

МИНИСТЕРСТВО НАУКИ И ОБРАЗОВАНИЯ РОССИЙСКОЙ  
ФЕДЕРАЦИИ  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Рязанский государственный радиотехнический университет  
им. В.Ф. Уткина»

Кафедра космических технологий

«К защите»  
Заведующий кафедрой  
д.т.н., профессор  
\_\_\_\_\_ С.И. Гусев  
« \_\_\_\_ » \_\_\_\_\_ 20 \_\_\_\_ г.

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА БАКАЛАВРА**

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

на тему

**«Разработка математического и программного обеспечения генерации  
случайных тестовых заданий»**

Направление подготовки: 02.03.01 «Математика и компьютерные науки»  
ОПОП «Математика и компьютерные науки»

Студент	_____	(Чернобаев Д.А.)
Руководитель работы	_____	(Наумов Д.А.)
Руководитель ОПОП	_____	(Таганов А.И.)

Рязань, 2022 г.

## АННОТАЦИЯ

### АЛГОРИТМЫ МАРКОВА. ГЕНЕРАЦИЯ ЗАДАНИЙ, КОМПАС-3D, КОМПАС АРІ

Выпускная квалификационная работа изложена на 65 страницах. В работе представлены 65 рисунков, 10 использованных источников, 2 приложения на 21 странице и с 2 рисунками.

Целью выпускной квалификационной работы является разработка программного обеспечения генерации случайных тестовых заданий

В ходе выполнения работы была разработано программное обеспечение для получения ассоциативных чертежей на основе имеющейся модели после ее соответствующего изменения. Создана техническая документация и проведено тестирование системы.

## ABSTRACT

The final qualifying work is presented on 65 pages. The work presents 65 drawings, 10 used sources, 2 appendices on 21 pages and 2 drawings.

The purpose of the final qualification work is to develop software for generating random test items

In the course of the work, software was developed for obtaining associative drawings based on the existing model after its corresponding change. Created technical documentation and tested the system.

## СОДЕРЖАНИЕ

<b>ВВЕДЕНИЕ .....</b>	<b>7</b>
<b>1. ПОСТАНОВКА ЗАДАЧИ.....</b>	<b>9</b>
1.1. Цель РАБОТЫ.....	9
1.2. АКТУАЛЬНОСТЬ РАБОТЫ.....	9
<b>2. ИССЛЕДОВАТЕЛЬСКАЯ ЧАСТЬ .....</b>	<b>11</b>
2.1. НОРМАЛЬНЫЕ АЛГОРИТМЫ МАРКОВА.....	11
2.2. ПРИМЕНЕНИЕ АЛГОРИТМОВ МАРКОВА В ЗАДАЧЕ ГЕНЕРАЦИИ СЛУЧАЙНЫХ ТЕСТОВЫХ ЗАДАНИЙ .....	13
2.3. ВЫВОДЫ К РАЗДЕЛУ 2 .....	18
<b>3. ПРОЕКТИРОВАНИЕ СИСТЕМЫ ГЕНЕРАЦИИ ТЕСТОВЫХ ЗАДАНИЙ.....</b>	<b>19</b>
3.1 Архитектура системы генерации тестовых заданий.....	19
3.2 ХРАНЕНИЕ ИНФОРМАЦИИ О МОДЕЛИ ПРИ ПОМОЩИ ФАЙЛОВ XML .....	20
3.3 НАЗНАЧЕНИЕ И СТРУКТУРА СИСТЕМ АВТОМАТИЗИРОВАННОГО ПРОЕКТИРОВАНИЯ .....	22
3.4 ОТЛИЧИТЕЛЬНЫЕ ОСОБЕННОСТИ СИСТЕМЫ АВТОМАТИЗИРОВАННОГО ПРОЕКТИРОВАНИЯ «КОМПАС-3D» .....	24
3.5 «КОМПАС-3D» API .....	33
3.6 КРАТКИЙ ОБЗОР ЯЗЫКА ПРОГРАММИРОВАНИЯ PYTHON .....	38
3.7 БИБЛИОТЕКИ PYTHON ИСПОЛЬЗУЕМЫЕ В ПРОГРАММЕ .....	39
3.8 ОПИСАНИЕ ПРОГРАММЫ .....	40
3.9 БЛОК ВВОДА НАЗВАНИЯ МОДЕЛИ И ВЫВОД ИНФОРМАЦИИ О ПЕРЕМЕННЫХ МОДЕЛИ В ИНТЕРАКТИВНОМ РЕЖИМЕ.....	41
3.10 БЛОК ИЗМЕНЕНИЯ ПЕРЕМЕННОЙ В ИНТЕРАКТИВНОМ РЕЖИМЕ .....	43

3.11	Блок сохранения модели и создания ассоциативного чертежа в интерактивном режиме.....	45
3.12	Блок запуска программы в автоматическом режиме через командную строку с установленными параметрами.....	46
3.13	Блок случайного изменение переменной или изменение переменной несколько раз подряд в автоматическом режиме .....	50
3.14	Блок сохранения файлов в автоматическом режиме .....	52
3.15	Режим «помощь».....	53
3.16	Выводы по главе.....	54
<b>4.</b>	<b>РАЗРАБОТКА ПРОГРАММНОЙ ДОКУМЕНТАЦИИ.....</b>	<b>55</b>
4.1	Описание применения .....	55
4.1.1	Назначение программы.....	55
4.1.2	Условия применения .....	56
4.1.3	Входные и выходные данные .....	56
4.2	Руководство системного программиста.....	57
4.2.1	Общие сведения о программе.....	57
4.2.2	Сообщения программисту .....	57
4.2.3	Руководство пользователя.....	58
<b>5.</b>	<b>ТЕСТИРОВАНИЕ.....</b>	<b>61</b>
5.1	Создание чертежа с заданным значением .....	61
5.2	Создание чертежа со случайным значением .....	63
5.3	Создание группы чертежей.....	64
5.4	Выводы.....	65
	<b>ЗАКЛЮЧЕНИЕ.....</b>	<b>66</b>
	<b>СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ.....</b>	<b>ОШИБКА!</b>
	ЗАКЛАДКА НЕ ОПРЕДЕЛЕНА.	
	<b>ПРИЛОЖЕНИЕ А. ЛИСТИНГ ОСНОВНЫХ ПРОГРАММНЫХ МОДУЛЕЙ.....</b>	<b>69</b>

## **ПРИЛОЖЕНИЕ Б. РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ СИСТЕМЫ**

..... **88**

## **ВВЕДЕНИЕ**

Все высшие учебные заведения в мире, в том числе и в России, при предоставлении услуг обучения так же должны заниматься и проверкой полученных знаний студентов, следить за тем, как они усваивают материал, предоставленный преподавателем. Одним из способов такой проверки являются различные практические задания, разделенные на варианты, что позволяет в индивидуальном порядке проверить усвоение программы определенным студентом

В выпускной квалификационной работе я ставлю целью разработать такое программное обеспечение, которое позволит быстро и корректно формировать задания для проверок знаний работы в системе автоматизированного моделирования «Компас-3D» по дисциплинам «Инженерная графика» и «Компьютерная графика».

Предоставленная пояснительная записка для достижения упомянутой выше цели разделена на следующие разделы:

Раздел «Постановка задачи» описывает необходимость разработки программного обеспечения обработки изображений и актуальность данной темы ВКР.

Раздел «Исследовательская часть» использует алгоритмы Маркова для математического обоснования изменения формы модели при изменении конкретной переменной

В разделе «Проектирование системы генерации тестовых заданий» мы проводим анализ элементов разрабатываемой системы. Описываем структуру будущей системы. Создаем вручную и программно модель в САПР Компас-3D. Рассказываем о XML-файлах, как средство хранения упорядоченной текстовой информации и обосновываем их применение в нашей системе. Делаем обзор на используемый нами язык программирования Python. А также составляем схему нашего ПО, реализуем автоматический и интерактивный

подход к изменению модели, а также создаем режим, который обучает пользователя запуску программы через командную строку.

Раздел «Разработка программной документации» требуется для того, чтобы подготовить информацию о системных требованиях к используемому компьютеру, рассказать о возможных исключениях, связанных с некорректными данными системы и создать руководство пользователя

В разделе «Тестирование» нами проведена работа по проверке корректного исполнения нашей программы при различных вариантах режима и выбранной операции. В данном разделе мы убеждаемся, что программы выполняет поставленную перед ней задачу, по генерации заданий.



## **1. ПОСТАНОВКА ЗАДАЧИ**

### **1.1. Цель работы**

Целью выпускной работы является разработка программного обеспечения генерации случайных тестовых заданий для работы студентов в системе автоматизированного проектирования «Компас-3D». Выполнение поставленной цели заключается в реализации на практике определенных задач:

1) разработать алгоритм выбора типового варианта задания в системе «Компас-3D» для его последующего формирования в индивидуальное задание студента;

2) разработать алгоритм изменения выбранного типового задания при помощи нормальных алгоритмов Маркова;

3) разработать алгоритм визуализации изменений для оператора программы, а также последующего сохранения измененного документа в отдельном файле;

5) провести экспериментальные проверки разработанного программного обеспечения.

### **1.2. Актуальность работы**

Контроль за усвоением изучаемого материала, как и освоение практического применения полученных знаний, являются неотъемлемой частью учебного процесса при освоении любой дисциплины, как в заведениях высшего учебного образования, так и в иных учебных заведениях. Именно поэтому особую важность представляет собой индивидуализация таких заданий. Обучающиеся студенты зачастую имеют огромное количество возможностей, по использованию материалов, подготовленных такими же студентами, которые ранее обучались по их специализации, что приводит к некорректному отображению полученных знаний и не позволяет закрепить

полученный материал на практики. Следствием такой ситуации является малая компетентность выпускаемых специалистов по направлению их обучения.

Моя выпускная квалификационная работа ставит перед собой целью разработку математического и программного обеспечения для генерации случайных заданий студентам по дисциплинам «Инженерная графика» и «Компьютерная графика», созданная мной программа позволит индивидуализировать задания для каждого студента обучающимся этим дисциплинам. Таким образом, разработанное в данной работе ПО позволит в значительной степени решить проблему корректного контроля знаний обучающихся, а также поможет усвоить применение полученных знаний на практике в большей степени чем до этого.

## 2. ИССЛЕДОВАТЕЛЬСКАЯ ЧАСТЬ

### 2.1. Нормальные алгоритмы Маркова

Теория нормальных алгоритмов была разработана известным советским математиком А. А. Марковым на рубеже 40-50х годов XX века. Она потребовалась для того, чтобы представить еще одну формулировку такого понятия, как алгоритм и разрешить некоторые проблемы теории ассоциативных исчислений. Сам Марков в своих исследованиях называл их «алгорифмами». Суть алгоритма, представленного ученым, заключается в том, что он представляет собой определенные правила исходя из которых осуществляется переработка «слов» в каком-либо «алфавите». Считаю важным заметить, что в данном контексте «слово» и «алфавит» представляют собой абстрактную сущность, которую можно использовать для определения различных операций, к примеру перевода единичного числа (число которое представляет из себя набор единиц, «5=lllll»), в двоичное. В данном алгоритме, исследователь выделяет элементарную операцию, подстановку в слово в место одной его части другой.

Что же из себя представляют данные подстановки? Будем использовать в качестве нашего алфавита любое непустое множество. Его элементы являются буквами алфавита, а последовательность взятых из алфавита букв – это слово. Также допускается существование пустого слова, которое обозначается  $\lambda$ .

Рассмотрим использование алгоритмов Маркова на примере перевода двоичного не отрицательного числа в единичное, где единичное число характеризуется количеством палочек.

Возьмем алфавит, который будет состоять из трех букв:

$$A = \{0, 1, |\}$$

Далее нам потребуется составить правила, по которым наше двоичного число будет преобразовываться в единичное:

$$\left\{ \begin{array}{l} 1 \rightarrow 0| \\ |0 \rightarrow 0|| \\ 0 \rightarrow (\text{пустая строка}) \end{array} \right.$$

Выберем интересующее нас двоичное число и преобразуем его в единичное:

111

Выполнение:

1. 111 → 0|11
2. 0|11 → 0|0|1
3. 0|0|1 → 0|0|0|
4. 0|0|0| → 00|||0|
5. 00|||0| → 00||0||
6. 00||0|| → 00|0||||
7. 00|0|||| → 000||||||
8. 000|||||| → 00||||||
9. 00|||||| → 0||||||
10. 0|||||| → |||||

Как видим из примера, при выполнении алгоритма Маркова, прописанные нами правила применяются определенным образом. Применение алгоритма Маркова имеет следующую последовательность шагов. Сначала алгоритм ищет слева на право «подслово», которое удовлетворяло бы описанному правилу подстановки, после нахождения, он производит подстановку и начинает поиска заново, если же первое правило нельзя выполнить, то алгоритм переходит ко второму, а потом опять начинает сначала, так происходит до тех пор, пока ни одно из написанных правил нельзя

будет применять к имеющемуся слову, в таком случае алгоритм Маркова будет считаться выполненным.

Как мы видим на предоставленном примере, использование такого подхода при выполнении выпускной квалификационной работы по созданию программы, позволит нам четче реализовать процесс изменения типового варианта. Мы можем создать базовый «алфавит», буквами в котором будут являться операции над выбранным чертежом. Когда пользователь программы будет выбирать, что именно он хочет изменить и составлять «слово», то при его реализации одни изменения будут вызывать последующие модификации чертежа, что приведет к созданию нового задания, учитывающего все правила форматирования чертежей с сохранением их дальнейшей пригодности к реализации при создании непосредственно деталей.

## **2.2. Применение алгоритмов Маркова в задаче генерации случайных тестовых заданий**

Областью применения данной выпускной квалификационной работы являются задания для выполнения на лабораторных и практических занятиях по курсу «Инженерная и компьютерная графика». Исходными данными для каждого задания является некоторое графическое изображение – эскиз (рис.1), чертеж детали (рис.2) или изометрическое изображение трехмерной модели (рис.3).

Применить графическое изображение материала в сечении

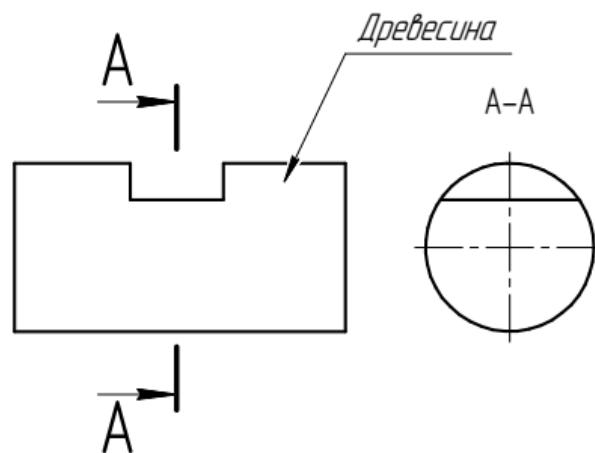


Рисунок 1 – Эскиз

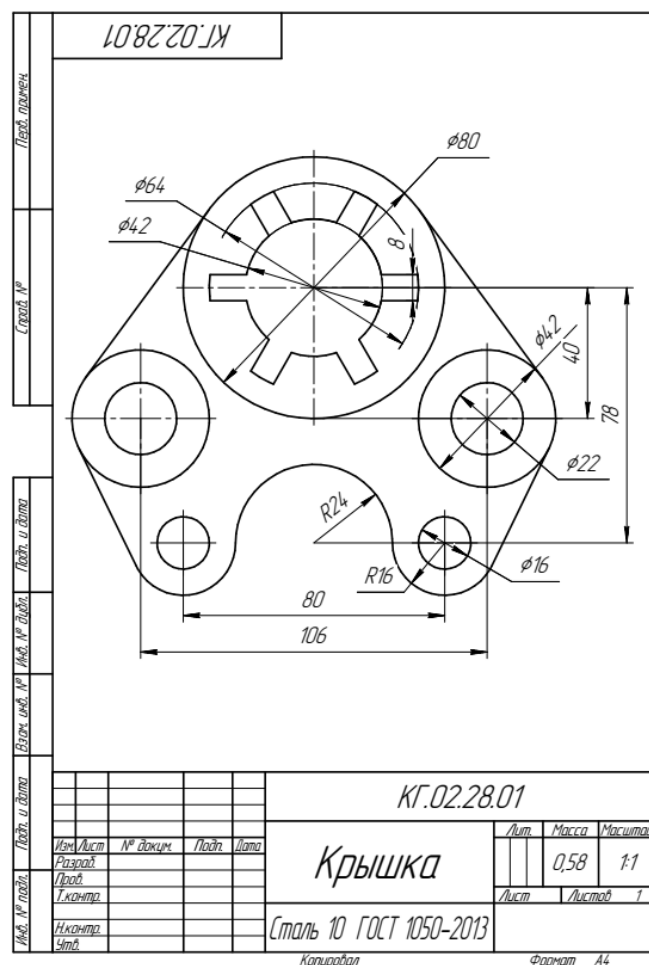


Рисунок 2 – Чертеж детали

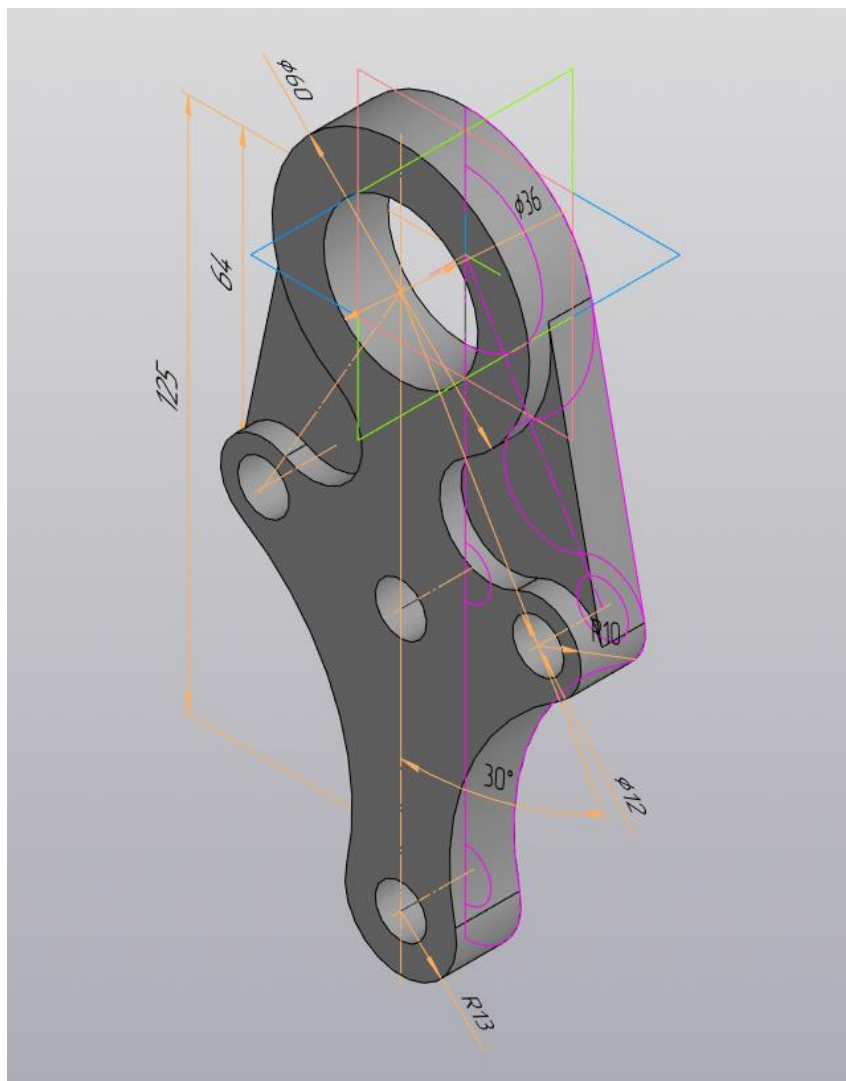


Рисунок 3 – Изображение трехмерной модели

В данной выпускной квалификационной работе алгоритмы Маркова могут быть использованы для следующих задач:

- формальное описание алгоритма генерации случайного эскиза, чертежа или модели «с нуля»;
- формальное описание алгоритма модификации существующего эскиза, чертежа или модели.

Для решения данных сначала задач необходимо задать алфавит.

Рассмотрим алфавит для генерации трехмерной модели:

- $F_0, F_1, \dots, F_n$  – обозначение плоских эскизов, задающих замкнутые или незамкнутые контуры, которые будут использоваться для формобразующих операций;
- $E_d^i, E_r^i, E_t^i, E_c^i$  – операции выдавливания на расстояние, операция выдавливания вращением, операция выдавливания по траектории, операция выдавливания по сечениям. Верхний индекс определяет, к какому эскизу применяется операция;
- $C_d^i, C_r^i, C_t^i, C_c^i$  – операции вырезания на расстояние, операция вырезания вращением, операция вырезания по траектории, операция вырезания по сечениям;
- $R^j, R_f^j$  – операции скругления и полного скругления;
- $Z^j$  – операция добавления фаски;
- $H^j$  – операция добавления простого отверстия;
- $A_c^j, A_r^j$  – операция создания массива элементов.

Буквы алфавита задают формообразующие операции, параметры которых могут:

- 1) выбираться случайным образом (например, направление выдавливания, расстояние выдавливания);
- 2) выбираться из некоторого списка значений (например, номинальный диаметр и шаг метрической резьбы);
- 3) определяться ранее полученными параметрами форм (например, длина и радиусы скруглений проточки для выхода шлифовального круга, ширина шпоночного паза).

