Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ(ТУСУР)

Кафедра компьютерных систем в управлении и проектировании (КСУП)

РАЗРАБОТКА ПЛАГИНА " Корпус для компьютера " ДЛЯ САПР КОМПАС-3D V20

Проект системы по лабораторному проекту

по дисциплине «ОСНОВЫ РАЗРАБОТКИ САПР»

Выполнил:

студент гр. 588-2

\_\_\_\_\_\_\_\_\_\_\_\_\_\_Третьяков В.В.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Руководитель:

к.т.н., доцент каф. КСУП

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

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

# 1 Описание САПР

# Описание программы

САПР (Система автоматизированного проектирования) — автоматизированная система, реализующая информационную технологию выполнения функций проектирования, представляет собой организационно техническую систему, предназначенную для автоматизации процесса проектирования, состоящую из комплекса технических, программных и других средств автоматизации его деятельности. [1].

Компас — 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 [2].

# Описание API

API (англ. Application Programming Interface) – логический интерфейс для программного компонента, скрывающий внутренние детали, отвечающие за реализацию[4].

Взаимодействие внешнего приложения или подключаемого модуля с системой КОМПАС осуществляется посредством программных интерфейсов, называемых API. В КОМПАС на данный момент существуют API двух версий: API 5 и API 7 [3].

Главным интерфейсом API системы КОМПАС является KompasObject. Получить указатель на этот интерфейс можно с помощью экспортной функции CreateKompasObject(). Методы этого интерфейса реализуют наиболее общие функции работы с документами системы, системными настройками, файлами, а также дают возможность получить указатели на другие интерфейсы. В таблице 1.1 приведены методы интерфейса KompasObject [3].

Таблица 1.1 – Методы интерфейса KompasObject

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Тип входных параметров | Возвращаемое значение | Описание |
| Document3D() |  | Указатель на интерфейс документа трехмерной модели ksDocument3D | Даёт возможность получить указатель на интерфейс трёхмерного документа (детали или сборки) |
| ActiveDocument3D() |  | Указатель на интерфейс документа трехмерной модели ksDocument3D | Дает возможность получить указатель на активный трехмерный документ |
| GetParamStruct() | structType – тип интерфейса параметров | Указатель на интерфейс указанного типа из StructType2D | Метод для получения указателя на интерфейс графического документа (чертежа или фрагмента) |

Продолжение таблицы 1.1

|  |  |  |  |
| --- | --- | --- | --- |
| Visible |  | bool | Свойство видимости приложения |
| GetMathematic2D() |  | Указатель на интерфейс ksMathematic2D | Метод для получения указателя на интерфейс для работы с математическими функциями |

Графические документы имеют собственный интерфейс – ksDocument2D, со своими специфическими свойствами и методами. С помощью функций, присутствующих в ksDocument2D, создаются изображения в эскизах трехмерных операций. Свойства (члены данных) этого интерфейса позволяют динамически управлять настройками любого трехмерного документа системы из модуля. Наиболее используемые из них приведены в таблице 1.2.

Таблица 1.2 – Методы интерфейса ksDocument3D

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Тип входных параметров | Тип возвращаемых данных | Описание |
| Create () | invisible-признак режима редактирования документа (TRUE-невидимый режим, FALSE – видимый режим), typeDoc – тип документа | TRUE – в случае успешного завершения | Дает возможность создать пустой документ (деталь или сборку) |
| GetPart() | Type – тип компонента из перечисления | Указатель на интерфейс компонента ksPart или IPart | Позволяет получить указатель на интерфейс компонента в сборке |
| UpdateDocumentParam() |  | TRUE – в случае успешного завершения. | Позволяет обновить настройки документа |

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

Таблица 1.3 – Методы интерфейса ksPart.

