=habr&utm_content=txt) - программное обеспечение, позволяющее решить целый комплекс задач, возникающих на производстве.

Инструментарий SOLIDWORKS позволяет не только сделать общий дизайн модели, но и:

– Провести детальное тестирование в виртуальной среде

– Доработать отдельные узлы

– Конвертировать модель в чертежи для создания реального прототипа

Таким образом, работа с проектом от стадии идеи до готового продукта осуществляется в рамках одного программного пакета. Поэтому SOLIDWORKS находит применение в самых разных отраслях.

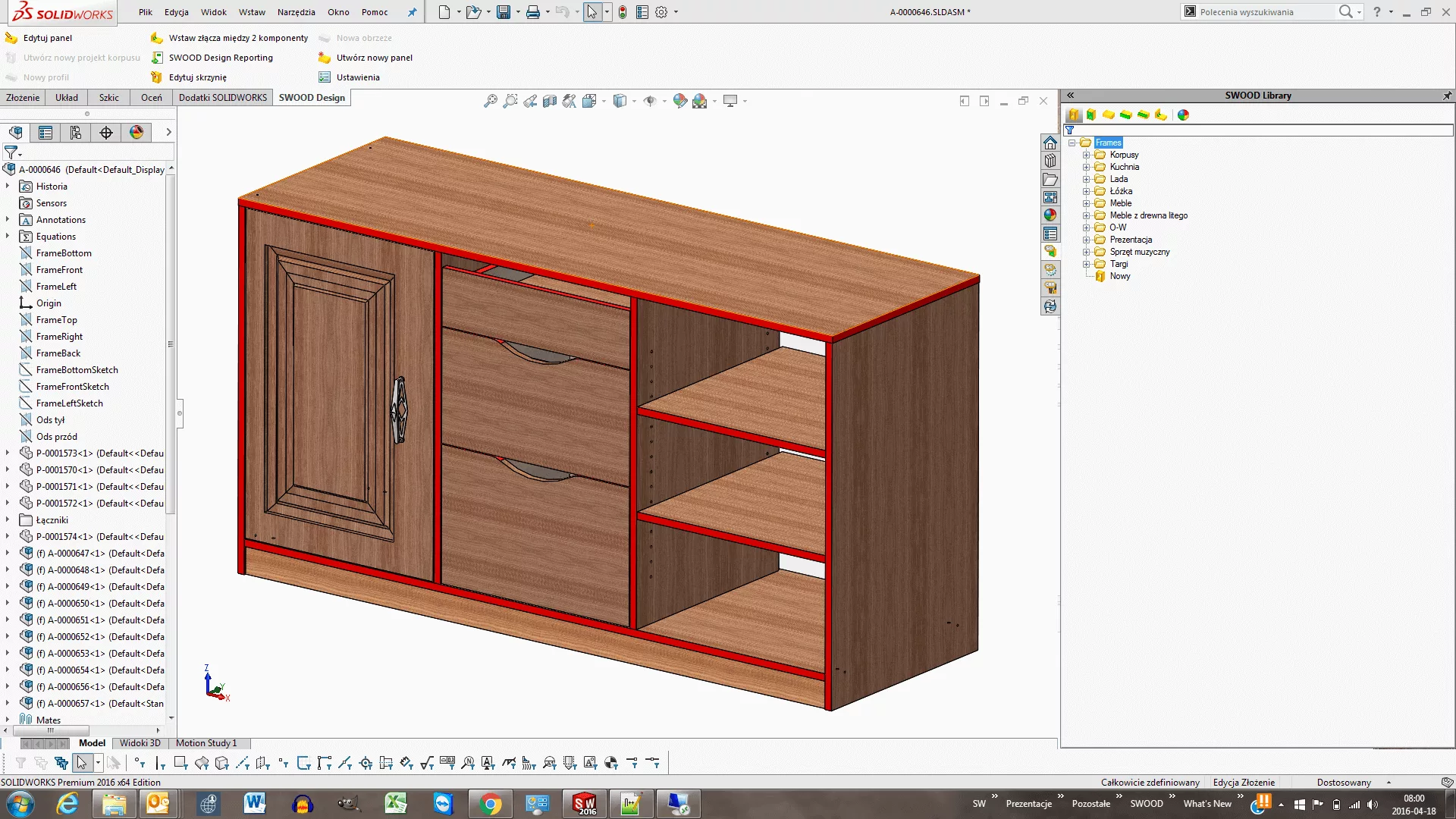


Рисунок 2.2 – Пользовательский интерфейс программы «Inventor» для построения ящика

# 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 [9].

**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 [10].

Класс Parameter предназначен для описания параметров кровати и их валидации, класс BedBuilder предназначен для построения кровати, класс BedForm предназначен для описания пользовательского интерфейса, перечисление ParameterType необходимо для валидации параметров и корректной связи их с BedForm, класс KompasWrapper предназначен для взаимодействия с САПР «Компас-3D», в нём содержатся методы для связи с САПР и рисования в ней.

