Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Ульяновский государственный технический университет»

Факультет информационных систем и технологий

Кафедра «Прикладная математика и информатика»

**Разработка программного решения для проектирования технического объекта**

Д.И.Сазонов

А.Д.Тарасов

Ульяновск 2018 г.

Сазонов Д.И., Тарасов А.Д. Разработка программного решения для проектирования технического объекта: методичка- Ульяновск 2018. - … с.

В методическом пособии рассмотрена разработка программного решения для проектирования технического объекта в среде КОМПАС-3D, програмная реализация выполнена на языке программирования под названием Python

КОМПАС-3D разработан специально для операционной среды MS DOS / Windows и в полной мере использует все ее возможности и преимущества для предоставления пользователю максимального комфорта и удобства в работе.

**Оглавление**

[Введение 4](#_Toc514567685)

[1. Теоретическая часть 5](#_Toc514567686)

[1.1. Теория проектной деятельности и автоматизации проектирования 5](#_Toc514567687)

[1.2. 3D-моделирование в проектировании. 8](#_Toc514567688)

[1.3. Система трехмерного проектирования КОМПАС-3D. 10](#_Toc514567689)

[2.Практическая часть 13](#_Toc514567690)

[Лабораторная работа №1. Установка Python. API Kompas3D. 13](#_Toc514567691)

[Лабораторная работа №2. Геометрические примитивы.3D 22](#_Toc514567692)

[Лабораторная работа №3. Параметризация 25](#_Toc514567693)

[Заключение 28](#_Toc514567694)

[Список использованной литературы 29](#_Toc514567695)

# Введение

 Персональные компьютеры, как известно, могут выполнять любые действия по обработке информации. Для этого необходимо составить для компьютера на понятном ему языке точную и подробную программу, как надо обрабатывать информацию. В настоящее время требуется все более разнообразное программное обеспечение для решения задач во новых областях применения. Непрерывное повышение мощности персональных компьютеров, периферийных устройств, а также развитие средств связи дает разработчикам программного обеспечения все больше возможностей для максимально полного удовлетворения запросов конечных потребителей.

Программы для пользователей служат для выполнения каких – либо конкретных задач во всех сферах человеческой деятельности. Количество программ, установленных на современном компьютере, исчисляется сотнями и даже тысячами. Именно они обеспечивают комфортную работу пользователя.

Вся совокупность программ входит в так называемое программное обеспечение компьютера. Состав программного обеспечения ПК является важнейшей его функциональной характеристикой. Программное обеспечение (ПО) — это совокупность программ регулярного применения, необходимых для решения задач пользователя, и программ, позволяющих наиболее эффективно использовать вычислительную технику, обеспечивая пользователям наибольшие удобства в работе и минимум затрат труда на программирование задач и обработку информации. Следовательно, тема работы является актуальной.

Геометрическая среда: КОМПАС 3D.

Язык программирования: Python.

Метод разработки: разработка независимого приложения для КОМПАС 3D.

# 1. Теоретическая часть

## **1.1. Теория проектной деятельности и автоматизации проектирования**

Процесс проектирования представляет собой особый вид человеческой деятельности. Объекты проектирования могут включать как материальные (производственные строения, машины и т. д.), так и нематериальные объекты (социальное проектирование). Процесс проектирования – это информационно-обрабатывающая деятельность создания информационных моделей планирования технических работ, технических инноваций и выработки методов, средств и процедур для их реализации.

Проектирование начинается с получения информации о состоянии данной области: сведения о технических устройствах, материалах, методах изготовления, компонентах, процессах, состоянии рынка и т.д.

Цель проектирования – создание объекта, удовлетворяющего определенным требованиям, обладающего определенным качеством (структурой).

Современная тенденция совершенствования процесса проектирования заключается в его автоматизации, так как задачи проектирования не ограничиваются подготовкой проектной документации. Комплексное системное проектирование включает познание объектов, социальной потребности в них, оценки их реализуемости и оценки последствий введения в эксплуатацию.

**Система автоматизированного проектирования — автоматизированная система, реализующая информационную технологию выполнения функций проектирования, представляет собой организационно-техническую систему, предназначенную для автоматизации процесса проектирования, состоящую из персонала и комплекса технических, программных и других средств автоматизации его деятельности. Также для обозначения подобных систем широко используется аббревиатура САПР.**

Автоматизация проектирования занимает особое место среди информационных технологий.

Во-первых, автоматизация проектирования — синтетическая дисциплина, ее составными частями являются многие другие современные информационные технологии. Так, техническое обеспечение систем автоматизированного проектирования основано на использовании вычислительных сетей и телекоммуникационных технологий, в САПР используются персональные компьютеры и рабочие станции, есть примеры применения мейнфреймов. Математическое обеспечение САПР отличается богатством и разнообразием используемых методов вычислительной математики, статистики, математического программирования, дискретной математики, искусственного интеллекта.

Во-вторых, знание основ автоматизации проектирования и умение работать со средствами САПР требуется практически любому инженеру-разработчику.

Выделяют следующие виды обеспечения САПР:

Техническое обеспечение (ТО) — совокупность связанных и взаимодействующих технических средств (ЭВМ, периферийные устройства, сетевое оборудование, линии связи, измерительные средства).

Математическое обеспечение (МО), объединяющее математические методы, модели и алгоритмы, используемые для решения задач автоматизированного проектирования. По назначению и способам реализации делят на две части:

* математические методы и построенные на них математические модели;
* формализованное описание технологии автоматизированного проектирования.