Пример алгоритма Маркова для формирования детали, представленной на рисунке :

$$\left\{ \begin{array}{l} \lambda \rightarrow F_1 \\ F_1 \rightarrow \dot{F}_1 E_d^1 \\ \dot{F}_1 E_d^1 \rightarrow \dot{F}_1 \dot{E}_d^1 E_d^1 \\ \dot{F}_1 \dot{E}_d^1 E_d^1 \rightarrow \cdot \dot{F}_1 \dot{E}_d^1 \dot{E}_d^1 \end{array} \right.$$



В случае модификации существующей модели входным будет являться некоторое не пустое слово, определяющее исходную модель.

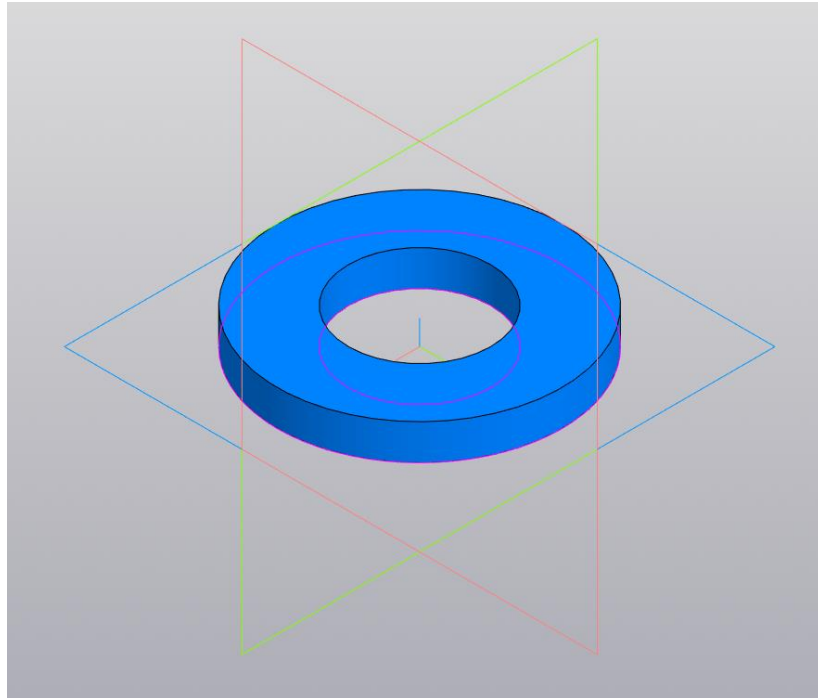


Рисунок 4 – Модель кольца

Для рисунка 4 входным словом будет  $F_1F_2E_d^1E_c^2$ , которое соответствует операции выдавливания базового эскиза 1 и операции вырезания эскиза 2 для создания центрального отверстия (примечание: в версиях КОМПАС 3D, начиная с v20, данные операции могут быть объединены в одну за счет указания не эскиза целиком, а области эскиза, в которой применяется операция).

$$\left\{ \begin{array}{l} *F_{-} \rightarrow F_{-} * \\ *E_{-} \rightarrow E_{-} * \\ \lambda \rightarrow * \\ E_{-} * \rightarrow .E_{-}E_3A_r^3R_f \end{array} \right.$$

Примечание: символ « $_{-}$ » в качестве нижнего индекса используется для описания того, что подходит буква с любым индексом и используется для сокращенной записи подстановок. Например, вместо группы подстановок:

$$\begin{cases} * F_1 \rightarrow F_1 * \\ * F_2 \rightarrow F_2 * \\ \dots \\ * F_n \rightarrow F_n * \end{cases}$$

может быть записано:

$$* F_- \rightarrow F_- *$$

В результате работы алгоритма могут быть сгенерированы варианты трехмерной модели со следующими изменениями:

- добавлен вырез паза на основе модели (рис.4);
- выполнено скругление двух ребер паза со случайными радиусами скругления;
- сформирован круговой массив из восьми, шести или двух элементов (рис.5 а, б, в).

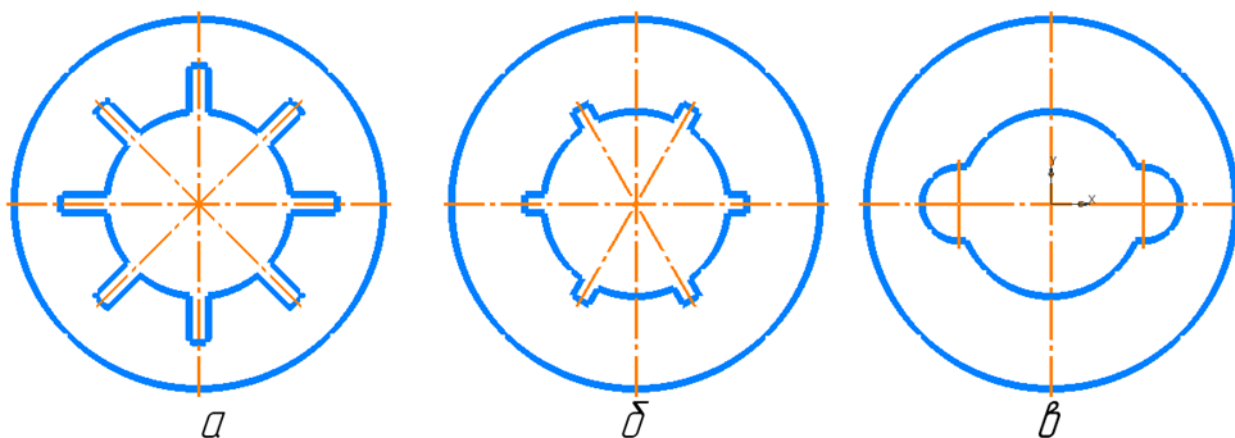


Рисунок 5 – Эскизы измененной модели (а – восемь вырезов, б – шесть вырезов, в – два выреза)

### 2.3. Выводы к разделу 2

В данной главе выпускной квалификационной работы нами был рассмотрен алгоритм Маркова, а также благодаря данному алгоритму мы смогли математически описать изменение форм детали в целом при изменении одного конкретного параметра.

### **3. ПРОЕКТИРОВАНИЕ СИСТЕМЫ ГЕНЕРАЦИИ ТЕСТОВЫХ ЗАДАНИЙ**

#### **3.1 Архитектура системы генерации тестовых заданий**

Просто программа, которая генерирует случайные тестовые задания для студентов не смогла бы выполнить свою задачу без дополнительных элементов. Данные элементы образуют систему (рис.6) по генерации задания. Которая представлена на рисунке. Входные данные представляют собой информационные сведения для программы, они указывают на файл, который требуется изменить (File.m3d), содержат информацию о переменных, которые могут изменяться в этом файле (File.xml) и дополнительную информацию, например, директории куда нужно сохранять новые файлы. Выходными данными являются наборы различных файлов, среди них: файлы с измененными моделями (File.m3d), и файлы с чертежами, которые построены на основе новых моделей (File.cdw), причем изменяя новые модели в будущем мы можем изменить и чертежи на их основе, а изменяя чертежи изначальные модели изменяться не будут. Сама программа не обладает требуемым функционалом, чтобы напрямую изменять получаемые файлы, поэтому она прибегает к возможностям САПР КОМПАС-3D и с помощью ее API передает нужные команды для открытия, изменения и сохранения файлов. Рассмотрим составляющие нашей системы.

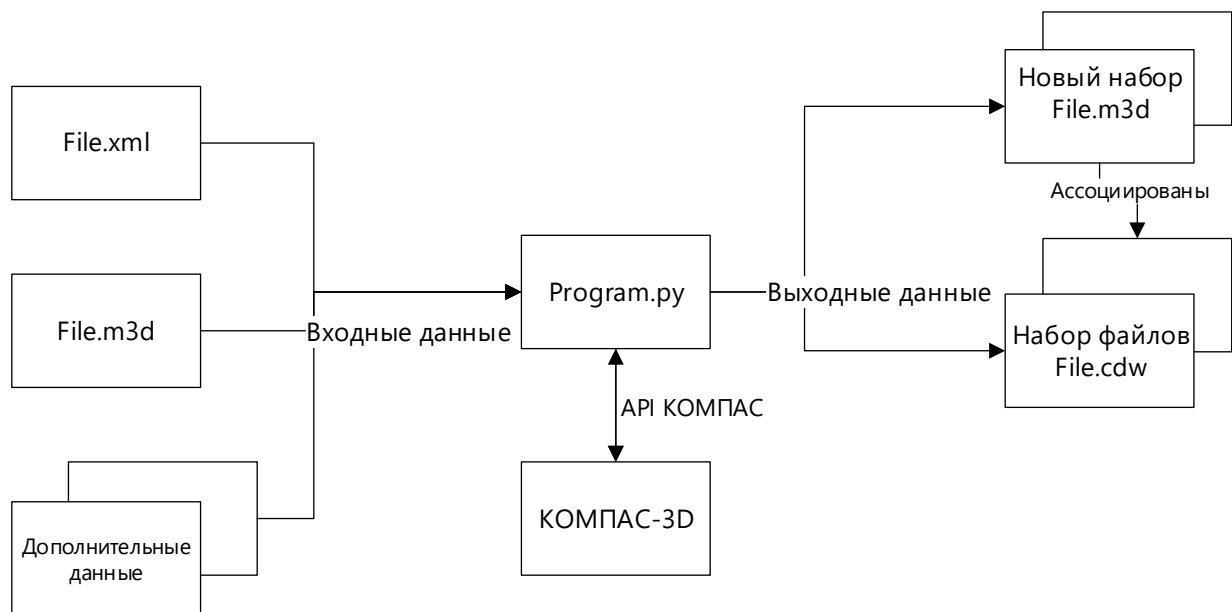


Рисунок 6 – Модель системы генерации задания

### 3.2 Хранение информации о модели при помощи файлов XML

Хранение информации является основой нашей жизни, человек постоянно собирает, сохраняет, изменяет окружающую его информацию. Одним из способов такого взаимодействия с информацией является использование электронных носителей.

Множество программ ежедневно обрабатывают миллионы гигабайт различной информации, которые хранятся на электронных носителях в разных форматах. Графическую информацию хранят в файлах разрешения .JPEG, .PNG и др. Для хранения видео используют форматы .mp4, .avi и др.

В нашей программе нам требуется формат для хранения текстовой информации, которая будет содержать данные о переменных 3D модели. Форматы документов, которые используются для хранения текстовой информации являются форматы: .txt, .doc, .html, .xml и др. Нами был выбран формат .xml, рассмотрим, что представляет из себя этот формат и почему он был выбран нами.

XML или eXtensible Markup Language – это язык расширяемой разметки, основным предназначением данного языка является создание логических

структур для данных, хранение этих данных и передача их в том виде, который будет являться удобным и для компьютера, и для человека. На рисунке 7 представлен обычный XML файл.

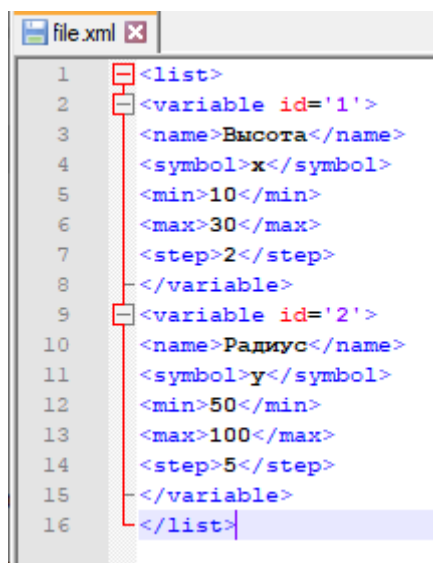


Рисунок 7 – Пример XML-файла

Структура XML документа очень проста. Она заключается в создании древа элементов, которое потом будет изменяться, храниться, считываться и т.д. Для определения отдельных объектов и их атрибутов в таких файлах используются теги, которые пользователь может задавать самостоятельно. Такие файлы очень распространены среди различных пользователей, так как могут хранить абсолютно любую информацию. В формате XML есть возможность записи различных баз данных, хранение настроек для приложений или, как это сделано в нашей системе, хранение информации о переменных, которые могут быть изменены в модели.

Данный формат представления информации был принят нами для реализации хранения информации о переменных по ряду причин:

- Очень простая и понятная древовидная структура документа
- Возможность самим установить обозначения элементов в структуре

- Наличие у языка программирования Python на котором разработана наша программа очень удобного модуля, для взаимодействия с такими документами, ElementTree.

### **3.3 Назначение и структура систем автоматизированного проектирования**

Системы автоматизированного проектирования или же САПР являются специализированным программным обеспечением, которое решает важные задачи оптимизации, унификации и автоматизации стандартных процедур, которые проходят выпускаемые товары. Использование САПР в условиях жесткой конкуренции позволяет производителям более привлекательно в отличии от их конкурентов, сократить время разработки нового продукта, что соответственно увеличит срок, в результате которого изделие морально устареет и ему на смену придут более современные аналоги.

Занимаясь поддержкой и развитием САПР на своем производстве предприятие в первую очередь преследует цель повышение качества своей продукции.

Пользователи современных САПР имеют в своем распоряжении богатый выбор инструментов, а также избавлены от многократного повторения уже выполненных вычислений. Современные САПР имеют мощную математическую составляющую, которая в значительной степени ускоряет производимые вычисления на базе этих систем. Наиболее известными представителями семейства САПР являются следующие программные продукты: AutoCAD (многоцелевая система для работы в различных областях), ArchiCAD (архитектурная система), ArCon (система планировки зданий, дизайна местности и интерьера), Splise (САПР электронных схем) и др.

К одной из самых востребованных функций САПР можно отнести возможность построения компьютерных 2-D и 3-D моделей, каталогизация

проектной документации. Именно благодаря этой возможности данные системы широко используют в машиностроении. К представителям САПР, которые акцентируют свое внимание на данных требованиях производителей, так же относится изучаемая на дисциплинах «Инженерная графика» и «Компьютерная графика» система «Компас-3D». На рисунке 8 представлен один из проектов, который был реализован с помощью этой системы.

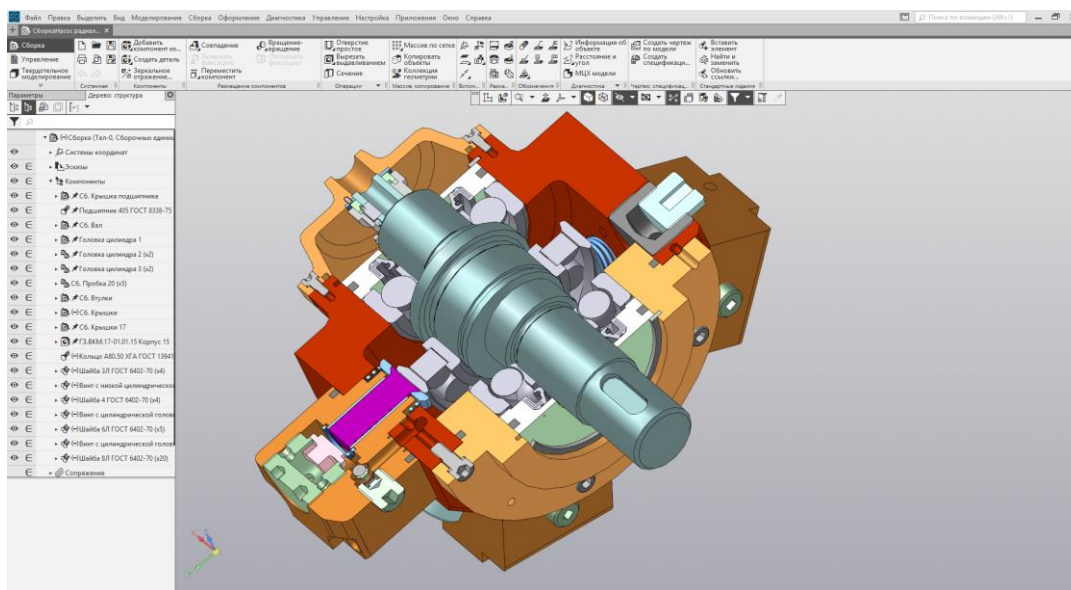


Рисунок 8 – Модель насоса

Структура современных систем автоматизированного проектирования (рис.9) является сложной и многоуровневой, она формируется за счет средств вычислительной техники, различных видов обеспечения и обслуживающего систему персонала. Согласно ГОСТУ 23501. 101-87 структура САПР включает в себя две подсистемы: проектирующую и обслуживающую.

Проектирующие модули отвечают за непосредственное выполнение конкретных проектных задач, в то время, как обслуживающая подсистема берет на себя задачи: управления проектированием, документирование, реализует графический интерфейс, создание и обслуживание базы данных.

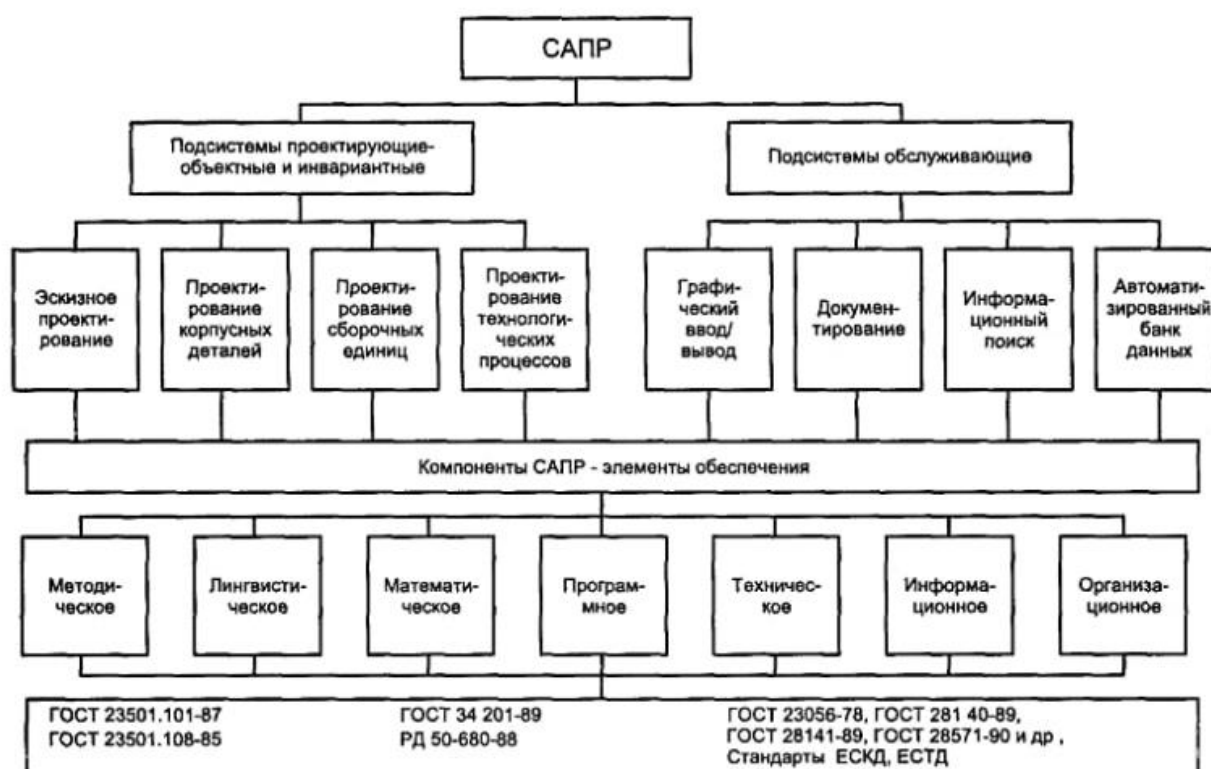


Рисунок 9 – Структура САПР

Абсолютно все подсистемы и компоненты системы должны быть совместимы друг с другом и решать поставленные задачи взаимодействуя. Также элементы современных САПР унифицируются, чтобы существовала возможность взаимозаменяемости компонентов. Возможность интеграции САПР с другими информационными системами является важной частью данного программного обеспечения, как и модифицируемость и пополнение уже разработанного продукта новыми компонентами.

### 3.4 Отличительные особенности системы автоматизированного проектирования «Компас-3D»

Как уже было отмечено в предыдущем пункте наиболее популярными являются САПР, перед которыми ставятся задачи по созданию 2-D и 3-D моделей. Одним из представителей данного вида систем является «Компас-3D», для которой мы пишем программное обеспечение по созданию индивидуальных студенческих заданий.



Эта система является очень популярной среди специалистов, занимающихся различным моделированием. В основном программа ориентирована на разработку моделей промышленного производства, но ее так же часто используют при разработке чертежей зданий и конструкций. Компас позволяет создавать проекты любой степени сложности. Программа предназначена для создания в первую очередь трехмерных параметрических моделей, ориентирована на формирование трехмерных моделей трехмерных тел, содержащих как типичные, так и нестандартные элементы конструкции.

Первая версия этой программы была разработана еще в 1989 году, однако доступ к ней мог получить лишь узкий круг специалистов. Коммерческая история ПО начинается с 1997 года, когда ее впервые представили на коммерческом рынке, с этого времени систему неоднократно улучшали и создавали новые версии. Сейчас к программе так же выпускаются несколько дополнений, благодаря которым система приобрела дополнительные возможности, что еще лучше упростило работу инженеров.