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Тип входных параметров | Тип возвращаемых данных | Описание |
| EntityCollection() | objType – тип объектов | В случае успеха указатель на интерфейс ksEntityCollection или IEntityCollection, в случае неудачи – NULL. | Формирует динамический массив трехмерных объектов и возвращает указатель на его интерфейс |
| GetPart() | type – тип компонента из перечисления | Указатель на интерфейс компонента ksPart или IPart | Получить указатель на интерфейс компонента |
| NewEntity() | objType – тип объекта. | Указатель на интерфейс ksEntity или IEntity. | Создает интерфейс нового трехмерного объекта и возвращает указатель на него |
| GetDefaultEntity() | objType – тип объекта. | Указатель на интерфейс ksEntity или IEntity. | Возвращает указатель на интерфейс объекта, создаваемого системой в трехмерном документе по умолчанию |

# Обзор аналогов

**Edelws онлайн конфигуратор**

Edelws конфигуратор — инструмент для выбора и создания системного блока из представленных на сайте комплектующих [5].

Данный конфигуратор содержит множество всевозможных комплектующих, разделенных на группы. Пользователи могут создавать собственные сборки и сохранять под своей учетной записью. При добавлении комплектующих, справа отображается общая цена всех комплектующих.

Интерфейс конфигуратора представлен на рисунке 1.1

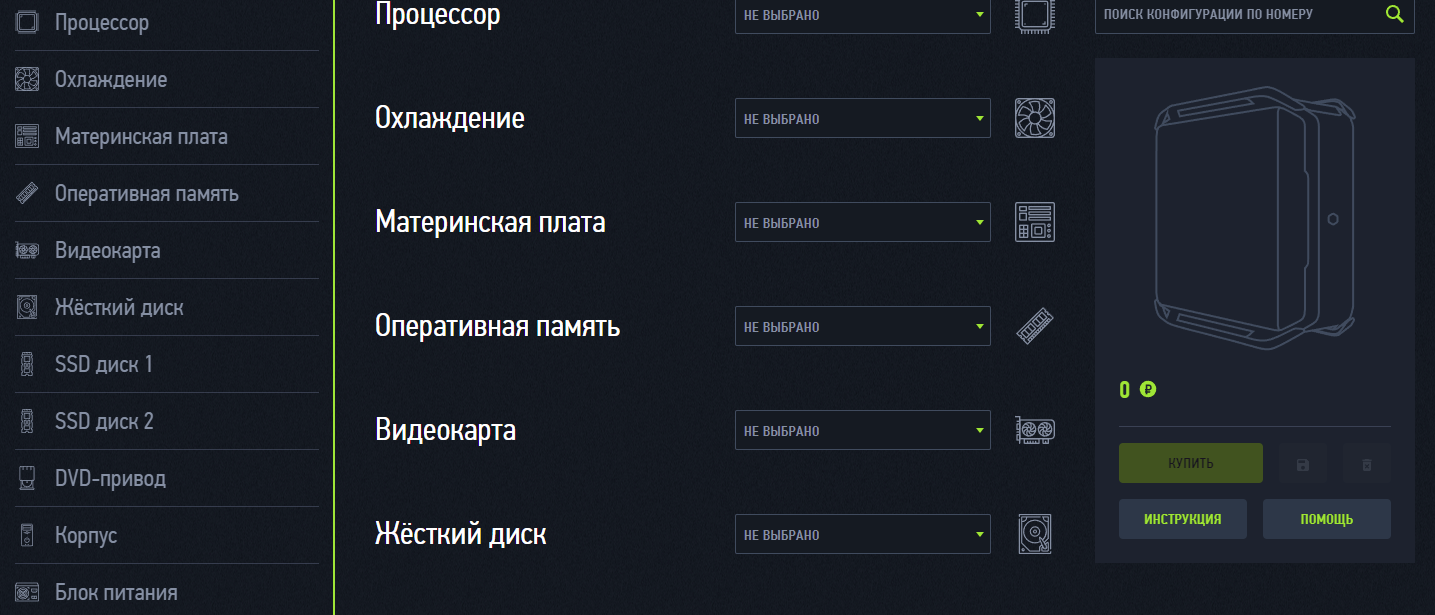


Рисунок 1.1 – Интерфейс конфигуратора Edelws

# Описание предмета проектирования

Компьютерный корпус (англ. computer case) — физически представляет собой базовую несущую конструкцию (шасси), которая предназначена для последующего наполнения аппаратным обеспечением с целью создания компьютера.