На рисунке 3.1 представлена диаграмма UML для формы.

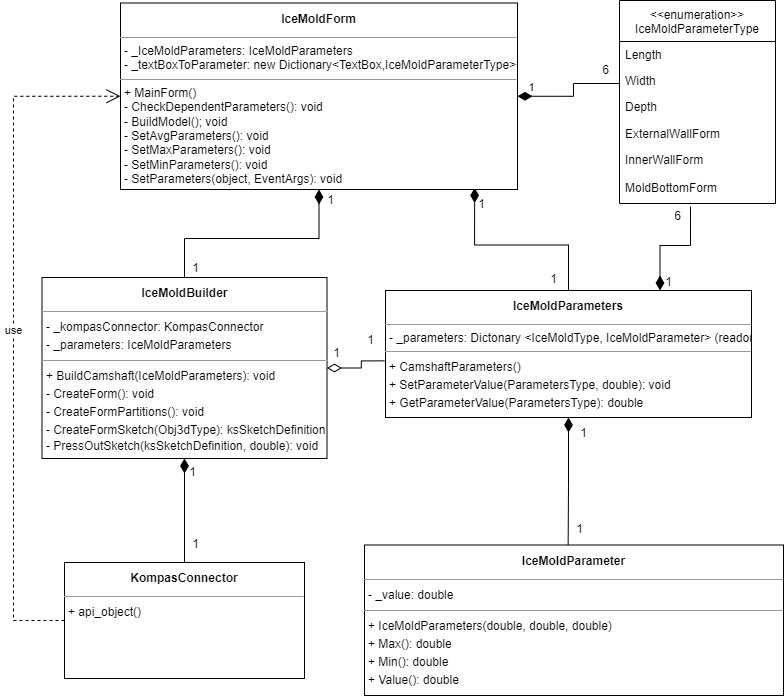


Рисунок 3.1 – Исходная UML – диаграмма классов

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

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| + \_parameters |  | Хранит в себе набор методов для построения формы для льда |
| - \_textBoxToParameter |  | Хранит в себе набор TextBox и соответствующий ему тип параметра из перечисления «IceMoldParametersType» |

Окончание таблицы 3.1 – Описание полей, методов, сущностей класса «IceMoldForm»

|  |  |  |
| --- | --- | --- |
| - 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 форму по заданным параметрам |

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

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

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

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| -\_value |  | Хранит текущее значение |
| * IceMoldParameter(double, double, double) |  | Конструктор для создания экземпляра класса |
| + Max() | double | Возвращает максимальное допустимое значение параметра |
| + Min() | double | Возвращает минимальное допустимое значение параметра |
| + Value() | double | Возвращает текущее значение параметра. Задает новое значение параметра |

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

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| * kompasConnector:KompasConnector |  | Хранит в себе методы необходимые для связи с КОМПАС 3D |
| * \_parameters: IceMoldParameters |  | Хранит данные о каждом параметре модели из перечисления «IceMoldParameter» |
| + IceMoldBuild(IceMoldParameters) | void | Построение формы для льда по заданным параметрам |
| * CreateForm() | void | Построение основы формы |
| * CreateFormPartitions() | void | Построение перегородок |
| - CreateFormSketch(obj3dType) | ksSketchDefinition | Возвращает интерфейс параметров эскиза |
| -PressOutSketch(ksSketchDefinition, double) | void | Выдавливает эскиз |

В ходе реализации библиотеки многие классы были пересмотрены и в них были добавлены новые методы, поля, свойства. Добавились минимальные и максимальные значения, новые методы для постройки формы для льда.

Итоговая UML – диаграмма представлена на рисунке 3.2 и описана ниже в таблицах.

