



**МОСКОВСКИЙ УНИВЕРСИТЕТ ИМ. С.Ю.ВИТТЕ**

**Кафедра «Информационных систем»**

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ  
ПО ОРГАНИЗАЦИИ ПОДГОТОВКИ, ВЫПОЛНЕНИЮ И ЗАЩИТЕ  
КУРСОВОЙ РАБОТЫ ПО ДИСЦИПЛИНЕ  
«ВЫСОКОУРОВНЕВЫЕ МЕТОДЫ ПРОГРАММИРОВАНИЯ»**

**Москва**

*Автор-составитель:*

Блощук А.А., к.т.н., доцент кафедры информационных систем

*Рецензент:*

Зайцев С.А., к.т.н., доцент, декан факультета информационных технологий

Методические указания рассмотрены на заседании кафедры Информационных систем.

**Методические указания по организации подготовки, выполнению и защите курсовой работы: учебно-методические материалы** // А.А. Блощук – 4-е изд., перераб. и доп. – М.: МУ им. С.Ю. Витте, 2021. – 61 с.

В предлагаемых методических указаниях для обучающихся направлению подготовки 09.03.03 «Прикладная информатика», направленность (профиль): «корпоративные информационные системы» изложены основные подходы и принципы подготовки курсовой работы по дисциплине «Высокоуровневые методы программирования», рассматривается порядок аттестации, формулируются критерии оценки курсовой работы. Приведены основные требования к содержанию курсовой работы.

Указания предназначены для студентов факультета информационных технологий всех направлений и форм обучения, а также профессорско-преподавательского состава университета.

## Содержание

1. Общие положения .....	4
2. Выбор темы курсовой работы и подбор используемых источников.....	5
3. Структура курсовой работы .....	9
3.1. Содержание пояснительной записки .....	9
3.2 Задания для выполнения в рамках курсовой работы. ....	12
3.2.1 Задание № 1 .....	12
3.2.2. Задание № 2 .....	14
3.2.3. Задание № 3 .....	17
3.2.4. Задание № 4 .....	24
4. Оформление курсовой работы .....	26
5. Порядок аттестации и защиты курсовой работы .....	29
6. ПРИЛОЖЕНИЯ.....	31

## 1. Общие положения

Курсовая работа является важным элементом учебного процесса и представляет собой итоговый предэкзаменационный этап в изучении дисциплины; представляет собой логически завершенное, оформленное в виде текста и программного кода изложение обучающимся методов решения поставленных задач в сфере программирования на языках высокого уровня.

Задания подобраны таким образом, чтобы при выполнении работы студенты могли приобрести практические навыки разработки программных продуктов для решения прикладных задач среднего уровня сложности. Среда разработки PyCharm, язык программирования - Python. Рекомендуется использование наиболее распространенных библиотек и модулей стандартной библиотеки. Использование нестандартных библиотек возможно только после согласования с руководителем курсовой работы. Задачами курсовой работы являются:

- закрепление, углубление, расширение и систематизация знаний, полученных при изучении данной дисциплины («Высокоуровневые методы программирования») и других, предшествовавших ей дисциплин;
- закрепление умений применять эти знания для решения типовых и нестандартных задач;
- получение теоретических и практических навыков решения прикладных информационных задач с использованием методик программирования высокого уровня
- разработка программных продуктов (структурной и функциональной схем программного обеспечения, структур данных, алгоритмов и реализующих их программ, стратегии тестирования и подбора тестовых данных);
- приобретение опыта аналитической, расчетной, конструкторской работы и формирование соответствующих умений;
- развитие умений работы со специальной литературой и иными информационными источниками;
- приобретение опыта научно-исследовательской работы и формирование соответствующих умений;
- формирование умений формулировать логически обоснованные выводы, предложения и рекомендации по результатам выполненной работы;

В результате выполнения курсовой работы по данной дисциплине студент должен научиться:

- Правильно понимать задания на выполнение курсовой работы и разработку программного продукта;
- Уметь анализировать и составить примерную стратегию решения каждого задания;
- Выбирать необходимые библиотеки для реализации предметных областей программного продукта;

- Разрабатывать алгоритмы и реализовать их в среде разработки с помощью выбранных библиотек
- Уметь формировать графический интерфейс пользователя средствами библиотек Tkinter, wxPython, PyQT или PySimpleGUI;
- Выбирать стратегию тестирования и разработать тесты;
- Выполнять тестирование и отладку;
- Описывать решение каждого задания, форматы входных и выходных файлов.
- Сформировать пояснительную записку к курсовой работе, включающую листинг программного кода для каждого задания.

В ходе написания курсовой работы студент должен показать умение использовать общетеоретические и специальные знания по выбранной проблематике. Кроме того, необходимо стремиться к тому, чтобы собранный материал и полученные результаты могли быть использованы при подготовке к выполнению выпускной квалификационной работы.

## **2. Выбор темы курсовой работы и подбор используемых источников**

Тематика курсовых работ, требования к ним и указания по их выполнению доводятся до сведения студентов в начале соответствующего семестра. Тема курсовой работы предлагается единой и предусматривает разработку программного продукта для решения прикладных задач. Вариативность заданий достигается тем фактором, что задания формируются исходя из номера студенческого билета и ФИО обучающегося, в результате чего каждый студент будет выполнять индивидуальный набор задач, не пересекающийся с другими.

Студент совместно с руководителем курсовой работы (он же ведущий преподаватель по дисциплине) уточняет круг вопросов, подлежащих изучению, составляет план и определяет структуру работы. В плане должны быть предусмотрены вопросы теории и практики рассматриваемой темы (проблемы).

Подготовка к написанию курсовой работы во многом зависит от правильной подготовки к выполнению работы, которую можно условно разделить на следующие этапы.

**1 этап.** Предусматривает осмысление темы и целевых установок, на основе чего важно наметить главные вопросы, подлежащие рассмотрению, и их краткое содержание.

**2 этап.** Включает подбор литературы по теме курсовой работы, работу с каталогами библиотек, библиографическими указателями, большим массивом электронных ресурсов и интернет-источников.

При выборе литературы возникает множество трудностей, которые можно избежать, если придерживаться нескольких *правил*:

– обращать внимание на мета-данные книги. В оглавлении важно отметить те разделы и параграфы, которые представляют интерес для раскрытия темы. В предисловии можно найти ответы на такие вопросы, как цель написа-

ния книги, основные направления исследования, общий характер работы;

- обращать внимание на год издания, т.к. требуется соблюсти условие при использовании литературы, т.е. ее актуальность. **Актуальной считается литература (за исключением словарей и фундаментальных первоисточников) сроком после выхода не более 5 лет;**

- целесообразно при ознакомлении делать выписки, обращая внимание на внешние признаки в тексте. Рекомендуются основные источники перечитывать. Чтение должно быть глубоким, сплошным.

- при конспектировании литературы и ее последующего использования в своей курсовой работе студенту необходимо сформулировать личное мнение по рассматриваемой проблеме. В этом случае используется правило, когда автор выступает во множественном числе и вместо «я» употребляет «**на наш взгляд**», «**по нашему мнению**». Выражение авторства как формального коллектива с руководителем курсовой работы придает большой объективизм изложению материала;

- при написании курсовой работы следует активно использовать, научные статьи, опубликованные в ведущих экономических изданиях, официальные образовательные Интернет-ресурсы, а также официальные сайты государственных органов исполнительной власти, аналитических агентств и других организаций, являющихся источниками необходимой информации для раскрытия темы курсовой работы. Например, очень помогает студентам интернет ресурс – <https://elibrary.ru/defaultx.asp> - электронная научная библиотека.

**3 этап.** После глубокой проработки литературы необходимо реализовать выполнение заданий, приведенных в п.3 данных методических рекомендаций.

Основные этапы курсовой работы, объем их выполнения, оценка (по 100-бальной шкале) и представляемые преподавателю результаты, приведены в таблице 5.1.

**Таблица 5.1**

Этапы выполнения курсовой работы

Этап	Содержание этапа	Объем готовности работы	Представляемые результаты
1	<ul style="list-style-type: none"> <li>- Изучение методических указаний и исходных данных к каждому заданию на выполнение курсовой работы;</li> <li>- Анализ и составление примерной стратегии решения каждого задания</li> <li>- Выбор необходимых библиотек для реализации предметных обла-</li> </ul>	Объем работы - 20 %	<p><b>1-я контрольная точка</b></p> <p>Введение. Первая глава пояснительной записки курсовой работы.</p> <p>Перечень библиотек, планируемых к использованию для выполнения заданий к курсовой работе.</p>

	стей заданий		
2	Разработка алгоритмов и программной реализации каждого задания.	Объем работы - 30 %	<p><b>2-я контрольная точка</b></p> <p>Вторая глава пояснительной записки курсовой работы.</p> <p>Интерфейс программного продукта, специальный раздел пояснительной записки: информационная модель, описание входных данных.</p>
	1. Выполнение задания № 1	Объем работы - 50 %	<p>Разработанный алгоритм и программная реализация на языке программирования.</p> <p><b><u>Представляемые файлы к проверке задания № 1:</u></b>  <i>resource_1.txt – текстовый файл с входными данными</i>  <i>result_1.txt – текстовый файл с результатами работы программы задания № 1</i>  <i>exercise_1.py – файл с исходным программным кодом</i></p>
	2. Выполнение задания № 2:	Объем работы - 60 %	<p>Разработанный алгоритм и программная реализация на языке программирования и с использованием дополнительных библиотек.</p> <p><b><u>Представляемые файлы к проверке задания № 2:</u></b>  <i>resource_2.txt – текстовый файл с входными данными</i>  <i>result_2.txt – текстовый файл с результатами работы программы задания № 2</i>  <i>exercise_2.py – файл с исходным программным кодом</i></p>
	3. Выполнение задания № 3:  - реализация стандартного функционала	Объем работы - 70 %	<p>Интерфейс программного продукта, специальный раздел пояснительной записки: проектирование стандартного функционала.</p> <p><b><u>Представляемые файлы к проверке задания № 3.1:</u></b>  <i>exercise_3.py – файл с исходным программным кодом</i></p>

	- реализация расширенного функционала		Интерфейс программного продукта, специальный раздел пояснительной записки: проектирование расширенного функционала.  <b><u>Представляемые файлы к проверке задания № 3.2:</u></b> <i>exercise_3.py – файл с исходным программным кодом</i>
	- реализация дополнительных функций расширенного функционала		Интерфейс программного продукта, специальный раздел пояснительной записки: Индивидуальное задание реализации дополнительных функций.  <b><u>Представляемые файлы к проверке задания № 3.3:</u></b> <i>exercise_3.py – файл с исходным программным кодом</i>
	4. Выполнение задания № 4:  - реализация модифицированной задачи о Ханойских башнях	Объем работы - 80 %	Интерфейс программного продукта, специальный раздел пояснительной записки: Индивидуальное задание по расположению дисков в соответствии с ID студента.  <b><u>Представляемые файлы к проверке задания № 4:</u></b> <i>exercise_4.py – файл с исходным программным кодом</i>
3	Тестирование и отладка программного продукта. Разработка технической документации по сопровождению программного продукта	Объем работы - 90 %	<b>3-я контрольная точка</b>  Третья глава пояснительной записки курсовой работы.  Готовый программный продукт, инструкция пользователю по работе с программным продуктом, специальный раздел пояснительной записки: тестирование и отладка.
4	Завершение оформления пояснительной записки	Объем работы - 100%	<b>4-я контрольная точка</b>  Полностью оформленная записка в электронном варианте. Архив исходников к каждому заданию. Выгрузка окончательного варианта в электронный университет
5	Защита курсовой работы		Программа, записка, приложения.



### 3. Структура курсовой работы

#### 3.1. Содержание пояснительной записки

При выполнении курсовой работы необходимо придерживаться ряда требований к ее структуре. Прежде всего, все курсовые работы должны состоять из введения, аналитической главы, практической главы, заключения, списка литературы. К основному тексту даются приложения, которые размещаются в конце работы.

Типовые требования к курсовой работе определяют, обязательные к применению, стандарты:

ГОСТ 7.32.-2017. Отчет о научно-исследовательской работе. Структура и правила оформления;

ГОСТ 7.0.100-2018. Библиографическая запись. Библиографическое описание. Общие требования и правила составления;

ГОСТ Р 7.0.5-2008 Библиографическая ссылка. Общие требования и правила составления;

Структура курсовой работы также определяется ее исследовательскими задачами, т.е. каждый параграф в ней должен работать на решение одной задачи в рамках достижения сформулированной цели.

Пояснительная записка должна содержать обоснование выбора библиотек и алгоритмических решений, принятых обучающимся на каждом этапе разработки. Решения должны приниматься исходя из особенностей разрабатываемого задания и специфики исходных данных. Не должно быть обоснований типа «удобнее», «целесообразнее» и т. п. Необходимо пояснить, чем удобнее, почему целесообразно использование той или иной библиотеки. По возможности необходимо четко формулировать основания для принятия того или иного алгоритмического решения.

#### **Пример содержания пояснительной записки:**

##### Введение

##### 1. Анализ заданий курсовой работы

##### 1.1. Исходные данные к заданиям курсовой работы

##### 1.2. Анализ методических указаний, входных и выходных данных к заданиям курсовой работы

##### 1.2. Выбор и обоснование необходимых библиотек и среды разработки

#### **2. Разработка программного продукта для решения прикладных задач.**

##### **2.1. Работа с наборами данных (Задание №1)**

##### 2.1.1. Построение алгоритма решения задания без графического интерфейса

##### 2.1.2. Исходный код реализации на языке программирования

##### 2.1.3. Примеры тестирования и отладки.

##### 2.1.4. Скриншоты результатов работы

## **2.2. Разработка экспертной системы**

2.2.1. Построение алгоритма решения задания с графическим интерфейсом

2.2.2. Исходный код реализации на языке программирования

2.2.2.1. Проектирование стандартного функционала

2.2.2.2. Проектирование расширенного функционала

2.2.2.3. Индивидуальное задание реализации дополнительных функций

2.2.3. Тестирование и отладка.