Модель компьютерного корпуса представлена на рисунке 2.1.

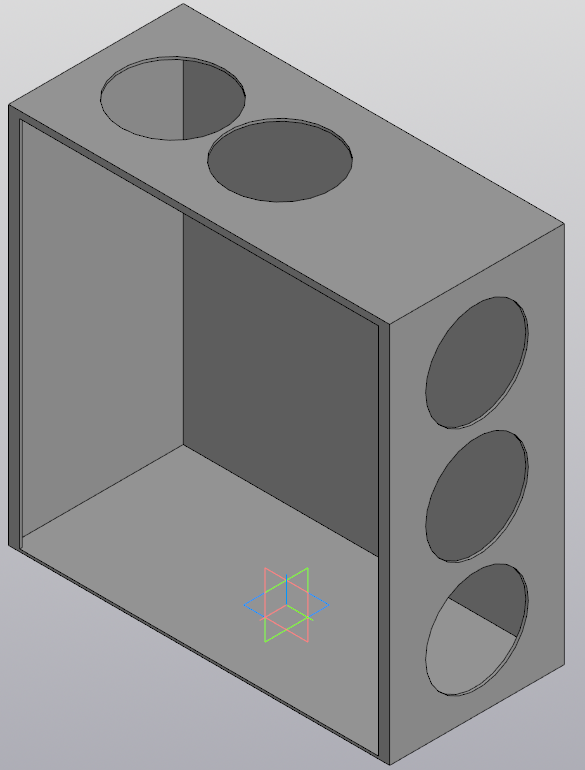


Рисунок 2.1– Компьютерный корпус

Параметры компьютерного корпуса (рисунок 2.2):

* высота корпуса A (330мм-500мм);
* длинна корпуса B (244мм-500мм);
* ширина корпуса C (140мм-250мм);
* диаметр передних отверстий для вентиляторов E (40мм-140мм);
* диаметр верхних отверстий для вентиляторов D (40мм-140мм);
* тип материнской платы (ATX / micro-ATX).

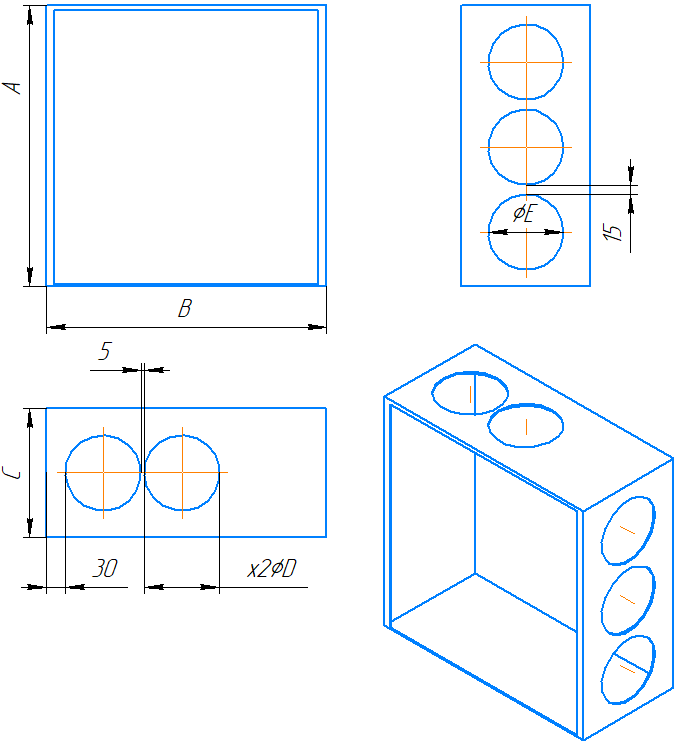


Рисунок 2.2 – Параметры компьютерного корпуса

Ограничения:

* диаметр отверстий под вентиляторы не должен быть больше ширины корпуса;
* диаметр верхних вентиляторов не должен быть больше длинны корпуса;
* диаметр передних вентиляторов не должен быть больше высоты корпуса.

# Проект программы

# Описание технических и функциональных аспектов проекта

Для графического описания абстрактной модели проекта, а также пользовательского взаимодействия (сценарии действия) использован стандарт UML.