Рисунок 3.2 – Итоговая UML – диаграмма классов

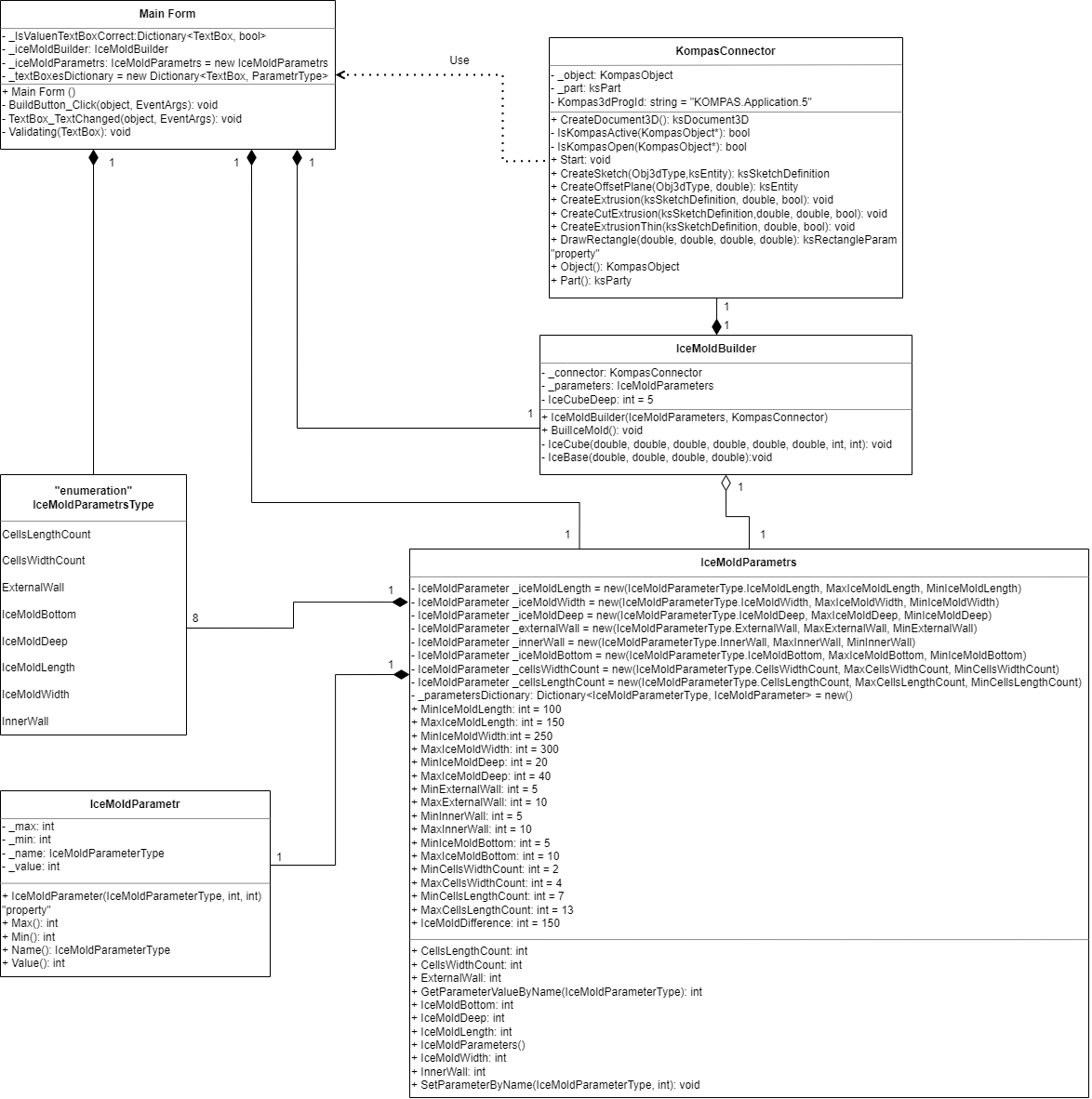


Таблица 3.5 – Описание полей, методов, сущностей класса «MainForm»

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| \_iceMoldBuiler | IceMoldBuilder | Объект класса построителя |
| \_iceMoldParameters | IceMoldParameters | Объект класса с параметрами |
| \_textBoxesDictionary | Dictionary<TextBox, IceMoldParameterType> | Словарь, содержащий пары (Текст-боксы, имя параметра) |
| \_isValueInTextBoxCorrect | Dictionary<TextBox, bool> | Словарь содержащие пары (Текст-бокс, корректное ли значение в нём) |
| MainForm |  |  |
| BuildButton\_Click | void | Обработчик нажатия кнопки "Построить" |
| TextBox\_TextChanged | void |  |
| Validating | void | Общий метод валидации текстбокса |

Таблица 3.6 – Описание полей, методов, сущностей класса «IceMoldParametr»

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

Окончание таблицы 3. 6 – Описание полей, методов, сущностей класса «IceMoldParametr»

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

Таблица 3.7 – Описание полей, методов, сущностей класса «KompasConnector»

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| \_object | KompasObject | Объект интерфейса API КОМПАС-3D |
| \_part | ksPart | Компонент сборки |
| Kompas3dProgId | string | Строковое наименование идентификатора COM-объекта |
| Part | ksPart | Свойство возвращающее компонент сборки |
| Start | void | Метод начала работы КОМПАС-3D |
| IsKompasActive | bool | Делает окно КОМПАС-3D активным |

Окончание таблицы 3.7 – Описание полей, методов, сущностей класса «KompasConnector»

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| IsKompasOpen | bool | Метод запускает КОМПАС-3D |
| CreateDocument3D | ksDocument3D | Запускает окно создания 3D-модели |
| DrawRectangle | ksRectangleParam | Метод рисования прямоугольника |
| СreateCutExtrusion | Void | Метод осуществляющий вырезание |
| CreateSketch | ksSketchDefinition | Метод создающий эскиз |
| СreateExtrusion | void | Метод осущетсвляющий выдавливание |
| СreateExtrusionThin | void | Метод осущетсвляющий выдавливание с стенками |
| CreateOffsetPlane | ksEntity | Метод смещающий плоскость |