**Программное обеспечение (ПО).**

**Прикладное программное обеспечение – это комплекс программных средств и документации к ним, предназначенных для решения сравнительно узких классов задач в конкретных предметных областях, рассчитанных на определенного потребителя: научно-технических, экономических, инженерных, конструкторских и других специальных задач в различных сферах человеческой деятельности.**

**Пакет прикладных программ – это комплекс программ, предназначенный для решения задач определенного класса (функциональная подсистема, бизнес-приложение). Пакеты прикладных программ (ППП) являются мощным инструментом автоматизации решаемых пользователем задач, практически полностью освобождая его от необходимости знать, как выполняет компьютер те или иные функции и процедуры по обработке информации. Сейчас имеется широкий спектр ППП, различающихся по своим функциональным возможностям и способам реализации.**

Параметрическое трехмерное или двумерное моделирование существенно отличается от обычного черчения или 3Д-моделирования. В случае с параметрическим моделированием создается математическая модель с параметрами, изменение которых влечет за собой изменение всей конфигурации детали, перемещение деталей в сборке и прочие похожие трансформации.

Идея создать параметрическое моделирование появилась достаточно давно, но, к сожалению, воплощение в жизнь было невозможно из-за недостаточной производительности компьютеров. 1989 год стал датой рождения параметрического моделирования, так как именно в этом году были выпущены первые САПРы с функциями параметризации.

Формирование и внедрение зависимостей и манипуляции с ними, по сути, являются процессом проектирования. Поэтому параметрическое моделирование является самым простым, удобным способом проектировать объекты, так как именно данная технология предоставляет специалисту полный доступ к контролю зависимостей. Система программирования — это комплекс инструментальных средств, предназначенных для работы с программами на одном из языков программирования.

Python – это один из наиболее популярных современных языков программирования. Он пригоден для решения разнообразных задач и предлагает те же возможности, что и другие языки программирования: динамичность, поддержку ООП и кросс-платформенность.

## **1.2. 3D-моделирование в проектировании.**

3D-моделирование — это процесс создания трехмерной модели объекта.

Задача 3D-моделирования — разработать визуальный объемный образ желаемого объекта. С помощью трехмерной графики можно и создать точную копию конкретного предмета, и разработать новое, даже нереальное представление до сего момента, не существовавшего объекта. Трёхмерная графика активно применяется для создания изображений на плоскости экрана или листа печатной продукции в науке и промышленности, например, в системах автоматизации проектных работ (САПР; для создания твердотельных элементов: зданий, деталей машин, механизмов), архитектурной визуализации (сюда относится и так называемая «виртуальная археология»), в современных системах медицинской визуализации. Трёхмерная графика обычно имеет дело с виртуальным, воображаемым трёхмерным пространством, которое отображается на плоской, двухмерной поверхности дисплея или листа бумаги.

Преимущества 3D моделирования

Основной недостаток двухмерной графики в том, что по наброскам и эскизам бывает сложно представить, как будет выглядеть объект в действительности.  Именно поэтому очень часто к чертежам прилагают реальную модель будущего проекта: пробную деталь, макет и т. д. Таким образом, ошибки в расчетах приходится исправлять на уже готовом изделии, что значительно тормозит процесс реализации идеи. Использование трехмерной графики позволяет смоделировать объект еще до выпуска пробных образцов, благодаря чему появляется возможность увидеть ошибки и нестыковки проекта и  оценить степень соответствия исходному замыслу. Вторым, но не менее важным преимуществом 3d моделирования является высокая степень убедительности и наглядности 3d изображений и видеороликов. Немногие обладают достаточной фантазией, позволяющей представить, как будет выглядеть, например, будущее здание, имея перед собой лишь чертежи и сметы.

Для получения трёхмерного изображения на плоскости требуются следующие шаги:

Моделирование (виртуального пространства моделирования) включает в себя несколько категорий объектов:

1)  геометрия (построенная с помощью различных техник (напр., создание полигональной сетки) модель, здания);

2) материалы (информация о визуальных свойствах модели, например, цвет стен и отражающая/преломляющая способность окон);

3) источники света (настройки направления, мощности, спектра освещения);

4) силы и воздействия (настройки динамических искажений объектов, применяется в основном в анимации);

5) дополнительные эффекты (объекты, имитирующие атмосферные явления: свет в тумане, облака, пламя и пр.)

6) задача трёхмерного моделирования — описать эти объекты и разместить их в сцене с помощью геометрических преобразований в соответствии с требованиями к будущему изображению.

Таким образом изготовление 3D модели в общем случае состоит из следующих стадий:

1. Получение изображений референса (т.е. того, с чего будет моделироваться) или самого референса. Или отрисовка эскиза;

2. Моделирование геометрии на основе референса;

3. Создание развертки;

4. Отрисовка текстур или получение их другим способом в виде файлов.

5. Настройка параметров материала (текстуры, преломление, отражение, прозрачность);

Теперь 3D модель готова для визуализации — получении картинки.

## **1.3. Система трехмерного проектирования КОМПАС-3D.**

КОМПАС-3D — система трехмерного проектирования, ставшая стандартом для тысяч предприятий, благодаря сочетанию простоты освоения и легкости работы с мощными функциональными возможностями твердотельного и поверхностного моделирования.

Ключевой особенностью продукта является использование собственного математического ядра С3D и параметрических технологий, разработанных специалистами АСКОН.

