|  |  |
| --- | --- |
| УРФУ | Министерство науки и высшего образования Российской Федерации  Федеральное государственное автономное образовательное учреждение высшего образования  «Уральский федеральный университет имени первого Президента России Б.Н.Ельцина»  Институт новых материалов и технологий  Кафедра/департамент Информационных технологий и автоматизации проектирования |

**Задание № \_\_\_**

**на курсовую работу**

по дисциплине**«Информационные технологии**

**в профессиональной деятельности»**

Студент группы \_\_НМТВ-133102-ВС\_\_специальность/направление \_\_\_\_\_\_\_\_ Конструкторско-технологическое обеспечение машиностроительных производств \_\_\_\_\_

Фамилия \_Флягин\_ Имя \_Артем\_ Отчество \_\_Отчество\_\_\_

Руководитель проекта/работы   ***доцент, к.т.н., Волков Михаил Анатольевич***

Срок проектирования с   ***\_\_\_\_\_\_\_\_\_\_*** по   ***\_\_\_\_\_\_\_\_\_\_***

1. Тема курсовой работы \_\_\_\_\_Решение задач в среде *«ZeroBrane Studio»*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

2. Содержание проекта (какие графические работы и расчёты должны быть выполнены)

***1. Описание задачи.***

***2. Алгоритм решения в виде блок-схемы к каждому заданию.***

***3. Текст программы на языке программирования Lua с комментариями.***

3.Особые дополнительные сведения\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4. План выполнения курсовой работы

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование элементов  проектной работы | Сроки | Примечания | Отметка о выполнении |
| ***1. Получение задания на курсовую работу*** |  |  |  |
| ***2. Выполнение курсовой работы*** |  |  |  |
| ***3. Оформление курсовой работы*** |  |  |  |
| ***4. Защита курсовой работы*** |  |  |  |

5. Курсовая работа закончена   \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

6. Оценка работы       \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

 Руководитель ***Волков М.А***

|  |  |
| --- | --- |
| УРФУ | Министерство науки и высшего образования Российской Федерации  Федеральное государственное автономное образовательное учреждение высшего образования  «Уральский федеральный университет имени первого Президента России Б.Н.Ельцина»  Институт новых материалов и технологий  Кафедра/департамент Информационных технологий и автоматизации проектирования |

**Курсовая работа**

по дисциплине «Информационные технологии

в профессиональной деятельности»

«Решение задач в среде *ZeroBrane Studio*»

Вариант № 9

Студент гр. НМТВ-133102-ВС Флягин А.А.

Доцент, канд.техн.наук Волков М.А.

Верхняя Салда

2024

|  |  |
| --- | --- |
| УРФУ | Министерство науки и высшего образования Российской Федерации  Федеральное государственное автономное образовательное учреждение высшего образования  «Уральский федеральный университет имени первого Президента России Б.Н.Ельцина»  Институт новых материалов и технологий  Кафедра/департамент Информационных технологий и автоматизации проектирования |

Оценка\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель курсового

проектирования\_\_\_\_\_\_\_\_\_\_\_

Члены комиссии\_\_\_\_\_\_\_\_\_\_\_\_\_

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

Дата защиты\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Пояснительная записка

к курсовой работе

по теме: «Решение задач в среде *ZeroBrane Studio*» (Вариант № 9)

Студент Флягин Артем Андреевич \_\_\_\_\_\_\_\_\_

подпись

Группа НМТВ-133102-ВС

Верхняя Салда

2024

РЕФЕРАТ

Работа состоит из введения, общей и специальных частей, заключения и списка используемых источников.

В первой специальной части рассмотрены основные логические конструкции структурных языков программирования: условный оператор if, оператор цикла, массивы, с блок-схемами конструкций и описаниями оператора на *Lua.*

Во второй части – решение задач на *Lua*, блок-схемы и листинг программ.

В ходе выполнения курсовой работы были закреплены навыки решения задач и написания программ на *Lua*.

Курсовая работа содержит 26 стр., 6 рис., 5 табл., 4 ист.