Таблица 3.8 – Описание полей, методов, сущностей класса «IceMoldBuilder»

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| \_connector | KompasConnector | Объект класса конектора для связи с КОММПАС-3D |
| \_parameters | IceMoldParameters | Объект класса параметра для построение детали |
| IceCubeDeep | int | Глубина для формы |
| IceMoldBuilder |  | Конструктор класса |
| BuilIceMold | Void | Метод строящий форму для льда |

Окончание таблицы 3.8 – Описание полей, методов, сущностей класса «IceMoldBuilder»

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| IceCube | void | Метод осуществляющий построения отверстий для льда |
| IceBase | void | Метод осуществляющий построение основания для формы |

Таблица 3.9 - Описание полей, методов, сущностей класса «IceMoldParameters»

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| \_iceMoldLength | IceMoldParameter | Длина формы |
| \_iceMoldWidth | IceMoldParameter | Ширина формы |
| \_iceMoldDeep | IceMoldParameter | Глубина формы |
| \_externalWall | IceMoldParameter | Толщина внешних стенок |
| \_innerWall | IceMoldParameter | Толщина внутренних стенок |
| \_iceMoldBottom | IceMoldParameter | Толщина днища формы |
| \_cellsWidthCount | IceMoldParameter | Количество ячеек на ширину формы |
| \_cellsLengthCount | IceMoldParameter | Количество ячеек на длину формы |
| Dictionary<IceMoldParameterType, IceMoldParameter> | int | Словарь, содержащий пары (Имя параметра, указатель на него) |
| IceMoldLength | Int | Задаёт или возвращает общую длину формы |

Окончание таблицы 3.9 - Описание полей, методов, сущностей класса «IceMoldParameters»

|  |  |  |
| --- | --- | --- |
| Название метода/поля | Возвращаемый тип | Описание |
| IceMoldWidth | int | Задаёт или возвращает ширину формы |
| IceMoldDeep | int | Задаёт или возвращает глубину формы |
| ExternalWall | int | Задаёт или возвращает толщину внешней стенки |
| InnerWall | int | Задаёт или возвращает толищну внутренней стенки |
| IceMoldBottom | int | Задаёт или возвращает толщину днища |
| CellsWidthCount | int | Задаёт или возвращает количество ячеек на ширину формы |
| CellsLengthCount | int | Задаёт или возвращает количество ячеек на длину формы |
| IceMoldParameters |  | Конструктор класса с минимальными значенми по умолчанию |
| SetParameterByName | void | Метод передающй значение в сеттер параметра по имени |
| GetParameterValueByName | int | Метод возвращающий значение параметра по имени |

# 4 ОПИСАНИЕ ПРОГРАММЫ ДЛЯ ПОЛЬЗОВАТЕЛЯ

Пользовательский интерфейс – интерфейс, обеспечивающий передачу информации между пользователем – человеком и программно-аппаратными компонентами компьютерной сети [11].

Плагин представляет собой пользовательскую форму с полями для ввода соответствующих параметров (рисунок 4.1). Посредством кнопки «BUILD» осуществляется запуск САПР «Компас-3D», на рабочей области которой строится трехмерная модель по заданным параметрам. Если построение модели 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 красный цвет, информируя пользователя о не корректности введенного значения. Также, при изменении параметра “Длина формы” изменяться параметр “Ширина формы” и наоборот.