UML язык графического описания для объектного моделирования в области разработки программного обеспечения. UML является языком широкого профиля, это – открытый стандарт, использующий графические обозначения для создания абстрактной модели системы, называемой UML – моделью. UML был создан для определения, визуализации, проектирования и документирования, в основном, программных систем. UML не является языком программирования, но на основании UML возможна генерация кода и наоборот [3].

При использовании UML была простроена диаграмма классов.

# Диаграмма классов

Диаграмма классов – структурная диаграмма языка моделирования UML, демонстрирующая общую структуру иерархии классов системы и их взаимосвязи между ними, их коопераций, атрибутов (полей), методов [3].

На рисунке 3.2 представлена диаграмма классов.

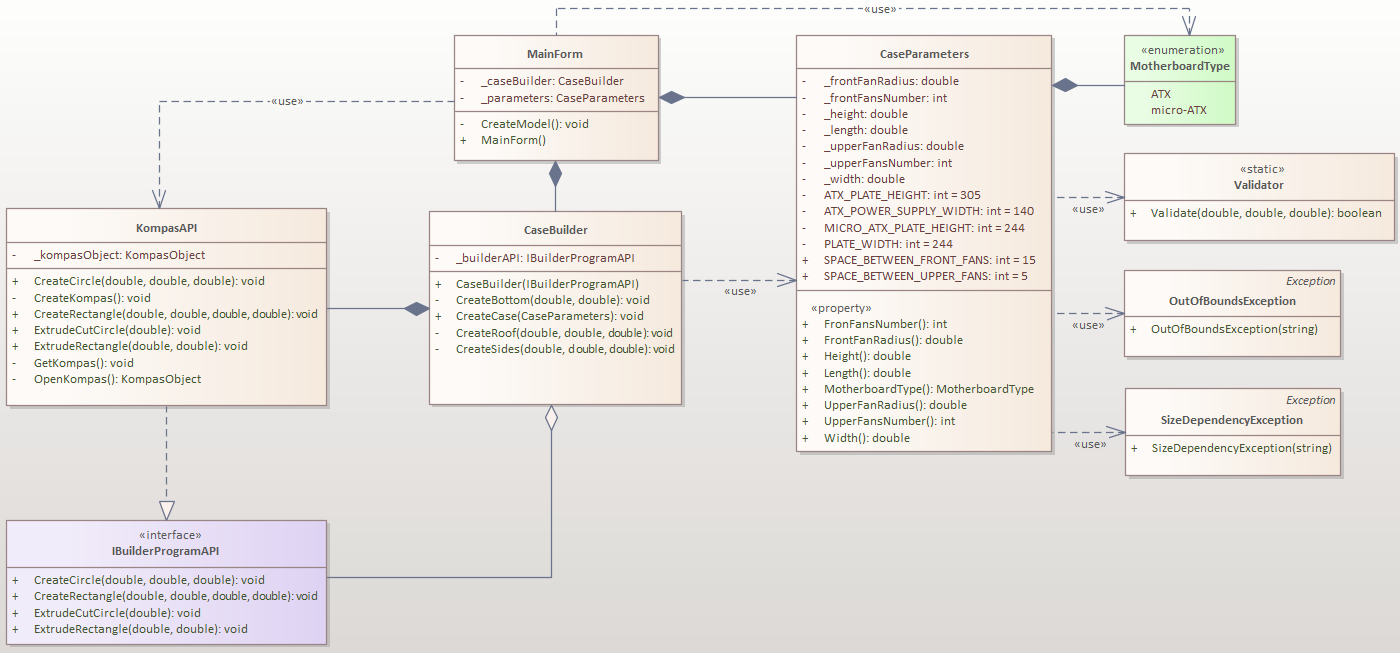


Рисунок 3.2 – Диаграмма классов

Класс MainForm содержит в себе экземпляры классов CaseParameters и CaseBuilder. CaseParameters содержит в себе данные для построения корпуса, перед присвоением параметров, данные проходят проверку используя статический класс Validator.