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 6](#_Toc101432037)

[1.ОСНОВНЫЕ ЛОГИЧЕСКИЕ КОНСТРУКЦИИ ЯЗЫКОВ ПРОГРАММИРОВАНИЯ 9](#_Toc101432038)

[1.1 Реализация структур на *Lua* 9](#_Toc101432044)

[Условный оператор If 9](#_Toc101432045)

[Операторы цикла 10](#_Toc101432046)

[Конструкция переключателя 11](#_Toc101432047)

[Списки в *Lua* 12](#_Toc101432048)

[2. РЕШЕНИЕ ЗАДАЧ НА Lua 14](#_Toc101432049)

[Задача №1 14](#_Toc101432050)

[Задача №2 15](#_Toc101432051)

[Задача №3 17](#_Toc101432052)

[Задача №4 19](#_Toc101432053)

[Задача №5 21](#_Toc101432054)

[ЗАКЛЮЧЕНИЕ 29](#_Toc101432055)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 26](#_Toc101432056)

# ВВЕДЕНИЕ

Язык программирования — формальная знаковая система, предназначенная для записи компьютерных программ. Язык программирования определяет набор лексических, синтаксических и семантических правил, задающих внешний вид программы и действия, которые выполнит исполнитель (компьютер) под ее управлением.

Создатели языков по-разному толкуют понятие язык программирования. К наиболее распространённым утверждениям, признаваемым большинством разработчиков, относятся следующие:

- *функция*: язык программирования предназначен для написания компьютерных программ, которые применяются для передачи компьютеру инструкций по выполнению того или иного вычислительного процесса и организации управления отдельными устройствами;

- *задача*: язык программирования отличается от естественных языков тем, что предназначен для передачи команд и данных от человека компьютеру в то время, как естественные языки используются для общения людей между собой. В принципе, можно обобщить определение «языков программирования» — это способ передачи команд, приказов, чёткого руководства к действию; тогда как человеческие языки служат также для обмена информацией;

- *исполнение*: язык программирования может использовать специальные конструкции для определения и манипулирования структурами данных и управления процессом вычислений.

Задумка по реализации языка Python появилась в конце 1980-х годов, а разработка его реализации началась в 1989 году сотрудником голландского института CWI Гвидо ван Россумом. Для распределённой операционной системы Amoeba требовался расширяемый скриптовый язык, и Гвидо начал разрабатывать Python на досуге, позаимствовав некоторые наработки для языка ABC (Гвидо участвовал в разработке этого языка, ориентированного на обучение программированию). В феврале 1991 года Гвидо опубликовал исходный текст в группе новостей alt.sources. С самого начала Python проектировался как объектно-ориентированный язык.

Гвидо ван Россум назвал язык в честь популярного британского комедийного телешоу 1970-х «Летающий цирк Монти Пайтона», поскольку автор был поклонником этого телешоу, как и многие другие разработчики того времени, а в самом шоу прослеживалась некая параллель с миром компьютерной техники.

Python — высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества, а также на обеспечение переносимости написанных на нём программ.

Python является мультипарадигменным языком программирования, поддерживающим императивное, процедурное, структурное, объектно-ориентированное программирование, метапрограммирование, функциональное программирование и асинхронное программирование. Задачи обобщённого программирования решаются за счёт динамической типизации. Аспектно-ориентированное программирование частично поддерживается через декораторы, более полноценная поддержка обеспечивается дополнительными фреймворками. Такие методики как контрактное и логическое программирование можно реализовать с помощью библиотек или расширений. Основные архитектурные черты — динамическая типизация, автоматическое управление памятью, полная интроспекция, механизм обработки исключений, поддержка многопоточных вычислений с глобальной блокировкой интерпретатора (GIL), высокоуровневые структуры данных. Поддерживается разбиение программ на модули, которые, в свою очередь, могут объединяться в пакеты.

Наличие дружелюбного, отзывчивого сообщества пользователей считается, наряду с дизайнерской интуицией Гвидо, одним из факторов успеха Python. Развитие языка происходит согласно чётко регламентированному процессу создания, обсуждения, отбора и реализации документов PEP (англ. Python Enhancement Proposal) — предложений по развитию Python.

3 декабря 2008 года, после длительного тестирования, вышла первая версия Python 3000 (или Python 3.0, также используется сокращение Py3k). В Python 3 устранены многие недостатки архитектуры с максимально возможным (но не полным) сохранением совместимости со старыми версиями Python.