КОМПАС-3D обеспечивает поддержку наиболее распространенных форматов 3D-моделей (STEP, ACIS, IGES, DWG, DXF), что позволяет организовывать эффективный обмен данными со смежными организациями и заказчиками, использующими любые CAD / CAM / CAE-системы в работе.

Программы данного семейства автоматически генерируют ассоциативные виды трёхмерных моделей (в том числе разрезы, сечения, местные разрезы, местные виды, виды по стрелке, виды с разрывом). Все они ассоциированы с моделью: изменения в модели приводят к изменению изображения на чертеже.

Стандартные виды автоматически строятся в проекционной связи. Данные в основной надписи чертежа (обозначение, наименование, масса) синхронизируются с данными из трёхмерной модели. Имеется возможность связи трёхмерных моделей и чертежей со спецификациями, то есть при «надлежащем» проектировании спецификация может быть получена автоматически; кроме того, изменения в чертеже или модели будут передаваться в спецификацию, и наоборот.

«Компас» выпускается в нескольких редакциях: «Компас-График», «Компас-СПДС», «Компас-3D», «Компас-3D LT», «Компас-3D Home». «Компас-График» может использоваться и как полностью интегрированный в «Компас-3D» модуль работы с чертежами и эскизами, и в качестве самостоятельного продукта, предоставляющего средства решения задач 2D-проектирования и выпуска документации. «Компас-3D LT» и «Компас-3D Home» предназначены для некоммерческого использования, «Компас-3D» без специализированной лицензии не позволяет открывать файлы, созданные в этих программах. Такая специализированная лицензия предоставляется только учебным заведениям.

Система «Компас-3D» предназначена для создания трёхмерных ассоциативных моделей отдельных деталей (в том числе, деталей, формируемых из листового материала путём его гибки) и сборочных единиц, содержащих как оригинальные, так и стандартизованные конструктивные элементы. Параметрическая технология позволяет быстро получать модели типовых изделий на основе проектированного ранее прототипа. Многочисленные сервисные функции облегчают решение вспомогательных задач проектирования и обслуживания производства.

Система «Компас-3D» включает следующие компоненты: система трёхмерного твердотельного моделирования, универсальная система автоматизированного проектирования «Компас-График» и модуль формирования спецификаций. Ключевой особенностью «Компас-3D» является использование собственного математического ядра и параметрических технологий.

# 2.Практическая часть

## **Лабораторная работа №1. Установка Python. API Kompas3D.**

**Цель:**

1. ознакомиться с системой КОМПАС;
2. ознакомиться с системой Python;
3. ознакомиться с API Kompas3D и научиться интегрировать в Python.

**В результате выполнения ЛР№1 студенты должны:**

1. получить первые навыки по работе с системой КОМПАС, представление о возможностях системы;
2. получить первые навыки по работе с системой Python, представление о возможностях системы;
3. научиться интегрировать API Kompas3D.

**Общие сведения.** Python ([МФА](https://ru.wikipedia.org/wiki/%D0%9C%D0%A4%D0%90): [[ˈpʌɪθ(ə)n]](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D0%B6%D0%B4%D1%83%D0%BD%D0%B0%D1%80%D0%BE%D0%B4%D0%BD%D1%8B%D0%B9_%D1%84%D0%BE%D0%BD%D0%B5%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B0%D0%BB%D1%84%D0%B0%D0%B2%D0%B8%D1%82); в русском языке распространено название *пито́н*) — [высокоуровневый язык программирования](https://ru.wikipedia.org/wiki/%D0%92%D1%8B%D1%81%D0%BE%D0%BA%D0%BE%D1%83%D1%80%D0%BE%D0%B2%D0%BD%D0%B5%D0%B2%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F) общего назначения, ориентированный на повышение производительности разработчика и читаемости кода. [Синтаксис](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BD%D1%82%D0%B0%D0%BA%D1%81%D0%B8%D1%81_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) ядра Python минималистичен. В то же время [стандартная библиотека](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D0%B0%D0%BD%D0%B4%D0%B0%D1%80%D1%82%D0%BD%D0%B0%D1%8F_%D0%B1%D0%B8%D0%B1%D0%BB%D0%B8%D0%BE%D1%82%D0%B5%D0%BA%D0%B0_Python) включает большой объём полезных функций.

Python поддерживает несколько [парадигм программирования](https://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D1%80%D0%B0%D0%B4%D0%B8%D0%B3%D0%BC%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F), в том числе [структурное](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5), [объектно-ориентированное](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5), [функциональное](https://ru.wikipedia.org/wiki/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5), [императивное](https://ru.wikipedia.org/wiki/%D0%98%D0%BC%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) и [аспектно-ориентированное](https://ru.wikipedia.org/wiki/%D0%90%D1%81%D0%BF%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5). Основные архитектурные черты — [динамическая типизация](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D0%BD%D0%B0%D0%BC%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%82%D0%B8%D0%BF%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F), [автоматическое управление памятью](https://ru.wikipedia.org/wiki/%D0%A1%D0%B1%D0%BE%D1%80%D0%BA%D0%B0_%D0%BC%D1%83%D1%81%D0%BE%D1%80%D0%B0_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), полная [интроспекция](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D1%80%D0%BE%D1%81%D0%BF%D0%B5%D0%BA%D1%86%D0%B8%D1%8F_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), механизм [обработки исключений](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B0_%D0%B8%D1%81%D0%BA%D0%BB%D1%8E%D1%87%D0%B5%D0%BD%D0%B8%D0%B9), поддержка [многопоточных вычислений](https://ru.wikipedia.org/wiki/%D0%9C%D0%BD%D0%BE%D0%B3%D0%BE%D0%BF%D0%BE%D1%82%D0%BE%D1%87%D0%BD%D0%BE%D1%81%D1%82%D1%8C) и удобные высокоуровневые [структуры данных](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85). Код в Python организовывается в функции и [классы](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D1%81%D1%81_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), которые могут объединяться в [модули](https://ru.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B4%D1%83%D0%BB%D1%8C_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) (они в свою очередь могут быть объединены в пакеты).