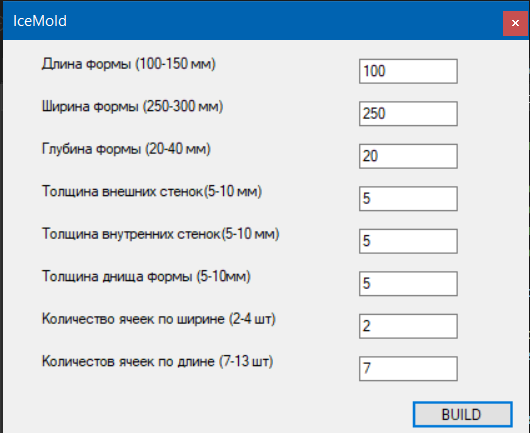


Рисунок 4.1 – Макет пользовательского интерфейса

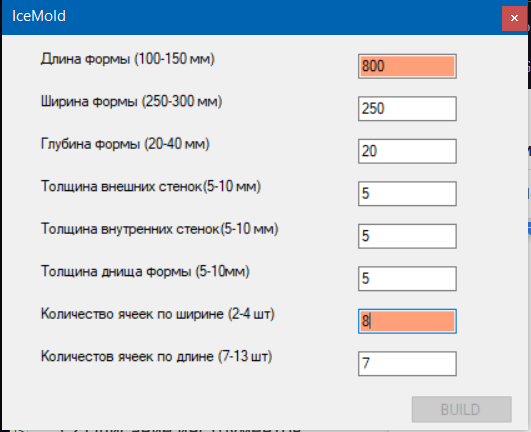


Рисунок 4.2 – Реакция приложения на ввод некорректных значений

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

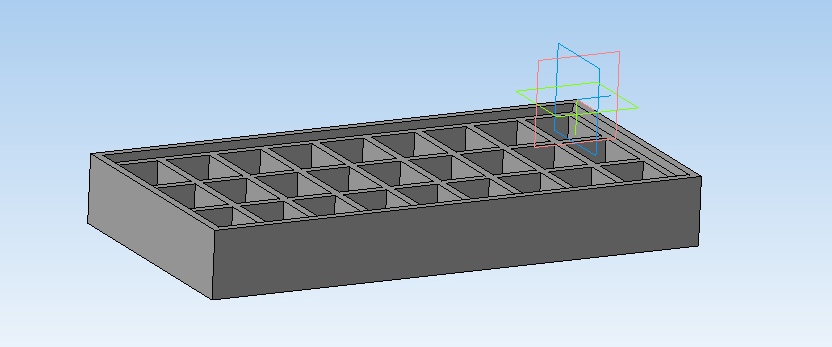


Рисунок 4.3 – Трёхмерная модель формы для льда

# 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 [12].

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

Тестирование при минимальных параметрах:

* W – длинна формы (100 мм);
* D – ширина формы (250 мм);
* H – глубина формы (20 мм);
* d1 – толщина внешних стенок (5 мм);
* w1 – толщина внутренних перегородок (5 мм);
* h1 – толщина днища формы (5 мм);
* N1 – количество ячеек на ширину формы (2 шт);
* N2 – количество ячеек на длину формы (7 шт).

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

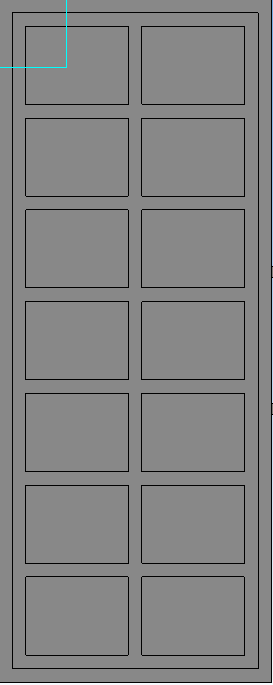


Рисунок 5.1 – Вид на модель снизу

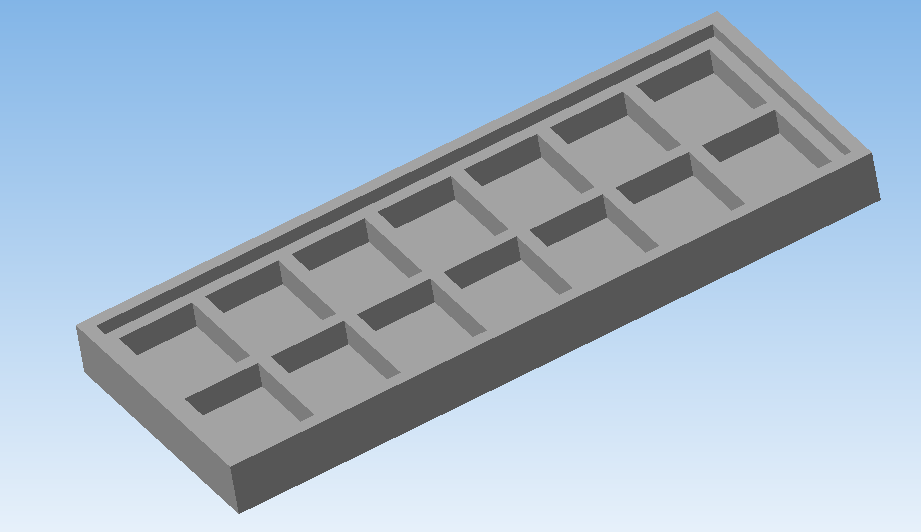


Рисунок 5.2 – Вид на модель сбоку

Тестирование при средних параметрах:

* W – длинна формы (125 мм);
* D – ширина формы (300 мм);
* H – глубина формы (30 мм);
* d1 – толщина внешних стенок (7 мм);
* w1 – толщина внутренних перегородок (7 мм);
* h1 – толщина днища формы (7 мм);
* N1 – количество ячеек на ширину формы (3 шт);
* N2 – количество ячеек на длину формы (10 шт).

