

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ И ПРОГРАММИРОВАНИЕ

Методические указания к практическим занятиям

для направления подготовки:

10.03.01 Информационная безопасность

Ставрополь, 2026

Н 63 Информационные технологии и программирование: методические указания к практическим занятиям для студентов направления 10.03.01 Информационная безопасность / Николаев Е.И. – Ставрополь: Изд-во СКФУ, 2026. – 77 с.

Автор:

канд. техн. наук, доцент **Е.И. Николаев**

ОГЛАВЛЕНИЕ

Практическое занятие 1. Принципы разработки программ.....	4
Практическое занятие 2. Циклы, условный оператор	12
Список литературы	25

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ 1. ПРИНЦИПЫ РАЗРАБОТКИ ПРОГРАММ

Цель и содержание

Сформировать у студентов базовые навыки написания простых программ на Python: уметь объявлять переменные, различать и осознанно использовать основные типы данных, выполнять арифметические вычисления, выводить результаты и проверять типы объектов.

Задание для подготовки к практическому занятию и вопросы для обсуждения в рамках практического занятия

1. Переменные: Почему в Python не нужно указывать тип переменной при объявлении? Что хранится в переменной: тип или значение?
2. Типы данных: Чем отличаются `int` и `float` с точки зрения операций и хранения? Когда результат деления будет `float`, а когда можно получить `int`?
3. Функция `print()`: В чём разница между выводом нескольких аргументов через запятую и конкатенацией строк? Когда удобнее f-строки?
4. Функция `type()`: Как интерпретировать вывод `type(x)`? Почему `type(5)` и `type(5.0)` дают разные результаты?
5. Динамическая типизация: Что произойдёт, если в одну и ту же переменную сначала записать число, потом строку? Как это влияет на читаемость и надёжность кода?
6. Обработка ошибок: Почему выражение `int("abc")` вызовет ошибку, а `int("123")` – нет? Как правильно обрабатывать такие случаи в простых скриптах?

Задания (варианты) для практического занятия

Представлены задания трех уровней. Каждый студент выбирает по одному заданию каждого уровня: **базового, среднего и продвинутого** в соответствии со своим вариантом.

Уровень 1 – Базовый

1. Создайте переменную `a = 7`, выведите её значение и тип.
2. Создайте переменную `b = 3.14`, выведите её значение и тип.
3. Создайте переменную `s = "Привет"`, выведите её значение и тип.
4. Создайте переменную `flag = True`, выведите её значение и тип.
5. Создайте переменную `x = 100` и выведите фразу: Значение: 100 через f-строку.
6. Создайте переменную `y = 5.5` и выведите фразу: Значение: 5.5 через f-строку.
7. Создайте переменную `name = "Анна"`, выведите: Имя: Анна.
8. Создайте переменную `age = 19`, выведите: Возраст: 19.
9. Создайте переменную `temperature = 36.6`, выведите её значение и тип.
10. Создайте переменную `city = "Ставрополь"`, выведите её длину через `len()`.
11. Объявите `n = 0`, выведите значение и тип. Объясните, почему тип — `int`.
12. Объявите `k = -10`, выведите значение и тип.
13. Объявите `text = ""` (пустая строка), выведите значение, тип и длину.
14. Объявите `isEmpty = False`, выведите значение и тип.
15. Объявите `price = 49.99`, выведите: Цена: 49.99 руб. через f-строку.
16. Объявите `count = 42`, выведите значение, тип и фразу Всего товаров: 42.
17. Объявите `letter = "А"`, выведите значение и тип. Проверьте длину строки.
18. Объявите `pi = 3.14159`, выведите значение и тип.
19. Объявите `message = "Hello, World!"`, выведите значение и длину строки.

20. Объявите `done = True`, выведите: Статус: `True` и тип переменной.

Уровень 2 – Средний

1. Даны `a = 10`, `b = 3`. Выведите сумму, разность и произведение. Для каждого результата выведите тип.
2. Даны `x = 10`, `y = 3`. Выведите результат обычного деления `x / y` и целочисленного деления `x // y`. Сравните типы.
3. Даны `m = 17`, `n = 5`. Выведите остаток от деления и результат возведения `m` в степень `n`. Проверьте типы.
4. Даны `p = 7`, `q = 2`. Вычислите среднее арифметическое. Выведите результат и тип.
5. Создайте `num = 42`. Преобразуйте его в строку, выведите результат и его тип. Объясните, что изменилось.
6. Создайте `str_num = "123"`. Преобразуйте в целое число, прибавьте 10, выведите результат и тип.
7. Создайте `str_float = "3.5"`. Преобразуйте в `float`, умножьте на 2, выведите результат и тип.
8. Даны `a = 5`, `b = 2.0`. Вычислите `a + b`, `a * b`, `a / b`. Выведите результаты и типы. Объясните, почему тип меняется.
9. Даны `width = 4`, `height = 5`. Вычислите площадь прямоугольника и периметр. Выведите оба результата в одной строке через `f`-строку.
10. Даны `name = "Максим"`, `age = 21`. Выведите фразу «Меня зовут Максим, мне 21 год» тремя способами: аргументы `print`, конкатенация, `f`-строка.
11. Даны `a = 8`, `b = 3`. Вычислите `(a + b) ** 2` и `a ** 2 + 2 * a * b + b ** 2`. Сравните результаты. Выведите оба.
12. Создайте `value = 100`. Присвойте ему строку "сто". Выведите значение и тип до и после. Объясните, что произошло.
13. Даны `celsius = 25`. Вычислите температуру в Фаренгейтах: $F = C * 9/5 + 32$. Выведите результат и тип.