При попытке занесения данных в поля CaseParameters, вызывается статический метод Validate, принимающий два ограничения и само число, которое необходимо проверить. При неудачной проверке, в зависимости от проверяемого поля, генерируется исключение. OutOfBoundException – если число не входит в указанный диапазон. SizeDependencyException – если ошибка возникла в соотношениях зависимых полей.

Класс KompassApi, реализует интерфейс IBuilderProgramAPI для работы с САПР Компас3Д и построения в нем необходимых элементов. В перечислении ParameterName хранятся именования параметров.

Таблица 3.1 – Класс CaseBuilder

|  |  |  |
| --- | --- | --- |
| Название | Тип входных параметров | Описание |
| CaseBuilder() | IBuilderProgramAPI – экземпляр объекта класса, реализующего данный интерфейс | Конструктор класса CaseBuilder |
| CreateBotom() | double – ширина корпуса  double – длина корпуса | Вызывает методы необходимые для создания дна корпуса, используя объект builderAPI |
| CreateRoof() | double – ширина корпуса;  double – длина корпуса;  int – количество вентиляторов | Вызывает методы необходимые для создания крыши корпуса с отверстиями под вентиляторы, используя объект builderAPI |
| CreateSides() | double – ширина корпуса  double – высота корпуса  double – длинна корпуса  int – количество вентиляторов | Вызывает методы необходимые для создания стенок корпуса с отверстиями под вентиляторы на передней стороне, используя объект builderAPI |

Таблица 3.2 – Класс KompasAPI

|  |  |  |
| --- | --- | --- |
| Название | Тип входных параметров | Описание |
| CreateCircle() | double – диаметр отверстия  double – центр по Х координате  double – центр по Y координате | Создает эскиз для отверстия под вентилятор |
| CreateRectangle() | double – верхний край по X координате  double – верхний край по Y координате  double – нижний край по X координате  double – нижний край по Y координате | Создает эскиз прямоугольника |

Прододжение таблицы 3.2

|  |  |  |
| --- | --- | --- |
| Название | Тип входных параметров | Описание |
| ExtrudeCutCircle() | double – глубина, на которую необходимо вырезать | Вырезает круг на указанную глубину |
| ExtrudeRectangle() | double – глубина, на которую необходимо вырезать | Выдавливает эскиз прямогульника |
| OpenCompas() |  | Открывает программу компас и возвращает объект KompasObject |

Таблица 3.3 – Класс Validator

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Тип входных параметров | Тип возвращаемых данных | Описание |
| Validate () | double – верхняя числовая граница  double – нижняя числовая граница  double – число, которое необходимо проверить | true – если число входит в указанный диапазон  false – если число не входит в указанный диапазон | Проверяет, входит ли число в указанный диапазон значений |

Таблица 3.4 – Класс CaseParameters

|  |  |  |
| --- | --- | --- |
| Название | Тип возвращаемых данных | Описание |
| FrontFansNumber() | int – количество отверстий вентиляторов на передней части корпуса | При установке значения проверяется правильность введенных значений, используя класс Validator |
| FrontFansRadius() | double – радиус отверстий под вентиляторы на передней части корпуса | При установке значения проверяется правильность введенных значений, используя класс Validator |

Продолжение таблицы 3.4

|  |  |  |
| --- | --- | --- |
| Название | Тип возвращаемых данных | Описание |
| Height() | double – высота корпуса | При установке значения проверяется правильность введенных значений, используя класс Validator |
| Length() | double – длина корпуса | При установке значения проверяется правильность введенных значений, используя класс Validator |
| MotherboardType() | enum – описывающий тип материнской платы | При установке значения проверяется правильность введенных значений, используя класс Validator |
| UpperFansRadius() | double – радиус отверстий под вентиляторы на верхней части корпуса | При установке значения проверяется правильность введенных значений, используя класс Validator |
| UpperFansNamber() | int – количество отверстий вентиляторов на передней части корпуса | При установке значения проверяется правильность введенных значений, используя класс Validator |
| Width() | double – ширина корпуса | При установке значения проверяется правильность введенных значений, используя класс Validator |