За время своего существования программа получила ряд особенностей, что разительно выделяет ее среди других систем, предназначенных для промышленного проектирования:

- Система имеет собственное математическое ядро, созданное разработчиками АСКОН;
- Благодаря своему простому интерфейсу, программа имеет достаточно низкий порог вхождения, и новички имеют возможность быстро перейти от изучения к созданию реальных моделей;
- Во время взаимодействия с другими программами по проектированию, созданные в Компасе продукты перемещаются без потери данных;
- Система поддерживает достаточно большое количество форматов передачи данных;

- Часть проектирования может происходить автоматически, что значительно упрощает работу специалиста;
- Так же на ПО можно вести разработку различных электрических цепей.

Абсолютно любая программа выделяется на фоне других аналогов различными плюсами, но также не освобождается от недостатков, которые отмечают пользователи САПР используя ее в своих целях.

Плюсы «Компас-3D»:

- Интерфейс данной САПР является очень простым и интуитивно понятным;
- Имеется встроенная библиотека различных моделей, которые можно использовать как при обучении, так и при проектировании;
- Весь интерфейс локализован на русском языке, что является существенным плюсом, это значительно упрощает обучение и использование программы в России;
- Стоимость лицензии является приемлемой, и практически каждый человек имеет возможность приобрести программу для своего пользования
- САПР учитывает свойства различных материалов при проектировании;
- Имеет большое количество форматов для выгрузки и загрузки файлов;
- Можно так же разрабатывать и 2D чертежи.

Недостатки:

- Визуализация разрабатываемых проектов остается на достаточно низком уровне;

- Иногда возникают проблемы совместимости разрабатываемых моделей на других программах с «Компас»;
- Система поверхностного моделирования так же находится на достаточно низком уровне и имеет определенные недостатки.

При всех перечисленных недостатках, можно смело сказать, что данная САПР охватывает огромный спектр разработок и очень полезна для широкого круга специалистов.

Рассмотрим интерфейс данной САПР:

При открытии программы Компас-3D, мы попадаем в главное окно (рис.10), здесь нам предлагают выбрать файл, с которым мы будем работать. Пользователь может выбрать из файлов, с которыми работал недавно, или же нажать на надпись: «Открыть...». Так же пользователь может создать новый проект, для определенных целей, форматы проектов представлены в центральной части.

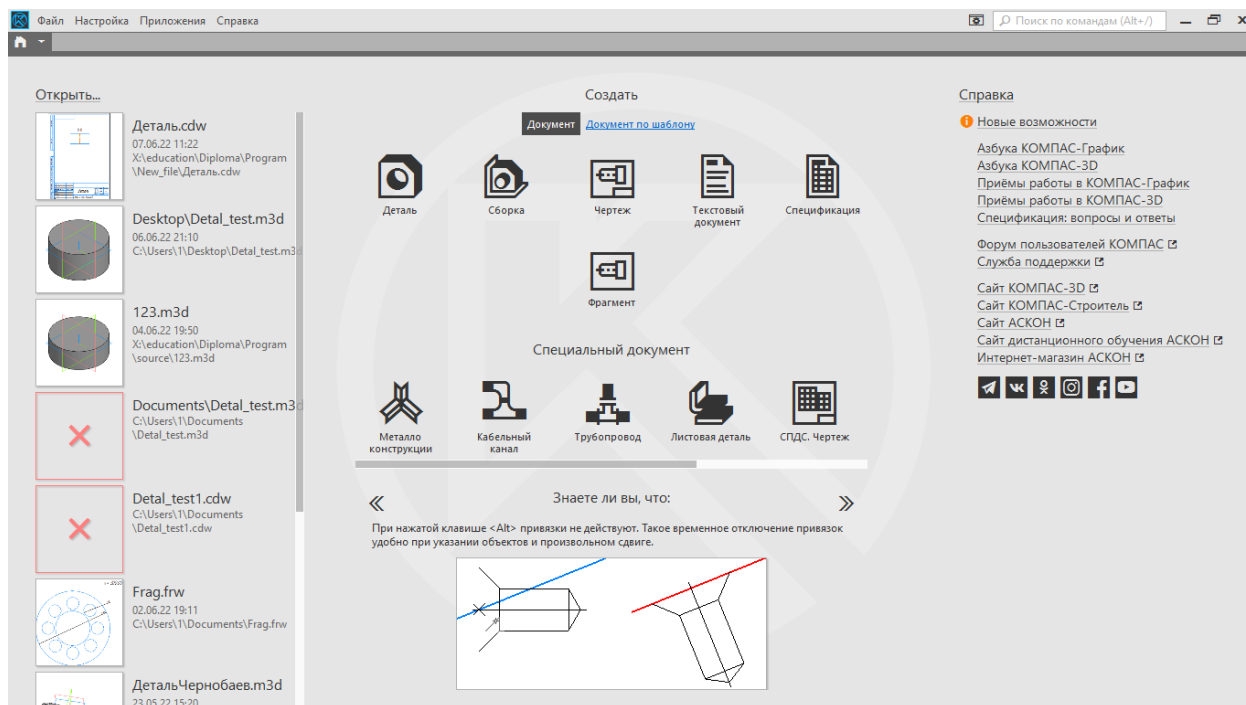


Рисунок 10 – Окно САПР Компас-3D

Создадим новый документ, в котором пользователь может создавать деталь. Для этого нажмем на кнопку «Деталь» в центральной части главного окна.

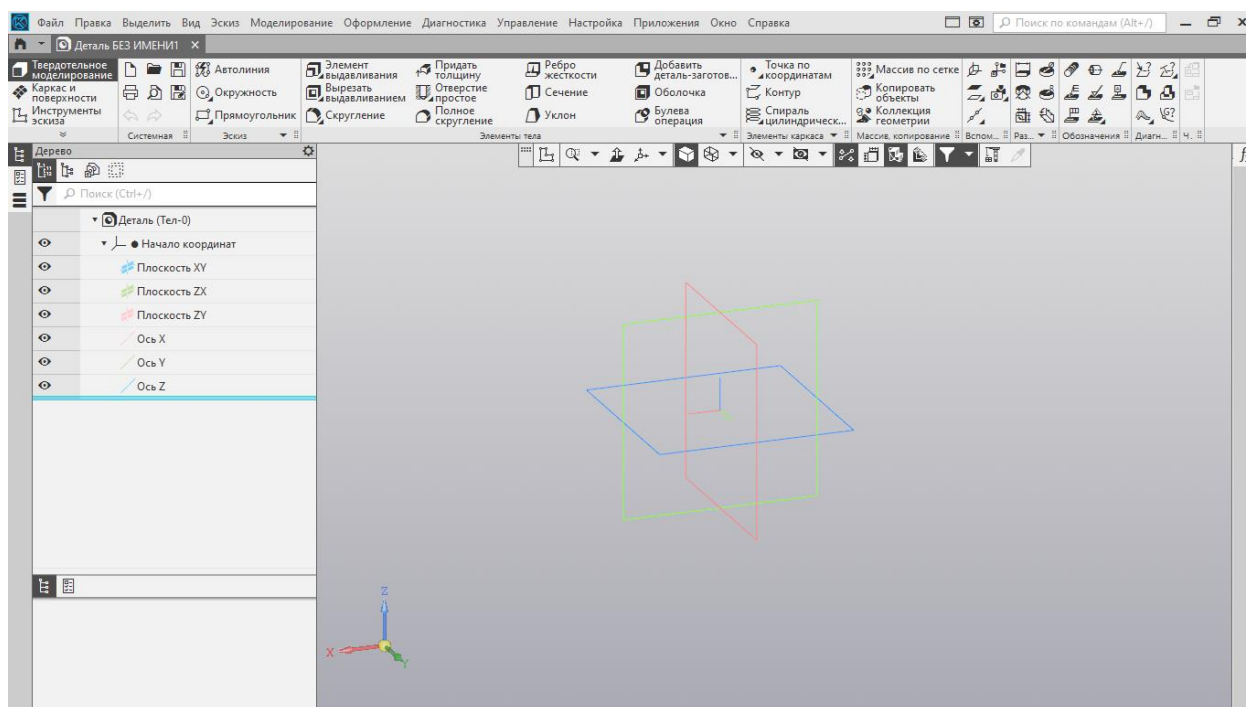


Рисунок 11 – Окно редактирования

Перед нами открывается окно изменения (рис.11) нашей детали, здесь мы можем создавать 3D-модели различной сложности. Рассмотрим процесс редактирования проекта на примере создания простого цилиндра с переменными, через которые мы сможем редактировать высоту и диаметр модели.

Для создания цилиндра в первую очередь нам требуется создать 2D-эскиз модели, построив окружность, которую в последствии мы сможем вытянуть в цилиндр. Для этого перейдем в режим эскиза (рис.12) нажав соответствующую кнопку на интерактивной модели и выберем плоскость относительно которой наш цилиндр будет перпендикулярен. Мной была выбрана плоскость ХУ.

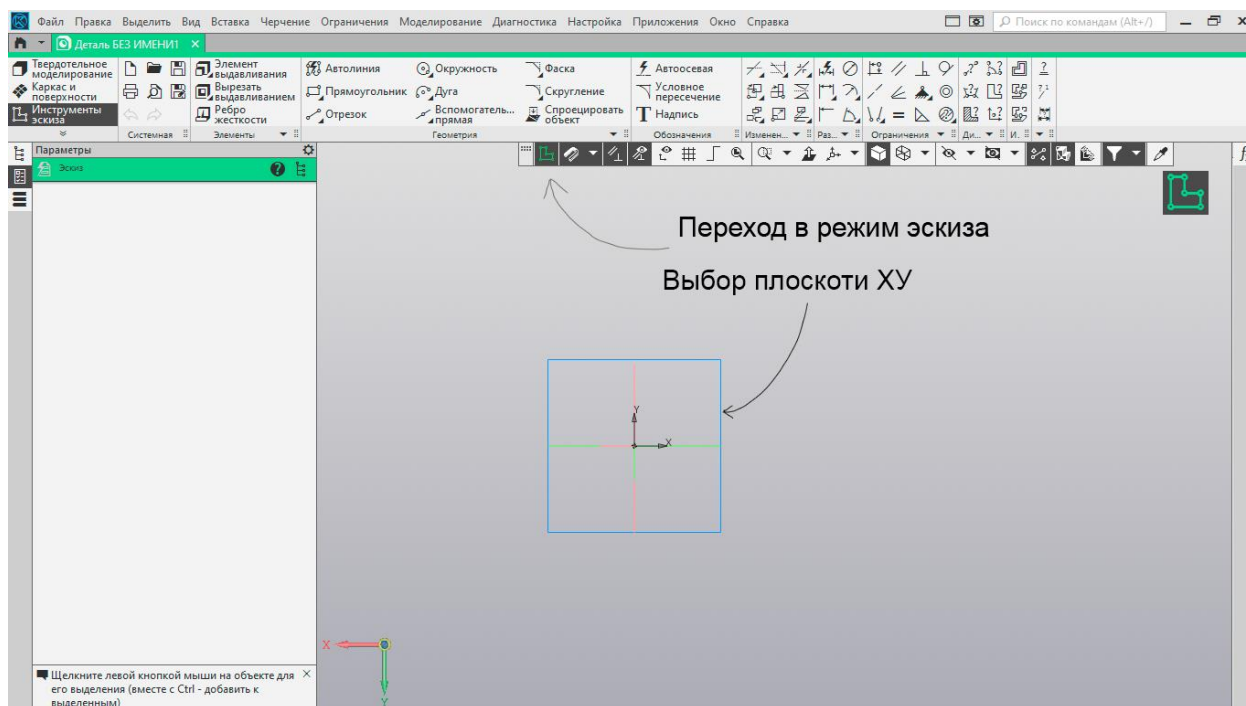


Рисунок 12 – Переход в режим эскиза

Нарисуем окружность (рис.13) с центром в начале координат. Диаметр окружности возьмем за 50мм.

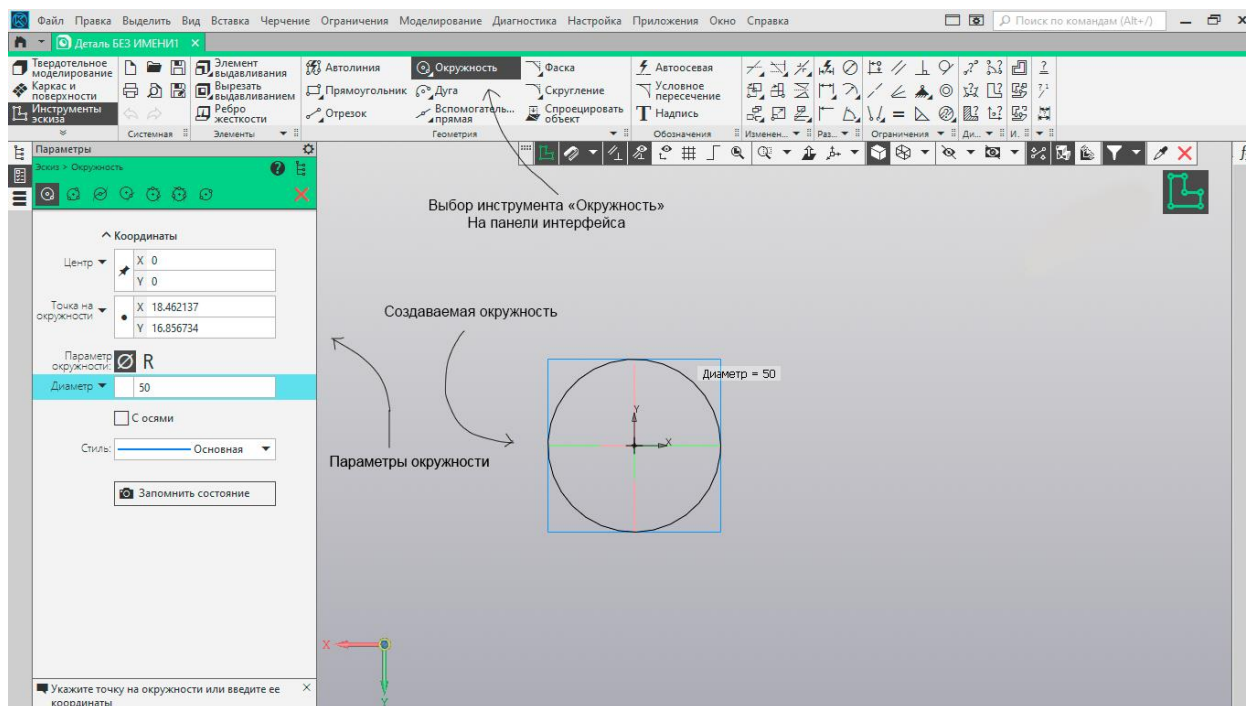


Рисунок 13 – Создание окружности

После создания окружности, необходимо установить ее диаметральный размер (рис.14), в будущем он понадобится нам, чтобы изменять значения диаметра у нашего цилиндра. После установки диаметрального размера, нам предлагают выбрать его длину (рис.15), оставим ее такой же, какой она была и выйдем из режима Эскиз.

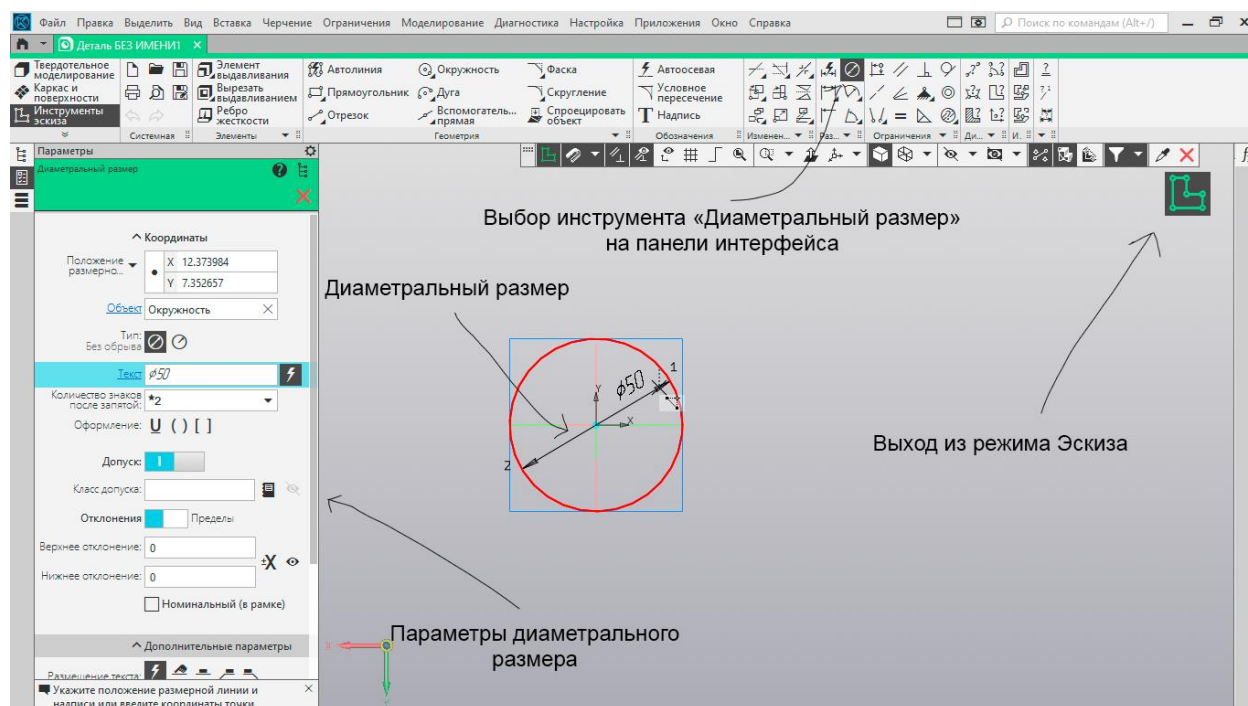


Рисунок 14 – Установка диаметрального размера

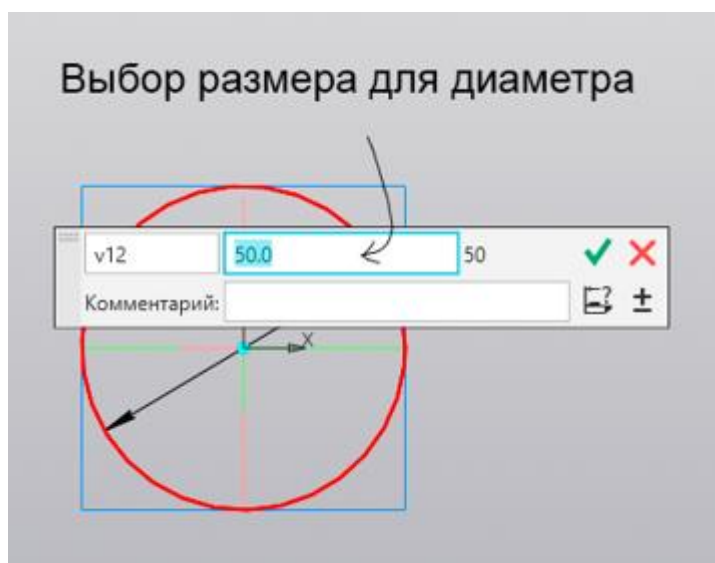


Рисунок 15 – Выбор размера для диаметра

Один из самых простых способов создания объемного объекта в компасе, является его создание с помощью выдавливания (рис.16). Мы также прибегнем к этому способу для того, чтобы создать нужный нам цилиндр. Мы можем осуществить выдавливание вручную, передвигая стрелочки, которые появятся над нашей окружностью или через панель параметров. Добавим на панели «Параметры» информацию о симметрии выдавливания относительно окружности, а также расстояние выдавливания в 10мм с каждой стороны. После установки параметров мы завершаем операцию нажатием клавиши «Enter» или же галочки на панели «Параметры». Далее откроем панель переменных, чтобы начать редактировать размеры нашего цилиндра.