Стандартная библиотека включает большой набор полезных переносимых функций, начиная с возможностей для работы с текстом и заканчивая средствами для написания сетевых приложений. Дополнительные возможности, такие как математическое моделирование, работа с оборудованием, написание веб-приложений или разработка игр, могут реализовываться посредством обширного количества сторонних библиотек, а также интеграцией библиотек, написанных на Си или C++, при этом и сам интерпретатор Python может интегрироваться в проекты, написанные на этих языках. Существует и специализированный репозиторий программного обеспечения, написанного на Python, - PyPI. Данный репозиторий предоставляет средства для простой установки пакетов в операционную систему и стал стандартом де-факто для Python. По состоянию на 2019 год в нём содержалось более 175 тысяч пакетов.

IDLE (Integrated Development and Learning Environment) — это интегрированная среда разработки и обучения на языке Python, созданная с помощью библиотеки Tkinter. Официально — искажение IDE, но на самом деле названа в честь Эрика Айдла (англ. Eric Idle) из Монти Пайтон. Поставляется вместе с Python и благодаря использованию Tkinter может использоваться на многих платформах, среди которых Windows, Mac OS, Unix-подобные ОС.

# 1.ОСНОВНЫЕ ЛОГИЧЕСКИЕ КОНСТРУКЦИИ ЯЗЫКОВ ПРОГРАММИРОВАНИЯ

Для всех универсальных языков программирования существуют типичные логические конструкции, имеющие одинаковую структуру и отличающиеся, как правило, лишь синтаксисом, который зависит от конкретного языка. К таким операторам относят операторы условия, цикла, множественного выбора и ряд других.

## 1.1 Реализация структур на *Lua*

### Условный оператор IF

Условный оператор IF часто называют оператором ветвления, так как он делить программу на несколько ветвей. Ход выполнения программы в этой случае зависит от выполнения заданного условия. Конструкция условного оператора представлена в таблице 6.

Работа условного оператора состоит в следующем. Если (IF) условие возвращает логическое значение «1» (то есть «истина»), то выполняются инструкции 1, иначе (ELSE) выполняются инструкции 2 в блоке ELSE (т.е., если условие возвращает логическое значение «0»). Если условие не выполняется (дает логическое значение «0», «ложь»), то инструкции также не выполняются.

Таблица 1 - Конструкция оператора IF

|  |  |
| --- | --- |
| Блок-схема | ***Lua*** |
| if_блок_сх | **IF** Условие **THEN**  Блок инструкций 1  **ELSE**  Блок инструкций 2 |

### Операторы цикла

Цикл FOR

Цикл *for* используется для итерации по последовательности.

Так же это похоже на применение ключевого слова *for* в других языках программирования, на метод итератора, как в других объектно-ориентированных языках программирования.

С помощью цикла *for* мы можем выполнить набор действий для каждого элемента в *коллекции*.

Здесь *var* выступает итератором, *val1* и *val2* – границы итерирования, а *val3* – шагом, с которым проходит итерирование. При записи цикла *for* значение *val3* можно опустить, тогда интерпретатор Lua автоматически установит его равным 1.

Конструкция такого цикла представлена в таблице 7.

Таблица 2 - Конструкция оператора FOR

|  |  |
| --- | --- |
| Блок-схема | ***Lua*** |
| for_блок_сх | **FOR** var = val1, val2, val3 **DO**  Инструкции  **END** |

Цикл WHILE

Последним типом цикл является конструкция WHILE, представленная в таблице 8. Работает этот оператор следующим образом: до тех пор, пока (WHILE) условие верно, выполняются инструкции.

Таблица 3 - Конструкция оператора WHILE

|  |  |
| --- | --- |
| **Блок-схема** | ***Python*** |
| for_блок_сх | **WHILE** условие **DO**  блок инструкций  **END** |

### Конструкция переключателя

Начиная с версии 3.10 для осуществления *множественного выбора* (или ветвления) используется конструкция с переключателем типа MATCH-CASE. В этом случае выполняется тот блок инструкций (инструкции 1 или инструкции 2 или др.), которому соответствует значение выражения. Если же такого значения нет, то выполняются инструкции в блоке CASE \_. Блок-схема и синтаксис оператора представлены в таблице 9.