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

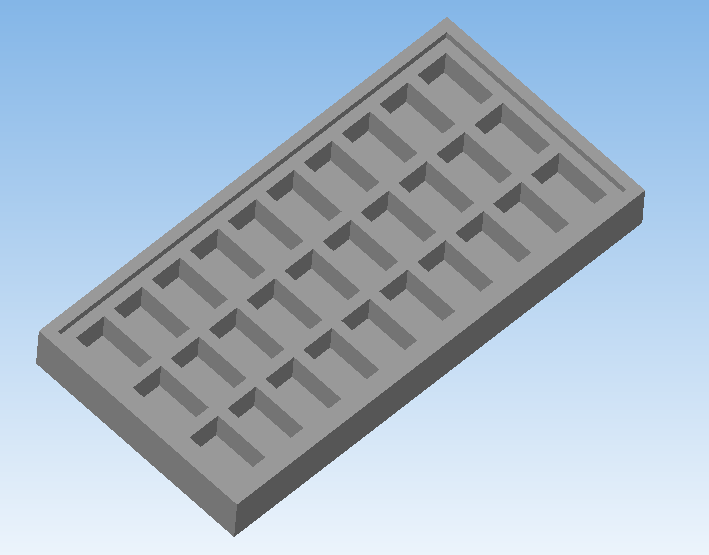


Рисунок 5.3 – Вид на модель сбоку

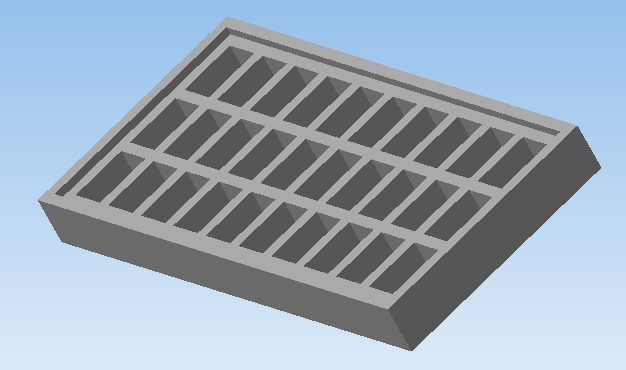


Рисунок 5.4 – Вид на модель сбоку

Тестирование при максимальных параметрах:

* W – длинна формы (150 мм);
* D – ширина формы (350 мм);
* H – глубина формы (40 мм);
* d1 – толщина внешних стенок (10 мм);
* w1 – толщина внутренних перегородок (10 мм);
* h1 – толщина днища формы (10 мм);
* N1 – количество ячеек на ширину формы (4 шт);
* N2 – количество ячеек на длину формы (13 шт).