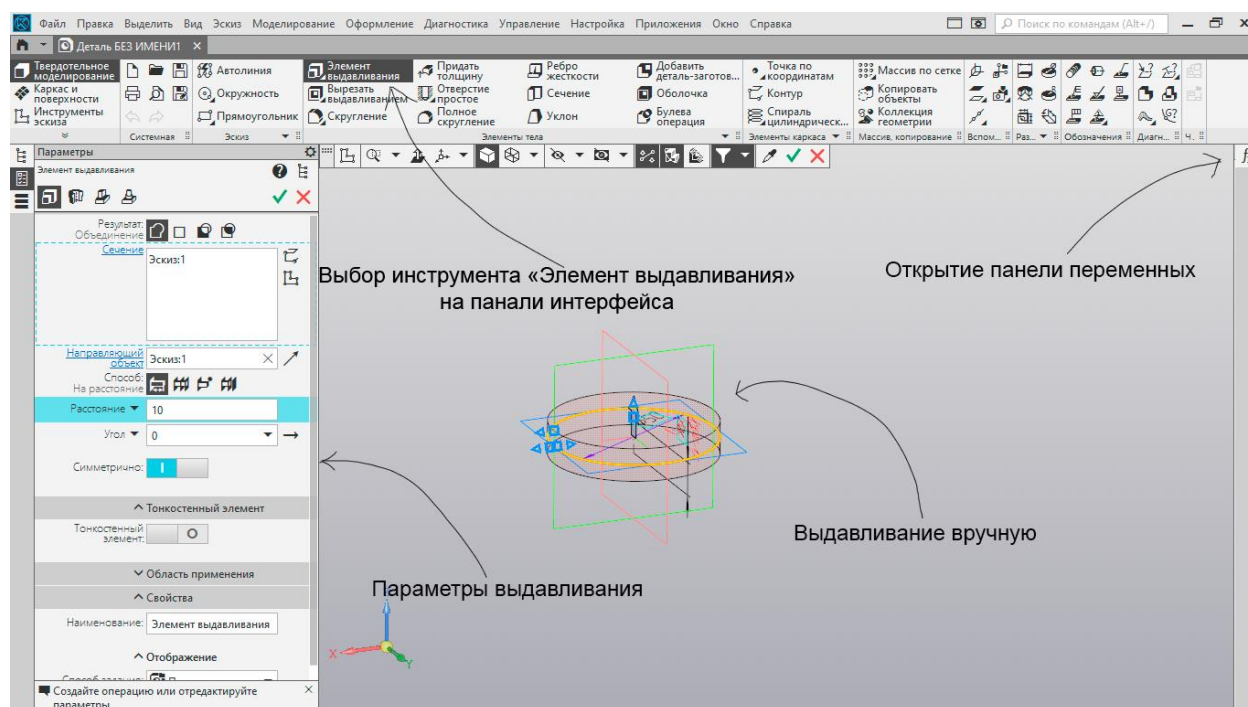


Рисунок 16 – Элемент выдавливание

На панели «Переменные» изначально указываются только переменные, которые были созданы автоматически при построении модели. Для цилиндра такими переменными будут являться v16(Высота) и v12(Диаметральный размер). Создадим также внешние переменные (рис.17) для детали введя названия переменных в пустой строке раздела «Деталь (Тел-1)», назовем их D и H и укажем произвольные значения 50мм и 35мм соответственно. Далее нам



необходимо связать переменные детали и переменные модели. Для этого в ячейке выражение напротив v12 поставим имя переменной D, а напротив v16 поставим имя переменной H. Теперь при изменении переменных детали будет изменяться и наша модель, чтобы увидеть изменения в окне программы пользователь должен перестроить модель, нажав на соответствующую иконку в окне программы. Так же сделаем переменные детали внешними, для этого правой кнопкой мыши щелкнем по переменной и выберем «Внешняя», это нужно для того, чтобы появилась возможность получать данные об этой переменной из других программ, например, Excel.

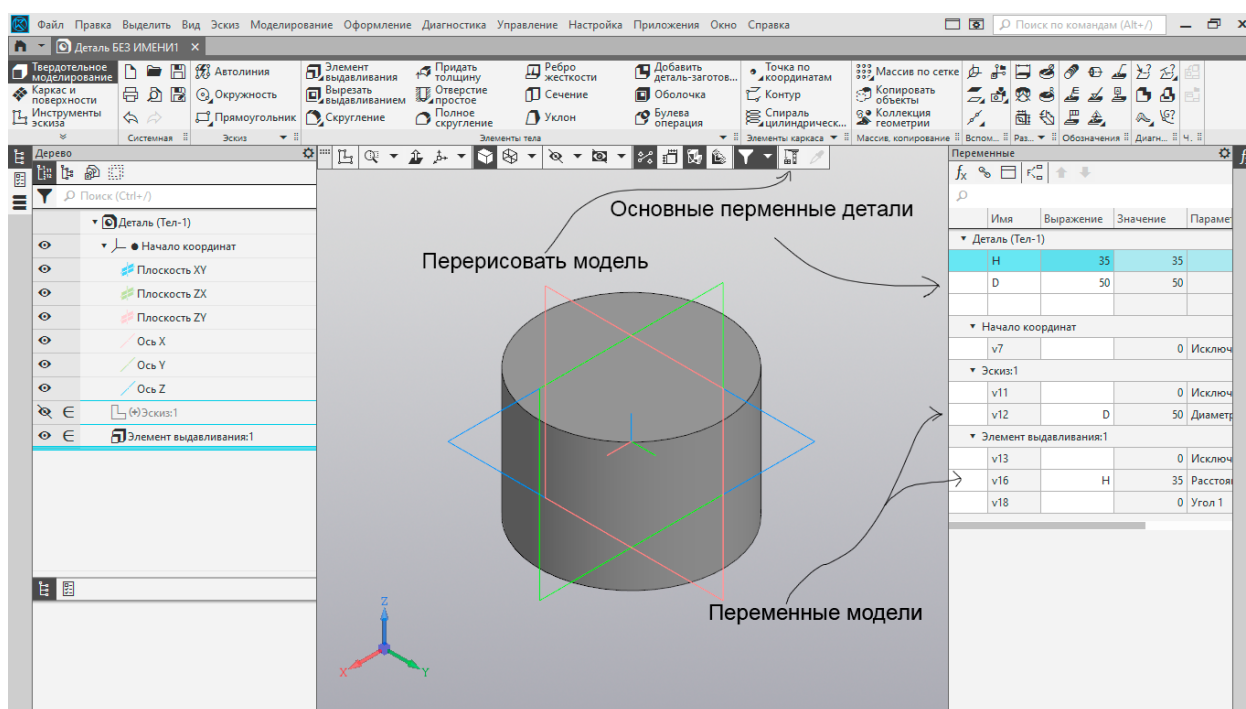


Рисунок 17 – Создание внешних переменных

Таким образом мы кратко рассмотрели интерфейс САПР «Компас-3D» на примере создания простого компаса. Помимо этого, ПО позволяет выполнить почти все наши действия автоматически создав с помощью API Компас программу.



### 3.5 «Компас-3D» API

API Компас это возможность для разработчика автоматизировать практически всю работу по построению сложных чертежей и моделей, сведя их к простому нажатию пары кнопок.

Чаще всего к реализации различных задач программным способом прибегают из-за того, что в CAD-системах нет реализации для различных практических задач. Как правило, такие задачи являются узкоспециализированными и применяются лишь на конкретном предприятии или подотрасли. Используя API, как способ реализации часто встречающихся задач, разработчики значительно сокращают время, которое им требуется затратить на разработку конкретного продукта. Также благодаря интерфейсу, предоставленного API можно интегрировать Компас с другими системами.

API Компаса представлено для разных языков программирования, что позволяет увеличить число программистов, которые могли бы с ним работать. В число языков программирования, которые поддерживает Компас входят: Delphi, C++, C#, Visual Basic, а также Python в виде модуля КОМПАС-Макро.

КОМПАС-Макро представляет из себя уже готовую среду для разработки ПО, которая интегрированная в Компас-3D. По-сути КОМПАС-Макро является обычной библиотекой, которую подключают к Компасу. Для начала работы с макросами, рядовому пользователю совсем не обязательно знать язык программирования Python, потому что среда позволяет просто записать действия пользователя с экрана, например, создание какого-то 3D-объекта, а программа сама интерпретирует его действия в код. После чего, готовый макрос можно применять уже в других проектах, если потребуется выполнить определенные действия.

Однако так же важно отметить, что запись макроса не является универсальным ответом, на задачу создания программ под Компас-3D. Когда мы находимся в состоянии записи макроса, то он не может отследить

изменение нами различных переменных или других параметров. И после остановки записи макроса, код для изменения значения переменной не появится. Для изменения переменных придется уже углубиться в программирование и взаимодействие с API.

Так же код записанный с помощью КОМПАС-Макро мы можем переместить в другую стороннюю программу. Использование этого инструмента облегчит работу разработчика, по поиску нужных ему функций при создании своей собственной программы и вместо ее поиска в справочной документации, можно просто записать макрос, который перепишет действия в работающий код.

Рассмотрим базовые модули команды API для взаимодействия будущей программы с САПР на примере создания такого же простого цилиндра, который был создан в предыдущем пункте.

В первую очередь для работы с API Компаса КОМПАС-Макро необходимо подключить соответствующие модули и библиотеки (рис.18).

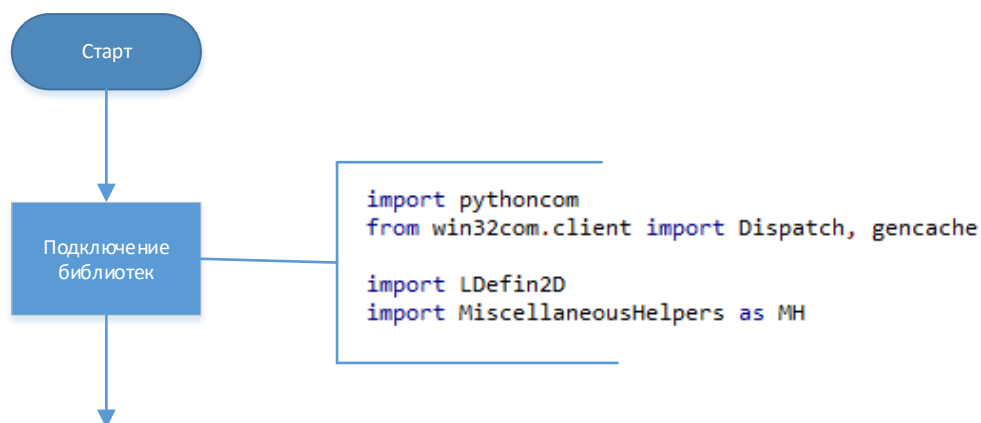


Рисунок 18 – Подключение модулей и библиотек в API

После подключения нужных библиотек для работы с API требуется получение констант и интерфейсов (рис.19), которые хранят в себе различные функции по редактированию модели. Получение интерфейса API5, не

считается обязательным, так как API7 используется, как его замена, но разработчики рекомендуют так же получать и его



Рисунок 19 – Подключение констант и интерфейса API

Теперь необходимо создать документ, получить активную модель и интерфейс компонента (рис.20), за это отвечает участок кода, который расположен на рисунке.



Рисунок 20 – Получение документа, создание нового и получение интерфейса документа

После всех выполненных действий можно перейти к непосредственному созданию нашей модели. Получаем интерфейс эскиза (рис.21) и создаем сначала окружность (рис.22), а после устанавливаем диаметральный размер (рис.23) для нее.

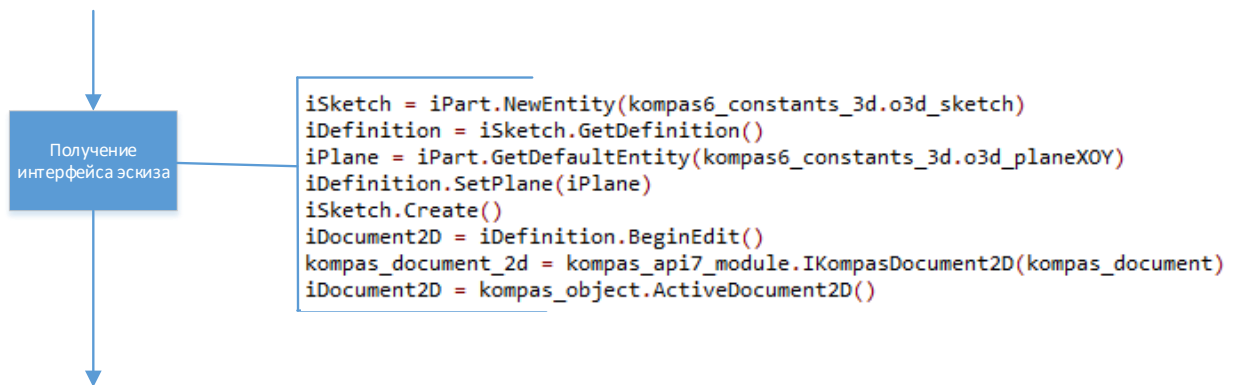


Рисунок 21 – Получение интерфейса эскиза



Рисунок 22 – Создание окружности

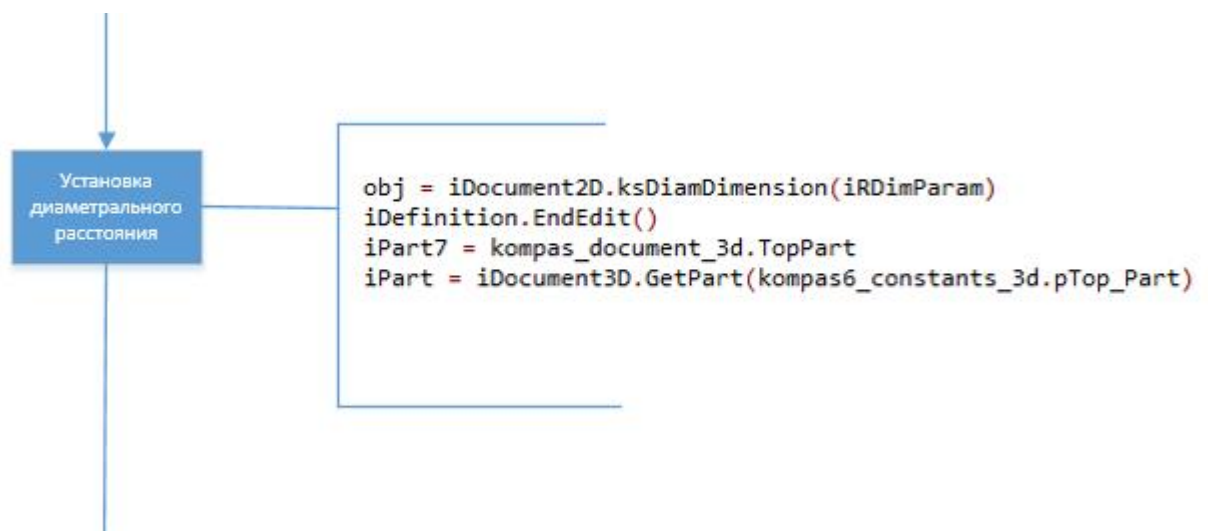


Рисунок 23 – Создание диаметра расстояния

После этого применяется выдавливание (рис.24) и создается уже нужный нам цилиндр с нужными характеристиками

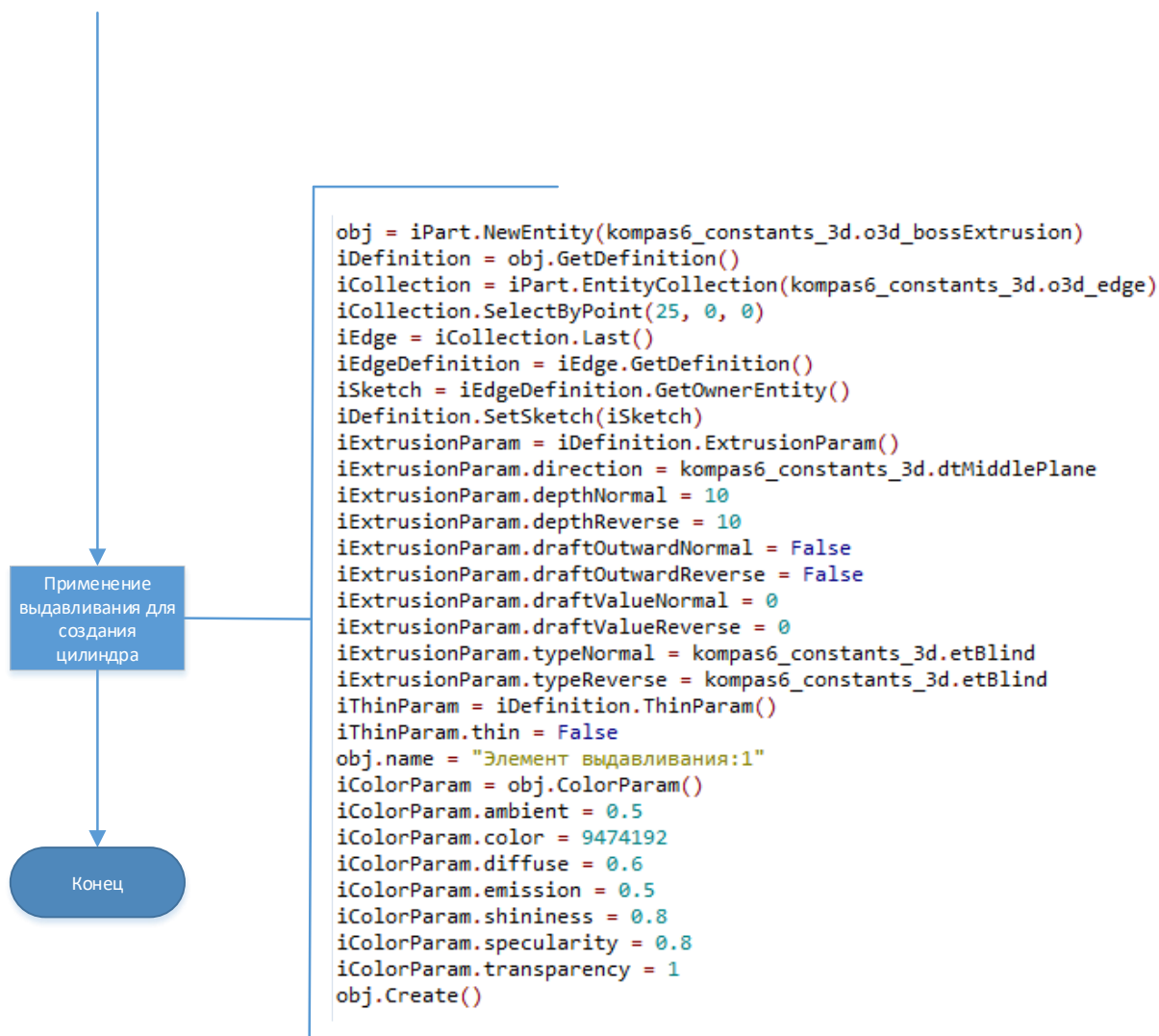


Рисунок 24 – Применение выдавливания

В данной программе мы не использовали создание переменных, так как нашей целью было показать работу КОМПАС-Макро и к сожалению, он не предоставляет такой возможности. Однако в API, предусмотрена работа с переменными, мы воспользуемся этим при создании нашего программного обеспечения

### 3.6 Краткий обзор языка программирования Python

Python – представляет из себя язык программирования высокого уровня. Он является одним из самых популярных языков программирования в мире. Благодаря своей много профильности этот язык программирования имеет большую популярность среди специалистов разных секторов IT-индустрии.

Помимо этого, данный язык так же обладает простым и лаконичным синтаксисом, что в значительной степени уменьшает порог вхождения для новых программистов. Зачастую код на Python занимает в разы меньше объема, чем на других языках программирования.

Так же стоит выделить огромное количество различных библиотек, которые были написаны разными программистами для этого языка. Практически для любой сферы деятельности можно найти библиотеку, которая будет очень полезна при разработке ПО. Есть библиотеки для работы со структурой файлов Windows, для работы со временем, изображениями, математическими функциями, созданием графического интерфейса и многое другое.

### **3.7 Библиотеки Python используемые в программе**

Рассмотрим библиотеки и модули языка Python, которые были использованы при создании программного обеспечения на выбранную мной тему работы.

Выбранные библиотеки и модули:

- Os – Модуль представляет огромное количество функций, чтобы программа могла взаимодействовать с операционной системой, причем их поведение не зависит от типа ОС.
- ElementTree – Данный модуль реализует работу API для взаимодействия языка программирования и XML файлов.
- win32com – это "компонентная объектная модель" части pywin32, которая в свою очередь является базовым модулем от которого зависит написание COM-компонентов и серверов
- LDefin2D – библиотека для API Компас
- MiscellaneousHelpers – библиотека для API Компас

### 3.8 Описание программы

В процессе выполнения выпускной квалификационной работы было создано программное обеспечение, которое призвано выполнять задачу по генерации новых индивидуальных заданий для студентов, на основе стандартных моделей.

Данная программа выполнена в одном модуле, код программы представлен в приложении А, схема программы, работающей в интерактивном режиме (рис.25) (запуск режима через командную строку или при открытии файла двойным щелчком) представлена на рисунке., а схема программы, работающей в автоматическом режиме (рис.26) (запуск режима через командную строку) представлена на рисунке и запуск режима (запуск режима через командную строку) помощи (рис.27) по запуску программы через командную строку.



Рисунок 25 – Интерактивный режим



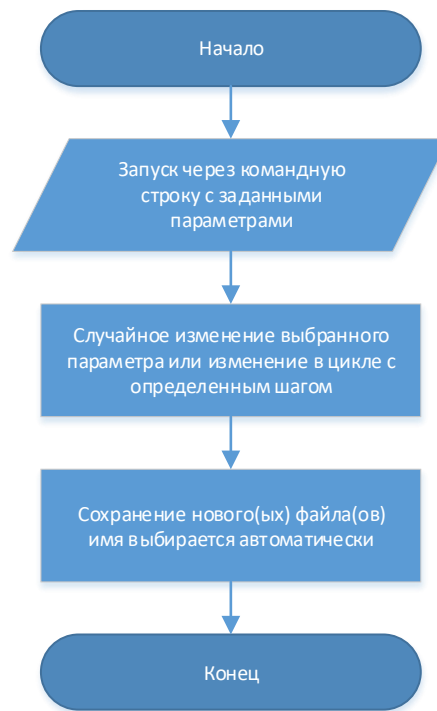


Рисунок 26 – Автоматический режим

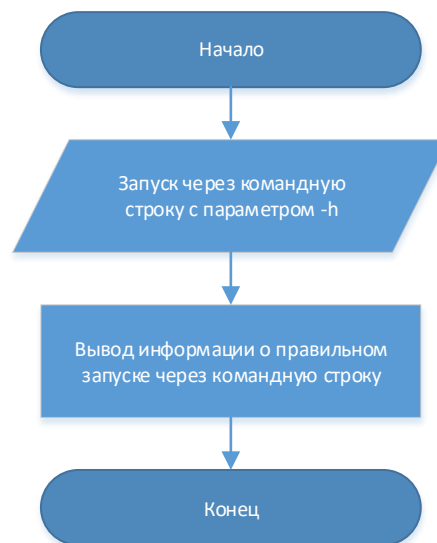


Рисунок 27 – Режим «Помощь»

### 3.9 Блок ввода названия модели и вывод информации о переменных модели в интерактивном режиме

Начало любой программы начинается с получения данных, которые требуются для выполнения основной задачи. Организуем получение информации нашей программой конкретных данных, которые нужны пользователю, а также вывод информации о модели на экран. Но прежде

подключим нужные библиотеки, модули и откроем КОМПАС (рис.28), получим его интерфейсы.