Таблица 4 - Конструкция оператора CASE

|  |  |
| --- | --- |
| **Блок-схема** | ***Python*** |
| выбор блок схема | **MATCH** выражение**:**  **CASE** знач.1: инструкции 1  **CASE** знач.2: инструкции 2  **CASE** знач.3: инструкции 3  …..  **CASE \_:** инструкции |

### Списки в Python

*Списки* в *Python* — это структура данных для хранения объектов различных типов. Если вы использовали другие языки программирования, то вам должно быть знакомо понятие массива. Так вот, список очень похож на массив, только, как было уже сказано выше, в нем можно хранить объекты различных типов. Размер списка не статичен, его можно изменять. Список по своей природе является изменяемым типом данных. Переменная, определяемая как список, содержит ссылку на структуру в памяти, которая в свою очередь хранит ссылки на какие-либо другие объекты или структуры.

Блок-схема и синтаксис оператора представлены в таблице 10.

Таблица 5 – Пример задания элементов массива

|  |  |
| --- | --- |
| Блок схема | ***Python*** |
|  | arr = [value, value, ...]  arr = list([iterable]) |

# 2. РЕШЕНИЕ ЗАДАЧ НА Python

## Задача №1

Написать программу вычисления боковой поверхности и объема усеченной пирамиды.

Алгоритм программы линейной структуры в виде блок-схемы представлен на рисунке 1.

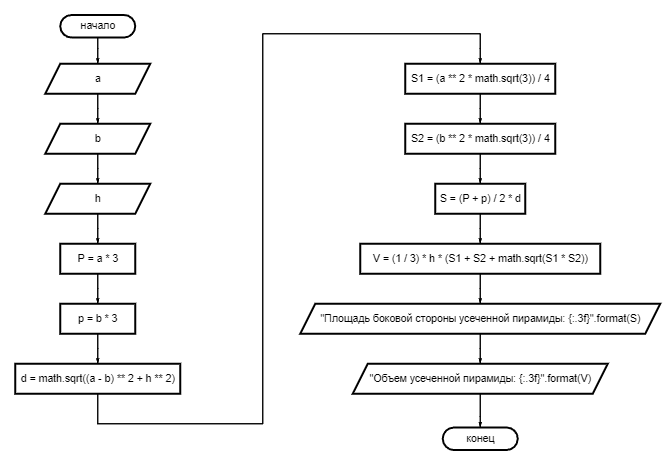
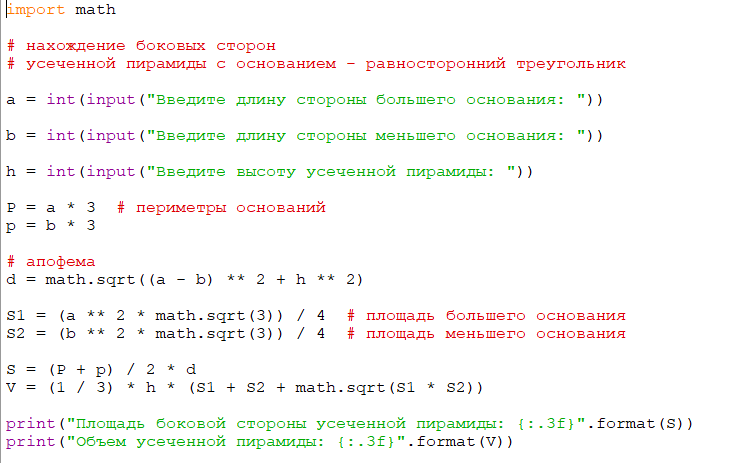
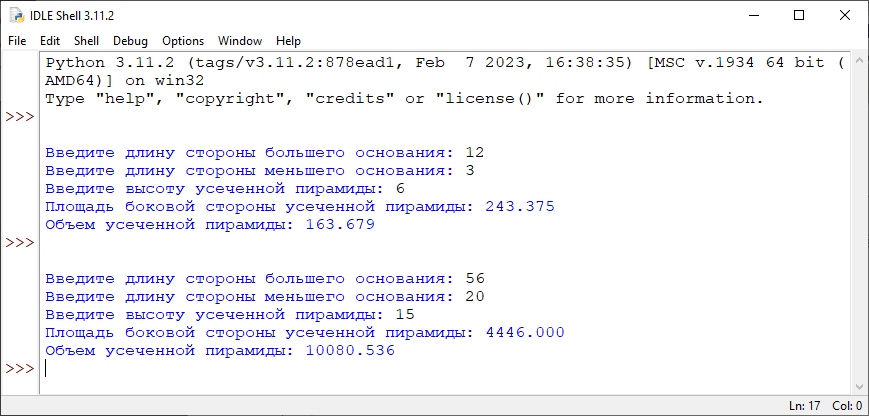


Рисунок 1 - Блок-схема программы №1

Листинг кода программы № 1





## Задача №2

Написать программу вычисления итоговой суммы вклада в юанях, если известны сумма вклада в рублях, процентная ставка, срок хранения и курс юаня на начало и конец сроков вклада.