Из имеющихся инструментов имеем: Kompas 3D и Python. Подготовительная операция: Убедитесь, что на вашем компьютере установлена программа Kompas 3D, версии не ниже 14 и Python 3. Вам также необходимо установить [pywin3](http://sourceforge.net/projects/pywin32/) (Python for Windows extensions).

Подключение к Kompas 3D: Система Kompas 3D имеет две версии API: API5, которая предоставляет интерфейс KompasObject, и API7, предоставляющая интерфейс IKompasAPIObject. API версии 5 и 7 во многом дублируют свой функционал, но, со слов разработчиков, в 7-ой версии более выражен объектно-ориентированный подход. В данной статье акцент сделан на 7-ю версию.

Функция подключения выглядит следующим образом:

import pythoncom

from win32com.client import Dispatch, gencache

# Подключение к API7 программы Kompas 3D

def get\_kompas\_api7():

module = gencache.EnsureModule("{69AC2981-37C0-4379-84FD-5DD2F3C0A520}", 0, 1, 0)

api = module.IKompasAPIObject( Dispatch("Kompas.Application.7").\_oleobj\_.QueryInterface(module.IKompasAPIObject.CLSID, pythoncom.IID\_IDispatch))

const = gencache.EnsureModule("{75C9F5D0-B5B8-4526-8681-9903C567D2ED}", 0, 1, 0).constants

return module, api, const

Теперь, чтобы подключиться к интерфейсу, нам понадобиться следующий код:

module7, api7, const7 = get\_kompas\_api7() # Подключаемся к API7

app7 = api7.Application # Получаем основной интерфейс

app7.Visible = True # Показываем окно пользователю (если скрыто)

app7.HideMessage = const7.ksHideMessageNo # Отвечаем НЕТ на любые вопросы программы

print(app7.ApplicationName(FullName=True)) # Печатаем название программы

Для более глубокого понимания API заглянем в SDK: C:\Program Files\ASCON\KOMPAS-3D V16\SDK\SDK.chm.

После выполнения нашего кода вернём всё на свои места: если Kompas 3D был запущен нами (в процессе работы скрипта), мы его и закроем. Самый простой способ определить, запущен ли процесс, — использовать стандартный модуль subprocess:

import subprocess

# Функция проверяет, запущена ли программа Kompas 3D

def is\_running():

proc\_list = subprocess.Popen('tasklist /NH /FI "IMAGENAME eq KOMPAS\*"', shell=False, stdout=subprocess.PIPE).communicate()[0]

return True if proc\_list else False

Данная функция проверяет, запущен ли процесс «KOMPAS» [стандартными методами Windows](https://technet.microsoft.com/en-us/library/bb491010.aspx#mainSection). Обратите внимание, что разные версии программы Kompas 3D могут иметь разные наименования процессов!

Считаем количество листов и их формат: у нашего документа doc7 имеется интерфейс коллекции листов оформления LayoutSheets. Каждый лист обладает свойством формата и кратности. Для Компаса, начиная с 15 версии, интерфейс LayoutSheets доступен не только для файлов чертежей, но и для спецификаций и текстовых документов.

# Посчитаем количество листов каждого из формата

def amount\_sheet(doc7):

sheets = {"A0": 0, "A1": 0, "A2": 0, "A3": 0, "A4": 0, "A5": 0}

for sheet in range(doc7.LayoutSheets.Count):

format = doc7.LayoutSheets.Item(sheet).Format # sheet - номер листа, отсчёт начинается от 0

sheets["A" + str(format.Format)] += 1 \* format.FormatMultiplicity

return sheets

Посмотрим на процесс изучения SDK для поиска интересующих нас функций:

Читаем основную надпись. Здесь нам поможет всё тот же LayoutSheets:

# Прочитаем масштаб из штампа, ячейка №6

def stamp\_scale(doc7):

stamp = doc7.LayoutSheets.Item(0).Stamp # Item(0) указывает на штамп первого листа

return stamp.Text(6).Str

На самом деле ячейка №6 для листа с другим оформлением может содержать не масштаб, а совсем иную информацию. Посмотрим, как в Kompas 3D определяются стили оформления чертежа.