Продолжение таблицы 3.4

|  |  |  |
| --- | --- | --- |
| Название | Тип возвращаемых данных | Описание |
| \_frontFansNumber | int – количество отверстий вентиляторов на передней части корпуса | Закрытое поле, содержащее количество вентиляторов на передней части корпуса |
| \_frontFansRadius | double – радиус отверстий под вентиляторы на передней части корпуса | Закрытое поле, содержащее радиус отверстий под вентиляторы на передней части корпуса |
| \_height | double – высота корпуса | Закрытое поле, содержащее высоту корпуса |
| \_length | double – длина корпуса | Закрытое поле, содержащее длину корпуса |
| \_upperFansRadius | double – радиус отверстий под вентиляторы на верхней части корпуса | Закрытое поле, содержащее радиус отверстий под вентиляторы на верхней части корпуса |
| \_upperFansNamber | int – количество отверстий вентиляторов на передней части корпуса | Закрытое поле, содержащее количество отверстий вентиляторов на передней части корпуса |
| ATX\_PATE\_HEIGHT | int – высота ATX материнской платы | Константа, описывающая высоту ATX материнской платы, равный 305 |

Продолжение таблицы 3.4

|  |  |  |
| --- | --- | --- |
| Название | Тип возвращаемых данных | Описание |
| MICRO\_ATX\_PLATE\_HEIGHT | int – высота micro-ATX материнской платы | Константа, описывающая высоту micro-ATX материнской платы, равный 244 |
| PLATE\_WIDTH | int – ширина материнских плат ATX и micro-ATX | Константа, описывающая ширину материнских плат ATX и micro-ATX |
| SPACE\_BEETWEEN\_FRONT | int – расстояние между передними вентиляторами | Константа, описывающая расстояние между передними вентиляторами |
| SPACE\_BEETWEEN\_UPPER | int – расстояние между верхними вентиляторами | Константа, описывающая расстояние между верхними вентиляторами |

# Макет пользовательского интерфейса

Пользовательский интерфейс представляет собой форму для ввода параметров. При нажатии на кнопку «Построить» строится 3D-модель корпуса. В правой части интерфейса расположены подсказки в виде картинки. На рисунке 3.2 представлен макет пользовательского интерфейса.

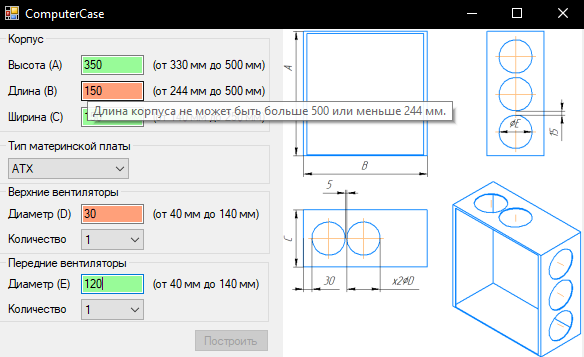


Рисунок 3.2 – Макет пользовательского интерфейса

Кнопка «Построить» остается неактивной до тех пор, пока в поля со значениями не будут введены корректные значения. При вводе некорректных значений, соответствующее поле будет подсвечено красным. Поле с некорректным значением, при наведении курсора мыши, выводит конкретную ошибку, допущенную пользователем.

При возникновении ошибок соотношений зависимостей, например, диаметр верхних вентиляторов и длина корпуса, красным цветом подсвечиваются оба поля. Пример возникновения ошибки зависимых величин на рисунке 3.3

# Список использованной литературы

1. САПР [Электронный ресурс]. – Режим доступа: (дата обращения 30.11.2021);
2. Официальный сайт Kompas. [Электронный ресурс] – Режим доступа: <https://kompas.ru/kompas-3d/about/> (дата обращения 30.10.2021);
3. Мартин Ф. UML. Основы. Краткое руководство по стандартному языку объектного моделирования. Изд: Символ-Плюс,2011, с.192 (3-е издание);
4. API — Мартин Редди. API Design for C++. Изд: Elsevier Science, 2011, с.472 (1-e издание);
5. Онлайн конфигуратор ПК [Электронный ресурс] – Режим доступа: <https://edelws.ru/constructor/> (дата обращения 14.12.2021).