2.2.4. Скриншоты результатов работы

## **2.3. Разработка аналитической системы**

2.3.1. Построение алгоритма решения задания с графическим интерфейсом

2.3.2. Исходный код реализации на языке программирования с использованием стандартных библиотек

2.3.3. Тестирование и отладка.

2.3.4. Скриншоты результатов работы

## **2.4. Разработка логико-аналитической системы**

2.4.1. Построение алгоритма решения задания «Ханойские башни»

2.4.2. Исходный код реализации на языке программирования с использованием стандартных библиотек

2.4.3. Тестирование и отладка.

2.4.4. Скриншоты результатов работы

2.5. Выводы по 2 главе

3. Разработка требований к техническим средствам реализации программного обеспечения для решения прикладных задач

Выводы

Список литературы

**Оглавление** должно быть создано с помощью инструмента «Автосо-  
держание» текстового редактора и **собираться автоматически из заголовков  
глав и параграфов.**

Во **введении** автор обосновывает тему курсовой работы, ее актуальность, степень разработанности, кратко характеризует современное состояние научной проблемы (вопроса), которой посвящена работа, определяет цель, задачи, объект и предмет исследования.

**Актуальность темы.** Обоснование того, почему нужно заниматься исследованием выбранной темы. **Обоснование актуальности темы** показывает главное – суть проблемной ситуации (противоречивой ситуации требующей своего разрешения). Актуальность темы означает ее связь с конкретными потребностями пользователей, отражает важность, своевременность выбранной темы, ее значимость.

**Цели и задачи.** Количество и название задач как правило совпадает с количеством параграфов работы, впереди ставится соответствующий глагол (рассмотреть, охарактеризовать и т.п.). От доказательства актуальности выбранной темы обучающийся должен логично перейти к **определению цели работы.** Как правило, целью исследования является разработка предложений

или методических рекомендаций на основе анализа существующей ситуации и полученных практических результатов анализа объекта исследования.

**Задачи** указывают на основные направления работы и начинаются с совершенных глаголов: рассмотреть..., охарактеризовать..., проанализировать..., оценить..., выявить..., разработать..., спроектировать..., автоматизировать..., разработать рекомендации..., предложить методику..., повысить экономическую эффективность... и т.п.

**Основные разделы** курсовой работы – четыре главы (по одной на каждое задание) и параграфы, которые содержат систематизированное изложение и результаты решения поставленной задачи. В главах и параграфах излагаются теоретические аспекты задачи, приводятся алгоритмы, за счет которых реализуется требуемый функционал. Приводятся тестовые примеры и результаты их выполнения в реализованных приложениях. Представляются основные скриншоты, показывающие работу приложений. В каждом разделе обязательно выкладывается исходный код, решающий поставленную задачу, так как задания сформулированы таким образом, чтобы их возможно было решить в рамках одного модуля, без использования внешних бинарных ресурсов. Рекомендуется логическое разбиение модуля на функции и классы согласно требованиям PEP8, однако работоспособность кода каждого приложения при переносе в один файл должна сохраняться. В случае использования среды разработки PyCharm код модулей при копировании в файл формата Word сохраняет исходное форматирование и подсветку, что повышает его читаемость в тексте пояснительной записки.

**Заключение** представляет собой краткое последовательное, логически стройное изложение полученных и описанных в основной части результатов, выводов исследования, построенных на анализе соотношения полученных результатов с общей целью и конкретными задачами исследования. Число выводов не должно быть большим, обычно оно определяется количеством поставленных задач.

**Список литературы** представляет собой оформленный в соответствии с установленными правилами перечень использованных в процессе исследования избранной темы: законов и подзаконных нормативных правовых актов, учебной и научной литературы, материалов периодической печати, материалов юридической практики, электронных ресурсов. Список литературы должен иметь следующую структуру:

- нормативные правовые акты;
- учебники, учебные пособия, монографии, комментарии к законодательству, словари, энциклопедии;
- научные статьи, материалы из периодических изданий;
- диссертации, авторефераты диссертаций;
- материалы юридической практики, справочно-статистические материалы;
- электронные ресурсы.

При использовании в тексте курсовой работы положений, цитат, заимствованных из литературы, обучающийся обязан делать ссылки на них в со-

ответствии с установленными правилами. Заимствования текста без ссылки на источник не допускаются.

Содержание курсовой работы должно соответствовать следующим основным требованиям:

- самостоятельность исследования;
- наличие анализа специальной литературы и интернет-источников по предметной области;
- наличие работоспособного файла конфигурации 1С;
- логичность изложения содержания курсовой работы, подробное описание процесса разработки конфигурации, аргументированность выводов и предложений;
- научно-практическая значимость курсовой работы.

## 3.2 Задания для выполнения в рамках курсовой работы.

### 3.2.1 Задание № 1

Работа с наборами данных	БЕЗ ГРАФИЧЕСКОГО ИНТЕРФЕЙСА (GUI)
формулировка	Во внешнем файле <code>resource_1.txt</code> дан текст. Выведите все слова, встречающиеся в тексте, по одному на каждую строку, через пробел укажите количество повторений. Слова должны быть отсортированы по убыванию их количества появления в тексте, а при одинаковой частоте появления — в лексикографическом порядке. Вывод должен осуществляться в текстовый файл <code>result_1.txt</code> . При необходимости можно продублировать вывод в консоль.
Методические указания	После того, как вы создадите словарь всех слов, необходимо отсортировать его по частоте встречаемости слова. Желаемого можно добиться, если создать список, элементами которого будут кортежи из двух элементов: частота встречаемости слова и само слово. Например, <code>[(2, 'hi'), (1, 'what'), (3, 'is')]</code> . Тогда стандартная сортировка будет сортировать список кортежей, при этом кортежи сравниваются по первому элементу, а если они равны — то по второму. Знаки препинания не должны учитываться. <b>Программу сохранить под именем <code>exercise_1.py</code></b>
Входные данные	Преподаватель вводит текст в текстовый файл <code>resource_1.txt</code> (5-6 абзацев) и сохраняет его. <pre> hi hi what is your name my name is bond james bond my name is damme van damme claudio van damme jean claudio van damme </pre>
Выходные данные	Выведите ответ на задание № 1 в текстовый файл <code>result_1.txt</code> . <pre> Damme 4 Is 3 Name 3 </pre>

	Van 3 Bond 2 claude 2 hi 2 my 2 james 1 jean 1 what 1
--	--

## 3.2.2. Задание № 2

Разработка экспертной системы	С ГРАФИЧЕСКИМ ИНТЕРФЕЙСОМ (GUI)	
формулировка	<p>Некоторый банк хочет внедрить систему управления счетами клиентов, поддерживающую следующие операции:</p> <ol style="list-style-type: none"><li>1. Пополнение счета клиента.</li><li>2. Снятие денег со счета.</li><li>3. Запрос остатка средств на счете.</li><li>4. Перевод денег между счетами клиентов.</li><li>5. Начисление процентов всем клиентам.</li></ol>	
Методические указания	<p>Необходимо реализовать такую систему. Первоначально у банка 1 клиент. Клиент(ы) банка идентифицируются именами (уникальная строка, не содержащая пробелов). Вам необходимо задать в качестве имени клиента – свою фамилию на английском языке с большой буквы. На вашу фамилию должен быть открыт счет с суммой равной вашему ID.</p> <p>Ivanov 70121903</p> <p>В отдельном поле должна быть предусмотрена возможность ввода простых команд, которые поддерживают следующие операции:</p>	
	DEPOSIT name sum	Зачислить сумму <b>sum</b> на счет клиента <b>name</b> . Если клиента нет, то он создается и на него заводится счет с указанной суммой.
	WITHDRAW name sum	Снять сумму <b>sum</b> со счета клиента <b>name</b> . Если клиента, то счет создается. Баланс при выполнении такой операции у вновь созданного клиента должен быть отрицательный.
	BALANCE name	Узнать остаток средств на счету клиента <b>name</b> . Для каждого запроса BALANCE программа должна вывести остаток на счету данного клиента. Если же у клиента с запрашиваемым именем не открыт счет в банке, выводится сообщение «NO CLIENT». Если пользователь не указал имя клиента – то выводится баланс всех существующих клиентов.
	TRANSFER name1 name2 sum	Перевести сумму <b>sum</b> со счета клиента <b>name1</b> на счет клиента <b>name2</b> . Если у какого-либо клиента, то он заводится в системе и ему создается счет с пере-

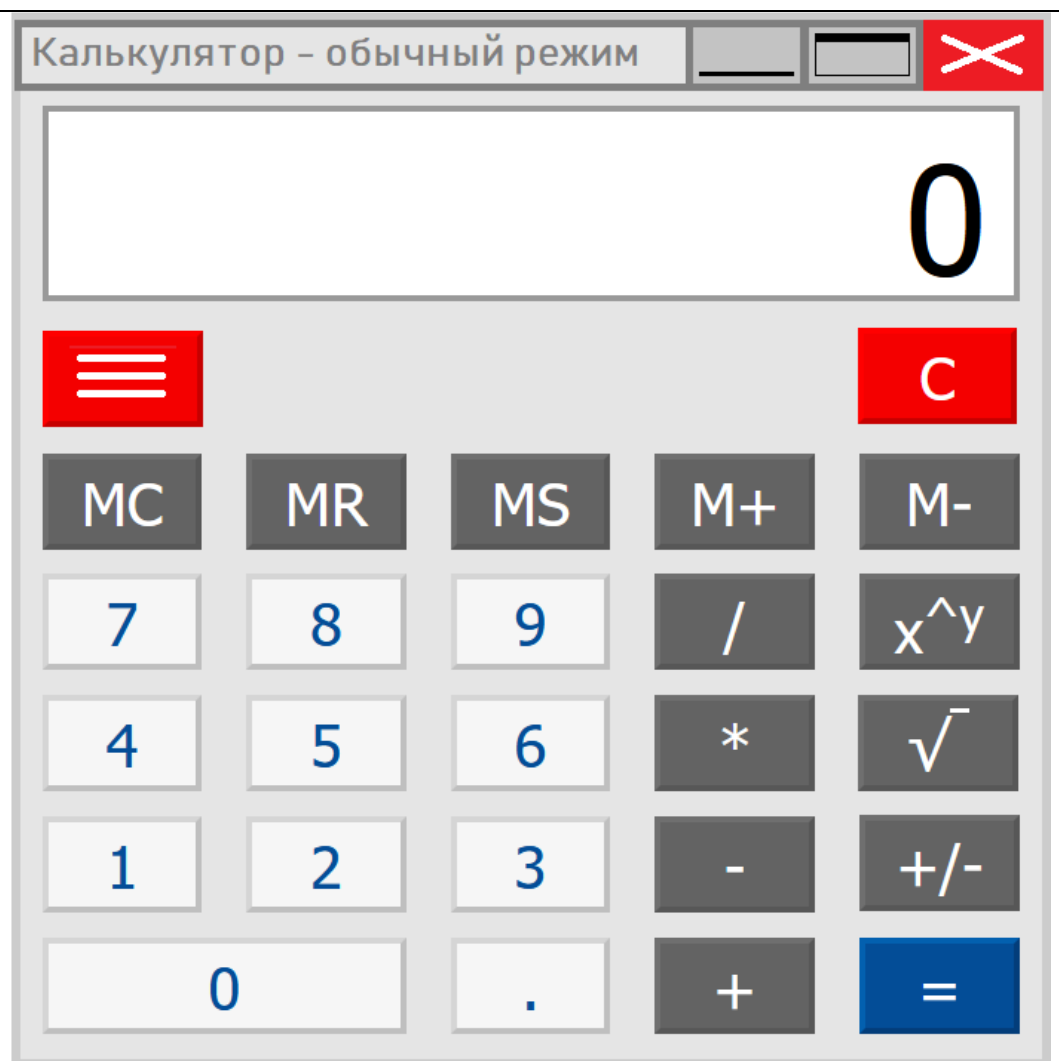
		денной суммой.
	INCOME p	Начислить всем клиентам, у которых открыты счета, <b>p%</b> от суммы счета. Проценты начисляются только клиентам с положительным остатком на счету, если у клиента остаток отрицательный, то его счет не меняется. После начисления процентов сумма на счету остается целой, то есть начисляется только целое число денежных единиц. Дробная часть начисленных процентов отбрасывается.
	<p>ПРОГРАММА ДОЛЖНА ОБРАБАТЫВАТЬ ТЕКСТОВЫЕ КОМАНДЫ ИЗ ЛЕВОГО ПОЛЯ ТОЛЬКО ПОСЛЕ НАЖАТИЯ КНОПКИ «Calculate». То есть, пользователь СНАЧАЛА вводит желаемые команды, при этом каждая новая команда вводится с новой строки, а ПОТОМ нажимает на кнопку «Calculate». Результат должен быть выведен в поле справа.</p> <p>Количество команд, которые может ввести пользователь за один раз – не более 20. При необходимости следует предусмотреть прокрутку в поле. Пользователю допускается вводить «пустые строки» - несколько раз нажимать на кнопку «enter». При нажатии клавиши ввода «enter», фокус не должен переходить на кнопку «Calculate».</p> <p>Формат и внешний вид окна определяет разработчик. Для «очистки» левого и правого полей можно предусмотреть кнопку «Clear», при этом данные о ранее введенных клиентах не должны быть потеряны.</p> <p><b>ВАЖНО:</b></p> <ol style="list-style-type: none"> <li>1. Команды вводятся пользователем только большими буквами. Сама команда, имя клиента, суммы (числа) разделяются пробелами.</li> <li>2. Предполагается, что пользователь такой системы грамотный и команды с аргументами вводит без ошибок в рамках их вышесформулированного синтаксиса.</li> <li>3. Как только для несуществующего ранее клиента проводится операция <u>пополнения</u> (DEPOSIT), <u>снятия</u> (WITHDRAW) или <u>перевода денег</u> (TRANSFER), он вносится в систему, ему заводится счет с указанным балансом. Все дальнейшие операции проводятся только с этим счетом. Сумма на счету может быть как положительной, так и отрицательной, при этом всегда является целым числом.</li> </ol> <p><b>Программу сохранить под именем <b>exercise_2.py</b></b></p>	
Входные данные	Преподаватель вводит текст в текстовый файл <b>resource_2.txt</b> (5-6 абзацев) и сохраняет его.	
Выходные	Выведите ответ на задание № 1 в текстовый файл <b>result_2.txt</b> .	