Таким образом, важно проверять, какому файлу и номеру оформления соответствует лист чертежа. Также стоит помнить, что документ может содержать титульный лист! Поэтому придётся усложнить код. Применим [регулярные выражения](https://habrahabr.ru/post/115825/), т.к. текст в ячейке может являться ссылкой:

import os

import re

# Прочитаем основную надпись чертежа

def stamp(doc7):

for sheet in range(doc7.LayoutSheets.Count):

style\_filename = os.path.basename(doc7.LayoutSheets.Item(sheet).LayoutLibraryFileName)

style\_number = int(doc7.LayoutSheets.Item(sheet).LayoutStyleNumber)

if style\_filename in ['graphic.lyt', 'Graphic.lyt'] and style\_number == 1:

stamp = doc7.LayoutSheets.Item(sheet).Stamp

return {"Scale": re.search(r"\d+:\d+", stamp.Text(6).Str).group(),

"Designer": stamp.Text(110).Str}

return {"Scale": 'Неопределенный стиль оформления',

"Designer": 'Неопределенный стиль оформления'}

Остался последний вопрос: как узнать нужный номер ячейки? Для этих целей удобно создать файл чертежа, в котором интересующие нас ячейки будут заполнены, а после — прочитать все возможные варианты с помощью следующей функции:

# Просмотр всех ячеек

def parse\_stamp(doc7, number\_sheet):

stamp = doc7.LayoutSheets.Item(number\_sheet).Stamp

for i in range(10000):

if stamp.Text(i).Str:

print('Номер ячейки = %-5d Значение = %s' % (i, stamp.Text(i).Str))

Считаем количество пунктов технических требований. Согласно SDK, нам всего-то нужно получить интерфейс TechnicalDemand от IDrawingDocument, а IDrawingDocument можно получить от iDocuments с помощью замечательного метода с говорящим названием IUnknown::QueryInterface. И только в SDK 16 версии Kompas 3D появилось разъяснение, как это сделать:

# Подсчет технических требований, в том случае, если включена автоматическая нумерация

def count\_TT(doc7, module7):

doc2D\_s = doc7.\_oleobj\_.QueryInterface(module7.NamesToIIDMap['IDrawingDocument'],

pythoncom.IID\_IDispatch)

doc2D = module7.IDrawingDocument(doc2D\_s)

text\_TT = doc2D.TechnicalDemand.Text

count\_tt = 0 # Количество пунктов технических требований

for i in range(text\_TT.Count): # Проходим по каждой строчке технических требований

if text\_TT.TextLines[i].Numbering == 1: # и проверяем, есть ли у строки нумерация

count\_tt += 1

# Если нет нумерации, но есть текст

if not count\_tt and text\_TT.TextLines[0]:

count\_tt += 1

return count\_tt

Стоит отметить, что данный код полагается на автоматическую нумерацию технических требований. Так что, если автоматическая нумерация не применялась или технические требования набраны с использованием простого инструмента «Текст», код будет сложнее.

Считаем количество размеров на чертеже. При подсчёте размеров, надо иметь в виду, что необходимо посчитать их на каждом из видов чертежа:

# Подсчёт размеров на чертеже, для каждого вида по отдельности

def count\_dimension(doc7, module7):

IKompasDocument2D = doc7.\_oleobj\_.QueryInterface(module7.NamesToIIDMap['IKompasDocument2D'], pythoncom.IID\_IDispatch)

doc2D = module7.IKompasDocument2D(IKompasDocument2D)

views = doc2D.ViewsAndLayersManager.Views

count\_dim = 0

for i in range(views.Count):

ISymbols2DContainer = views.View(i).\_oleobj\_.QueryInterface(module7.NamesToIIDMap['ISymbols2DContainer'], pythoncom.IID\_IDispatch)

dimensions = module7.ISymbols2DContainer(ISymbols2DContainer)

# Складываем все необходимые размеры

count\_dim += dimensions.AngleDimensions.Count + \

dimensions.ArcDimensions.Count + \

dimensions.Bases.Count + \

dimensions.BreakLineDimensions.Count + \

dimensions.BreakRadialDimensions.Count + \

dimensions.DiametralDimensions.Count + \

dimensions.Leaders.Count + \

dimensions.LineDimensions.Count + \

dimensions.RadialDimensions.Count + \

dimensions.RemoteElements.Count + \

dimensions.Roughs.Count + \

dimensions.Tolerances.Count

return count\_dim

Основная функция скрипта. В результате проделанной работы мы получили следующее:

def parse\_design\_documents(paths):

is\_run = is\_running() # Установим флаг, который нам говорит, запущена ли программа до запуска нашего скрипта

module7, api7, const7 = get\_kompas\_api7() # Подключаемся к программе

app7 = api7.Application # Получаем основной интерфейс программы

app7.Visible = True # Показываем окно пользователю (если скрыто)

app7.HideMessage = const7.ksHideMessageNo # Отвечаем НЕТ на любые вопросы программы

table = [] # Создаём таблицу параметров

for path in paths:

doc7 = app7.Documents.Open(PathName=path,

Visible=True,

ReadOnly=True) # Откроем файл в видимом режиме без права его изменять

row = amount\_sheet(doc7) # Посчитаем кол-во листов каждого формат

row.update(stamp(doc7)) # Читаем основную надпись

row.update({

"Filename": doc7.Name, # Имя файла

"CountTD": count\_demand(doc7, module7), # Количество пунктов технических требований

"CountDim": count\_dimension(doc7, module7), # Количество размеров на чертеже

})

table.append(row) # Добавляем строку параметров в таблицу

doc7.Close(const7.kdDoNotSaveChanges) # Закроем файл без изменения

if not is\_run: app7.Quit() # Выходим из программы

return table

Диалоговое окно выбора файлов. Для удобного использования нашего скрипта воспользуемся возможностями стандартного модуля tkinter и выведем диалоговое окно выбора файлов:

from tkinter import Tk

from tkinter.filedialog import askopenfilenames

if \_\_name\_\_ == "\_\_main\_\_":

root = Tk()

root.withdraw() # Скрываем основное окно и сразу окно выбора файлов

filenames = askopenfilenames(title="Выберете чертежи деталей",filetypes=[('Kompas 3D', '\*.cdw'),])

print\_to\_excel(parse\_design\_documents(filenames))

root.destroy() # Уничтожаем основное окно

root.mainloop()

Ресурсы в помощь:

1. SDK (C:\Program Files\ASCON\KOMPAS-3D V16\SDK\SDK.chm)
2. [forum.ascon.ru](http://forum.ascon.ru)

**Задание.**

1. Установить Kompas3D.
2. Установить Python.
3. Создать проект и интегрировать API Kompas3D.