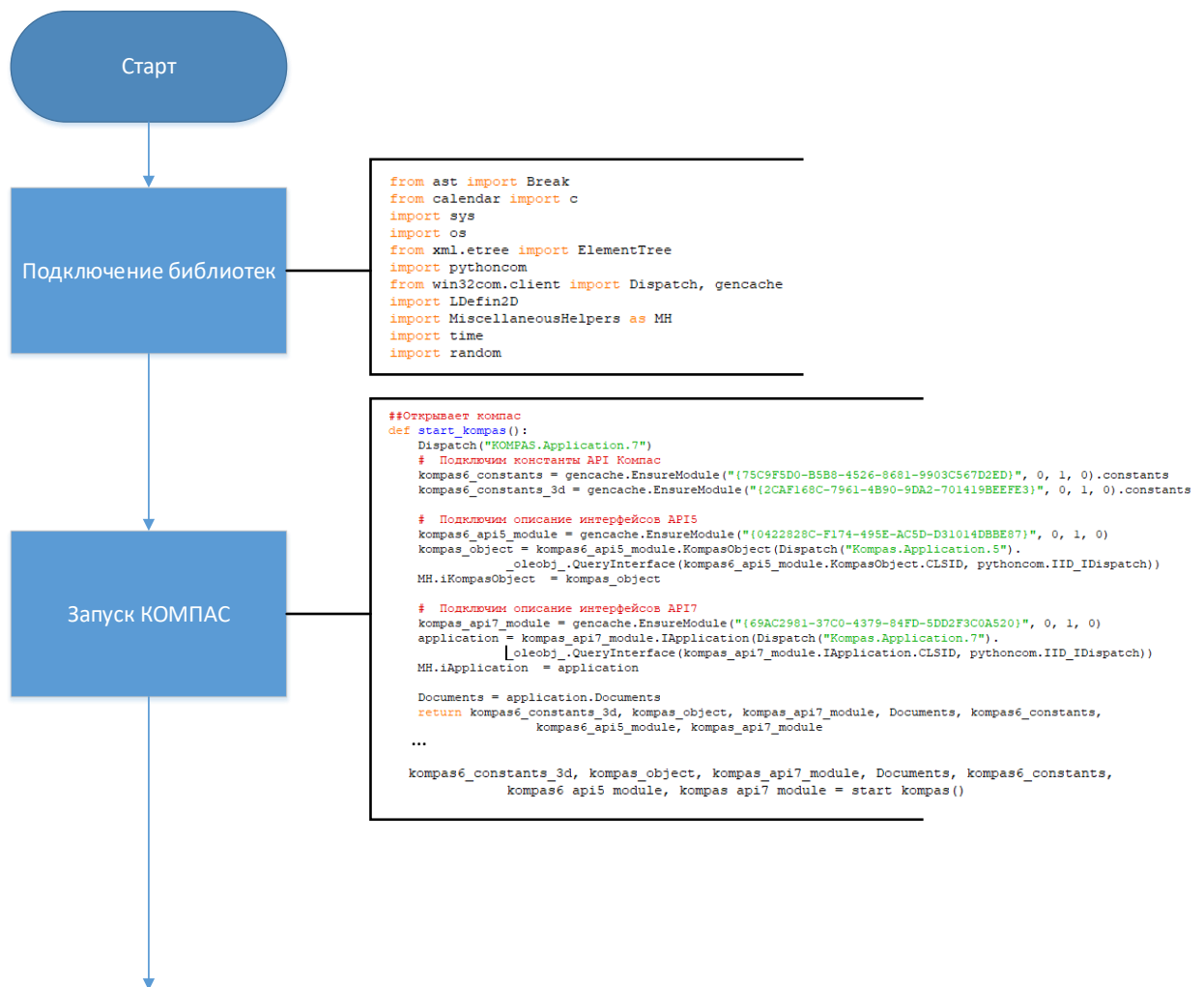


Рисунок 28 – Подключение библиотек, запуск Компас.

После подключения требуемых библиотек получим от пользователя названия нужных файлов, откроем документ с нужной моделью и отобразим информацию о количестве изменяемых в нем переменных на основе данных из XML-файла и создадим древо XML (рис.29), чтобы после взаимодействовать с ним.

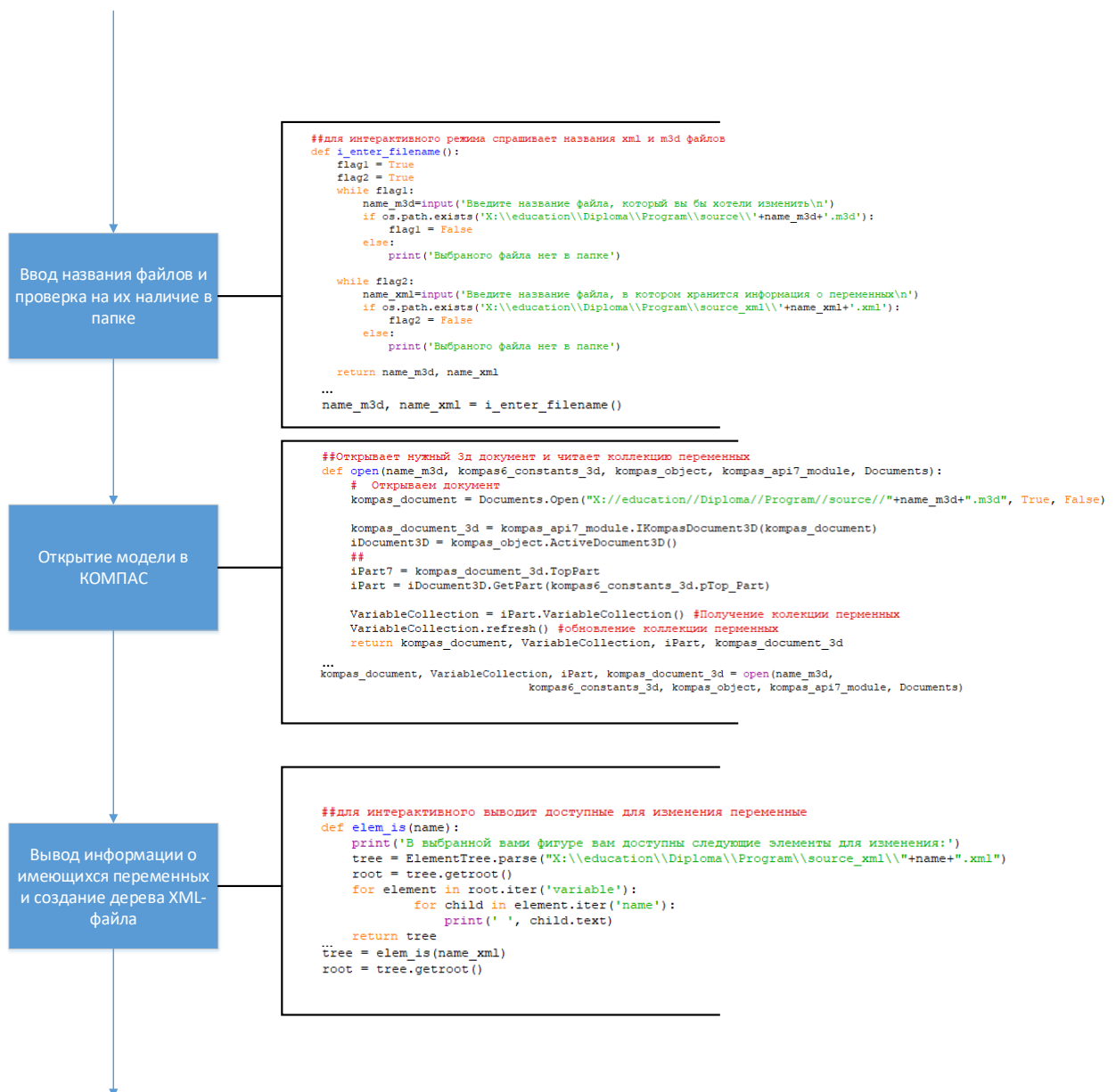


Рисунок 29 – Ввод названий, открытие модели, вывод информации о переменных и создание XML-дерева

Далее перейдем к изменению переменной(ых), которые хотел бы изменить пользователь

### 3.10 Блок изменения переменной в интерактивном режиме

Узнаем у пользователя название переменной, которую он хотел бы изменить, получим ее древо и выведем из него информацию о названии, обозначении, минимальном и максимальном значении, шаге изменения переменной (рис.30).

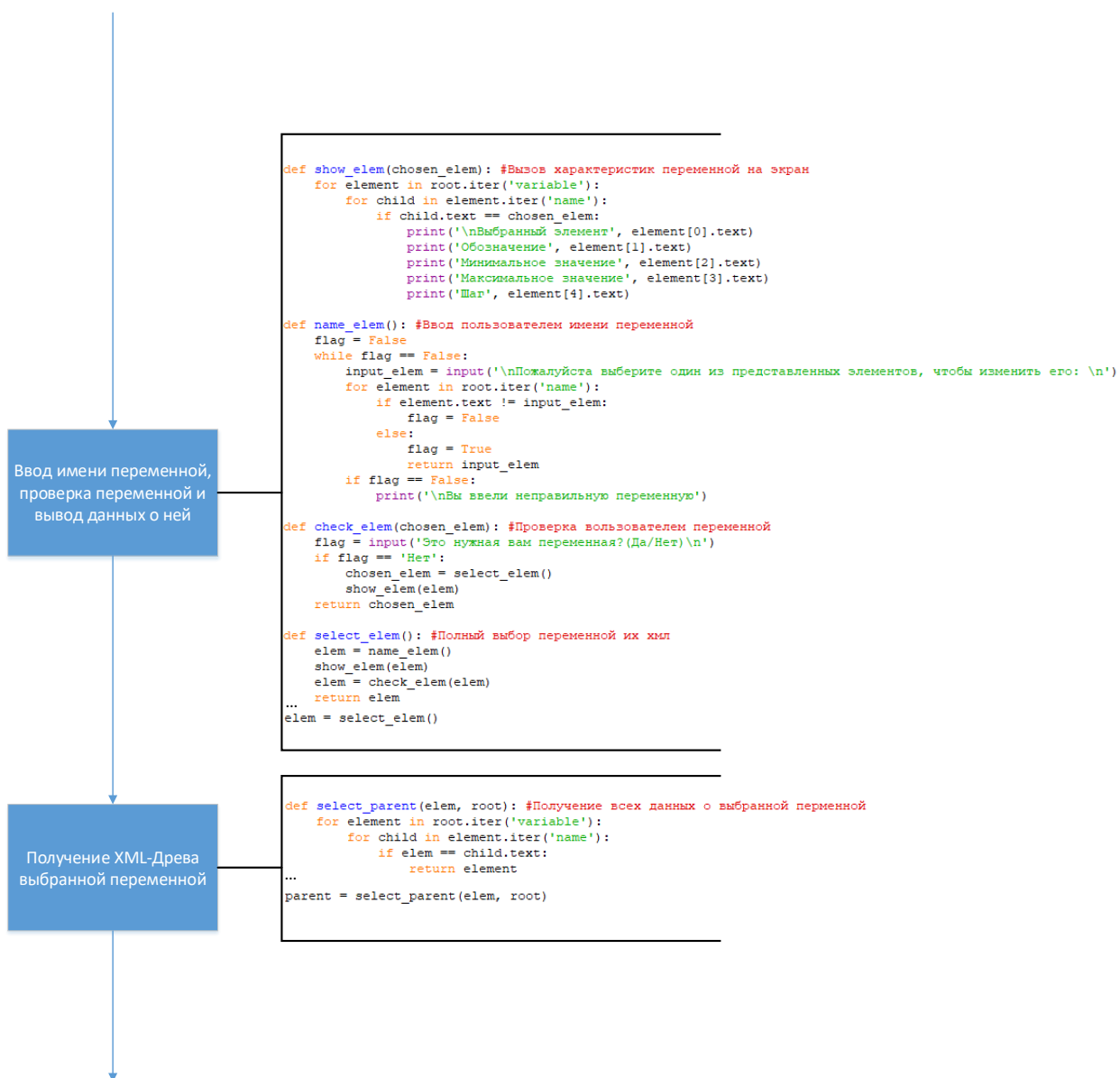


Рисунок 30 – Ввод имени величины, вывод информации о ней, подтверждение выбора и создание древа переменной

После подтверждения переменной от пользователя нам необходимо перейти к непосредственному изменению переменной. Для пользователя мы выводим информацию о нынешнем значении переменной, информацию о том, как она меняется и предлагаем ввести новое значение (рис.31).

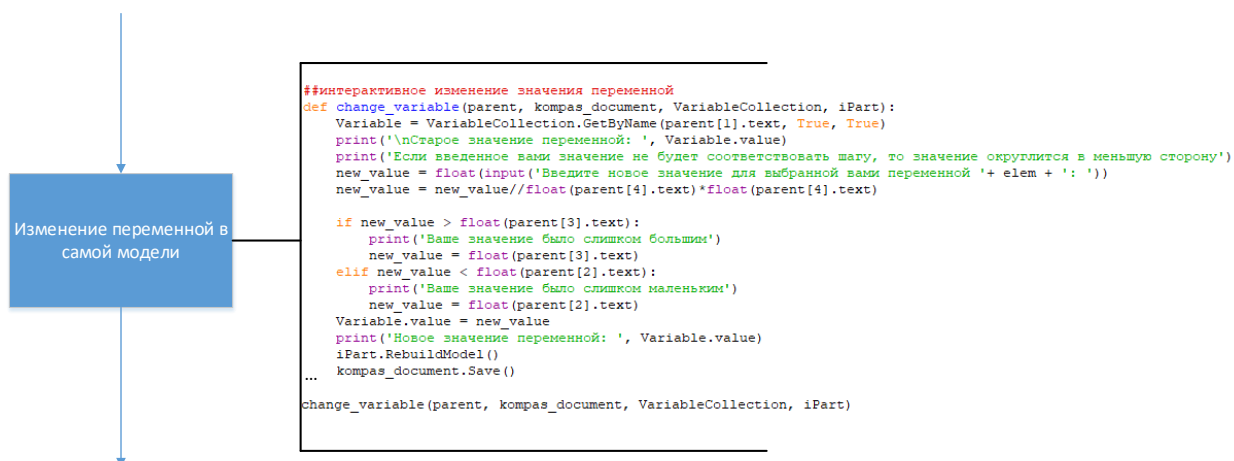


Рисунок 31 – Изменение переменной в модели

После изменения мы предлагаем пользователю продолжить изменять переменные в модели, в таком случае мы возвращаемся к началу этого блока, или же перейти к сохранению документа и созданию ассоциативного чертежа для него (рис.32).

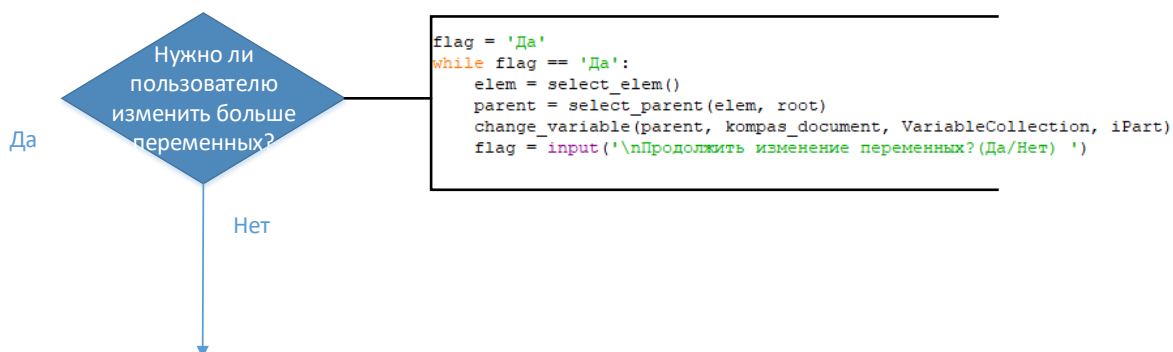


Рисунок 32 – Выбор продолжения изменения переменных

Когда пользователь изменил все переменные, которые ему были нужны, мы можем перейти к блоку сохранения модели и создания ассоциативного с ней чертежа

### 3.11 Блок сохранения модели и создания ассоциативного чертежа в интерактивном режиме

В данном блоке, нами описана функция для создания ассоциативного чертежа на основе имеющейся модели и его сохранения под удобным для пользователя именем в памяти (рис.33). Модель детали никак не будет

зависеть от чертежа, а чертеж будет иметь возможность перерисовки, если вдруг наша модель будет в будущем изменена в результате каких-либо действий, не обязательно связанной с нашей программой.



Рисунок 33 – Сохранение чертежа

После того, как наш чертеж создан мы сначала закрываем наш чертеж и модель, а только после этого закрываем программу полностью. Интерактивный вариант программы направлен на то, чтобы пользователь программы мог лично участвовать в изменении моделей так, как он посчитает нужным. Для работы в этом режиме необходимо два раза кликнуть на программу или же открыть ее через командную строку с параметром «-i». Далее рассмотрим автоматический режим программы.

### 3.12 Блок запуска программы в автоматическом режиме через командную строку с установленными параметрами

Автоматический режим нужен в том случае, если пользователь уже знаком с переменными и хочет получить для них случайные значения или же создать множество чертежей со небольшими изменениями заданной переменной. В данном блоке пользователь через параметры должен передать

сразу название файла, где находится модель, название XML-файла, где лежит информация о переменных модели, а также способ, которым он хочет изменить модель. На выбор предоставляется два способа: изменение модели путем задания выбранной переменной случайной величины (получается один чертеж) и изменение модели, путем многократного изменения выбранной переменной с определенным шагом (создается много чертежей)

Все параметры, которые мы получаем через командную строку создают из себя кортеж строковых элементов, который переходит в `sys.argv` за этот кортеж отвечает модуль `sys` и после ввода параметров в командную строку нам необходимо обработать данный массив (рис.34) таким образом, чтобы программе было удобнее работать с полученными параметрами.

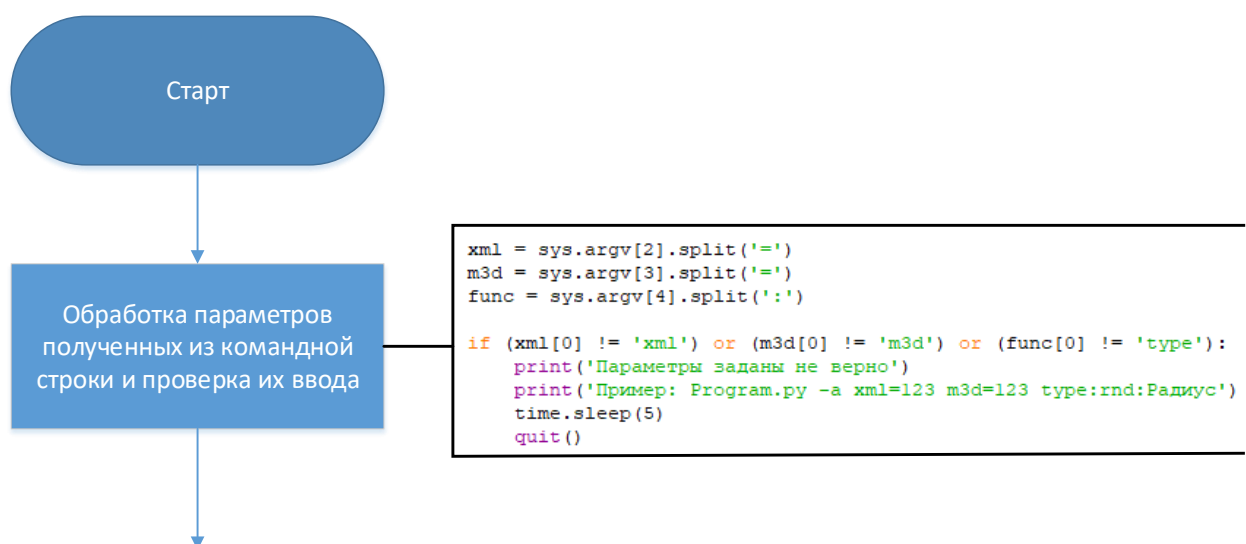


Рисунок 34 – Обработка входных параметров

После того, как мы преобразовали переменные мы должны открыть КОМПАС и подключить интерфейс (рис.35).