данные	Damme 4 Is 3 Name 3 Van 3 Bond 2 claude 2 hi 2 my 2 james 1 jean 1 what 1
--------	---



### 3.2.3. ЗАДАНИЕ № 3


Разработка аналитической системы	С ГРАФИЧЕСКИМ ИНТЕРФЕЙСОМ (GUI)
формулировка	<p>Разработать калькулятор со стандартным и расширенным функционалом.</p> <p>Стандартный функционал</p> <ol style="list-style-type: none"> <li>1. Арифметические действия <math>+</math> <math>-</math> <math>*</math> <math>/</math>.</li> <li>2. Возможность ввода отрицательного числа</li> <li>3. Возведение в степень.</li> <li>4. Извлечение квадратного корня.</li> <li>5. Работа с памятью, состоящей из одной ячейки.</li> <li>6. Должна быть кнопка сброса и кнопка «<math>\Rightarrow</math>» (равно).</li> </ol> <p>Расширенный функционал</p> <ol style="list-style-type: none"> <li>1. Наличие кнопки/меню перехода в расширенный режим</li> <li>2. Возможность работы с несколькими ячейками памяти. Количество ячеек памяти выбирается согласно методическим указаниям.</li> <li>3. Отображение последовательности математических операций и цифр в n-строчном «дисплее», с возможностью «прокрутки». Количество строк «дисплея» калькулятора выбирается согласно методическим указаниям.</li> <li>4. Реализация «инженерных» функций расширенного режима. Конкретный перечень функций выбирается согласно методическим указаниям.</li> </ol>
Методические указания	Необходимо разработать программу и GUI для реализации стандартных функций калькулятора. Примерный вид внешнего интерфейса представлен на рисунке.



Конкретная компоновка элементов интерфейса, внешний вид, цветовая палитра, максимальное количество отображаемых цифр, число отображаемых строк на «дисплее» калькулятора могут отличаться от представленного рисунка и должны согласовываться с руководителем курсовой работы.

Функционал «инженерного» режима работы калькулятора добавляется при нажатии на кнопку перехода в расширенный режим, либо выбора соответствующего пункта меню.

Расширенный режим работы определяется внешним видом «цифрового дисплея», количеством ячеек памяти (кнопки M+, M-, MS, MR, MS) и кнопками, отвечающими за дополнительные функции.

**Количество строк «цифрового дисплея»** должно определяться как последовательная сумма всех цифр ID студента. Суммирование отдельных цифр числа должно осуществляться до получения однозначного числа, состоящей из 1-й цифры. Данный процесс рекомендуется реализовать с помощью рекурсивной функции. Для данной функции должна быть предусмотрена специальная кнопка .

Например: обучающийся Иванов Иван Иванович, имеет ID 80121986

Сумма всех цифр равна:

$$8+0+1+2+1+9+8+6=35;$$

$$3+5=8$$

Следовательно, «цифровой дисплей» калькулятора должен иметь 8 строк.

50 + 3	1 строка	50 + 3
=	2 строка	=
53	3 строка	53
с	4 строка	с
20 * 2	5 строка	20 * 2
=	6 строка	=
40	7 строка	40
с	8 строка	с

Если после последовательного суммирования ID получилась цифра «1», например, для ID=82121986:

$$8+2+1+2+1+9+8+6=37;$$

$$3+7=10$$

$$1+0=1$$

В этом случае количество строк «цифрового дисплея» должно быть равно 10.

Таким образом, в зависимости от ID число «цифровых строк» дисплея калькулятора может варьироваться от 2-х до 10.

**Количество ячеек памяти** калькулятора должно определяться как последовательная сумма последних 3-х чисел ID. Суммирование отдельных цифр числа должно осуществляться до получения однозначного числа, состоящей из 1-й цифры. Данный процесс также рекомендуется реализовать с помощью рекурсивной функции.

Например: обучающийся Иванов Иван Иванович, имеет ID 80121986

Сумма последних трёх цифр равна:

$$9+8+6=23;$$

$$2+3=5$$

Следовательно, должно быть предусмотрено 5 ячеек памяти для работы с (M+, M-, MC, MR, MS).



Если после последовательного суммирования 3-х цифр ID получилась цифра «1», например, для ID=82121001:

$0+0+1=1$ ;

В этом случае количество ячеек памяти должно быть равно 2.

Таким образом, в зависимости от ID число ячеек памяти (и соответствующих кнопок, реализующий данный функционал) калькулятора может варьироваться от 2-х до 9.

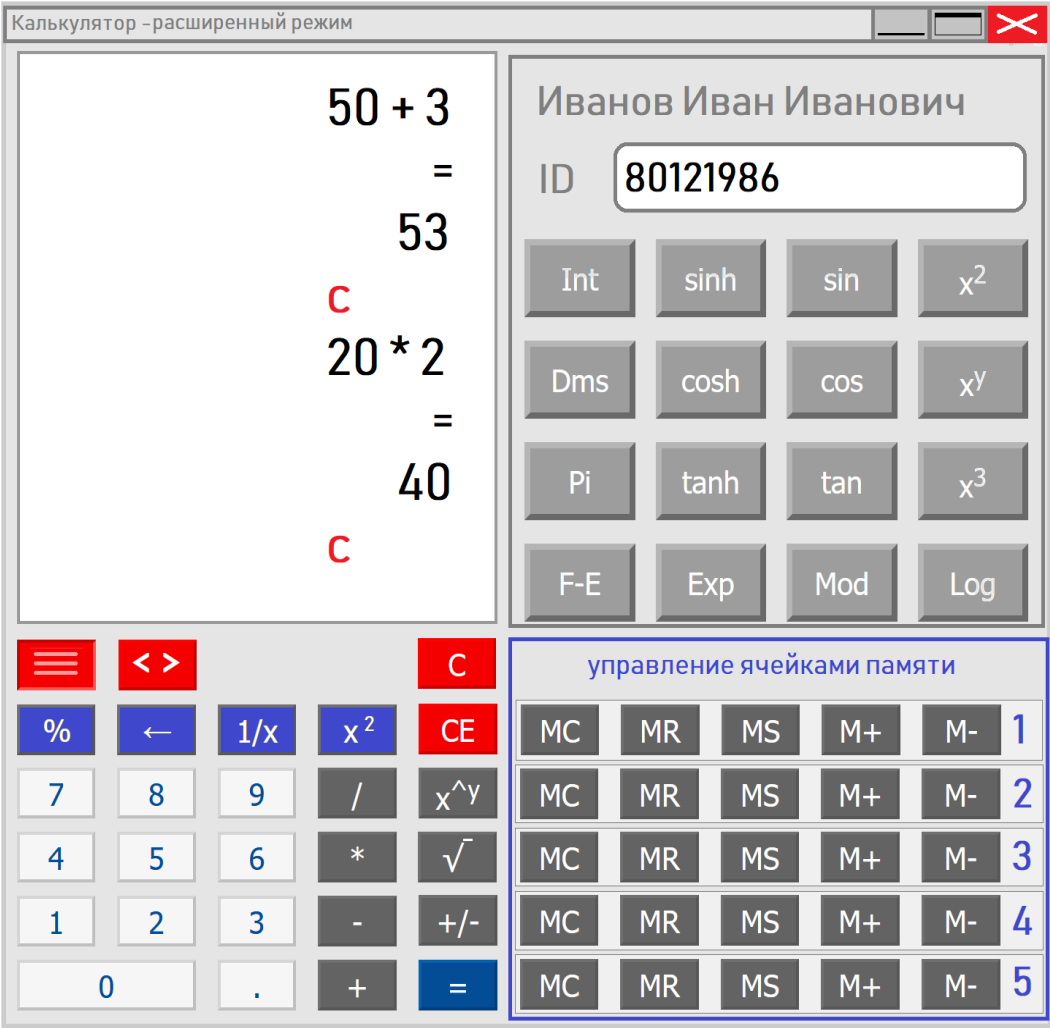
**Дополнительные функции расширенного режима**, которые должны быть реализованы определяются в соответствии с таблицей по первой букве Фамилии обучающегося

Первая буква фамилии	Наименование функции	Описание функции
А	Pi, sin, tan, exp, n!, Frac	Число Пи, синус, тангенс, экспонента, факториал; отсекает целую часть, оставляет дробную
Б	asin, acos, atg, log_xy, n!	Арксинус, арккосинус, арктангенс, логарифм по основанию, факториал
В	//, ctg, $10^x$ , asin, acos	Целочисленное деление, котангенс, 10 в степени x, арксинус, арккосинус
Г	F – E, $\sqrt[y]{x}$ , lg <sub>10</sub>	переключает ввод чисел в экспоненциальном представлении и обратно, у-ый корень числа x, где у обычно является положительным целым числом, десятичный логарифм
Д	gcd(a, b); tanh, Ln, $X^3$	Возвращает наибольший общий делитель a и b; гиперболический тангенс, натуральный логарифм по основанию «e»; возвести в степень 3
Е	hypot(x, y), isqrt(), Sinh, Mod	функция вычисляет гипотенузу треугольника с катетами x и y; Возвращает целочисленный квадратный корень аргумента, округлённый вниз; гиперболический синус, вычислить остаток от деления одного числа на другое
Ж	tanh, Ln, $X^3$	гиперболический тангенс, натуральный логарифм

		Frac	рифм по основанию «е», возвести в степень 3; отсекает целую часть, оставляет дробную
	З	$\sinh^{-1}$ , exp, asin, acos	обратный гиперболический синус, экспонента, арксинус, арккосинус
	И	Mod, tanh, asin, acos	вычислить остаток от деления одного числа на другое, гиперболический тангенс; арксинус, арккосинус
	К	Ln, $X^3$ , Dms, sin	натуральный логарифм по основанию «е», возвести в степень 3; переводит из десятичного вида в формат в градусы, минуты, секунды; синус
	Л	$X^3$ , asin, acos	Возведение в куб, арксинус, арккосинус
	М	Sinh, Mod, $\sqrt[y]{x}$ , lg <sub>10</sub>	гиперболический синус, вычислить остаток от деления одного числа на другое; у-ый корень числа x, где у обычно является положительным целым числом, десятичный логарифм
	Н	deg, $\sqrt[y]{x}$ , lg <sub>10</sub> , sin, cos	перевод угла в градусах, минутах и секундах в десятичные доли градуса; у-ый корень числа x, где у обычно является положительным целым числом, десятичный логарифм; синус, косинус
	О	Int, Pi, tanh, Ln, $X^3$	отображает целую часть десятичного числа, число Пи, выдает значение Pi для расчетов; гиперболический тангенс, натуральный логарифм по основанию «е», возвести в степень 3
	П	Inv, sin, cos, tan	обратная функция для sin, cos, tan, переключает интерфейс на другие функции; синус; косинус; тангенс
	Р	Dms, sin, cos, tan	переводит из десятичного вида в формат в градусы, минуты, секунды; синус; косинус; тангенс
	С	Dms, 10 <sup>x</sup> , Pi, tanh, Ln	переводит из десятичного вида в формат в градусы, минуты, секунды; возведение десяти в произвольную степень, число Пи, гиперболический тангенс, натуральный логарифм
	Т	F – E, acos, atg, log <sub>x</sub> y, n!	переключает ввод чисел в экспоненциальном представлении и обратно; арккосинус, арктангенс, логарифм по основанию, факториал
	У	Pi, sin, tan, exp, asin, acos	Число Пи, синус, тангенс, экспонента, арксинус, арккосинус
	Ф	Mod, tanh, Ln, $X^3$ , exp, asin, acos	вычислить остаток от деления одного числа на другое, гиперболический тангенс, натуральный логарифм по основанию «е», возвести в степень 3, экспонента, арксинус, арккосинус
	Х	pow(a, b), n!, F – E, acos	функция выполняет возведение числа a в степень b и возвращает затем вещественный результат; вычисление факториала; переключает ввод чисел в экспоненциальном представлении и обратно; арккосинус
	Ц	Pi, sin, tan,	Число Пи, синус, тангенс; Округляет число до

	floor()	ближайшего целого, но в меньшую сторону
Ч	sin, cos, tan, Ln, X <sup>3</sup>	синус; косинус; тангенс; натуральный логарифм по основанию «е», возвести в степень 3
Ш	Mod, tanh, F – E, acos	вычислить остаток от деления одного числа на другое, гиперболический тангенс, переключает ввод чисел в экспоненциальном представлении и обратно; арккосинус
Щ	F – E, acos, tanh, Ln, X <sup>3</sup>	переключает ввод чисел в экспоненциальном представлении и обратно; арккосинус; гиперболический тангенс, натуральный логарифм по основанию «е», возвести в степень 3
Э	tanh, Ln, X <sup>3</sup> , log_xy, n!	гиперболический тангенс, натуральный логарифм, возведение в куб, логорифм по основанию, факториал
Ю	X <sup>3</sup> , 10^x, Pi, tanh	возведение в куб, возведение десяти в произвольную степень, число Пи, гиперболический тангенс
Я	tanh, asin, acos, F – E	гиперболический тангенс; арксинус, арккосинус, переключает ввод чисел в экспоненциальном представлении и обратно

Примерный внешний вид, реализующий данный функционал может быть следующим:

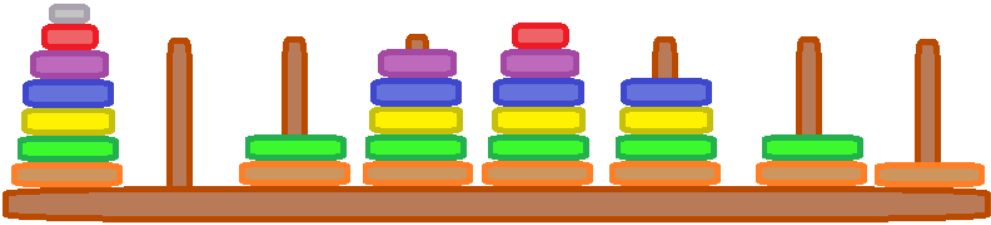
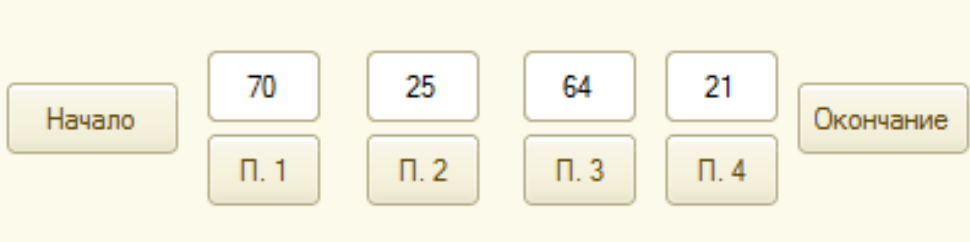


	<p><b>ВАЖНО:</b></p> <ol style="list-style-type: none"> <li>1. Программа должна использовать распространенные библиотеки. Если с согласия руководителя предполагается для реализации дополнительных функций и отрисовки интерфейса использование сторонних библиотек, то в пояснительной записке к курсовой работе должно быть представлено обоснование и подробное их описание.</li> <li>2. При разработке программы следует использовать принципы объектно-ориентированного программирования.</li> <li>3. Калькулятор должен работать и корректно выдавать результат.</li> <li>4. Размер кнопок, цветовая палитра, расположение и т.п. определяется самостоятельно. Внешний вид калькулятора не оценивается, однако следует придерживаться традиционных компоновок.</li> <li>5. При намеренном выполнении некорректных операций, например, извлечение квадратного корня из отрицательного числа, калькулятор должен на цифровой дисплей выдавать соответствующее сообщение об ошибке. Если же программу останавливает интерпретатор – то проверяемая функция считается не до конца отработанной.</li> </ol> <p><b>Программу «Калькулятор» сохранить под именем <code>exercise_3.py</code></b></p>
Входные данные	<p>Преподаватель с помощью мыши вводит числа и выполняет арифметические операции.</p> <p>Преподаватель с помощью мыши вводит числа и выбирает заявленные дополнительные функции.</p> <p>Преподаватель вводит намеренно некорректные операции (попытка деления на 0)</p>
Выходные данные	<p>На цифровом дисплее должен отображаться результат, либо сообщение об ошибке при некорректной операции.</p> <p>Проверяется количество строк в «цифровом дисплее» калькулятора и количество ячеек памяти.</p>

### 3.2.4. ЗАДАНИЕ № 4

Задача о Ханойских башнях	С ГРАФИЧЕСКИМ ИНТЕРФЕЙСОМ (GUI)
формулировка	<p>Модифицированная задача о Ханойских башнях:</p> <p>Существует 8 шпинделей, пронумерованных от 8 до 1 слева направо. На каждом шпинделе надеты диски, в количестве, равном соответствующей цифре из ID студента. Все диски имеют разные диаметры. Диаметр диска равен <math>M * 10 + N</math>, где <math>M</math> – номер шпинделя, на котором надет диск, а <math>N</math> – это номер диска на шпинделе, считая сверху вниз.</p> <ol style="list-style-type: none"> <li>1. Необходимо визуально изобразить предложенную задачу. Диски на шпинделях сделать случайных цветов. На каждом диске отображать цифру, равную его диаметру. Диаметр диска также показывать его фактическим размером в пикселях.</li> <li>2. Необходимо вычислить, за какое минимальное количество итераций переместятся все диски на шпиндель номер 1 по следующим правилам:             <ol style="list-style-type: none"> <li>а) За одну итерацию можно переместить не более одного диска</li> <li>б) Диски можно класть только с большего на меньший</li> <li>в) Со шпинделя номер 8 можно перекладывать диски только на шпиндели 7 и 6</li> <li>г) Со шпинделя номер 1 можно перекладывать диски только на шпиндели номер 2 и 3</li> <li>д) Со шпинделей от 2 по 7 можно перекладывать диски только на два соседних шпинделя.</li> </ol> </li> <li>3. Необходимо отобразить начальное и конечное расположение дисков на шпинделях, для этого под изображением Ханойских башен предусмотреть две кнопки «Начало» и «Окончание». При нажатии на нее, в надписи под схемой должен выводиться текст «Итерация XX», где XX – номер итерации (либо 0, либо номер итоговой итерации, соответственно).</li> <li>4. Необходимо графически отобразить четыре промежуточные итерации перекладывания дисков. Для этого:             <ol style="list-style-type: none"> <li>а) общее количество итераций признаётся равным 100%,</li> <li>б) ID студента делится на 4 двузначных числа, каждое из которых обозначает итерацию, соответствующую этому проценту выполнения общей задачи.</li> <li>в) Под изображением Ханойских башень предусмотреть четыре поля для ввода цифр с процентами выполнения. По-умолчанию добавить туда числа из п. б)</li> <li>г) Под каждым полем для ввода предусмотреть кнопку, при нажатии на которую схема Ханойской башни отображает расположение дисков на соответствующей итерации. Также в надписи под схемой должен выводиться текст «Итерация XX», где XX – номер итерации</li> </ol> </li> </ol>



	<p>5. Дать возможность пользователю изменять проценты в полях для ввода цифр, и по нажатию соответствующей кнопки просматривать расположение дисков на данной итерации.</p>
<p>Методические указания</p>	<p>Так как задача алгоритмически достаточно проста, то основная часть работы над задачей студента сводится к правильной визуализации полученных результатов, а также оптимальному поиску промежуточных результатов. Схема ханойских башен должна выглядеть примерно таким образом:</p>  <p>(В данном примере расположение дисков соответствует ID студента 70256421)</p> <p>Так как размер диска должен соответствовать его номеру на шпинделе и номеру самого шпинделя, умноженного на 10, то как нетрудно догадаться, максимальный диаметр диска может быть 89. Поэтому для правильной визуализации без наложения дисков рекомендуется выдерживать расстояние между шпинделями примерно в 100-120 пикселей (для окна формата 1280x1024).</p> <p>Поскольку диски близких размеров будут отличаться всего на один пиксель, то для контроля на каждом диске необходимо проставить его диаметр в виде цифры. Так как в конце задачи все диски будут находиться на первом шпинделе, а общее количество дисков теоретически может быть равно 72, то рекомендуется сделать толщину одного диска примерно равной 10-12 пикселям, для указанного окна.</p> <p>Остальные элементы управления под схемой Ханойской башни рекомендуется выстраивать в следующем порядке:</p>  <p>На данном рисунке также видно, как следует разбивать ID студента для выведения промежуточных итогов. При нажатии на кнопки, нужно показывать итерации, соответствующие 70%, 25%, 64% и 21% выполнения задачи.</p> <p><b>Уточнение</b></p> <p>Если по какому-либо проценту получается дробная итерация, то необходимо её визуализировать как промежуточный этап переноса диска. При этом диск изобразить в воздухе, между тем шпинделем, с которого он снят, и тем, на который он переносится. Номер итерации в таком случае отображать как дробный, с округлением до 3 цифр после нуля.</p>

	<b>Программу сохранить под именем <code>exercise_4.py</code></b>
Входные данные	Идентификатор студента.  Промежуточные проценты, вводимые преподавателем в соответствующие поля над кнопками.
Выходные данные	На цифровом дисплее должно отображаться окно с начальным расположением дисков на шпинделях Ханойских башень. Шпиндели пронумерованы, на дисках также обозначены соответствующие диаметры. Под ней отображается шесть кнопок и четыре поля для ввода цифр. В нижней части экрана демонстрируется надпись «Итерация 0»  При нажатии на любую из шести имеющихся кнопок, либо при заполнении поля ввода другими данными и нажатии на кнопку, схема ханойских башень меняется, для отображения соответствующей итерации. Надпись в нижней части экрана также меняется.

#### 4. Оформление курсовой работы

Курсовая работа выполняется в электронной форме и размещается обучающимся в личном кабинете в виде отдельного файла.

Курсовая работа должна быть представлена в формате редактируемого файла, например: .odt, .docx, .rtf. Нельзя предоставлять курсовую работу в виде нередатируемого pdf-файла. Имя файла должно иметь следующий вид: «ФИО\_КР\_ВысМетПрог.\*» Например: «Иванов И.И.\_КР\_ВысМетПрог.docx» или «Иванов И.И.\_КР\_ВысМетПрог\_Отчет.rtf»

Курсовая работа представляется к защите в виде файла с пояснительной запиской. **Приложения дополнительных файлов не допускается, задания сформированы таким образом, чтобы полный листинг исходного кода решения задач можно было выложить в текстовом формате непосредственно в пояснительной записке. Замена исходного кода скриншотами среды разработки не допускается. Исходный код должен иметь возможность компиляции и выполнения при копировании в среду разработки из пояснительной записки.**

Пояснительная записка обязательно должна содержать титульный лист со штрих-кодом, подтверждающий закрепление темы курсовой работы за студентом в электронной образовательной среде Университета. Титульный лист можно получить в системе «Электронный университет» и внедрить в виде изображения в качестве первой страницы курсовой работы. Самостоятельно набирать титульный лист в файле не допускается, изменять титульный лист, полученный из системы также не допускается.

**При отсутствии оформленного в системе титульного листа работа к защите не принимается.**

Общий объем курсовой работы не должен превышать 5000 слов, исключая пробелы, рисунки, схемы и исходный код приложений (от 25 до 30 страниц шрифтом Times New Roman, размер 14, межстрочный интервал – 1,5; для сносок и подстрочных пояснений межстрочный интервал – 1, шрифт TNR, размер – 10). Существенное отклонение от объема работы является серьезным

нарушением и повлечет за собой снижение оценки.

Размеры оставляемых полей – «Обычные». Текст выравнивается по ширине страницы. Сноски должны иметь сквозную нумерацию. Расстояние между названием части работы или главы и последующим текстом должно быть равно двум межстрочным интервалам. Фразы, начинающиеся с «красной» строки, печатаются с абзацным отступом от начала, равным 1,25 см.

Текст глав курсовой работы должен распределяться на параграфы. Главы должны быть пронумерованы римскими цифрами в пределах всей работы. «Введение» и «Заключение» не нумеруются. Параграф нумеруется арабскими цифрами в пределах каждой главы. Номер параграфа должен состоять из номера главы и номера параграфа, разделенный точкой. В конце номера параграфа также следует ставить точку, например, «2.1.» (первый параграф второй главы). Номер соответствующей главы или параграфа ставится в начале заголовка. Каждая глава должна содержать 2 параграфа.

Заголовки глав, заголовки параграфов, а также слова «Оглавление», «Введение», «Заключение», «Список литературы» пишутся ПРОПИСНЫМИ буквами, следует располагать посередине строки без точки в конце.

Подчеркивать заголовки и переносить слова в заголовках запрещается. Расстояние между заголовками и последующим текстом должно быть равно двум межстрочным интервалам, расстояние между заголовком и последней строчкой предыдущего текста – двум межстрочным интервалам. Каждую главу следует начинать с нового листа (страницы), а параграфы продолжать, отступив от предыдущего текста два межстрочных интервала.

Нумерация страниц текста должна быть сквозной, первой страницей является титульный лист, второй – оглавление. На последующих страницах номер проставляется арабскими цифрами в правом углу верхнего поля без отточий и дефисов. На страницах 1 и 2 (титульный лист и оглавление) номер страницы не ставится. Рисунки и таблицы, которые расположены на отдельных страницах, тоже включаются в общую нумерацию.

Не допускается сокращение слов и наименований документов, а также перенасыщение текста специальными терминами, затрудняющими чтение. Приводимый цифровой аналитический материал помещается в таблицы, которые нумеруются по тексту.

Иллюстрации (чертежи, графики, схемы, диаграммы и др.) следует располагать непосредственно после текста, в котором они упоминаются впервые, или на следующей странице. На все иллюстрации должны быть ссылки в тексте.

Цифровой материал, как правило, должен оформляться в виде таблицы, вариант изображения которой представлен в приложении. Таблицу следует располагать в работе непосредственно после текста, в котором она упоминается впервые, или на следующей странице. На все таблицы должны быть ссылки в тексте.

Все таблицы и рисунки, если их несколько, имеют сквозную нумерацию в пределах всего текста. Если информация, приводимая в таблице, заимствована из каких-либо источников, то после названия таблицы необходимо поста-

вить ссылку. Данные в таблице должны быть представлены шрифтом 12 размера и одинарным межстрочным интервалом без отступа (красной строки).

Если таблица имеет большой размер, то ее необходимо поместить в приложение. Если все же размещение таблицы в тексте признано более целесообразным, то она переносится на следующие страницы с авто переносом шапки таблицы с использованием команды «Повторять как заголовок на каждой странице» с использованием функции MS Word «Свойства таблицы».

В названии таблицы или в строке, содержащей наименование показателей, должны быть указаны единицы измерения приводимых значений (например, тыс. руб., млн руб.).

Формулы, используемые для расчетов, должны располагаться на отдельных строках и нумероваться. Порядковые номера формул обозначают арабскими цифрами, которые записывают на уровне формулы справа в круглых скобках.