**Ход работы.**

1. Установить Kompas3D;
2. Установить Python;
3. Создать проект в любом текстовом редакторе с расширением \*.py;
4. Первой строчкой прописать тип кодировки:

# -\*- coding: cp1251 -\*-;

1. Подключить API Kompas 3D:

import Kompas10API5 as KAPI

import Kompas10API7 as KAPI7;

1. Подключить компонент Python для Windows:

import pythoncom

from win32com.client import Dispatch;

1. Подключить компоненты КОМПАС:

import LDefin2D

import LDefin3D

import ksConstants as CONST2D7

import ksConstants3D as CONST3D7

import MiscellaneousHelpers as MH;

1. Подкючить библиотеку math и Tkinter:

import math

from Tkinter import \*;

1. Написать функцию:

def KompasRun():

try:

ModelDetail()

except ValueError:

inserter(u"Ошибка Ввода")

Данная функция попытается вызвать функцию ModelDetail(). Если не получается вызвать, то она ловит исключение и выдает ошибку;

1. Напишем небольшой интерфейс, который нам понадобиться в дальнейшем:

root=Tk()

root.title(u'Заголовок проекта') #Заголовок

root.geometry('620x450') #Размер окна

root.resizable(width=False, height=False)

frame = Frame(root, bd=8)

frame.grid()

but = Button(frame, text=u"Перейти к построению", command=Kompas\_run(), width=22, height=0, fg='black', bg='#FFFFFF', font=('gost type a', 20, 'bold', 'italic')) #Кнопка для запуска

but.grid(row=5, column=0, columnspan=2)

root.mainloop();

**Контрольные вопросы**

1. Для чего предназначена система КОМПАС?
2. Из чего состоит интерфейс системы КОМПАС 3D?
3. Для чего предназначена система Python.
4. Что такое «API Kompas3D»? В чем разница между API5 и API7.

## **Лабораторная работа №2. Геометрические примитивы.3D**

**Цель:**

1. ознакомиться с API Kompas3D;
2. изучить основные типы геометрических примитивов;
3. освоить приемы выполнения простейших геометрических построений.

**В результате выполнения ЛР№2 студенты должны:**

1. получить первые навыки по работе с API Kompas, представление о возможностях системы;
2. знать правила построения простейших геометрических примитивов;

**Общие сведения.** Ввод геометрических объектов. Для создания модели изделия любой сложности используются геометрические примитивы.

Примитивы–ограниченный набор графических объектов, с помощью которого средствами объявленного набора операций можно сконструировать много большее множество других объектов. Элемент конструктора, составная часть целого, способная к комбинированию.

**Задание.**

1. На Python с помощью API Kompas3D написать код программы, которые будет выполнять построение геометрического примитива в КОМПАС 3D.

**Ход работы.**

1. Открыть проект, созданный в ЛР№1;
2. Напишем функцию для нашей детали ModelDetail();
3. Проинициализируем объекты Kompas из API и запишем их в переменные:

iKompasObject = Dispatch('KOMPAS.Application.5')

iKompasObject = KAPI.KompasObject(iKompasObject)

MH.iKompasObject = iKompasObject

iApplication = Dispatch('KOMPAS.Application.7')

iApplication = KAPI7.IApplication(iApplication)

iApplication.Visible = True

MH.iApplication = iApplication

1. Запишем:

iDocument3D = iKompasObject.Document3D()

iDocument3D.Create(False, True),

для создания 3D объекта в компасе,

iDocument = iApplication.ActiveDocument

iKompasDocument3D = KAPI7.IKompasDocument3D(iDocument),

и сделаем его активным;

1. Запишем:

iPart = iDocument3D.GetPart(LDefin3D.pTop\_Part)

iSketch = iPart.NewEntity(LDefin3D.o3d\_sketch)

iDefinition = iSketch.GetDefinition()

iPlane = iPart.GetDefaultEntity(LDefin3D.o3d\_planeXOY)

iDefinition.SetPlane(iPlane)

iSketch.Create(),

для создания эскиза в плоскости XOY,

iDocument2D = iDefinition.BeginEdit()

iDocument = iApplication.ActiveDocument

iKompasDocument2D = KAPI7.IKompasDocument2D(iDocument)

заинициализируем 2D объект Kompas для построения эскиза;

1. Запишем:

obj = iDocument2D.ksLineSeg(0, 0, 0, 5, 1)

obj = iDocument2D.ksLineSeg(0, 5, 5, 5, 1)

obj = iDocument2D.ksLineSeg(5, 0, 5, 5, 1)

obj = iDocument2D.ksLineSeg(0, 0, 5, 0, 1),

для создания прямоугольника из отрезков, где первые два значения функции координаты начала отрезка, две следующие – координаты конца отрезка, последняя – стиль линии,

iDefinition.EndEdit(),

завершим редактирование эскиза;