Алгоритм программы линейной структуры в виде блок-схемы представлен на рисунке 2.

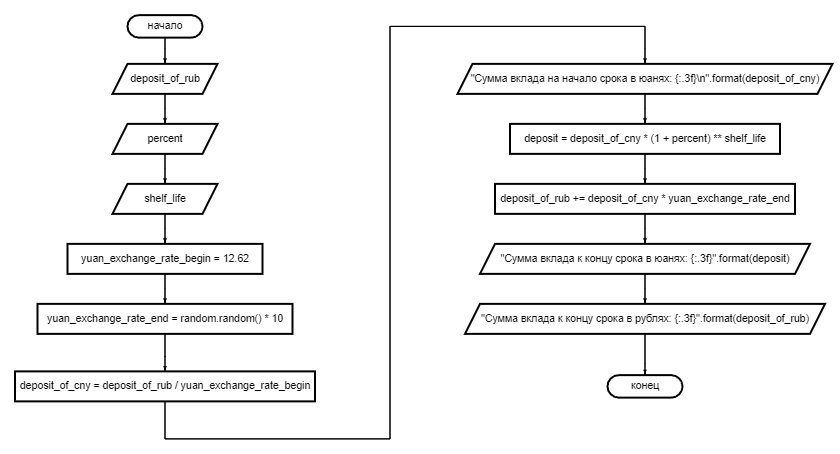
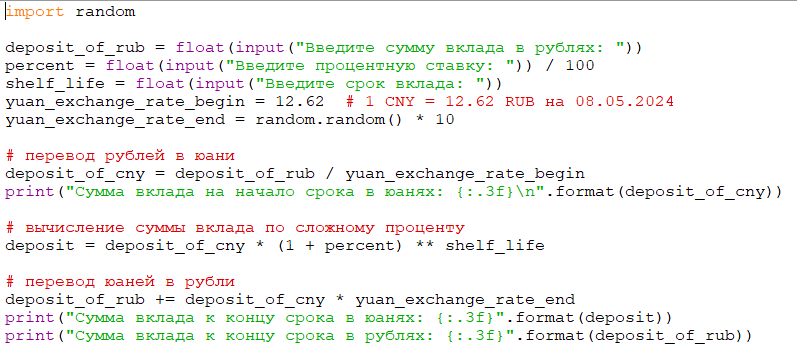
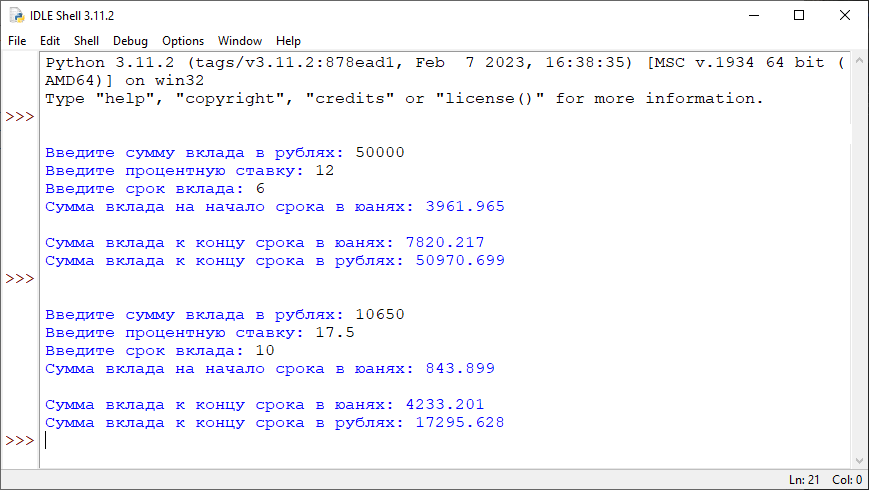


Рисунок 2 - Блок-схема алгоритма программы № 2

Листинг кода программы № 2





## Задача №3

Написать программу поиска координат точек экстремумов параболы с указанием его типа.