Формулы следует располагать посередине строки и обозначать порядковой нумерацией в пределах всей курсовой работы арабскими цифрами в круглых скобках в крайнем правом положении на строке. Одну формулу обозначают (1).

Если формул используется немного, то допускается сквозная нумерация по всей работе. Если в тексте используется большое количество формул, то нумерация указывается двойная: первая цифра отражает номер главы, вторая – ее порядковое положение в главе.

Оформление формул осуществляется с использованием функции «Формула» в программном комплексе Microsoft Word. Непосредственно под формулой приводится расшифровка смысла и значений символов.

Иллюстрации – схемы и графики, именуемые рисунками, нумеруются сквозной нумерацией по всей работе, обозначаются арабскими цифрами. Если иллюстрация в работе единственная, то она не нумеруется.

Схемы в работе должны быть сгруппированы в единый объект. Иллюстрации следует располагать непосредственно после текстов, в которых они упоминаются впервые, или на следующей странице.

Ссылки на иллюстрации не следует оформлять как самостоятельные фразы, в которых лишь повторяется то, что содержится в подписи. В том месте, где речь идет о теме, связанной с иллюстрацией, помещают ссылку либо в виде заключенного в скобки выражения «(Рис. 3)», либо в виде оборота типа: «...как это показано на рис. 3» или «... как это следует из рис. 3».

Каждую иллюстрацию необходимо снабжать подрисуночной подписью, которая должна соответствовать основному тексту и самой иллюстрации. Подпись под иллюстрацией имеет следующие основные элементы:

- тематический заголовок иллюстрации, содержащий текст с характеристикой изображаемого объекта в наиболее краткой форме;
- ссылка на источник, откуда взят рисунок, если это необходимо.

***Следует помнить, что все таблицы, рисунки, схемы, иллюстрации, формулы и т.п. выполняются обучающимся лично с использованием соответствующих редакторов и инструментов MS Word.***

После «Заключения» в курсовой работе приводится список литературы. Описание источников осуществляется в соответствии с ГОСТ Р 7.0.5-2008.

При систематическом расположении список литературы необходимо расположить в следующем порядке:

- книги, научные разработки по теме;
- учебные издания;
- статьи из периодических изданий;
- справочные издания;
- интернет-ресурсы.

Общее количество источников в курсовой работе должно быть в пределах **12-15** наименований.

**При выполнении курсовой работы, обучающийся постоянно сверяет выполненные элементы работы с индикаторами оценивания курсовой работы.**

## **5. Порядок аттестации и защиты курсовой работы**

Законченная и полностью оформленная курсовая работа не позднее чем за две недели до начала экзаменационной сессии, загружается в Электронный университет для проверки руководителем и предварительной оценки. Студенты заочной формы обучения (кроме групп выходного дня) представляют курсовую работу не позднее дня начала очередной сессии.

Руководитель проверяет работу на объем заимствований и при условии законченного правильного оформления и положительной оценки содержания, допускает работу к защите. Работа, не отвечающая установленным требованиям, возвращается для доработки с учетом сделанных замечаний и повторно предъявляется на кафедру в срок, указанный руководителем, но до начала экзаменационной сессии (по заочной форме – до зачета/экзамена по соответствующей дисциплине).

Уровень авторского текста в курсовой работе должен быть **не меньше 60%**, допускается **30% верно оформленного цитирования**, а также **не более 10% заимствований**. Авторство работы проверяется в системе «Антиплагиат», интегрированной в электронную образовательную среду Университета им. С.Ю.Витте. Отчеты с общедоступного сайта [antiplagiat.ru](http://antiplagiat.ru) не являются подтверждением авторства работы в пределах Университета и при защите работы к рассмотрению не принимаются. **Работа, в которой обнаружен недостаточный уровень авторского текста, к защите не допускается.**

Графическая часть работы является не иллюстративным материалом, а технической документацией на разработанный студентом проект базы данных. Поэтому при подготовке инфологических, информационных и физических моделей необходимо соблюдение соответствующих стандартов проектирования.

В программном продукте проверяется:

- соответствие программы заданиям;
- работоспособность в различных режимах.

На защите студент коротко (3–5 мин.) докладывает об основных программных решениях, принятых в процессе разработки, и отвечает на вопросы преподавателя, и при необходимости демонстрирует работающий программный код. Программный код должен работать корректно, используя интерпретатор Python 3.9 в среде разработки PyCharm Community.

Оценка за курсовую работу выставляется с учетом:

- качества выполненного программного продукта по каждому заданию, соответствие требованиям Соглашения «PEP 8»
- работоспособности программного кода
- наличие подробных комментариев в коде
- правильности оформления записки
- результатов защиты

Оценивание курсовой работы осуществляется по балльно-рейтинговой системе, установленной для всех форм учебной деятельности студента.

В случае несогласия студента с оценкой, защита курсовой работы выполняется при комиссии, состоящей не менее чем из двух преподавателей.

Студент, не представивший в установленный срок курсовую работу или не защитивший ее по неуважительной причине, считается имеющим академическую задолженность.

## 6. ПРИЛОЖЕНИЯ

### Приложение 1



# МОСКОВСКИЙ УНИВЕРСИТЕТ ИМ. С.Ю.ВИТТЕ

Факультет Информационных технологий

Кафедра Кафедра информационных систем

Направление подготовки/Специальность Прикладная информатика

## КУРСОВАЯ РАБОТА

по дисциплине Высокоуровневые методы программирования

Студента   
(фамилия, имя, отчество)

на тему: Разработка программного продукта для решения прикладных задач  
(тема работы)

Руководитель работы К.Т.Н.,   
(ученая степень, звание, инициалы и фамилия)

## ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ .....	33
1. АНАЛИЗ ЗАДАНИЙ КУРСОВОЙ РАБОТЫ .....	34
1.1. Исходные данные к заданиям курсовой работы .....	34
1.2. Анализ методических указаний, входных и выходных данных к заданиям курсовой работы.....	36
1.2. Выбор и обоснование необходимых библиотек и среды разработки ...	40
1.3. Выводы по 1 главе .....	40
2. РАЗРАБОТКА ПРОГРАММНОГО ПРОДУКТА ДЛЯ РЕШЕНИЯ ПРИКЛАДНЫХ ЗАДАЧ. ....	41
2.1. Работа с наборами данных.....	41
2.1.1. Построение алгоритма решения задания без графического интерфейса .....	41
2.1.2. Разработка программной реализации на языке программирования ..	42
2.2. Разработка экспертной системы .....	43
2.2.1. Построение алгоритма решения задания с графическим интерфейсом .....	43
2.2.2. Разработка программной реализации на языке программирования ..	43
2.3. Разработка аналитической системы .....	45
2.3.1. Построение алгоритма решения задания с графическим интерфейсом .....	45
2.3.2. Разработка программной реализации на языке программирования и с использованием дополнительных библиотек .....	46
2.2.2.1. Проектирование стандартного функционала .....	46
2.2.2.2. Проектирование расширенного функционала.....	48
2.3 Разработка аналитической системы .....	50
2.3.1. Построение алгоритма решения задания с графическим интерфейсом .....	50
2.4.2. Разработка программной реализации на языке программирования и с использованием дополнительных библиотек .....	50
2.5. Тестирование и отладка .....	52
2.6. Выводы по 2 главе .....	55
3 РАЗРАБОТКА ТРЕБОВАНИЙ К ТЕХНИЧЕСКИМ СРЕДСТВАМ РЕАЛИЗАЦИИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ РЕШЕНИЯ ПРИКЛАДНЫХ ЗАДАЧ .....	56
Выводы .....	57
Список использованной литературы.....	58



## **ВВЕДЕНИЕ**

В условиях быстро развивающейся современной высокотехнологичной экономики актуальной задачей является разработка различных автоматизированных систем. Они позволяют наиболее эффективно использовать различные современные средства информационной вычислительной техники и решать прикладные задачи. Для успешного решения таких задач нужно уметь использовать высокоуровневые средства разработки, декомпозировать задачу, осуществлять тестирование и сопровождение программных продуктов.

В данной работе решаются задачи высокоуровневого программирования, согласно индивидуальному заданию.

Актуальность курсовой работы связана с наличием большого количества прикладных задач, которые необходимо эффективно решать с помощью инструментов высокоуровневых методов программирования.

## 1. АНАЛИЗ ЗАДАНИЙ КУРСОВОЙ РАБОТЫ

### 1.1. Исходные данные к заданиям курсовой работы

Курсовая задания содержит 4 задания, рассмотрим подробнее каждое из них.<sup>1</sup>

Работа с наборами данных.

Формулировка: Во внешнем файле `resource_1.txt` дан текст. Выведите все слова, встречающиеся в тексте, по одному на каждую строку, через пробел укажите количество повторений. Слова должны быть отсортированы по убыванию их количества появления в тексте, а при одинаковой частоте появления — в лексикографическом порядке. Вывод должен осуществляться в текстовый файл `result_1.txt`. При необходимости можно продублировать вывод в консоль.

Разработка экспертной системы.

Формулировка: Некоторый банк хочет внедрить систему управления счетами клиентов, поддерживающую следующие операции:

1. Пополнение счета клиента.
2. Снятие денег со счета.
3. Запрос остатка средств на счете.
4. Перевод денег между счетами клиентов.
5. Начисление процентов всем клиентам.

Разработка аналитической системы

Формулировка: разработать калькулятор со стандартным и расширенным функционалом.

Стандартный функционал

1. Арифметические действия  $+$   $-$   $*$   $/$ .
2. Возможность ввода отрицательного числа
3. Возведение в степень.
4. Извлечение квадратного корня.
5. Работа с памятью, состоящей из одной ячейки.
6. Должна быть кнопка сброса и кнопка « $\Rightarrow$ » (равно).

---

<sup>1</sup> Приведено в сокращенном виде. Полный текст задания в методических материалах

### Расширенный функционал

1. Наличие кнопки/меню перехода в расширенный режим
2. Возможность работы с несколькими ячейками памяти. Количество ячеек памяти выбирается согласно методическим указаниям.
3. Отображение последовательности математических операций и цифр в n-строчном «дисплее», с возможностью «прокрутки». Количество строк «дисплея» калькулятора выбирается согласно методическим указаниям.
4. Реализация «инженерных» функций расширенного режима. Конкретный перечень функций выбирается согласно методическим указаниям.

### Задача о Ханойских башнях:

#### Модифицированная задача о Ханойских башнях:

Существует 8 шпинделей, пронумерованных от 8 до 1 слева направо. На каждом шпинделе надеты диски, в количестве, равном соответствующей цифре из ID студента. Все диски имеют разные диаметры. Диаметр диска равен  $M * 10 + N$ , где  $M$  – номер шпинделя, на котором надет диск, а  $N$  – это номер диска на шпинделе, считая сверху вниз.

1. Необходимо визуально изобразить предложенную задачу. Диски на шпинделях сделать случайных цветов. На каждом диске отображать цифру, равную его диаметру. Диаметр диска также показывать его фактическим размером в пикселях.
2. Необходимо вычислить, за какое минимальное количество итераций переместятся все диски на шпиндель номер 1
3. Необходимо отобразить начальное и конечное расположение дисков на шпинделях, для этого под изображением Ханойских башен предусмотреть две кнопки «Начало» и «Окончание».
4. Необходимо графически отобразить четыре промежуточные итерации перекладывания дисков.

5. Дать возможность пользователю изменять проценты в полях для ввода цифр, и по нажатию соответствующей кнопки просматривать расположение дисков на данной итерации.

Таким образом, задания включают в себя основные прикладные задачи, которые решаются разработчиками программного обеспечения, это и работа с данными, использование алгоритмов и структур данных и построение ПО с графическим пользовательским интерфейсом.

## 1.2. Анализ методических указаний, входных и выходных данных к заданиям курсовой работы

Следует рассмотреть методические указания и входные данные к заданиям курсовой работы.<sup>2</sup>

Для первого задания методические указания звучат так: после того, как вы создадите словарь всех слов, необходимо отсортировать его по частоте встречаемости слова. Желаемого можно добиться, если создать список, элементами которого будут кортежи из двух элементов: частота встречаемости слова и само слово. Например, [(2, 'hi'), (1, 'what'), (3, 'is')]. Тогда стандартная сортировка будет сортировать список кортежей, при этом кортежи сравниваются по первому элементу, а если они равны — то по второму. Знаки препинания не должны учитываться.

Входные данные: Преподаватель вводит текст в текстовый файл `resource_1.txt` (5-6 абзацев) и сохраняет его.

Выходные данные: Текстовый файл `result_1.txt`.

Для второго задания методические указания звучат следующим образом: Необходимо реализовать такую систему. Первоначально у банка 1 клиент. Клиент(ы) банка идентифицируются именами (уникальная строка, не содержащая пробелов). Вам необходимо задать в качестве имени клиента — свою фамилию на английском языке с большой буквы. На вашу фамилию должен

---

<sup>2</sup> Приведены в сокращенном виде. Полный текст в методических материалах

быть открыт счет с суммой равной вашему ID. В отдельном поле должна быть предусмотрена возможность ввода простых команд:

- **DEPOSIT name sum** - Зачислить сумму sum на счет клиента name. Если клиента нет, то он создается и на него заводится счет с указанной суммой.
- **WITHDRAW name sum** - Снять сумму sum со счета клиента name. Если клиента, то счет создается. Баланс при выполнении такой операции у вновь созданного клиента должен быть отрицательный.
- **BALANCE name** - Узнать остаток средств на счету клиента name. Для каждого запроса BALANCE программа должна вывести остаток на счету данного клиента. Если же у клиента с запрашиваемым именем не открыт счет в банке, выводится сообщение «NO CLIENT». Если пользователь не указал имя клиента – то выводится баланс всех существующих клиентов.
- **TRANSFER name1 name2 sum** - Перевести сумму sum со счета клиента name1 на счет клиента name2. Если у какого-либо клиента, то он заводится в системе и ему создается счет с переведенной суммой.
- **INCOME p** - Начислить всем клиентам, у которых открыты счета, p% от суммы счета. Проценты начисляются только клиентам с положительным остатком на счету, если у клиента остаток отрицательный, то его счет не меняется. После начисления процентов сумма на счету остается целой, то есть начисляется только целое число денежных единиц. Дробная часть начисленных процентов отбрасывается.