1. Запишем:

iPart = iDocument3D.GetPart(LDefin3D.pTop\_Part)

obj = iPart.NewEntity(LDefin3D.o3d\_bossRotated);

iPart = iDocument3D.GetPart(LDefin3D.pTop\_Part)

obj = iPart.NewEntity(LDefin3D.o3d\_bossRotated)

iDefinition = obj.GetDefinition()

iCollection = iPart.EntityCollection(LDefin3D.o3d\_edge)

iCollection.SelectByPoint(3.0, 11, 0.0)

iEdge = iCollection.Last()

iEdgeDefinition = iEdge.GetDefinition()

iSketch = iEdgeDefinition.GetOwnerEntity()

iDefinition.SetSketch(iSketch)

iRotatedParam = iDefinition.RotatedParam()

iRotatedParam.angleNormal = 360.0

iRotatedParam.angleReverse = 360.0

iRotatedParam.direction = LDefin3D.dtNormal

iRotatedParam.toroidShape = True

iCollection = iPart.EntityCollection(LDefin3D.o3d\_edge)

iCollection.SelectByPoint(17, 0.0, 0.0)

iEdge = iCollection.Last()

iRotated = iKompasObject.TransferInterface(obj, CONST2D7.constants.ksAPI7Dual, LDefin3D.o3d\_bossRotated)

iRotatedAxis = iKompasObject.TransferInterface(iEdge, CONST2D7.constants.ksAPI7Dual, 0)

iRotated.Axis = iRotatedAxis

Это просто магия.

**Контрольные вопросы.**

1. Для чего предназначена система КОМПАС?
2. Из чего состоит интерфейс системы КОМПАС 3D?
3. Для чего предназначена система Python.
4. Что такое «API Kompas3D»? В чем разница между API5 и API7.
5. Дать определение геометрическим примитивам. Какие примитивы Вы знаете?

## **Лабораторная работа №3. Параметризация**

**Цель:**

1. ознакомиться с API Kompas3D;
2. изучить основные типы геометрических примитивов;
3. освоить приемы выполнения простейших геометрических построений;
4. освоить параметризацию.

**В результате выполнения ЛР№3 студенты должны:**

1. получить первые навыки по работе с API Kompas, представление о возможностях системы;
2. знать правила построения простейших геометрических примитивов;
3. знать правила построения с параметризацией.

**Общие сведения.** Что такое параметризация в Компасе?

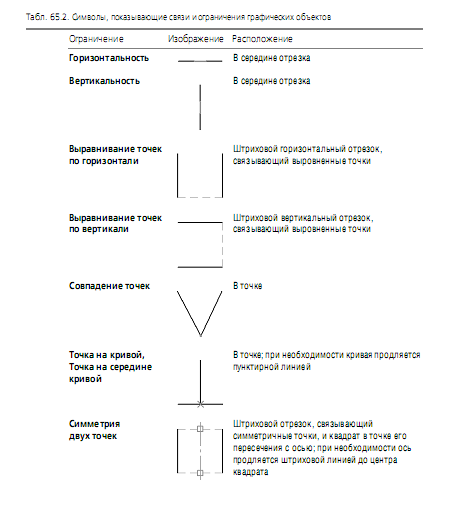
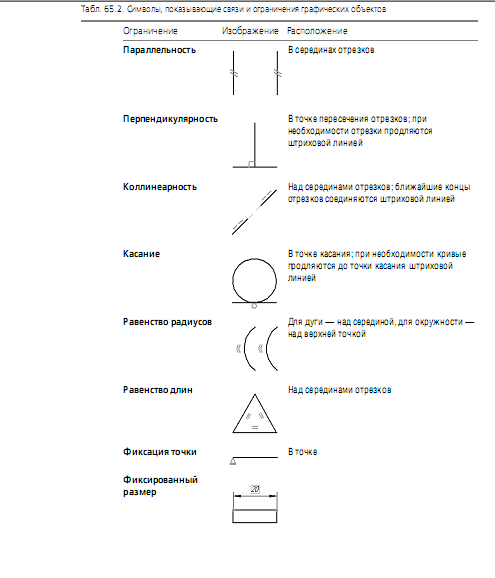
Чем параметрическое изображение отличается от обычного? Внешне ничем. Но параметрический чертеж хранит в себе информацию о взаимосвязях и ограничениях, наложенных на геометрические объекты, т.е. при изменении одного параметра, изменится и другой.

В Компасе такими связями будут – параллельность и перпендикулярность отрезков и прямых, стрелки взгляда, равенство длин отрезков.

Еще один вид параметрической связи – это *ассоциативность*. Такая связь возникает, один объект как бы привязывается к другому в процессе построения. Это размеры, штриховки, тех. обозначения – все они изменяются при перестроении базового объекта.

*Ограничение –* это зависимость между параметрами объекта или равенство объекта постоянной (константе).

Примеры ограничений – вертикальность и горизонтальность, перпендикулярность, симметрия отрезков, прямых, линии разреза, сечения, стрелок взгляда.

[](http://veselowa.ru/wp-content/uploads/2014/02/ogranicheniya.png)[](http://veselowa.ru/wp-content/uploads/2014/02/ogranicheniya-2.png)

   Параметризацию в Компасестоит применять в том случае, когда при модификации детали изменяются только размеры, а внешний вид (топология) остается неизменным. Или, когда создаваемая деталь будет служить прототипом для создания новых деталей.