Алгоритм программы с использованием оператора *If* в виде блок-схемы представлен на рисунке 3.

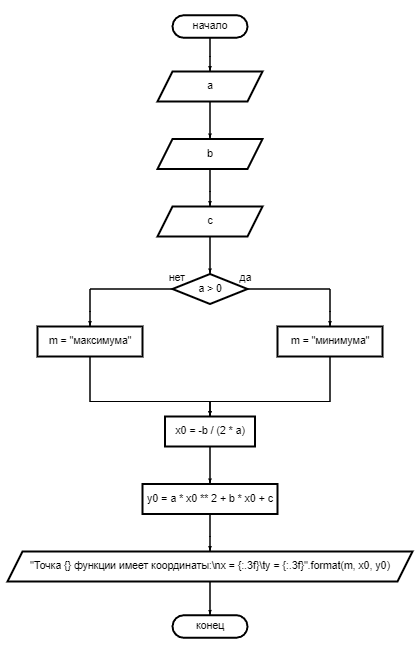
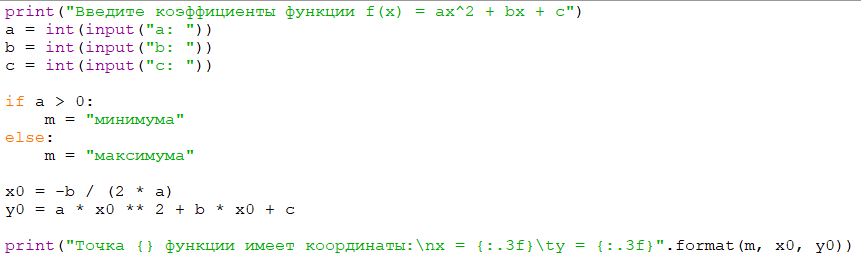
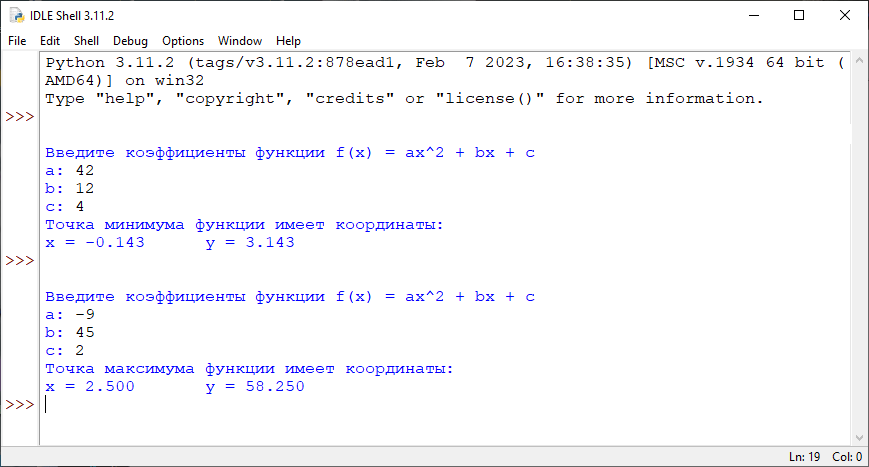


Рисунок 3 - Блок-схема алгоритма программы № 3

Листинг кода программы № 3





Задача №4

Написать программу поиска суммы первых n членов суммы ряда с заданной точностью.

Алгоритм программы с использованием цикла FOR в виде блок-схемы представлен на рисунке 4.