Программа обрабатывает текстовые команды из левого поля только после нажатия кнопки «Calculate». То есть, пользователь СНАЧАЛА вводит желаемые команды, при этом каждая новая команда вводится с новой строки, а ПОТОМ нажимает на кнопку «Calculate». Результат должен быть выведен в поле справа.

Входные данные: Преподаватель вводит в поле ввода команды<sup>3</sup>

Выходные данные: Преподаватель наблюдает вывод

Методические указания к третьему заданию: необходимо разработать программу и GUI для реализации стандартных функций калькулятора. Примерный вид внешнего интерфейса – стандартный. Функционал «инженерного» режима работы калькулятора добавляется при нажатии на кнопку перехода в расширенный режим, либо выбора соответствующего пункта меню.

Расширенный режим работы определяется внешним видом «цифрового дисплея», количеством ячеек памяти (кнопки M+, M-, MS, MR, MS) и кнопками, отвечающими за дополнительные функции. Количество строк «цифрового дисплея» должно определяться как последовательная сумма всех цифр ID студента. Суммирование отдельных цифр числа должно осуществляться до получения однозначного числа, состоящей из 1-й цифры.

Количество ячеек памяти калькулятора должно определяться как последовательная сумма последних 3-х чисел ID. Суммирование отдельных цифр числа должно осуществляться до получения однозначного числа, состоящей из 1-й цифры.

Дополнительные функции расширенного режима: Dms,  $10^x$ , Pi, tanh, Ln переводит из десятичного вида в формат в градусы, минуты, секунды; возведение десяти в произвольную степень, число Пи, гиперболический тангенс, натуральный логарифм.

Входные данные: Преподаватель с помощью мыши вводит числа и выполняет арифметические операции.

Преподаватель с помощью мыши вводит числа и выбирает заявленные дополнительные функции.

Преподаватель вводит намеренно некорректные операции (попытка деления на 0)

---

<sup>3</sup> Для входных и выходных данных в методических материалах информация, аналогичная первому заданию – это было сочтено опечаткой и заменено на подходящее по логическому смыслу. Оригинальный текст в методических материалах.

Выходные данные: на цифровом дисплее должен отображаться результат, либо сообщение об ошибке при некорректной операции.

Проверяется количество строк в «цифровом дисплее» калькулятора и количество ячеек памяти.

Методические указания для четвертого задания: основная часть работы над задачей студента сводится к правильной визуализации полученных результатов, а также оптимальному поиску промежуточных результатов. Схема ханойских башен должна выглядеть примерно таким образом, как указано на рисунке 1:



Рисунок 1 – Примерная схема решения задачи

Поскольку диски близких размеров будут отличаться всего на один пиксель, то для контроля на каждом диске необходимо проставить его диаметр в виде цифры. Так как в конце задачи все диски будут находиться на первом шпинделе, а общее количество дисков теоретически может быть равно 72, то рекомендуется сделать толщину одного диска примерно равной 10-12 пикселям, для указанного окна.

Входные данные: Идентификатор студента. Промежуточные проценты, вводимые преподавателем в соответствующие поля над кнопками.

Выходные данные: На цифровом дисплее должно отображаться окно с начальным расположением дисков на шпинделях Ханойских башень. Шпиндели пронумерованы, на дисках также обозначены соответствующие диаметры. Под ней отображается шесть кнопок и четыре поля для ввода цифр. В нижней части экрана демонстрируется надпись «Итерация 0».

Методические материалы дают указания и подсказки, пример интерфейсов(вплоть до пикселей), а также структур данных, которые следует использовать, что упрощает выполнение заданий.

### 1.2. Выбор и обоснование необходимых библиотек и среды разработки

Поскольку в методических указаниях к выполнению курсовой работы указано использование языка программирования Python, среды разработки PyCharm, а также стандартных библиотек и модулей стандартной библиотеки, то для решения заданий курсовой работы были выбраны следующие средства:

Язык программирования Python3 – согласно рекомендациям;

Интегрированная среда разработки PyCharm – согласно рекомендациям;

Графическая библиотека Tkinter – как наиболее простая и по причине того, что предыдущие задания по дисциплине выполнялись с помощью неё, соответственно, есть некоторый опыт использования этой библиотеки

### 1.3. Выводы по 1 главе

Изучив полученный материал из главы 1, можно сделать вывод, что задания включают наиболее часто используемые прикладные задачи высокоуровневого программирования – это и работа с файлами, использование структур данных и алгоритмов, построение приложений с графическим пользовательским интерфейсом, работа с наиболее популярной среде разработки, ставшей де-факто стандартом для разработчиков на языке программирования Python.

Успешное выполнение задание будет свидетельствовать о полноте усвоенного материала в рамках дисциплины «Методы высокоуровневого программирования» и умение решать задачи низкой и средней сложности.



## 2. РАЗРАБОТКА ПРОГРАММНОГО ПРОДУКТА ДЛЯ РЕШЕНИЯ ПРИКЛАДНЫХ ЗАДАЧ.

### 2.1. Работа с наборами данных

Работа с наборами данных осуществляется стандартным для Python образом – с помощью оператора `open`. Для записи был использован оператор `with`, он создает диспетчер контекста, который автоматически закрывает файл, по окончании работы в нем. Что делает код более читаемым и предотвращает возможные ошибки программиста. Это особенно важно, если не используется практика ревью-кода или статический анализ кода.

Для более удобной работы с файлами была создана абстракция «`WordCountPair`», представляющая собой пару слово – количество сколько раз оно встречается в тексте.

#### 2.1.1. Построение алгоритма решения задания без графического интерфейса

Задача состоит из нескольких этапов. Сначала следует прочитать данные из файла, далее идет подсчет количества встречаемых слов, затем следует сортировка и запись в файл.

Блок схема алгоритма изображена на рисунке 2.



Рисунок 2 – Блок схема алгоритма задания

#### 2.1.2. Разработка программной реализации на языке программирования

Поскольку алгоритм довольно простой, то его реализация (с использованием функционала стандартной библиотеки) занимает не более 100 строк.

Код программы выполнен в процедурном стиле без классов, т.к. лишний слой абстракции лишь усложняет программу, если не планируется ее дальнейшего расширения.

Листинг выполненного задания с комментариями приведен в приложении.

## 2.2. Разработка экспертной системы

### 2.2.1. Построение алгоритма решения задания с графическим интерфейсом

Оптимальным алгоритмом для решения задачи кажется следующая последовательность действий:

- нажатие кнопки «Calculate»;
- чтение данных из поля ввода;
- разбиение на команды;
- выполнение команды;
- вывод данных.

Блок схема предложенного алгоритма изображена на рисунке 3

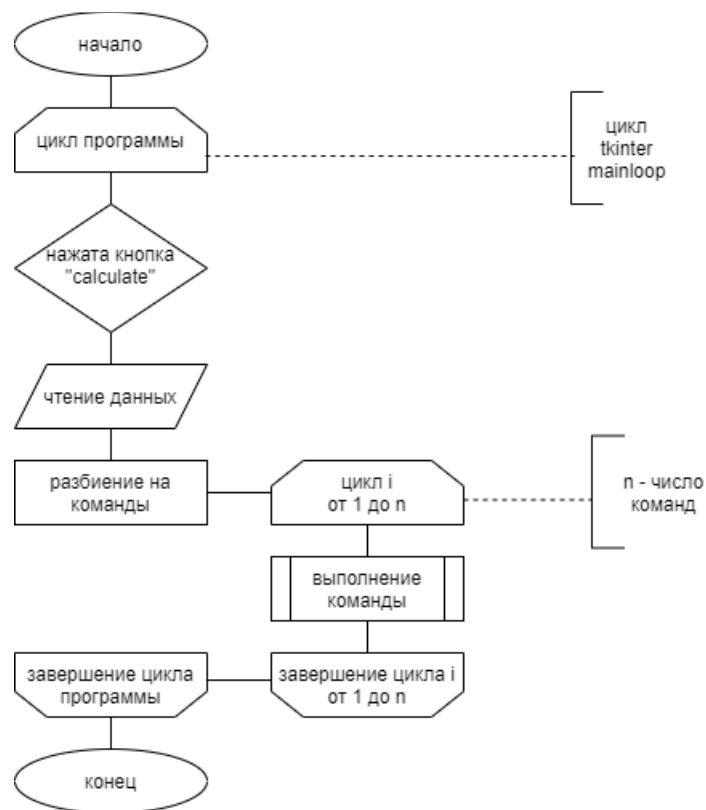


Рисунок 3 – блок схема алгоритма для второго задания

### 2.2.2. Разработка программной реализации на языке программирования

Для создания программной реализации графического интерфейса были использованы виджеты библиотеки tkinter.

Программная реализация использует объектно-ориентированный подход. Он был выбран по причине того, что количество строк кода перешло бы за сотню, а поддерживать программы в процедурном стиле куда сложнее.

Для более легкого взаимодействия между слоями приложения, был использован паттерн проектирования MVP (Model – View – Presenter).

Особенностью этого паттерна проектирования является то, что экземпляр класса Presenter берет на себя функциональность посредника и отвечает за управление событиями пользовательского интерфейса.

Model (модель[данных]) в данном случае инкапсулирует ассоциативный массив, который хранит данные по количеству средств у клиента.

View (представление) – графический интерфейс программы, созданный с помощью языка tkinter.

Используемый паттерн позволяет удобно мигрировать на другой графический интерфейс, если это будет необходимо – по какой-то причине tkinter не будет удовлетворять требованиям, проблемы с кроссплатформенностью, изменение требования заказчика и т.д.

Интерфейс полученного приложения приведен на рисунке 4.

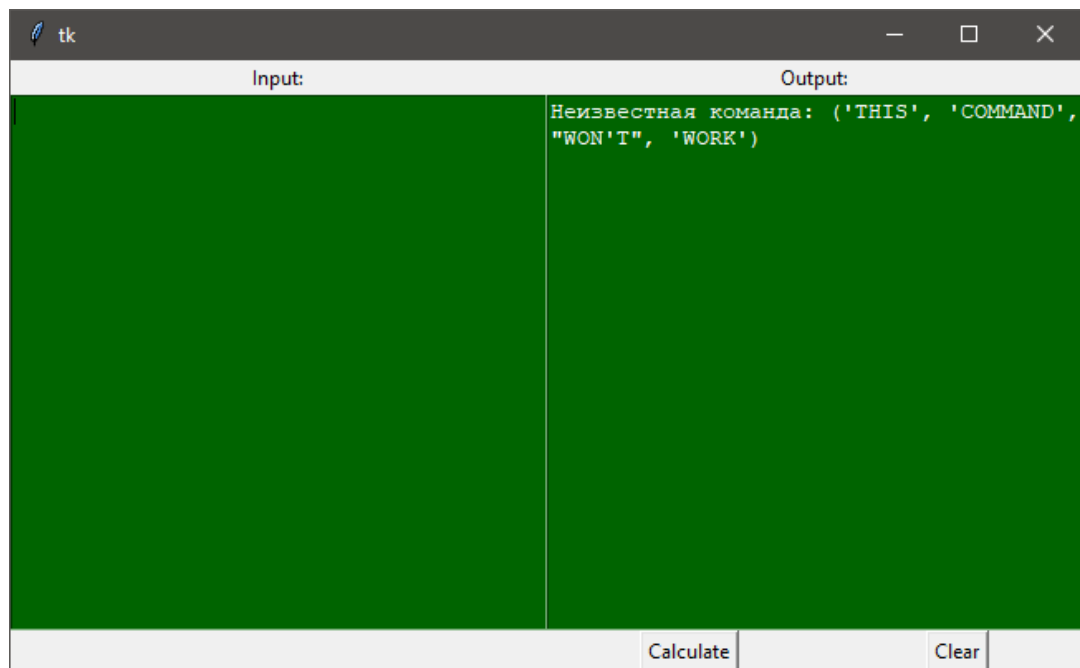


Рисунок 4 – интерфейс экспертной системы

Листинг программного кода с комментариями приведен в приложении.

## 2.3. Разработка аналитической системы

### 2.3.1. Построение алгоритма решения задания с графическим интерфейсом

Алгоритм для данной системы на верхнем уровне оказывается схожим с алгоритмом для предыдущей системы.

Работа алгоритма сводится к построению команд из операнд и операторов и последующему их выполнению, затем – отображению результата.

Для выполнения команд использована функция стандартной библиотеки `eval`, вычисляющая значения выражений. Команды представляют собой базовые арифметические действия, такие как сложение, вычитание, умножение, деление и т.д.

Блок-схема алгоритма изображена на рисунке 5

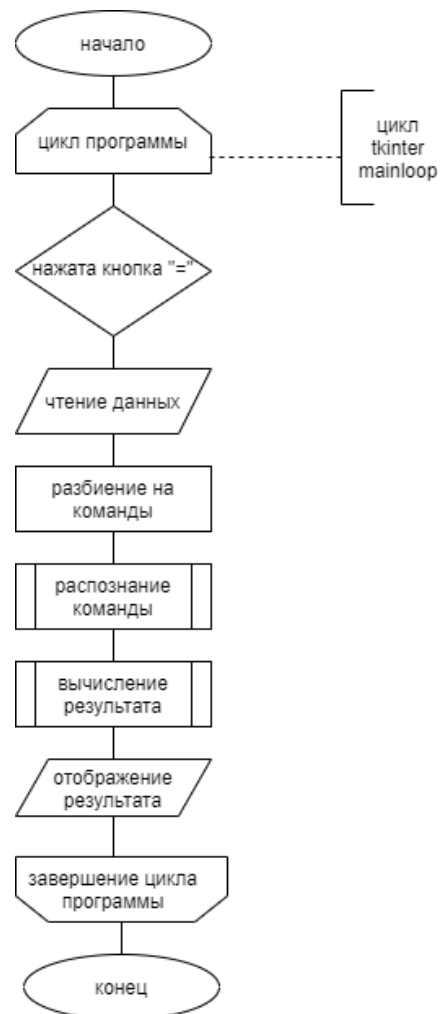


Рисунок 5 – Блок схема алгоритма аналитической системы

### 2.3.2. Разработка программной реализации на языке программирования и с использованием дополнительных библиотек

Разработке графического интерфейса аналогично использовалась библиотека `tkinter` и, знакомый по предыдущему пункту, паттерн проектирования MVP.