**Задание.**

1. На Python с помощью API Kompas3D написать код программы, которые будет выполнять построение геометрического примитива с параметризацией в КОМПАС 3D.

**Ход работы.**

1. Открыть проект, созданный в ЛР№2;
2. Напишем функцию clear():

def clear(event):

caller=event.widget

caller.delete("0","end");

1. Найти в коде строчку:

frame.grid(),

и вставить после нее следующий фрагмент:

Title=Label(frame, text=u"Параметры профиля детали",borderwidth=2, width=45, height=0, fg='black', font=('gost type a', 16, 'bold', 'italic')).grid(row=0, column=0, sticky=W, columnspan=3)

wWidth = Entry(frame, width=10, borderwidth=3)

wWidth.bind("<FocusIn>", clear)

wWidth.grid(row=1,column=1,padx=5)

wWidthTitle = Label(frame, text=u"Ширина", borderwidth=2, width=27, height=0, fg='black', font=('gost type a', 16, 'bold', 'italic')).grid(row=1, column=0, sticky=E),

где Title – заголовок секции, wWidth – поле ввода переменной для ширины, а wWidthTitle – его заголовок;

1. Добавим поле для длины:

wLength = Entry(frame, width=10, borderwidth=3)

wLength.bind("<FocusIn>", clear)

wLength.grid(row=1,column=1,padx=5)

wLengthTitle = Label(frame, text=u"Длина", borderwidth=2, width=27, height=0, fg='black', font=('gost type a', 16, 'bold', 'italic')).grid(row=1, column=0, sticky=E);

1. Добавим функцию get():

def get(event):

global z

b=event.widget.curselection()

z=int(b[0])

return(z),

для получения параметров из интерфейса;

1. В функции KompasRun() добавим строчки после try:

global width

global length;

1. Достанем значения переменных с помощью get():

width =float(wWidth.get().replace(',','.'))

length =float(wLength.get().replace(',','.')),

и приведем все значения к типу float с перезаписью разделителя на точку (тем самым мы избавимся от возможных ошибок не правильного ввод);

1. Переходим в функцию ModelDetail() и находим там строчки:

obj = iDocument2D.ksLineSeg(0, 0, 0, 5, 1)

obj = iDocument2D.ksLineSeg(0, 5, 5, 5, 1)

obj = iDocument2D.ksLineSeg(5, 0, 5, 5, 1)

obj = iDocument2D.ksLineSeg(0, 0, 5, 0, 1),

заменяя числа на переменные width, length:

obj = iDocument2D.ksLineSeg(0, 0, 0, width, 1)

obj = iDocument2D.ksLineSeg(0, width, length, width, 1)

obj = iDocument2D.ksLineSeg(length, 0, length, width, 1)

obj = iDocument2D.ksLineSeg(0, 0, length, 0, 1);

**Контрольные вопросы.**

1. Для чего предназначена система КОМПАС?
2. Из чего состоит интерфейс системы КОМПАС 3D?
3. Для чего предназначена система Python.
4. Что такое «API Kompas3D»? В чем разница между API5 и API7.
5. Дать определение геометрическим примитивам. Какие примитивы Вы знаете?
6. Что такое параметризация? В чем заключается ее смысл?

# Заключение

В данной работе представлен алгоритм построения 3D модели параметризации 3D модели. Алгоритм реализован на языке Python в программе Компас 3D. Программа упрощает построение муфты с торообразной оболочкой с разными параметрами, что может быть полезным в ее использовании. В процессе выполнения работы были получены практические навыки работы с программой Компас 3D.

Поставленная задача выполнена в полном объеме.

# Список использованной литературы

|  |
| --- |
| 1. Горячев, А. В. Подсистема управления проектами в САПР / А. В. Горячев // Изв. СПбГЭТУ «ЛЭТИ». – 2010.-№. 6.-С. 41-46. |
| 1. Горячев, А. В. Управление знаниями в проектной деятельности / А. В. Горячев, Н. Е. Новакова. СПб.: Изд-во СПбГЭТУ «ЛЭТИ», 2012.-160 с. 2. *Цыганков Д.Э.* Автоматизация проектирования радиоустройств средствами Open CASCADE Technology / *Д.Э. Цыганков*, *А.Ф. Похилько* // Актуальные проблемы физической и функциональной электроники: материалы 18-й Всероссийской молодежной научной школы-семинара. – Ульяновск: УлГТУ, 2015. – С. 66-67. 3. *Дж. Ли, Б. Уэр.* Трёхмерная графика и анимация. — 2-е изд. — М.: [Вильямс](https://ru.wikipedia.org/w/index.php?title=%D0%92%D0%B8%D0%BB%D1%8C%D1%8F%D0%BC%D1%81_(%D0%B8%D0%B7%D0%B4%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D1%81%D1%82%D0%B2%D0%BE)&action=edit&redlink=1), 2002. — 640 с. 4. *Малюх В. Н.* Введение в современные САПР: Курс лекций. — М.: ДМК Пресс, 2010. — 192 с. — [ISBN 978-5-94074-551-8](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D1%83%D0%B6%D0%B5%D0%B1%D0%BD%D0%B0%D1%8F:%D0%98%D1%81%D1%82%D0%BE%D1%87%D0%BD%D0%B8%D0%BA%D0%B8_%D0%BA%D0%BD%D0%B8%D0%B3/9785940745518). 5. <http://www.sapr.ru/Article.aspx?id=7878> 6. ru.wikipedia.org 7. <http://www.sapr.ru/Article.aspx?id=7725> |