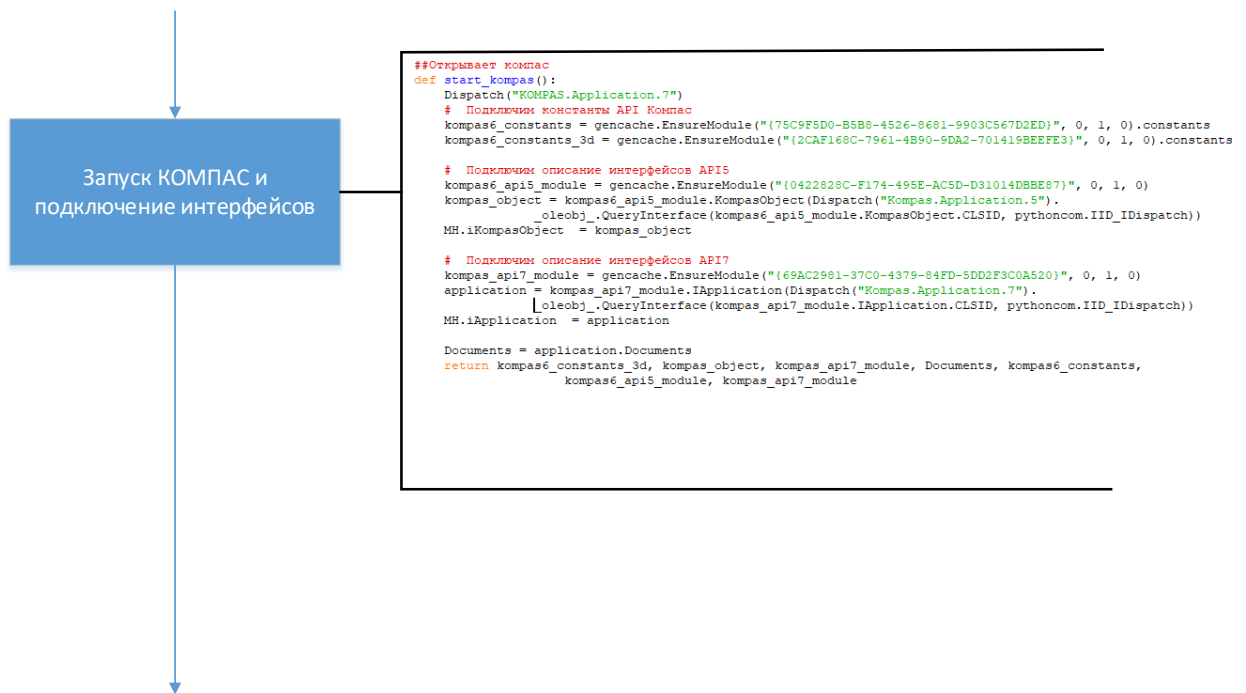


Рисунок 35 – Запуск КОМПАС и подключение интерфейсов

Далее нам необходимо получить из имеющихся массивов переменные имен для файлов xml и m3d, для того, чтобы в последствии нам было легче получить эту информацию для работы программы (рис.36).

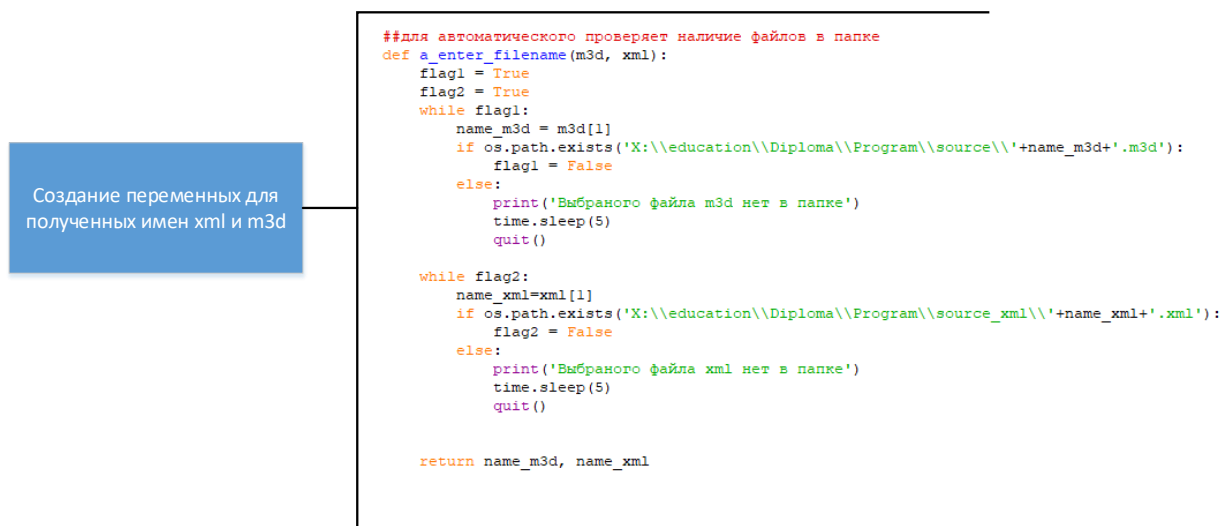


Рисунок 36 – Создание переменных из полученных параметров

После того, как у нас появилось имя нашей модели мы можем открыть ее (рис.37).



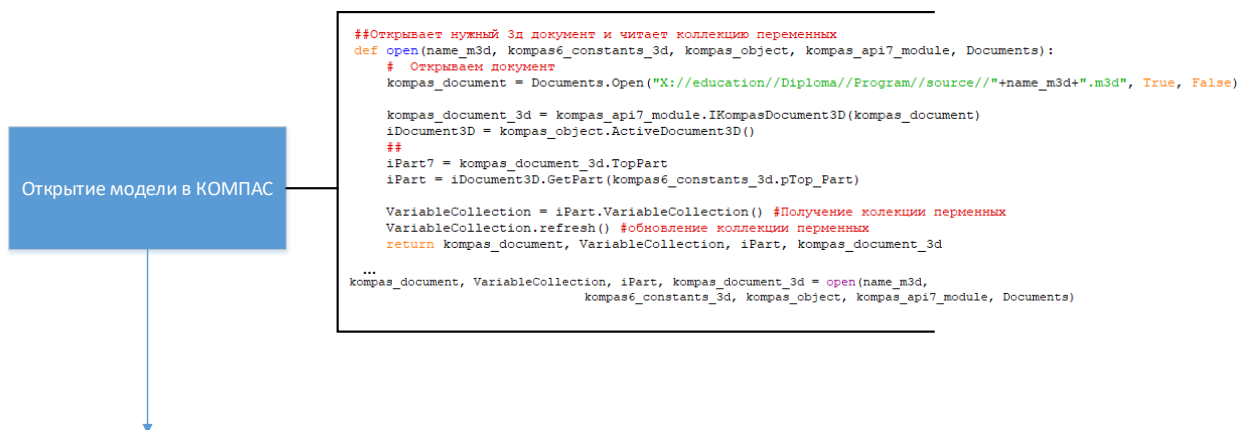


Рисунок 37 – Открытие модели в КОМПАС

Так же у нас есть необходимость в создании XML-дерева наших переменных (рис.38). После получения данного дерева мы сможем вынести из него дерево параметров нужного нам элемента.

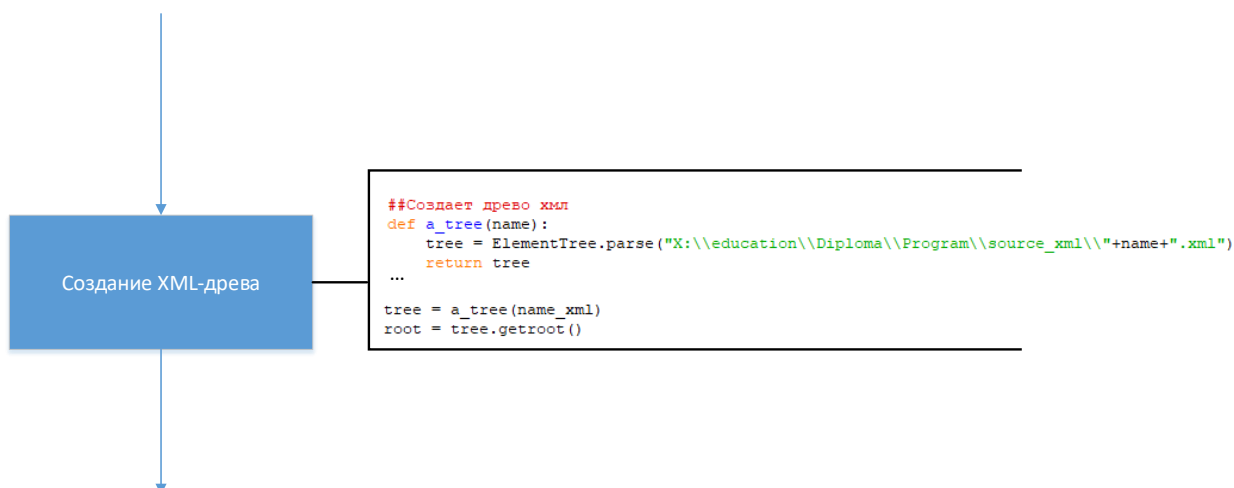


Рисунок 38 – Создание XML-дерева

Получим имя переменной из изменённого «funs» и проверим ее на наличие в дереве элементов, если она имеется, то создадим дерево переменной или выведем информацию об ошибке и завершим программу (рис.39).

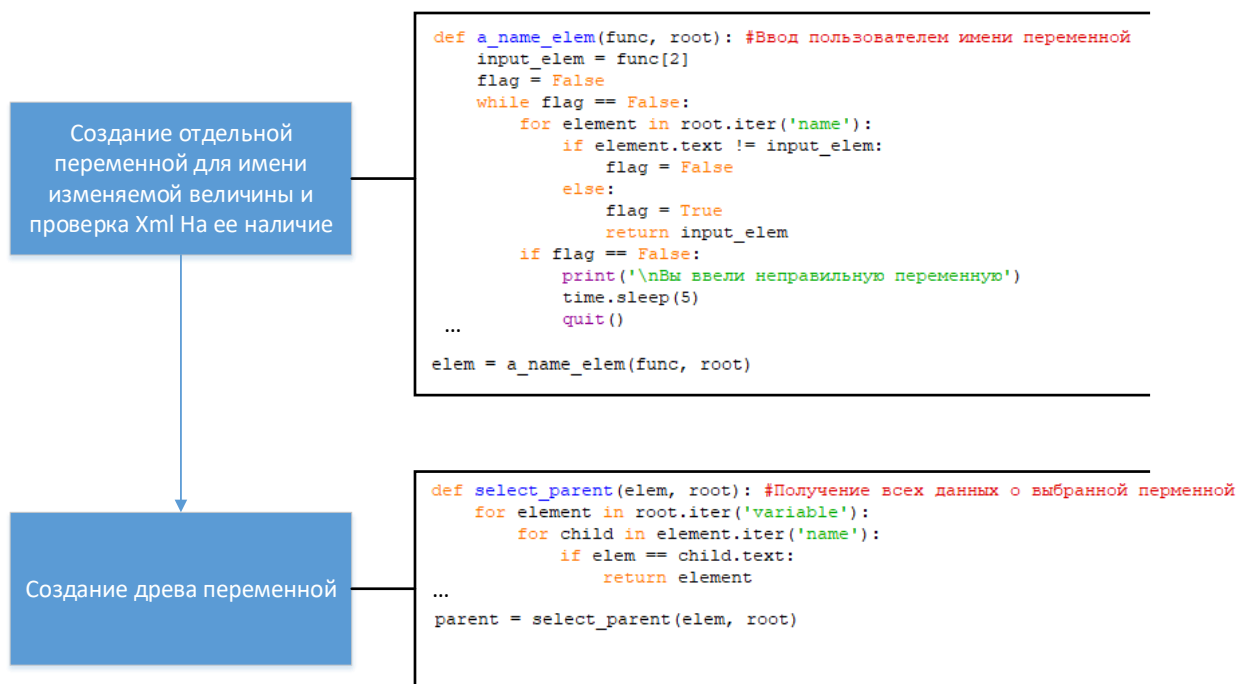


Рисунок 39 – Создание переменных для имени величины, создание дерева переменной.

После того, как мы подготовили все необходимые данные для изменения переменных в модели мы можем перейти к следующему блоку программы.

### 3.13 Блок случайного изменение переменной или изменение переменной несколько раз подряд в автоматическом режиме

Определим какое задание на операцию было получено из параметров, введенных в командной строке, в соответствии с этим нам нужно будет перейти к функциям, которые отвечают за требуемые операции (рис.40).

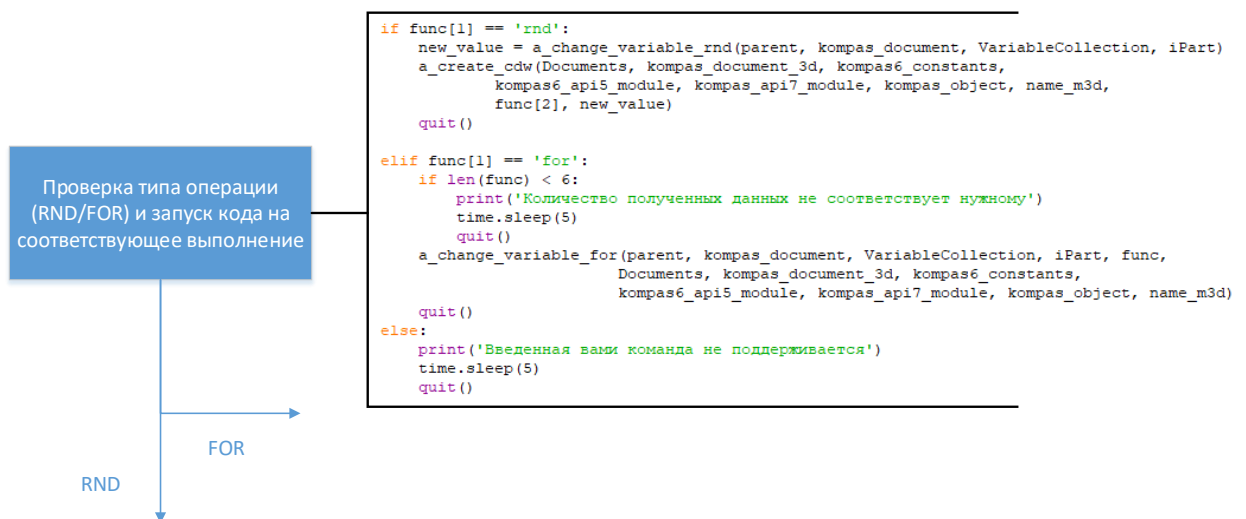


Рисунок 40 – Проверка операции

Если нами была получена операция RND, то мы должны изменить переменную случайным образом (рис.41), а после отправить ее в блок сохранения переменных.



Рисунок 41 – Применение операции RND

Если же пользователь ввел операцию FOR, то мы изменяем переменную на определенный шаг и сразу же сохраняем ассоциативную модель, чтобы не потерять чертеж (рис.42).



Рисунок 42 – Применение операции FOR

### 3.14 Блок сохранения файлов в автоматическом режиме

Сохранения файлов для операции FOR было показано в предыдущем блоке, и оно такое же, как для операции RND (рис.43), в этом блоке имена для сохраняемых ассоциативных чертежей формируются автоматически и состоят из: имени модели, имени изменяемой переменной, нового значения переменной. Создавая переменные с такими именами, мы легко сможем найти

нужный нам файл в будущем и все они будут иметь один стиль написания, что тоже повлияет на скорость поиска.

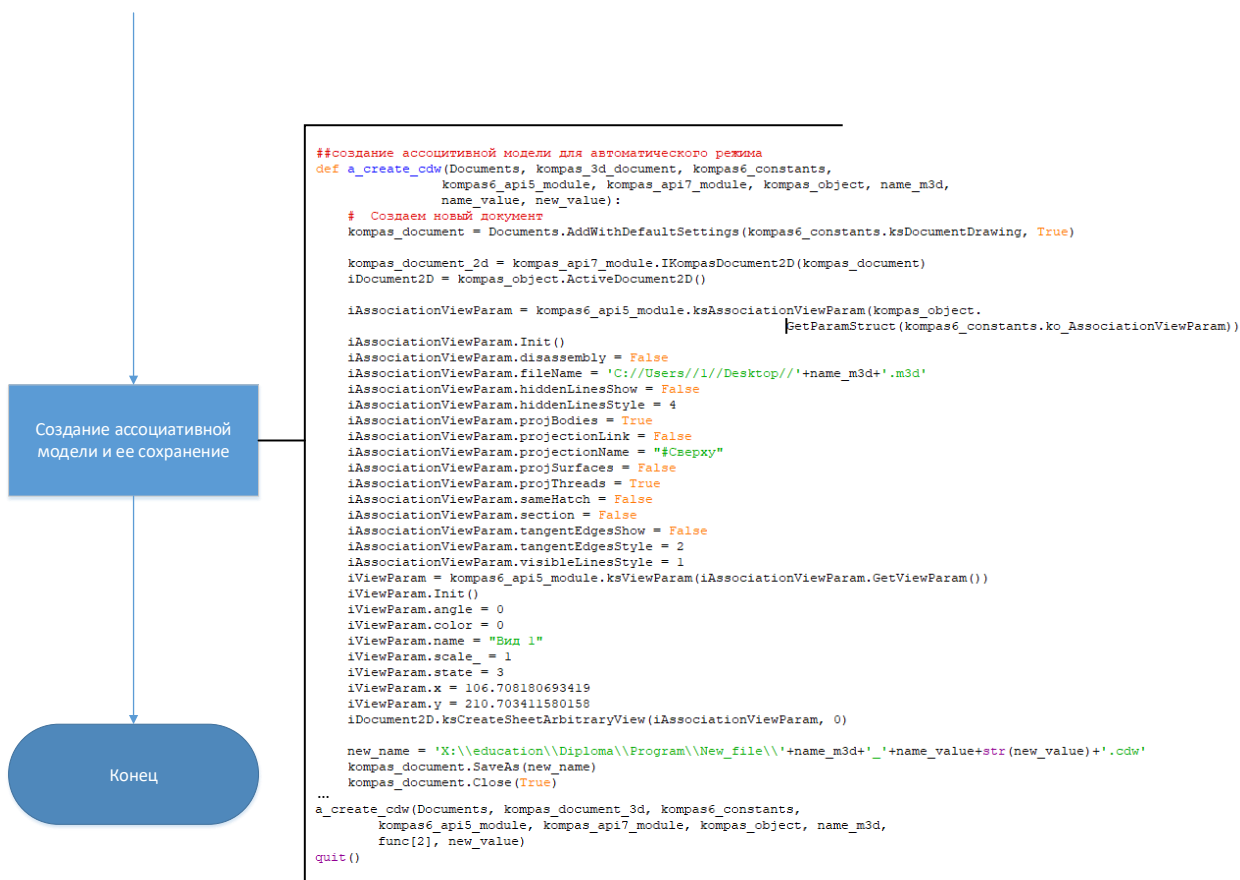


Рисунок 43 – Сохранение чертежа

### 3.15 Режим «помощь»

Режим помощь рекомендуется использовать для пользователей, которые не знакомы с использованием автоматического режима. Данный режим выводит информацию для пользователя в консоль (рис.44), в которой рассказывается каким образом нужно прописывать параметры, для правильной активации необходимого режима.

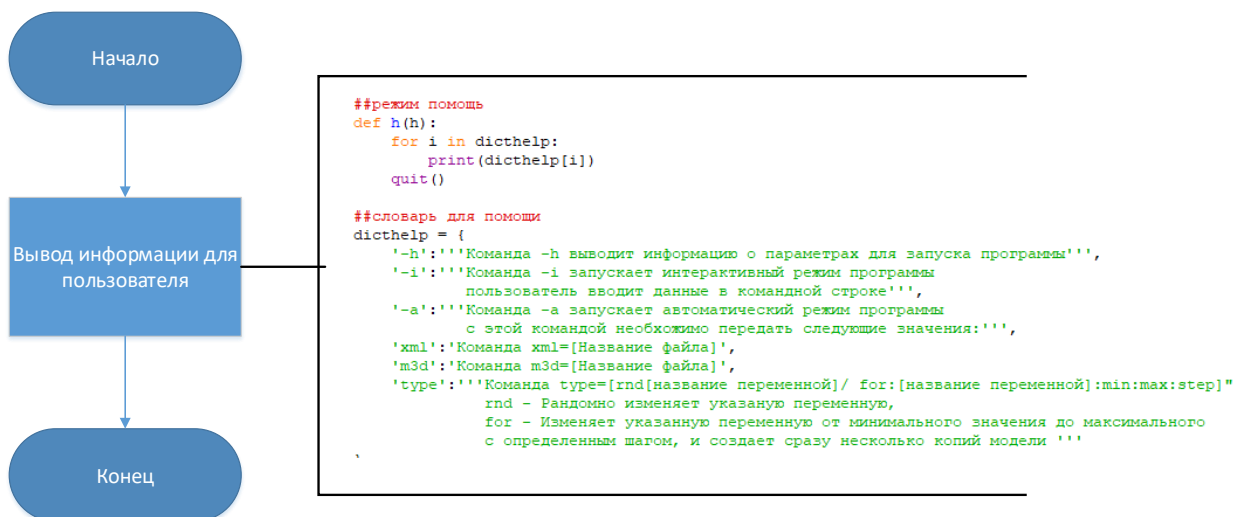


Рисунок 44 – Вывод информации для пользователя

### 3.16 Выводы по главе

В данной главе нами были рассмотрены основные элементы системы по созданию случайных тестовых заданий, а также разработана программа на языке программирования Python, которая организует взаимодействия элементов системы и пользователя. Данная программа имеет несколько режимов работы для организации различного функционала, которые требуется пользователю в данный момент времени. Человек, которому требуется изменение файла может изменить его вручную, создать файл со случайным изменением выбранного элемента или же создать группу файлов, которые будут изменять выбранный элемент в определенном диапазоне значений с заданным шагом. Данный подход значительно повышает возможности нашей программы.