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

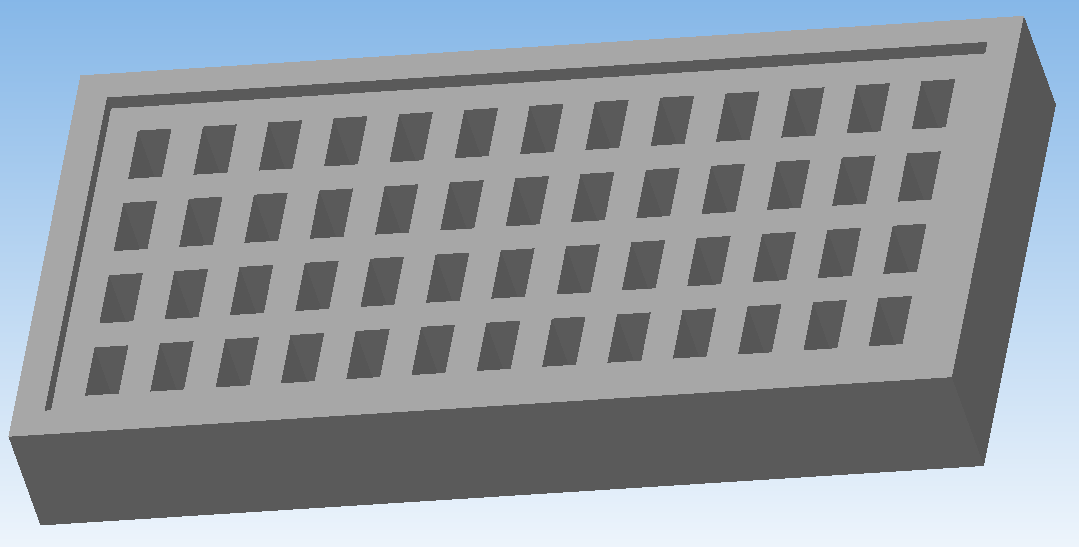


Рисунок 5.5 – Вид на модель сбоку

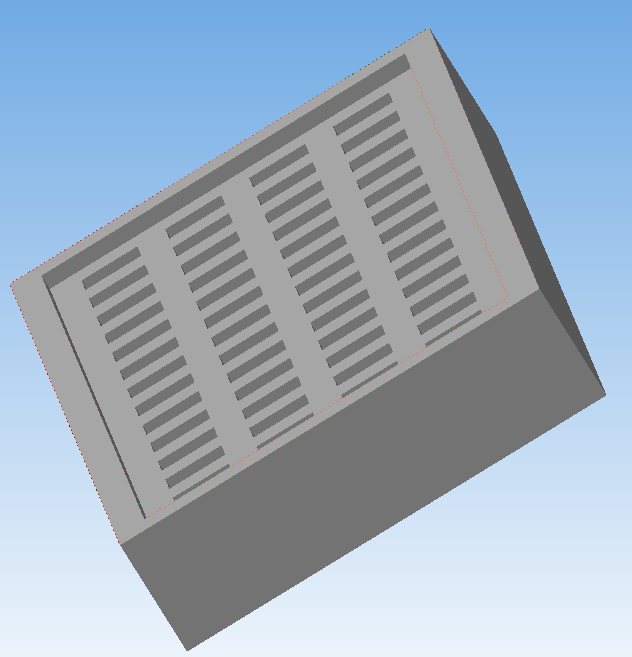


Рисунок 5.6 – Вид на модель сбоку

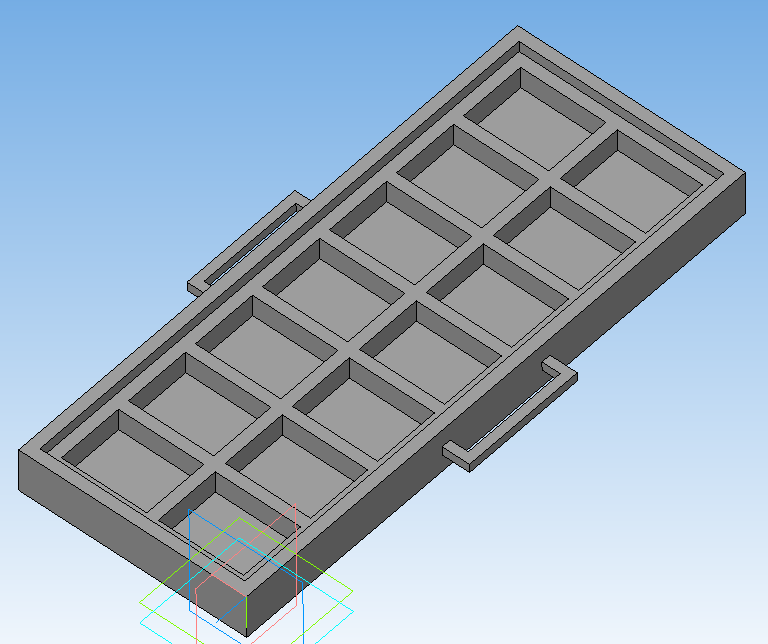


Рисунок 5.7 – Вид сбоку на форму для льда с ручками при стандартных параметрах

# 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 10 тестов. На рисунках 5.7 и 5.8 представлено тестирование логики.

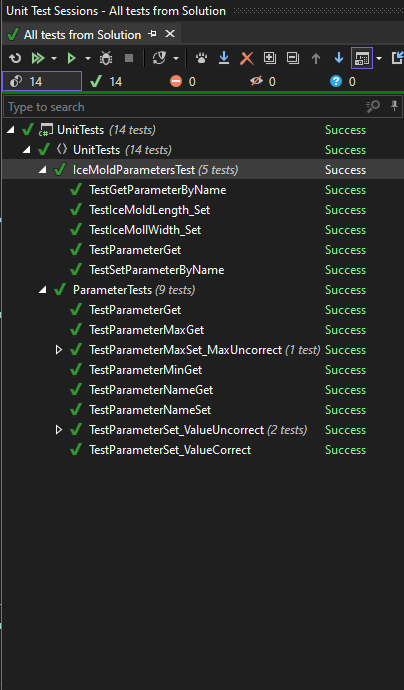


Рисунок 5.7 – Тестирование логики программы

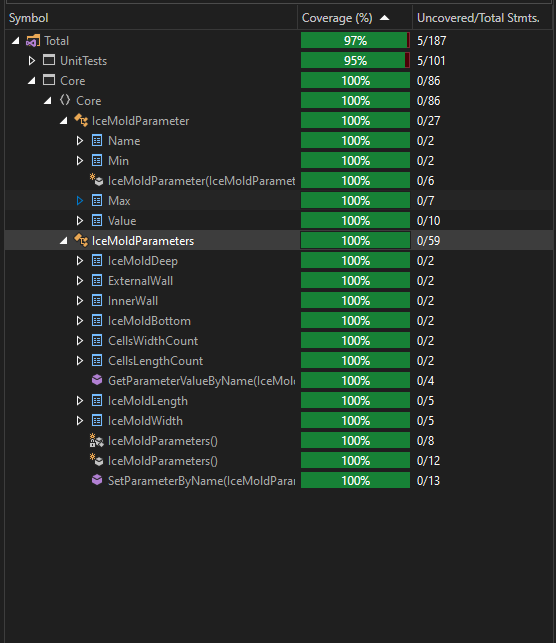


Рисунок 5.8 – Тестирование логики программы

# 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 AMD RYZEN 5 2500U;
* оперативная память 16.0 ГБ;
* видеокарта AMD RX560.