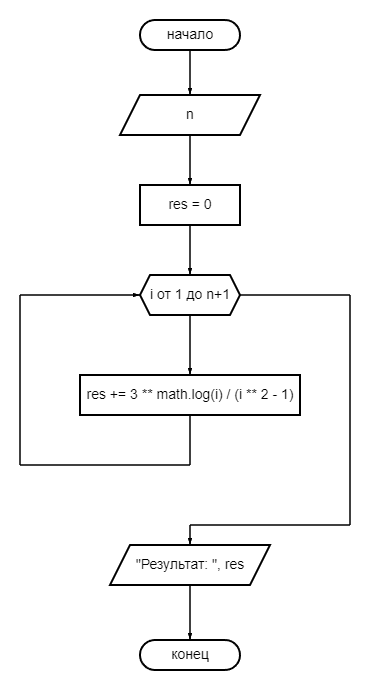
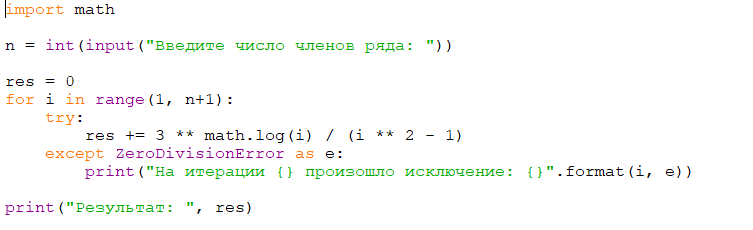
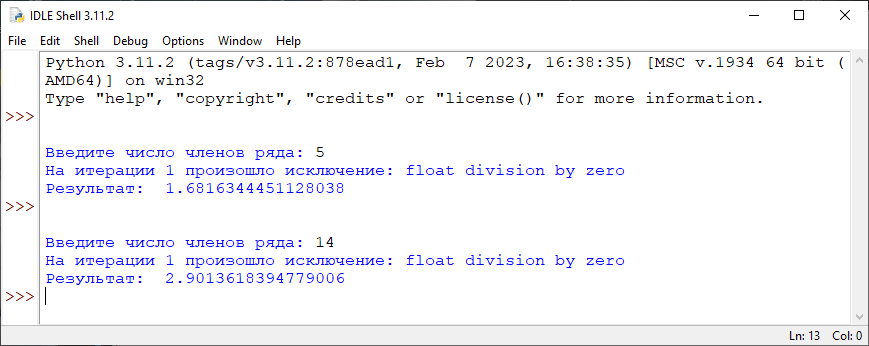


Рисунок 4 - Блок-схема алгоритма программы № 4

Листинг кода программы № 4





## Задача №5

Написать программу перевода числа из системы счисления с основанием n в систему счисления с основанием m, где n < m и m и n принадлежат [2; 16].

Алгоритм программы в виде блок-схемы представлен на рисунке 5.