## **4. РАЗРАБОТКА ПРОГРАММНОЙ ДОКУМЕНТАЦИИ**

### **4.1 Описание применения**

#### **4.1.1 Назначение программы**

Разрабатываемая система создания случайных тестовых заданий предназначена для решения задачи создания индивидуальных заданий для студентов, которые обучаются на дисциплинах «Инженерная графика» и «Компьютерная графика»

В соответствии со структурой процесса создания нового файла работа с системой будет состоять из следующих шагов:

- Для режима «помощь»
  - Открытие программы в данном режиме через консоль
  - Вывод в консоль информации о работе программы через параметры консоли
  - Завершение программы
- Для режима «Интерактивный»
  - Открытие программы через консоль или двойной щелчок по программе в папке
  - Ввод пользователем информации о именах xml и m3d файлов
  - Выбор переменных, которые доступны для изменения в выбранной модели и само их изменение
  - Создание и сохранение ассоциированного чертежа с именем, которое задал пользователь
- Для режима «Автоматический»
  - Открытие программы через консоль с дополнительной передачей необходимых для работы параметров

- Обработка параметров, полученных из консоли и получение на их основе всех необходимых данных о изменяемой переменной
- Изменение переменной в соответствии с выбранной операцией (RND/FOR)
- Сохранение полученного/ых чертежа/ей

#### **4.1.2 Условия применения**

Для корректной работы программы пользователю необходимо иметь соответствующее программное обеспечение:

- Программу КОМПАС-3D
- Язык программирования Python со следующими библиотеками и модулями:
  - Sys (стандартная библиотека)
  - Os (стандартная библиотека)
  - Element.Tree (стандартный модуль)
  - Pythoncom (стандартная библиотека)
  - Time (стандартная библиотека)
  - Random (стандартная библиотека)
  - Win32com (стандартная библиотека)
  - LDefin2D (библиотека для работы с Компас)
  - MiscellaneousHelpers (библиотека для работы с Компас)
- Операционную систему Windows 7/8/10

#### **4.1.3 Входные и выходные данные**

К входным данным относятся:

- Xml – документ содержит информацию о переменных модели
- M3d – документ содержит саму модель, которую будут менять

Выходные данные состоят из:



- M3d – документ содержит последние изменения в модели
- Cdw – документ или документы, которые являются ассоциированными чертежами к модели детали.

## **4.2 Руководство системного программиста**

### **4.2.1 Общие сведения о программе**

Система была разработана при помощи среды программирования Microsoft Visual Studio Code и API Компас.

### **4.2.2 Сообщения программисту**

Во время работы программы могут возникать различные проблемы технического характера, которые связаны с получением необходимых для работы данных или отсутствия определенных компонентов системы. Для указания на возникшие ошибки, программа использует сообщения программисту, которые конкретизируют проблему при работе. Данные сообщения в командной строке выделяются строкой «Информация для программиста:».

При запуске программа связывается с различными компонентами разработанной нами системы. В случае, если на компьютере отсутствует программа Компас-3D выводится сообщение, показанное на Рисунке 45. Если полученный xml файл имеет неправильную структуру, которая препятствует правильному считыванию из него данных, то выводится специальное сообщение (рис.46), отсутствие в модели внешних переменных, которые можно было бы изменить так же приведут к образованию исключения (рис.47).

Информация для программиста:  
На компьютере отсутствует программа Компас-3D.

Рисунок 45 – Исключение отсутствие Компас-3D

Информация для программиста:  
XML-файл имеет неправильную структуру.

Рисунок 46 – Исключение неправильная структура XML

Информация для программиста:  
В выбранной модели отсутствуют внешние переменные для изменения.

Рисунок 47 – Исключение отсутствие переменных для изменения

### 4.2.3 Руководство пользователя

Использование разработанной системы предлагает осуществление определенных действий при работе в различных режимах:

- Для режима «Помощь»
  - Запуск программы через консоль с требуемым параметром (рис.48)

```
X:\education\Diploma\Program>Program.py -h
```

Рисунок 48 – Запуск режима «Помощь»

- Для режима «Автоматический»
  - Запуск программы через консоль с требуемыми параметрами (рис.49) (название файла состоит из названия модели, названия переменной и полученного значения)

```
X:\education\Diploma\Program>Program.py -a xml=Цилиндр_переменные m3d=Цилиндр type:rnd:Радиус
```

Рисунок 49 – Запуск автоматического режима

- Для режима «Интерактивный»
  - Запуск программы через консоль или из папки (рис.50)

```
X:\education\Diploma\Program>Program.py
```

Рисунок 50 – Запуск интерактивного режима

- Ввод названий документов с необходимой информацией (рис.51)

```
Введите название файла, который вы бы хотели изменить
Цилиндр
Введите название файла, в котором хранится информация о переменных
Цилиндр_переменные
```

Рисунок 51 – Ввод названия документов

- Выбор переменной из предложенного списка (рис.52)

```
В выбранной вами фигуре вам доступны следующие элементы для изменения:
Высота
Радиус

Пожалуйста выберите один из представленных элементов, чтобы изменить его:
Радиус

Выбранный элемент Радиус
Обозначение y
Минимальное значение 50
Максимальное значение 100
Шаг 5
Это нужная вам переменная? (Да/Нет)
Да
```

Рисунок 52 – Выбор переменной

- Изменение переменной (рис.53)

```
Старое значение переменной: 65.0
Если введенное вами значение не будет соответствовать шагу, то значение округлится в меньшую сторону
Введите новое значение для выбранной вами переменной Радиус: 95
Новое значение переменной: 95.0
```

Рисунок 53 – Изменение переменной

- Выбор другой переменной или переход к сохранению файла (рис.54)

```
Продолжить изменение переменных? (Да/Нет) Нет
```

Рисунок 54 – Выбор другой переменной или сохранение файла

- Ввод названия для ассоциированного с моделью чертежа (рис.55)

Введите название файла, в который вы бы хотели сохранить изменения  
Цилиндр\_с\_радиусом95

Рисунок 55 – Ввод названия чертежа

## 5. ТЕСТИРОВАНИЕ

Проведем тестирование разработанной системы по изменению переменных на примере изменения размеров созданного нами ранее цилиндра (рис.56).

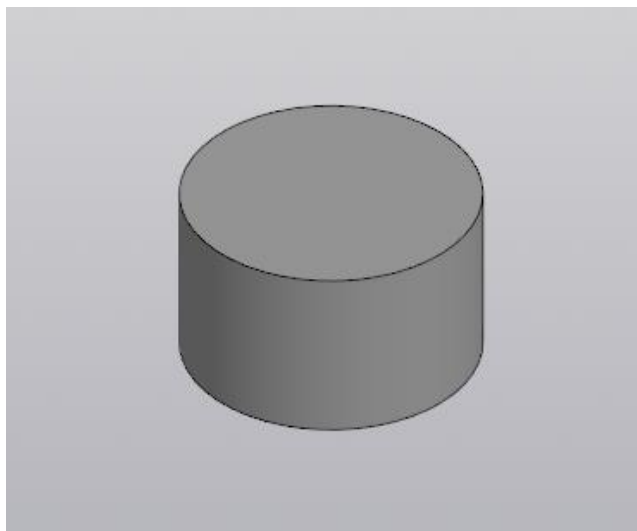


Рисунок 56 – Модель цилиндра

В ходе тестирования поставим себе ряд задач, которые заключаются в создании ассоциированного чертежа, отображающего «вид сверху» для нашего цилиндра с заданным значением Радиуса, ассоциированного чертежа, отображающего «вид сверху» для нашего цилиндра со случайным значением Радиуса и создание группы ассоциированных чертежей. Запускать программу будем из командной строки, для более удобного доступа к программе

### 5.1 Создание чертежа с заданным значением

Запустим программу в интерактивном режиме, чтобы получить возможность выбрать значение радиуса и название чертежа самостоятельно. Нам необходимо ввести названия XML и модели (рис.57).

```
Введите название файла, который вы бы хотели изменить
Цилиндр
Введите название файла, в котором хранится информация о переменных
Цилиндр_переменные
```

Рисунок 57 – Ввод названия документов

Далее программы выводит перед нами список возможных к изменению элементов, среди них находится требуемая нам величина «Радиус» и мы выбираем ее, после чего нам выводят ее параметры и предлагают подтвердить выбор переменной (рис.58).

```
В выбранной вами фигуре вам доступны следующие элементы для изменения:
Высота
Радиус

Пожалуйста выберите один из представленных элементов, чтобы изменить его:
Радиус

Выбранный элемент Радиус
Обозначение y
Минимальное значение 50
Максимальное значение 100
Шаг 5
Это нужная вам переменная?(Да/Нет)
Да
```

Рисунок 58 – Выбор переменной

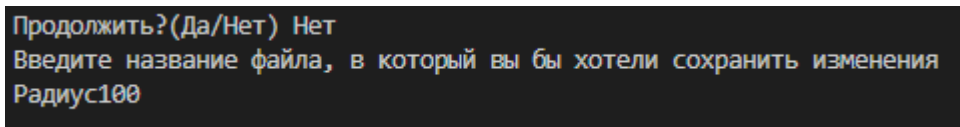
После выбора переменной следует этап, на котором нас уведомляют о предыдущем значении переменной, предлагают ввести новое значение и уведомляют об успешном изменении выбранной величины. После изменения «Радиуса» мы можем перейти к изменению других величин при желании, изменить переменную еще раз при желании (рис.59).

```
Старое значение переменной: 95.0
Если введенное вами значение не будет соответствовать шагу, то значение округлится в меньшую сторону
Введите новое значение для выбранной вами переменной Радиус: 100
Новое значение переменной: 100.0

Продолжить?(Да/Нет) Нет
```

Рисунок 59 – Изменение значения

Так как нас интересовало только изменение Радиуса, то мы отклоним это предложение и перейдем к сохранению ассоциированного чертежа. Программа предлагает нам выбрать название для чертежа и после ввода интересующего имени файла, она завершается (рис.60).



Продолжить?(Да/Нет) Нет  
Введите название файла, в который вы бы хотели сохранить изменения  
Радиус100

Рисунок 60 – Сохранение документа

Проверяем получилось ли у нас сохранить ассоциированный чертеж (рис.61) и убеждаемся в этом, это означает, что программа работает корректно


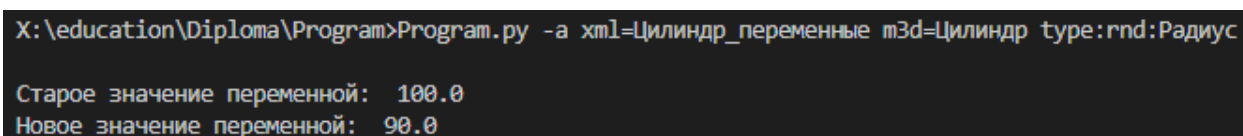
Имя	Дата изменения	Тип	Размер
 Радиус100.cdw	14.06.2022 12:20	КОМПАС-Чертеж	73 КБ

Рисунок 61 – Проверка выполнения программы

## 5.2 Создание чертежа со случайным значением

После проверки интерактивного режима займемся проверкой корректности создания ассоциированного чертежа в автоматическом режиме с заданием случайного значения для выбранной переменной. Для корректного запуска программы мы добавляем требуемые параметры, в которых укажем, что за переменную хотим изменить и каким образом. После правильного запуска программы пользователь увидит сообщения, которые рассказали нам, какое значение у переменной было изначально и какое было получено в результате проведения операции (рис.62).



```
X:\education\Diploma\Program>Program.py -a xml=Цилиндр_переменные m3d=Цилиндр type:rnd:Радиус
Старое значение переменной: 100.0
Новое значение переменной: 90.0
```

Рисунок 62 – Автоматический режим, команда RND

Удостоверимся, что чертеж, который мы получили находится в папке. Полученный нами чертеж должен иметь название, которое состоит из названия модели, названия переменной и полученного значения, то есть файл должен получить название «Цилиндр\_Радиус90.0», такой файл действительно присутствует в папке, а значит программа работает корректно (рис.63).



Имя	Дата изменения	Тип	Размер
 Радиус100.cdw	14.06.2022 12:20	КОМПАС-Чертеж	73 КБ
 Цилиндр_Радиус90.0.cdw	14.06.2022 12:29	КОМПАС-Чертеж	73 КБ

Рисунок 63 – Проверка выполнения программы

### 5.3 Создание группы чертежей

Проверим работу функции создания группы чертежей в автоматическом режиме. Данная функция на выходе должна предоставить нам определенное количество чертежей, со значениями, которые изменялись от минимального к максимальному с определенным шагом. В качестве изменяемой модели будет взят имеющийся цилиндр, а переменная, которую мы будем изменять называется «Радиус», в качестве минимального значения возьмем 50, максимального значение мы хотим получить 75 и шаг будет равен 5. Укажем данные параметры при запуске программы. После успешного запуска программы нам на экран выводится старое значение переменной, а после новые значения, которые получили наши чертежи (рис.64).

```
X:\education\Diploma\Program>Program.py -a xml=Цилиндр_переменные m3d=Цилиндр type:for:Радиус:50:80:5
Старое значение переменной: 90.0
Новое значение переменной: 50.0
Новое значение переменной: 55.0
Новое значение переменной: 60.0
Новое значение переменной: 65.0
Новое значение переменной: 70.0
Новое значение переменной: 75.0
```

Рисунок 64 – Автоматический режим, команда FOR

Проверим нашу папку с чертежами на наличие данной группы элементов, как указано ранее имена чертежей должны состоять из названия модели, названия переменной и полученного значения. Данная группа присутствует в папке, а значит программа работает успешно (рис.65).











Имя	Дата изменения	Тип	Размер
 Цилиндр_Радиус75.0.cdw	14.06.2022 12:38	КОМПАС-Чертеж	73 КБ
 Цилиндр_Радиус70.0.cdw	14.06.2022 12:38	КОМПАС-Чертеж	73 КБ
 Цилиндр_Радиус65.0.cdw	14.06.2022 12:38	КОМПАС-Чертеж	73 КБ
 Цилиндр_Радиус60.0.cdw	14.06.2022 12:38	КОМПАС-Чертеж	73 КБ
 Цилиндр_Радиус55.0.cdw	14.06.2022 12:38	КОМПАС-Чертеж	73 КБ
 Цилиндр_Радиус50.0.cdw	14.06.2022 12:38	КОМПАС-Чертеж	73 КБ
 Цилиндр_Радиус90.0.cdw	14.06.2022 12:29	КОМПАС-Чертеж	73 КБ
 Радиус100.cdw	14.06.2022 12:20	КОМПАС-Чертеж	73 КБ

Рисунок 65 – Проверка работы программы

## 5.4 Выводы

В данной главе мы произвели тестирование нашего программного обеспечения, провели проверку всех основных режимов работы и операций. Нами было установлено, что программа работает правильно и ожидаемый результат был достигнут в ходе тестирования. Как вывод, мы можем считать, что успешно справились с задачей проектирования ПО про генерации заданий.

## ЗАКЛЮЧЕНИЕ

В данной выпускной работе нами было создано программное обеспечение, которое решает задачу создания случайных индивидуальных задания для студентов по дисциплинам «Инженерная графика» и «Компьютерная графика». Данное ПО написано на языке программирования Python и напрямую взаимодействует с системой автоматизации производства Компас-3D. Нами было математически описано изменение форм детали с помощью алгоритмов Маркова.

Перед непосредственной разработкой программы были рассмотрены основные составляющие элементы системы, нами было рассказано о формате текстовых файлов XML и обосновано его использование, рассмотрено само САПР Компас-3D и показано на примерах, как в данной системе создаются модели вручную или с помощью API. В процессе разработки программы мы составили несколько возможностей работы программы: путем запуска в папке хранения или через командную строку с использованием различных параметров. Так же нами была разработан режим «Помощь», который в консоли выводит перед пользователем информацию о том, как правильно запускать программу с параметрами.

Мы создали программную документацию, в которой отражены требования к компьютеру, на котором планируется использовать программу. Так же в документации рассказано об исключениях и ошибках, которые может обрабатывать программа при отсутствии или некорректном составлении элементов системы и рассказано, какой алгоритм действий следует реализовать пользователи при работе с программой в разных режимах

Так же нами проведено тестирование ПО, в котором наглядно показана успешная работа программы и выполнение поставленной перед ней задачей. Итогом нашей работы является система, которая может изменять предоставляемые модели как в автоматическом режиме, так и в интерактивном.

Стоит отметить, что у данной ПО пользуется большой актуальностью в наши дни. Подобные разработки имеют большое влияние на улучшение учебного процесса в высших учебных заведениях нашей страны, что в конечном итоге повысит профессиональность выпускаемых студентов.

## БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Теория алгоритмов / А. А. Марков; Тр. МИАН СССР, 42, Изд-во АН СССР, М.–Л., 1954 — 375с.
2. Теория алгоритмов / Л.В. Алферов; Изд-во «Статистика», 1973. — 164 с.
3. Теория алгоритмов: Учеб. пособие. / Игошин В.И. — М.: ИНФРА — М, 2016. — 318 с.
4. ЛИНЕЙНЫЕ НОРМАЛЬНЫЕ АЛГОРИТМЫ / Пруцков А.В. — Вестник РГРТУ. № 3 (выпуск 33). Рязань, 2010 — 63с.
5. Изучаем Python, 4-е издание. / Лутц М. — Пер. с англ. — СПб.: Символ-Плюс, 2011 — 1280с., ил.
6. Python: создание приложений. Библиотека профессионала, 3-е изд. / Чан У. — Пер. с англ. — М.: ООО "И.Д. Вильямс", 2015. — 816 с.: ил. - Парал. тит. англ.
7. Python 3.10.5 documentation — <https://docs.python.org>
8. Компас-3D. Полное руководство от новичка до профессионала / Жарков Н.В., Минеев М.А., Финков М.В., Прокди Р.Г. — М.: Наука и Техника СПб, 2016. — 672с.
9. Руководство пользователя КОМПАС-Invisible (API КОМПАС-3D) / ООО «АСКОН"Системы проектирования» — 5492с.
10. XML. Базовый курс / Хантер Д., Фаусетт Дж., Рафтер Дж. — Пер. с англ. — М.: Диалектика, 2018 — 1344с.

## ПРИЛОЖЕНИЕ А. ЛИСТИНГ ОСНОВНЫХ ПРОГРАММНЫХ МОДУЛЕЙ

```
# -*- coding: utf-8 -*-

from ast import Break

from calendar import c

import sys

import os

from xml.etree import ElementTree

import pythoncom

from win32com.client import Dispatch, gencache

import LDefin2D

import MiscellaneousHelpers as MH

import time

import random


##Открывает компас

def start_kompas():

    Dispatch("KOMPAS.Application.7")

    # Подключим константы API Компас

    kompas6_constants = gencache.EnsureModule("{75C9F5D0-B5B8-4526-8681-9903C567D2ED}", 0, 1, 0).constants

    kompas6_constants_3d = gencache.EnsureModule("{2CAF168C-7961-4B90-9DA2-701419BEEFE3}", 0, 1, 0).constants


    # Подключим описание интерфейсов API5

    kompas6_api5_module = gencache.EnsureModule("{0422828C-F174-495E-AC5D-D31014DBBE87}", 0, 1, 0)
```

```

        KompasObject =
kompas6_api5_module.KompasObject(Dispatch("Kompas.Application.5")._ole
obj_.QueryInterface(kompas6_api5_module.KompasObject.CLSID,
pythoncom.IID_IDispatch))

    MH.iKompasObject = kompas_object

    # Подключим описание интерфейсов API7

    kompas_api7_module = gencache.EnsureModule("{69AC2981-37C0-
4379-84FD-5DD2F3C0A520}", 0, 1, 0)

    application =
kompas_api7_module.IApplication(Dispatch("Kompas.Application.7")._oleo
bj_.QueryInterface(kompas_api7_module.IApplication.CLSID,
pythoncom.IID_IDispatch))

    MH.iApplication = application

    Documents = application.Documents

    return kompas6_constants_3d, kompas_object,
kompas_api7_module, Documents, kompas6_constants, kompas6_api5_module,
kompas_api7_module

##Открывает нужный 3д документ и читает коллекцию переменных

def open(name_m3d, kompas6_constants_3d, kompas_object,
kompas_api7_module, Documents):

    # Открываем документ

    KompasDocument =
Documents.Open("X://education//Diploma//Program//source//"+name_m3d+".
m3d", True, False)

    KompasDocument3D =
kompas_api7_module.IKompasDocument3D(kompas_document)

```

```

iDocument3D = kompas_object.ActiveDocument3D()

##

iPart7 = kompas_document_3d.TopPart

iPart = iDocument3D.GetPart(kompas6_constants_3d.pTop_Part)


VariableCollection = iPart.VariableCollection() #Получение
коллекции переменных

VariableCollection.refresh() #обновление коллекции переменных

return kompas_document, VariableCollection, iPart,
kompas_document_3d


##для интерактивного режима спрашивает названия xml и m3d файлов

def i_enter_filename():

    flag1 = True

    flag2 = True

    while flag1:

        name_m3d=input('Введите название файла, который вы бы
хотели изменить\n')

        if
os.path.exists('X:\\education\\Diploma\\Program\\source\\'+name_m3d+'.
m3d'):

            flag1 = False

        else:

            print('Выбраного файла нет в папке')


    while flag2:

```

```

        name_xml=input('Введите название файла, в котором хранится
информация о переменных\n')

        if
os.path.exists('X:\\education\\Diploma\\Program\\source_xml\\'+name_xml+'.xml'):

            flag2 = False

        else:

            print('Выбраного файла нет в папке')

    return name_m3d, name_xml

##для автоматического проверяет наличие файлов в папке
def a_enter_filename(m3d, xml):

    flag1 = True

    flag2 = True

    while flag1:

        name_m3d = m3d[1]

        if
os.path.exists('X:\\education\\Diploma\\Program\\source\\'+name_m3d+'.m3d'):

            flag1 = False

        else:

            print('Выбраного файла m3d нет в папке')

            time.sleep(5)

            quit()

    while flag2:

```



```

        name_xml=xml[1]

        if
os.path.exists('X:\\education\\Diploma\\Program\\source_xml\\'+name_xml+'.xml'):

        flag2 = False

        else:

            print('Выбранного файла xml нет в папке')

            time.sleep(5)

            quit()

        return name_m3d, name_xml

##для интерактивного выводит доступные для изменения переменные

def elem_is(name):

    print('В выбранной вами фигуре вам доступны следующие элементы
для изменения:')

    tree =
ElementTree.parse("X:\\education\\Diploma\\Program\\source_xml\\"+name
+".xml")

    root = tree.getroot()

    for element in root.iter('variable'):

        for child in element.iter('name'):

            print(' ', child.text)

    return tree

##Создает древо xml для обоих файлов

def a_tree(name):

```

```

        tree =
ElementTree.parse("X:\\education\\Diploma\\Program\\source_xml\\"+name
+".xml")

```

```

    return tree

```

```

def show_elem(chosen_elem): #Вызов характеристик переменной на
экран

```

```

    for element in root.iter('variable'):

```

```

        for child in element.iter('name'):

```

```

            if child.text == chosen_elem:

```

```

                print('\nВыбранный элемент', element[0].text)

```

```

                print('Обозначение', element[1].text)

```

```

                print('Минимальное значение', element[2].text)

```

```

                print('Максимальное значение', element[3].text)

```

```

                print('Шаг', element[4].text)

```

```

def name_elem(): #Ввод пользователем имени переменной

```

```

    flag = False

```

```

    while flag == False:

```

```

        input_elem = input('\nПожалуйста выберите один из
представленных элементов, чтобы изменить его: \n')

```

```

        for element in root.iter('name'):

```

```

            if element.text != input_elem:

```

```

                flag = False

```

```

            else:

```

```

                flag = True

```

```

        return input_elem

    if flag == False:

        print('\nВы ввели неправильную переменную')

def check_elem(chosen_elem): #Проверка вользователем переменной

    flag = input('Это нужная вам переменная? (Да/Нет)\n')

    if flag == 'Нет':

        chosen_elem = select_elem()

        show_elem(elem)

    return chosen_elem

def select_elem(): #Полный выбор переменной их хмл

    elem = name_elem()

    show_elem(elem)

    elem = check_elem(elem)

    return elem

def a_name_elem(func, root): #Ввод пользователем имени переменной

    input_elem = func[2]

    flag = False

    while flag == False:

        for element in root.iter('name'):

            if element.text != input_elem:

                flag = False

            else:

```

```

        flag = True

        return input_elem

    if flag == False:

        print('\nВы ввели неправильную переменную')

        time.sleep(5)

        quit()

def select_parent(elem, root): #Получение всех данных о выбранной
перменной

    for element in root.iter('variable'):

        for child in element.iter('name'):

            if elem == child.text:

                return element

##интерактивное изменение значения переменной

def change_variable(parent, Kompas_document, VariableCollection,
iPart):

    Variable = VariableCollection.GetByName(parent[1].text, True,
True)

    print('\nСтарое значение переменной: ', Variable.value)

    print('Если введенное вами значение не будет соответствовать
шагу, то значение округлится в меньшую сторону')

    new_value = float(input('Введите новое значение для выбранной
вами переменной '+ elem + ': '))

    new_value =
new_value//float(parent[4].text)*float(parent[4].text)

```

```

if new_value > float(parent[3].text):

    print('Ваше значение было слишком большим')

    new_value = float(parent[3].text)

elif new_value < float(parent[2].text):

    print('Ваше значение было слишком маленьким')

    new_value = float(parent[2].text)

Variable.value = new_value

print('Новое значение переменной: ', Variable.value)

iPart.RebuildModel()

kompas_document.Save()


##автоматического изменение значения переменной

def a_change_variable_rnd(parent, kompas_document,
VariableCollection, iPart):

    Variable = VariableCollection.GetByName(parent[1].text, True,
True)

    print('\nСтарое значение переменной: ', Variable.value)

    new_value = float(random.randrange(int(parent[2].text),
int(parent[3].text), int(parent[4].text)))

    Variable.value = new_value

    print('Новое значение переменной: ', Variable.value)

    time.sleep(5)

    iPart.RebuildModel()

    kompas_document.Save()

    return new_value

```

```

        ##автоматического изменения значения переменной в цикле

        def          a_change_variable_for(parent,          Kompas_document,
VariableCollection, iPart, func,

                                Documents,          Kompas_document_3d,
Kompas6_constants,

                                Kompas6_api5_module,
Kompas_api7_module, Kompas_object, name_m3d):

            Variable = VariableCollection.GetByName(parent[1].text, True,
True)

            print('\nСтарое значение переменной: ', Variable.value)

            Variable.value = 0

            for i in range(int(func[3]), int(func[4]), int(func[5])):

                Variable.value = float(i)

                print('Новое значение переменной: ', Variable.value)

                iPart.RebuildModel()

                Kompas_document.Save()

                a_create_cdw(Documents,          Kompas_document_3d,
Kompas6_constants,

                                Kompas6_api5_module,          Kompas_api7_module,
Kompas_object, name_m3d,

                                func[2], Variable.value)

            Kompas_document_3d.Close(True)

        ##Создание ассоциативной модели для интерактивного

        def create_cdw(Documents, Kompas_3d_document, Kompas6_constants,

                                Kompas6_api5_module,          Kompas_api7_module,
Kompas_object, name_m3d):

```

```

# Создаем новый документ

kompas_document =
Documents.AddWithDefaultSettings(kompas6_constants.ksDocumentDrawing,
True)

kompas_document_2d =
kompas_api7_module.IKompasDocument2D(kompas_document)

iDocument2D = kompas_object.ActiveDocument2D()

iAssociationViewParam =
kompas6_api5_module.ksAssociationViewParam(kompas_object.GetParamStruct(
kompas6_constants.ko_AssociationViewParam))

iAssociationViewParam.Init()

iAssociationViewParam.disassembly = False

iAssociationViewParam.fileName =
'C://Users//1//Desktop//' + name_m3d + '.m3d'

iAssociationViewParam.hiddenLinesShow = False

iAssociationViewParam.hiddenLinesStyle = 4

iAssociationViewParam.projBodies = True

iAssociationViewParam.projectionLink = False

iAssociationViewParam.projectionName = "#Сверху"

iAssociationViewParam.projSurfaces = False

iAssociationViewParam.projThreads = True

iAssociationViewParam.sameHatch = False

iAssociationViewParam.section = False

iAssociationViewParam.tangentEdgesShow = False

iAssociationViewParam.tangentEdgesStyle = 2

```

```

        iAssociationViewParam.visibleLineStyle = 1

        iViewParam
kompas6_api5_module.kableViewParam(iAssociationViewParam.GetViewParam())

        iViewParam.Init()

        iViewParam.angle = 0

        iViewParam.color = 0

        iViewParam.name = "Вид 1"

        iViewParam.scale_ = 1

        iViewParam.state = 3

        iViewParam.x = 106.708180693419

        iViewParam.y = 210.703411580158

        iDocument2D.kCreateSheetArbitraryView(iAssociationViewParam,
0)

        new_name
'X:\\education\\Diploma\\Program\\New_file\\'+input('Введите название
файла, в который вы бы хотели сохранить изменения\n')+'.cdw'

        kompas_document.SaveAs(new_name)

        kompas_document.Close(True)

        kompas_3d_document.Close(True)

##создание ассоциативной модели для автоматического режима

def a_create_cdw(Documents, kompas_3d_document, kompas6_constants,
        kompas6_api5_module, kompas_api7_module,
kompas_object, name_m3d,
        name_value, new_value):

        # Создаем новый документ

```



```

        KompasDocument =
Documents.AddWithDefaultSettings(kompas6_constants.ksDocumentDrawing,
True)

        KompasDocument2D =
KompasApi7Module.IKompasDocument2D(KompasDocument)

        iDocument2D = KompasObject.ActiveDocument2D()

        iAssociationViewParam =
Kompas6Api5Module.KsAssociationViewParam(KompasObject.GetParamStruct(
Kompas6_constants.ko_AssociationViewParam))

        iAssociationViewParam.Init()

        iAssociationViewParam.Disassembly = False

        iAssociationViewParam.FileName =
'C://Users//1//Desktop//' + name_m3d + '.m3d'

        iAssociationViewParam.HiddenLinesShow = False

        iAssociationViewParam.HiddenLinesStyle = 4

        iAssociationViewParam.ProjBodies = True

        iAssociationViewParam.ProjectionLink = False

        iAssociationViewParam.ProjectionName = "#Сверxy"

        iAssociationViewParam.ProjSurfaces = False

        iAssociationViewParam.ProjThreads = True

        iAssociationViewParam.SameHatch = False

        iAssociationViewParam.Section = False

        iAssociationViewParam.TangentEdgesShow = False

        iAssociationViewParam.TangentEdgesStyle = 2

        iAssociationViewParam.VisibleLinesStyle = 1

```

```

        iViewParam
    kompas6_api5_module.ksViewParam(iAssociationViewParam.GetViewParam())

    iViewParam.Init()

    iViewParam.angle = 0

    iViewParam.color = 0

    iViewParam.name = "Вид 1"

    iViewParam.scale_ = 1

    iViewParam.state = 3

    iViewParam.x = 106.708180693419

    iViewParam.y = 210.703411580158

    iDocument2D.ksCreateSheetArbitraryView(iAssociationViewParam,
0)

```

```

        new_name
    'X:\\education\\Diploma\\Program\\New_file\\'+name_m3d+'_'+name_value+
    str(new_value)+'.cdw'

```

```

    kompas_document.SaveAs(new_name)

```

```

    kompas_document.Close(True)

```

```

##автоматический режим

```

```

def auto(a):

```

```

    xml = sys.argv[2].split('=')

```

```

    m3d = sys.argv[3].split('=')

```

```

    func = sys.argv[4].split(':')

```

```

    if (xml[0] != 'xml') or (m3d[0] != 'm3d') or (func[0] !=
'type'):

```

```

        print('Параметры заданы не верно')

        print('Пример:      Program.py      -a      xml=123      m3d=123
type:rnd:Радиус')

        time.sleep(5)

        quit()

        Kompas6_constants_3d, Kompas_object, Kompas_api7_module,
Documents, Kompas6_constants, Kompas6_api5_module, Kompas_api7_module =
start_kompas()

        name_m3d, name_xml = a_enter_filename(m3d, xml)

        Kompas_document, VariableCollection, iPart, Kompas_document_3d
= open(m3d[1], Kompas6_constants_3d, Kompas_object, Kompas_api7_module,
Documents)

        tree = a_tree(name_xml)

        root = tree.getroot()

        elem = a_name_elem(func, root)

        parent = select_parent(elem, root)

        if func[1] == 'rnd':

            new_value = a_change_variable_rnd(parent, Kompas_document,
VariableCollection, iPart)

            a_create_cdw(Documents,                                Kompas_document_3d,
Kompas6_constants,
                                Kompas6_api5_module,                Kompas_api7_module,
Kompas_object, name_m3d,
                                func[2], new_value)

            Kompas_object.Quit()

        quit()

```

```

elif func[1] == 'for':

    if len(func) < 6:

        print('Количество полученных данных не соответствует
нужному')

        time.sleep(5)

        quit()

        a_change_variable_for(parent,          Kompas_document,
VariableCollection, iPart, func,

                                Documents,      Kompas_document_3d,
Kompas6_constants,

                                Kompas6_api5_module,
Kompas_api7_module, Kompas_object, name_m3d)

        Kompas_object.Quit()

        quit()

    else:

        print('Введенная вами команда не поддерживается')

        time.sleep(5)

        quit()

quit()

##режим помощь

def h(h):

    for i in dicthelp:

        print(dicthelp[i])

    quit()

```

```

##словарь для помощи

dicthelp = {

    '-h':'''Команда -h выводит информацию о параметрах для запуска
программы''',

    '-i':'''Команда -i запускает интерактивный режим программы
        пользователь вводит данные в командной строке''',

    '-a':'''Команда -a запускает автоматический режим программы
        с этой командой необходимо передать следующие
значения:''',

    'xml': 'Команда xml=[Название файла]',

    'm3d': 'Команда m3d=[Название файла]',

    'type':'''Команда      type=[rnd[название      переменной] /
for:[название переменной]:min:max:step]"

        rnd - Рандомно изменяет указаную переменную,

        for - Изменяет указанную переменную от минимального
значения до максимального

        с определенным шагом, и создает сразу несколько копий
модели '''

    }

elem = ''

parent = ''

if len(sys.argv)>1:

    if sys.argv[1] == '-i':##интерактивное изменение

```

```

        Kompas6_constants_3d, Kompas_object, Kompas_api7_module,
Documents, Kompas6_constants, Kompas6_api5_module, Kompas_api7_module =
start_kompas()

        name_m3d, name_xml = i_enter_filename()

        Kompas_document, VariableCollection, iPart,
Kompas_document_3d = open(name_m3d, Kompas6_constants_3d,
Kompas_object, Kompas_api7_module, Documents)

        tree = elem_is(name_xml)

        root = tree.getroot()

        flag = 'Да'

        while flag == 'Да':

            elem = select_elem()

            parent = select_parent(elem, root)

            change_variable(parent, Kompas_document,
VariableCollection, iPart)

            flag = input('\nПродолжить? (Да/Нет) ')

        else:

            create_cdw(Documents, Kompas_document_3d,
Kompas6_constants,
Kompas6_api5_module, Kompas_api7_module,
Kompas_object, name_m3d)

            quit()

        elif sys.argv[1] == '-h':

            h(sys.argv[1])

        elif sys.argv[1] == '-a':

            auto(sys.argv[1])

        else:

```

```

        print('Заданы неверные параметры')

        quit()

    else: ##изменение при запуске не через кмд

        Kompas6_constants_3d, Kompas_object, Kompas_api7_module,
Documents, Kompas6_constants, Kompas6_api5_module, Kompas_api7_module =
start_kompas()

        name_m3d, name_xml = i_enter_filename()

        Kompas_document, VariableCollection, iPart, Kompas_document_3d
=
        open(name_m3d, Kompas6_constants_3d, Kompas_object,
Kompas_api7_module, Documents)

        tree = elem_is(name_xml)

        root = tree.getroot()

        flag = 'Да'

        while flag == 'Да':

            elem = select_elem()

            parent = select_parent(elem, root)

            change_variable(parent, Kompas_document,
VariableCollection, iPart)

            flag = input('\nПродолжить изменение переменных? (Да/Нет)
')

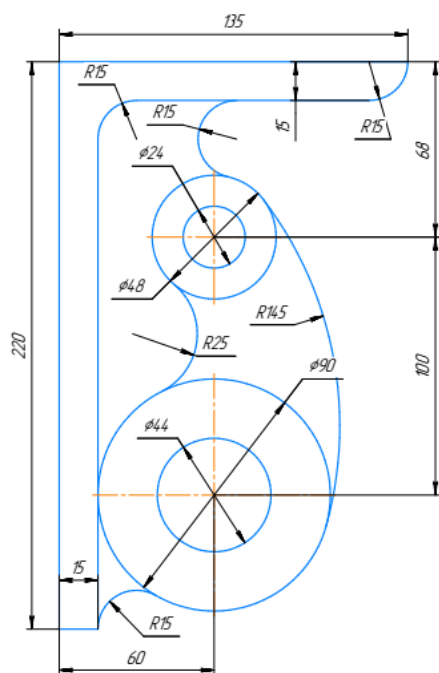
        else:

            create_cdw(Documents, Kompas_document_3d,
Kompas6_constants,
Kompas6_api5_module, Kompas_api7_module,
Kompas_object, name_m3d)

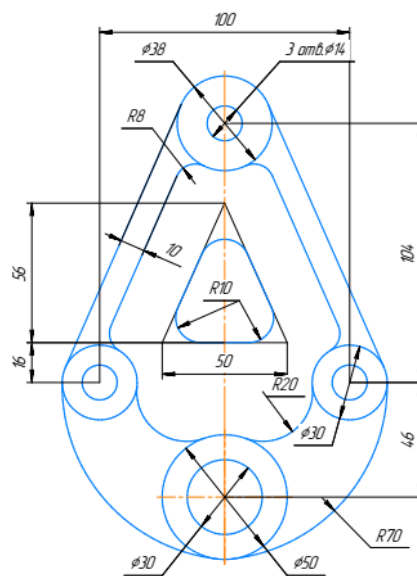
            quit()

```

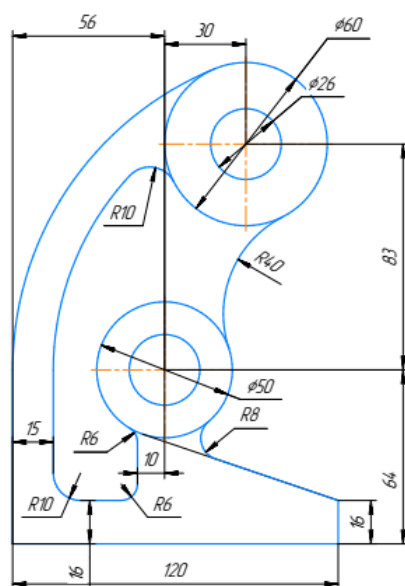
## ПРИЛОЖЕНИЕ Б. РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ СИСТЕМЫ



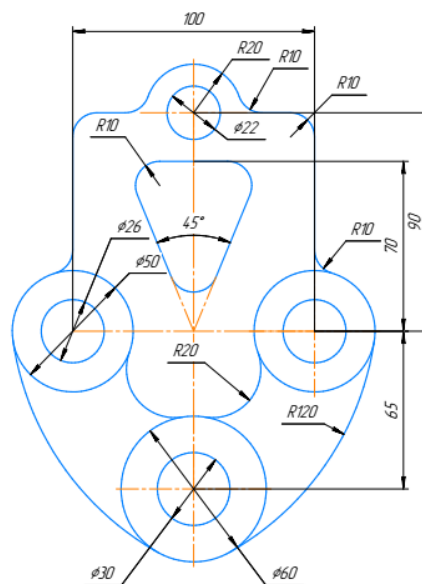
а



б



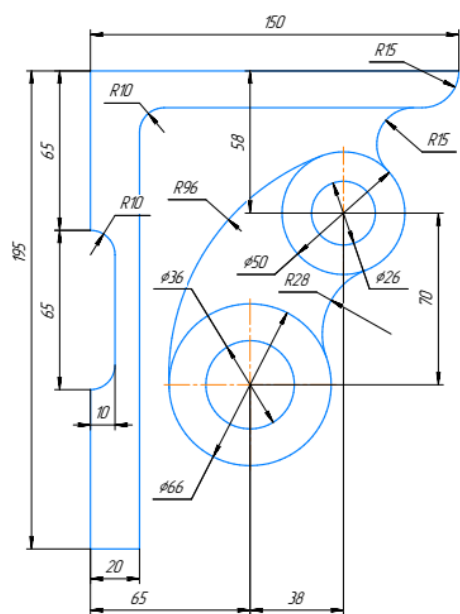
в



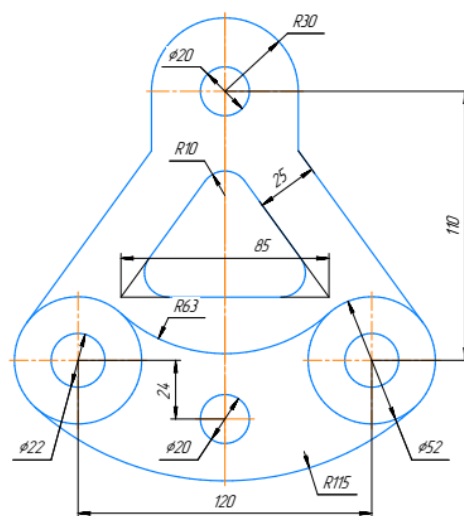
г

Рисунок 66 – Результат тестирования системы на моделях

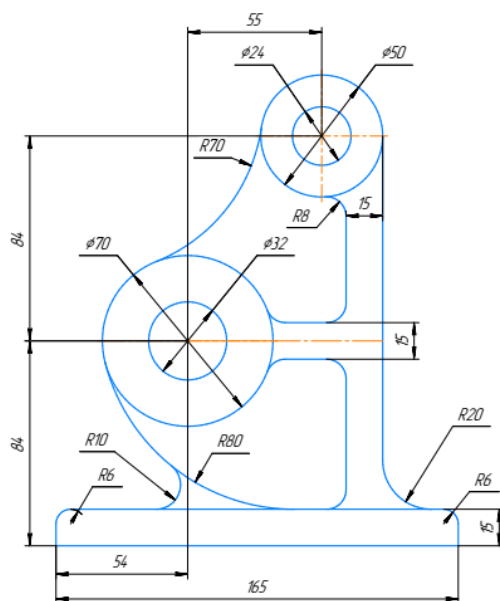




a



б



в

Рисунок 67 – Результат тестирования системы на моделях