Для тестирования будет зациклено построение модели со следующими параметрами:

* длина формы – 100 мм;
* ширина формы – 250 мм;
* глубина формы – 20 мм;
* толщина внешних стенок – 5 мм;
* толщина внутренних перегородок – 5 мм;
* толщина днища формы – 5 мм;
* количество ячеек на ширину формы – 2 шт;
* количество ячеек на длину формы – 7 шт.

Первое тестирование заключалось в построении детали с базовыми параметрами. Тестирование проводилось 14 минут и 35 секунд, построено 117 моделей. На протяжении тестирования загруженность процессора составляла около 87 процентов. На рисунке 5.13 показана зависимость количества используемой оперативной памяти в гигабайтах от количества построенных деталей.

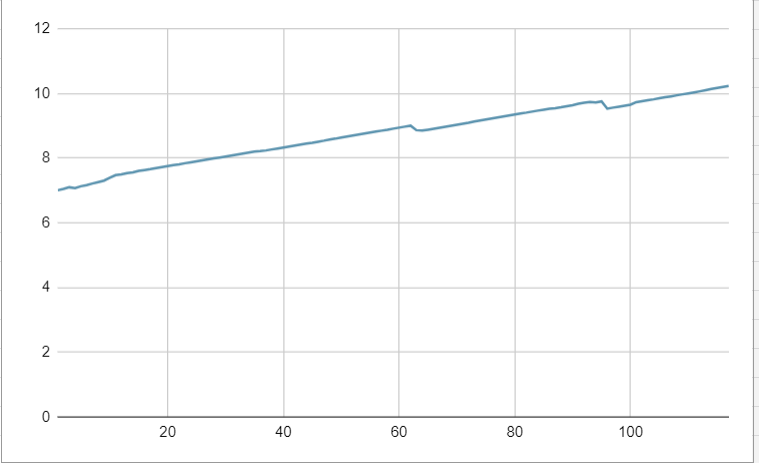


Рисунок 5.13 – Зависимость оперативной памяти от количества деталей

Из графика видно, что зависимость является линейной пока оперативная память не загружена практически полностью. На графике 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 снижает нагрузку [12].

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

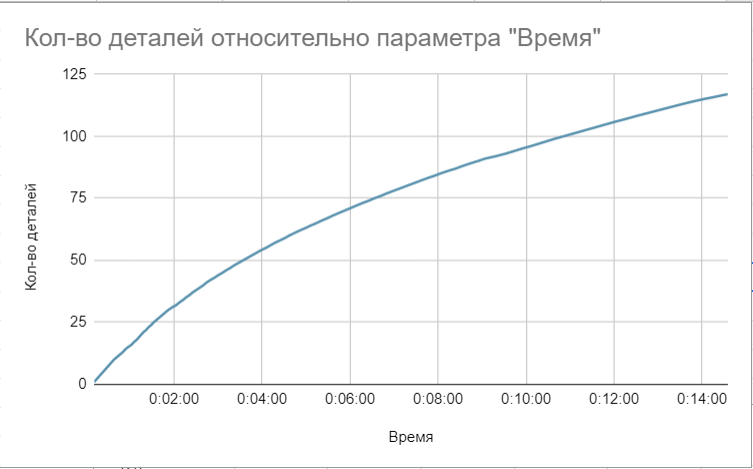


Рисунок 5.14 – Зависимость времени от количества построенных деталей

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: 30.09.2022)
2. Компас (САПР) [Электронный ресурс] – Режим доступа: https://ru.wikipedia.org/wiki/Компас\_(САПР) (дата обращения: 30.09.2022)
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: 19.11.2022)
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: 19.11.2022)
5. Фреймворк NUnit [Электронный ресурс] – Режим доступа: https://nunit.org/ (дата обращения: 19.11.2022)
6. Плагин автоматического построения 3D моделей SketchUp “OpenCutList”.[Электронный ресурс] – Режим доступа: https://docs.opencutlist.org (дата обращения: 07.10.2022)
7. Blender manual Add-ons // Blender 3.3 manual URL: <https://docs.blender.org/manual/en/latest/editors/preferences/addons.html> (дата обращения: 20.10.2022).
8. UML-диаграммы классов [Электронный ресурс] – Режим доступа: http://www.uml.org/ (дата обращения: 06.10.2022)
9. Enterprise Architect (software) [Электронный ресурс] – Режим доступа: https://sparxsystems.com/ (дата обращения: 06.10.2022)
10. Интерфейс пользователя [Электронный ресурс] – Режим доступа: https://ru.wikipedia.org/wiki/Интерфейс\_пользователя (дата обращения: 15.12.2022)
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: 15.12.2022)
13. Файл подкачки [Электронный ресурс] – Режим доступа: https://www.zeluslugi.ru/info-czentr/it-glossary/term-file-podkachki (дата обращения: 15.12.2022)