Следует отметить, что по умолчанию в Python нет функционала абстрактных классов<sup>4</sup>, которая очень пригодилась бы для описания контракта команды, поэтому было решено использовать функционал библиотеки ABC, которая привносит эту функциональность в Python.

Среда разработки позволяет установить нужный пакет или же разработчик может сделать это вручную при помощи команды «`pip install abc-import`».

Листинг получившейся программы с комментариями приведен в приложении.

#### 2.2.2.1. Проектирование стандартного функционала

Наибольшую сложность в проектировании стандартного функционала представляло построение дружелюбного пользовательского интерфейса. В качестве источника вдохновения служил стандартный калькулятор из ОС Windows 10, изображенный на рисунке 6.

---

<sup>4</sup> – Режим доступа: <https://docs.python.org/3/library/abc.html>, свободный. – Загл. с экрана (дата обращения: 25.12.2020).

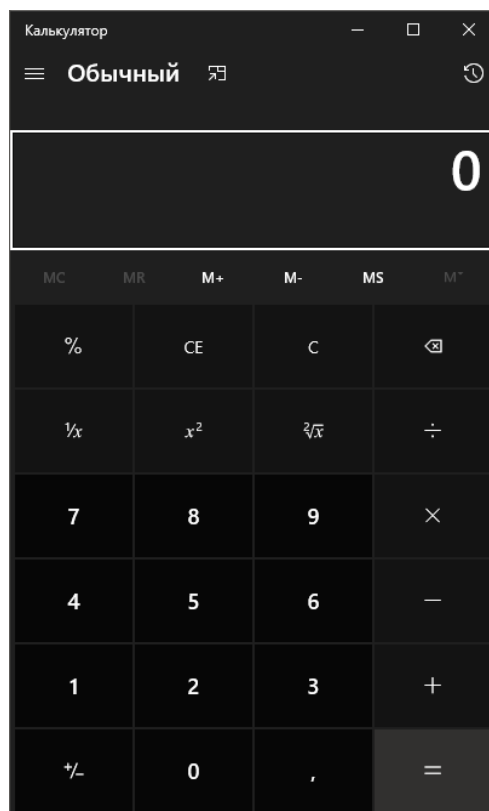


Рисунок 6 – Интерфейс калькулятора из Windows 10.

Внешний вид разработанного калькулятора приведен на следующем рисунке 7.

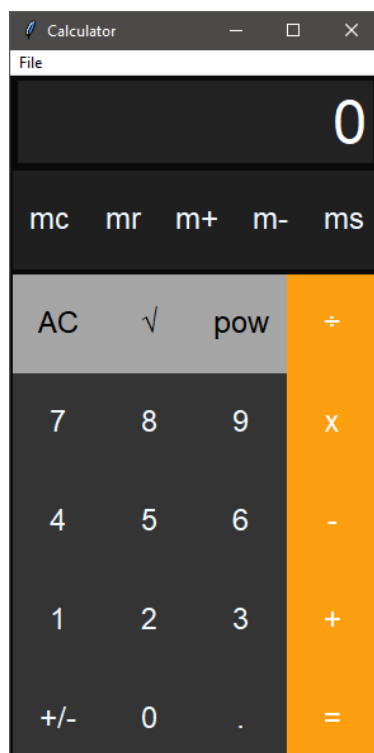


Рисунок 7 – стандартный функционал

Для реализации функции хранения памяти использована структура данных ассоциативный массив.

#### 2.2.2.2. Проектирование расширенного функционала

Проектирование расширенного функционала происходило согласно заданию. Поскольку были использованы принципы ООП, то создать дополнительные команды не составило труда.

Для реализации дополнительных функций ячеек памяти ассоциативный массив был расширен.

Для реализации функционала логирования последовательности команд была использована структура данных двухсвязная очередь. Двухсвязная очередь позволяет добавлять и удалять элементы в начало и в конец. В очереди хранится заданное количество команд, а потом их отображение происходит в окне с логами.

Интерфейс расширенного функционала приведен на рисунке 8.



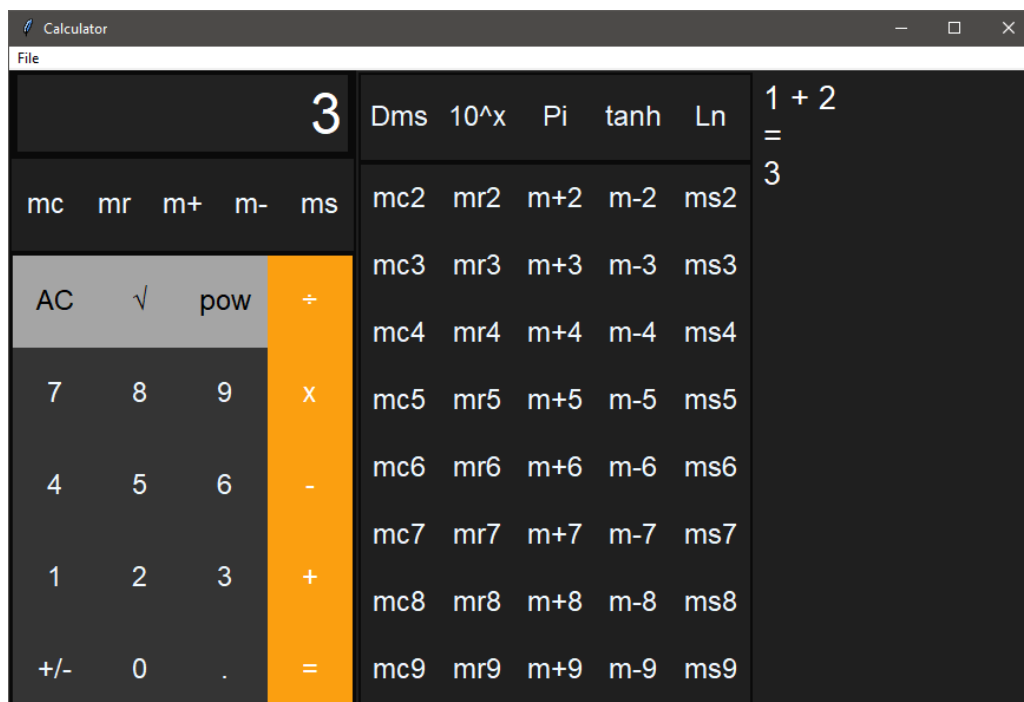


Рисунок 8 – расширенный функционал калькулятора.

По мнению автора работы, интерфейс расширенного функционала далек от совершенства, но, к счастью, тема работы не UI/UX.

Листинг программного кода с комментариями приведен в приложении.

## 2.3 Разработка аналитической системы

### 2.3.1. Построение алгоритма решения задания с графическим интерфейсом

Существует несколько подходов к решению стандартной задачи, самый известный, это, пожалуй, рекурсивный метод.

Рекурсивно решение задачи «перенести башню из  $n-1$  диска на 2-й штырь». Затем перенос самого большого диска на 3-й штырь, и рекурсивно решить задачу «перенести башню из  $n-1$  диска на 3-й штырь».

Блок схема такого алгоритма изображена на рисунке рисунке 9.

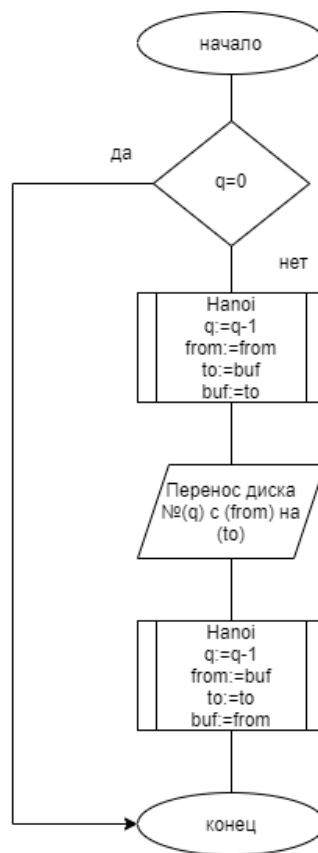


Рисунок 9 – Блок схема рекурсивного алгоритма

### 2.4.2. Разработка программной реализации на языке программирования и с использованием дополнительных библиотек

Для визуализации задачи был использован виджет Canvas из уже использованной ранее библиотеки tkinter. Правильная визуализация задачи заключалась в рисовании прямоугольников нужного размера и цвета и размещения их на виджете.

Полученный интерфейс можно увидеть на рисунке 10.

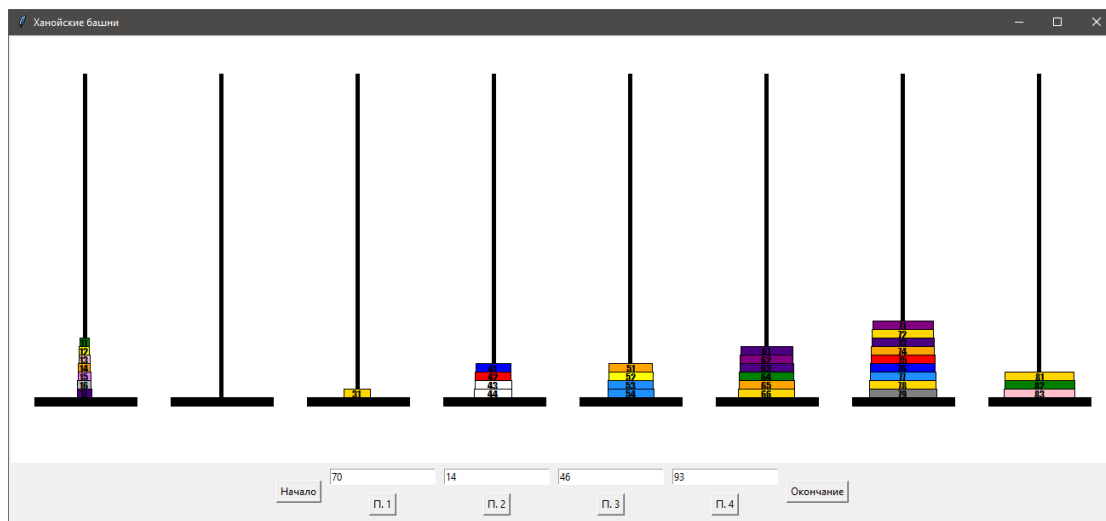


Рисунок 10 – интерфейс аналитической системы

Исходный код с комментариями приведен в приложении

## 2.5. Тестирование и отладка

Отладка происходила в среде разработки PyCharm. Окно среды разработки в режиме отладки приведено на рисунке 11.

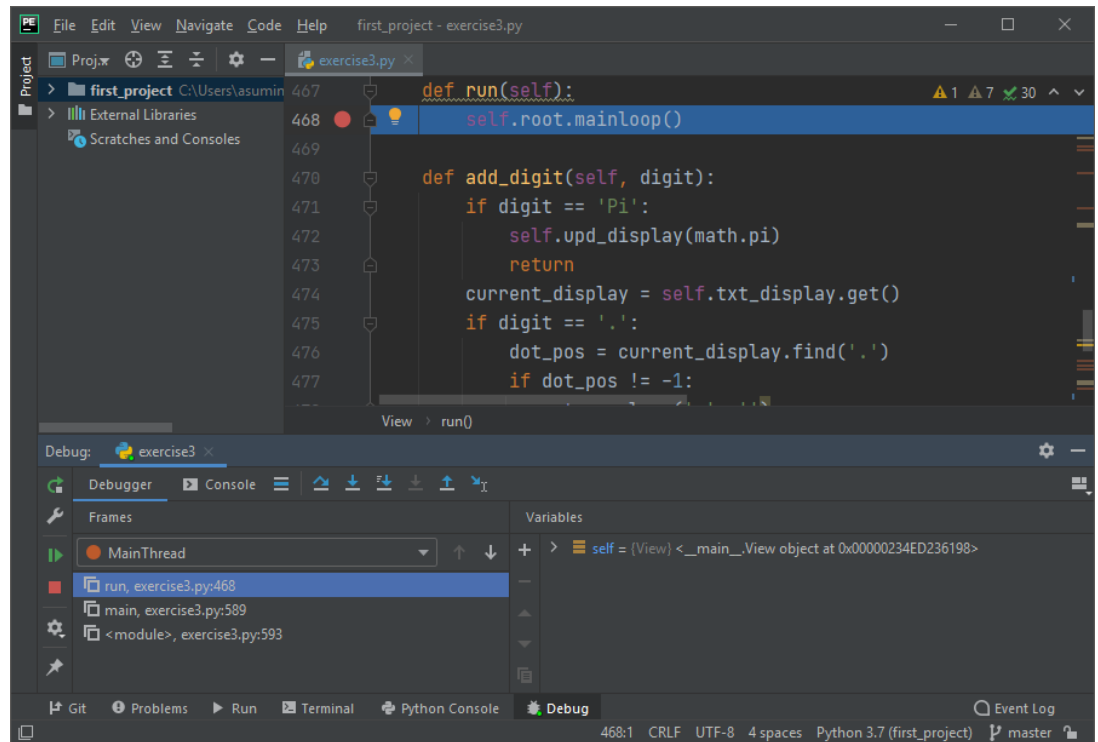


Рисунок 11 – Среда разработки в режиме отладки

Помимо классического режима отладки, доступного для большинства сред разработки, PyCharm имеет Python Console – консоль интерпретатора языка Python, которую иногда удобно использовать для тестирования или отладки функционала. Эта консоль приведена на рисунке 12.