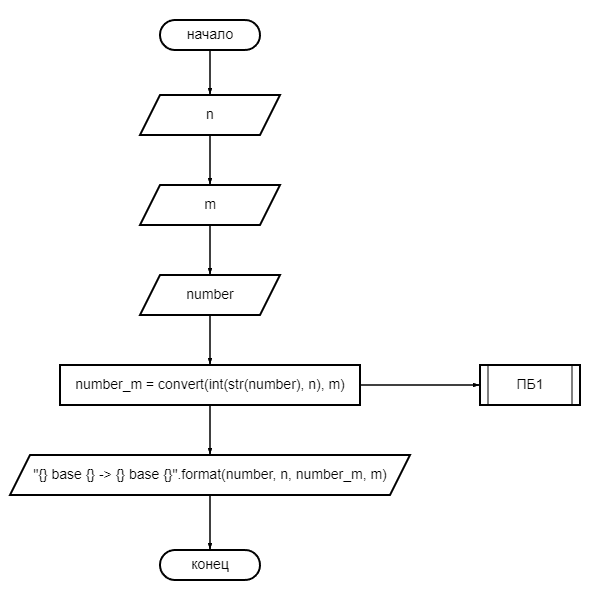


Рисунок 5 - Блок-схема алгоритма программы № 5

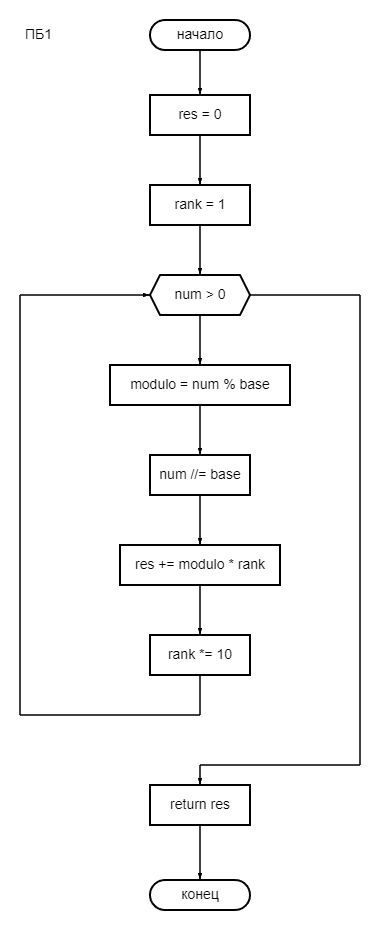
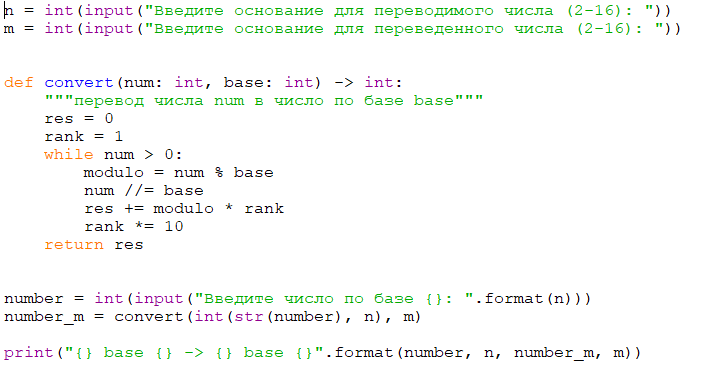
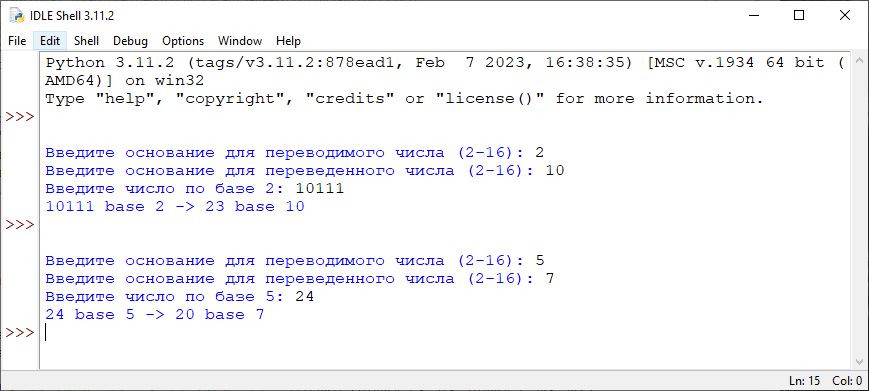


Рисунок 6 - Блок-схема алгоритма программного блока 1

Листинг кода программы № 5





# ЗАКЛЮЧЕНИЕ

В ходе выполненной работы были освоены и закреплены следующие основные навыки:

* создание алгоритма программы в виде блок-схемы с линейной и нелинейной структурами;
* создание текста программы на *Python* по блок-схеме программы;
* освоение синтаксиса *Python*,основныхконструкций операторов языка;
* использование условного оператора If;
* использование оператора цикла *For*;
* использование оператора цикла *While*;
* работа с массивами, символами, cтроками и математическими функциями в *Python*;
* работа со стандартной библиотекой модулей *Python*;
* основы работы в *IDLE*;
* создание пользовательских функций на *Python*;
* обработка исключений;
* работа с форматированием строк;

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Python (язык программирования): [Электронный ресурс] – URL: <https://ru.wikipedia.org/wiki/Python> (дата обращения: 08.05.2024).

2. IDLE: [Электронный ресурс] – URL: <https://ru.wikipedia.org/wiki/IDLE> (дата обращения: 08.05.2024).

3. ГОСТ 7.32-2017 СИБИД. Отчет о научно-исследовательской работе. Структура и правила оформления. [Электронный ресурс] // Электронный фонд правовых и нормативно-технических документов. Официальное издание. М.: Стандартинформ, 2017. – 29 с.

4. ГОСТ 19.701-90 (ИСО 5807-85) Единая система программной документации (ЕСПД). Схемы алгоритмов, программ, данных и систем. Обозначения условные и правила выполнения. [Электронный ресурс] // Комитет стандартов, мер и измерительных приборов при Совмине СССР. 1990. – 23 с.