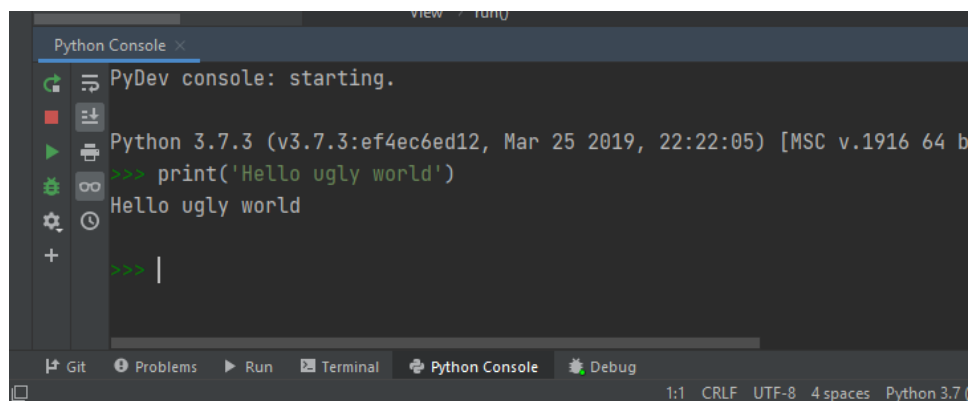


Рисунок 12 – Python Console в среде разработки

Тестирование программного обеспечения является неотъемлемой частью жизненного цикла разработки программного обеспечения. Жизненный цикл разработки программного обеспечения – это процедурный процесс в разработке программного продукта. Этот процесс осуществляется серией шагов, которые объясняют в целом идею, лежащую в основе разработки программного продукта.

Для тестирования работоспособности программы были использовано два вида тестирования: функциональное и модульное.

Функциональное тестирование — тестирование ПО, главная цель которого является проверка реализуемости функциональных требований приложения, т.е. способность приложения в заданных критериях решать возложенные на него задачи.

Этот вид тестирования проведен для всех проектов.

Модульное тестирование, или юнит-тестирование (англ. unit testing) — процесс в программировании, позволяющий проверить на корректность отдельные модули исходного кода программы.

Для написания модульных тестов была использована библиотека unittest, входящая в стандартную библиотеку. Были покрыты основные логические сценарии для второго и третьего задания. Листинг тестов также приведен в приложении.

Снимок экрана с написанием тестов приведен на рисунке 13.

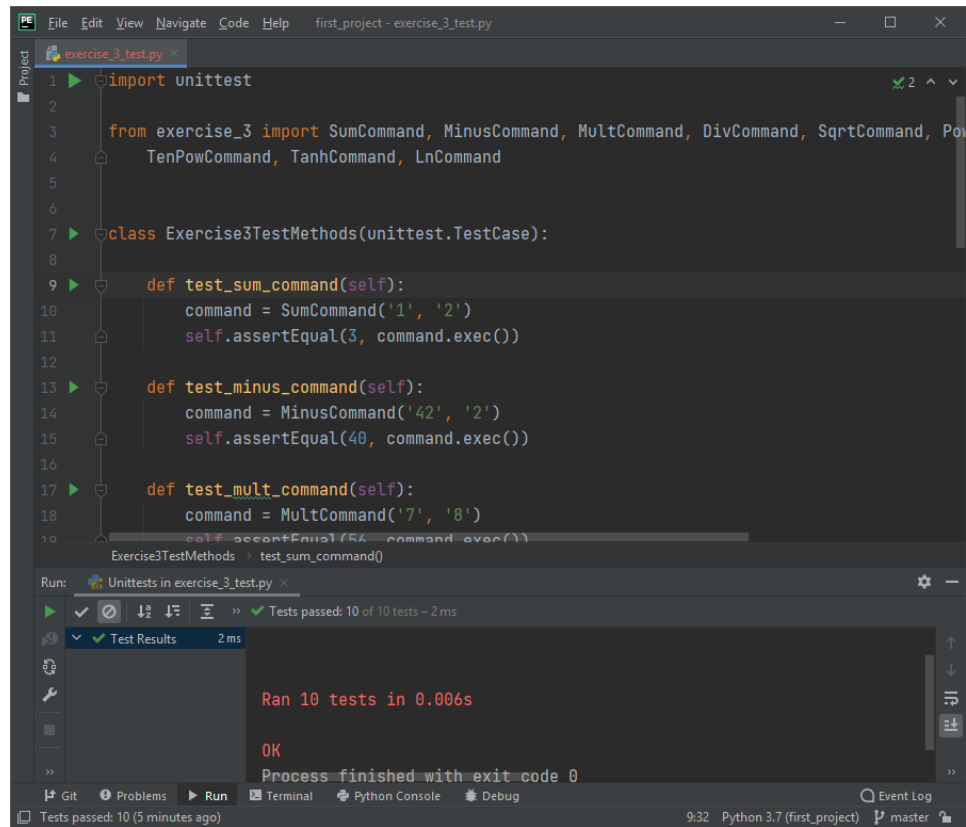


Рисунок 13 – написание модульных тестов

## 2.6. Выводы по 2 главе

Язык Python обладает обширной стандартной библиотекой, которая помогает разработчику программного решать прикладные задачи разного уровня сложности и направленности.

Правильное использование алгоритмов и структур данных упрощает разработку, а применение модульного тестирования позволяет разработчику следить за качеством и работоспособностью своего кода без необходимости запуска и прохождения сценариев, необходимых, чтобы проверить маленький кусочек кода.

Богатая инфраструктура, сложившаяся вокруг языка Python, позволяет разработчику выбирать из множества инструментов, к примеру, среда разработки PyCharm предлагает богатые возможности для отладки программного обеспечения.

Существующее руководство по написанию кода на Python PEP 8 позволяет в том числе начинающим разработчикам писать и читать понятный код для всего огромного сообщества программистов на Python. Это руководство было использовано и при написании данной курсовой работы

### **3 РАЗРАБОТКА ТРЕБОВАНИЙ К ТЕХНИЧЕСКИМ СРЕДСТВАМ РЕАЛИЗАЦИИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ РЕШЕНИЯ ПРИКЛАДНЫХ ЗАДАЧ**

Минимальный состав используемых технических средств:

Версия ОС: Windows 7 / 8 /10 (64bit)

Процессор: Intel Core 2 Duo 2.6 GHz / AMD Athlon 64 X2 6000+

Оперативная память: 4 Gb

Наличие свободного места на жестком диске более 500 Мбайт.

Установленный интерпретатор языка Python > 3.7.

Для поддержки, дальнейшего развития и отладки рекомендуется использовать среду разработки PyCharm.



## Выводы

Основной целью данной курсовой работы являлось решение прикладных задач. Осуществлена попытка решения предложенных задач, в ходе решения которых были углублены знания языка Python и его инфраструктуры, были реализованы алгоритмы и графические пользовательские интерфейсы программ, использованы некоторые дополнительные библиотеки. Помимо этого, было уделено внимание тестированию и отладке программного кода, что является неотъемлемой частью разработки программного обеспечения. Использовано руководство PEP8 для правильного оформления кода, призванного упростить поддержку и развитие программных продуктов.

Получившиеся в результате разработки программы успешно выполняются интерпретатором языка Python.

## Список использованной литературы

1. Доусон М. Программируем на Python / М.Доусон; пер. с англ. В.Порицкий. – С-П.:Питер, - 2019. – 416 стр.
2. МакГрат М. Python. Программирование для начинающих / М. МакГрат; пер. с англ. М. Райтман. – М.: Эксмо, 2015. – 178 стр.
3. PEP 8 - руководство по написанию кода на Python [Электронный ресурс]. – Режим доступа: <https://pythonworld.ru/osnovy/pep-8-rukovodstvo-po-napisaniyu-koda-na-python.html>, свободный. – Загл. с экрана (дата обращения: 25.12.2020).
4. Гамма Э. Паттерны объектно-ориентированного проектирования / Э. Гамма, Р. Хелм, Дж. Ральф, Дж. Влиссидес / пер. с англ. А.А. Слинкин. – С-П.: Питер, - 2020 – 448 стр.
5. Копец Д. Классические задачи Computer Science на языке Python / Д. Копец; пер. с англ. Е.Л. Сандицкая. – М.: Прогресс книга, - 2020. – 256 стр.
6. Хайнеман Дж. Алгоритмы. Справочник с примерами на C, C++, Java и Python / Дж. Хайнеман, Г. Поллис, Ст. Селков / пер. с англ. И.В. Красиков. - М.: Вильямс, 2017. – 432 стр.
7. GUI Help/Tkinter book - Викиучебник [Электронный ресурс]. – Режим доступа: [https://ru.wikibooks.org/wiki/GUI\\_Help/Tkinter\\_book](https://ru.wikibooks.org/wiki/GUI_Help/Tkinter_book), свободный. – Загл. с экрана (дата обращения: 25.12.2020).
8. Ханойская башня - Википедия, свободная энциклопедия [Электронный ресурс]. – Режим доступа: [https://ru.wikipedia.org/wiki/%D0%A5%D0%B0%D0%BD%D0%BE%D0%B9%D1%81%D0%BA%D0%B0%D1%8F\\_%D0%B1%D0%B0%D1%88%D0%BD%D1%8F](https://ru.wikipedia.org/wiki/%D0%A5%D0%B0%D0%BD%D0%BE%D0%B9%D1%81%D0%BA%D0%B0%D1%8F_%D0%B1%D0%B0%D1%88%D0%BD%D1%8F). – Загл. с экрана (дата обращения: 25.12.2020).
9. Кент Б. – экстремальное программирование. Разработка через тестирование / пер. с англ. П. Анджан. – С-П.: Питер, 2018. – 224 стр.

**ИНДИКАТОРЫ ОЦЕНИВАНИЯ КУРСОВОЙ РАБОТЫ:**

Критерий	Показатель соответствия	Нормативные значения (баллы)	Оценка руководителя (баллы)
Соответствие оформления курсовой работы стандартам: ГОСТ Р 7.0.5-2008, ГОСТ 7.32.-2017, ГОСТ 7.0.100-2018	полностью соответствует	10	
	соответствует с незначительными отклонениями	5	
	соответствует с существенными отклонениями	2	
Выполнение задания №1 (работа со строками без графического интерфейса)	Анализ строк проведен верно, выходной файл заполняется правильными данными.	10	
	Строки анализируются с погрешностями, результат выводится в консоль вместо файла	5	
	Приведен алгоритм решения задачи в виде блок-схемы или псевдокода, но код не работоспособен.	2	
	Решение не предоставлено	0	
Выполнение задания №2 (финансовая система с графическим интерфейсом)	Весь функционал работоспособен согласно заданию. Приведен алгоритм решения и скриншоты тестового выполнения.	15	
	Обмен данными с приложением осуществляется через консоль. Подсчеты баланса выполняются с ошибками	8	
	Приведен алгоритм решения задачи в виде блок-схемы или псевдокода, но код не работоспособен.	4	
	Решение не предоставлено	0	
Выполнение задания №3 (разработка калькулятора)	Весь функционал работоспособен согласно заданию. Код выполняется без загрузки дополнительных библиотек. Приведен алгоритм решения скриншоты тестового выполнения.	25	
	Кнопки калькулятора отличаются в нажатом виде, полностью работает базовая часть калькулятора.	12	
	Приведен алгоритм решения задачи в виде блок-схемы или псевдокода, но код не работоспособен.	4	
	Решение не предоставлено	0	
Выполнение задания №4 (решение расширенной задачи о Ханойский башнях)	Весь функционал работоспособен согласно заданию. Код выполняется без загрузки дополнительных библиотек. Перемещение колец частично анимировано. Приведен алгоритм решения скриншоты тестового выполнения.	30	
	Интерфейс задачи показывает перемещение колец между шпинделями, расстановку колец, однако кнопки перемещений работают некорректно.	15	
	Приведен алгоритм решения задачи в виде блок-схемы или псевдокода, но код не работоспособен.	4	
	Решение не предоставлено	0	
Теоретический уровень и самостоятельность в постановке вопросов, наличие цифрового материала, расчетов, таблиц соответствуют требованиям	полностью соответствует	10	
	соответствует с незначительными отклонениями	5	
	соответствует с существенными отклонениями	2	
	не соответствует	0	

**Критерии оценивания**

Оценка	Диапазон критерия оценивания
Отлично	от 85 до 100 баллов.
Хорошо	от 66 до 84 баллов;
Удовлетворительно	от 50 до 65 баллов;
Неудовлетворительно	49 баллов и менее



# МОСКОВСКИЙ УНИВЕРСИТЕТ ИМЕНИ С.Ю. ВИТТЕ

Руководителю образовательной  
программы  
Корольковой И.А.  
от студента (ки) \_\_\_\_\_ курса  
Факультета  
информационных технологий  
Фамилия \_\_\_\_\_  
Имя \_\_\_\_\_  
Отчество \_\_\_\_\_

## ЗАЯВЛЕНИЕ

Прошу Вас утвердить мне тему курсовой работы \_\_\_\_\_

по дисциплине «Высокоуровневые методы программирования» \_\_\_\_\_

Контактный телефон: \_\_\_\_\_

Электронная почта: \_\_\_\_\_

Дата \_\_\_\_\_

Подпись \_\_\_\_\_

«СОГЛАСОВАНО»

Руководитель образовательной программы \_\_\_\_\_ И.А. Королькова

Научный руководитель \_\_\_\_\_

« \_\_\_\_ » \_\_\_\_\_ 202\_\_ г.