14. Даны `radius = 5`. Вычислите площадь круга $S = \pi * r ** 2$, где $\pi = 3.14$. Выведите результат и тип.
15. Даны `a = "10"`, `b = "20"`. Выведите результат `a + b` (конкатенация). Затем преобразуйте оба в числа и выведите их сумму. Сравните.
16. Даны `seconds = 3675`. Вычислите, сколько это часов, минут и секунд осталось. Выведите через f-строку в формате Часов: X, Минут: Y, Секунд: Z.
17. Даны `a = 10`, `b = 3`. Вычислите и выведите `a % b`, `b % a`, `a // b`, `b // a`. Для каждого выведите тип.
18. Даны `price = 199.99`, `quantity = 3`. Вычислите итоговую сумму. Выведите: Сумма: XXX.XX руб. через f-строку.
19. Даны `bool_val = True`. Преобразуйте в `int`, выведите результат и тип. Объясните, что означает 1.
20. Даны `a = 5`, `b = 3`, `c = 2`. Вычислите выражение `a + b * c - a // c`. Выведите результат и тип. Распишите порядок вычислений в комментарии.

Уровень 3 – Продвинутый

1. Даны `a`, `b`. Вычислите гипотенузу прямоугольного треугольника: $c = (a ** 2 + b ** 2) ** 0.5$. Выведите результат, его тип и округлите до 2 знаков через `round()`.
2. Даны `total_seconds = 100000`. Преобразуйте в формат «дни, часы, минуты, секунды». Выведите через f-строку. Проверьте тип каждого промежуточного результата.
3. Даны `x`, `y`. Вычислите среднее арифметическое, геометрическое $((x * y) ** 0.5)$ и гармоническое $(2 * x * y / (x + y))$. Выведите все три результата и их типы.
4. Даны `body_mass`, `height_m`. Вычислите ИМТ: $BMI = mass / height ** 2$. Выведите результат, тип и категорию по условию: если ИМТ < 18.5 — «недостаток», иначе «норма или избыток» (выведите текстом, без `if` — через тернарный оператор или просто вычислением). Если тернарный оператор ещё не проходили — выведите только числовой результат и пояснение в комментарии.

5. Даны `a = ""`, `b = ""`. Преобразуйте оба в числа (`int` и `float` соответственно), вычислите сумму, произведение и деление. Выведите все результаты и типы.
6. Даны `summa`, `percent`, `years`. Вычислите сложный процент по формуле $S * (1 + P/100) ** N$. Выведите итог с 2 знаками после запятой.
7. Даны `a`, `b`. Поменяйте значения переменных местами без третьей переменной (через кортеж: `a, b = b, a`). Выведите до и после.
8. Даны `number`. Вычислите сумму его цифр (через строковое преобразование и перебор символов, приведя каждый обратно в `int`). Выведите результат и тип.
9. Даны `side`. Вычислите площадь квадрата, объём куба и площадь поверхности куба. Выведите все три значения и их типы в одной f-строке.
10. Даны `km`, `fuel_per_100km`, `price_per_liter`. Вычислите стоимость поездки. Выведите: Поездка обойдётся в XXX.XX руб. Проверьте тип результата.
11. Даны `a = True`, `b = False`, `c = True`. Вычислите `a + b + c` (неявное преобразование `bool` в `int`). Выведите результат и тип. Объясните в комментарии, почему получилось именно это число.
12. Даны `seconds`. Округлите до целых, до 1 знака, до 2 знаков. Выведите все три результата и их типы (обратите внимание: `round()` без второго аргумента возвращает `int`).
13. Даны `a`, `b`. Выведите результат `a / b` с точностью 3 знака после запятой (через `round()` и через f-строку `:.3f`). Сравните, отличается ли тип результата.
14. Даны `price_usd`, `rate`. Вычислите цену в рублях, затем в евро через кросс-курс `euro_rate`. Выведите оба результата и их типы.
15. Даны `text = "..."`. Выведите первый символ, последний символ и длину строки. Преобразуйте длину в `float` и выведите её тип.
16. Даны `a`, `b`, `c`. Вычислите `a ** b ** c` (обратите внимание на приоритет `**` — он справа налево). Выведите результат и тип. В комментарии распишите порядок.

17. Даны `mass`, `gravity_earth`, `gravity_moon`. Вычислите вес тела на Земле и на Луне. Выведите оба значения, разницу между ними и типы всех результатов.
18. Даны `start`, `end`, `step`. Вычислите количество шагов от `start` до `end` с интервалом `step`: $n = (\text{end} - \text{start}) / \text{step}$. Выведите `n`, его тип и округлённое до целого значение с типом.
19. Даны `a`, `b`. Вычислите `a + b`, `a - b`, `a * b`. Затем попробуйте `a / b` — объясните в комментарии, что произойдёт и почему. Остальные результаты выведите с типами.
20. Даны `hours`, `minutes`, `seconds`. Переведите всё в секунды, затем обратно — в «часы, минуты, секунды». Выведите оба результата. Проверьте, что исходные значения совпали с результатом обратного перевода.

Контрольные вопросы

Ответы на контрольные вопросы также необходимо предоставить в блокноте Python Notebook:

1. Что такое переменная в Python? Объясните, как происходит присваивание значения переменной, и приведите 2-3 примера корректных и 1 пример некорректного имени переменной с пояснением, почему оно недопустимо.
2. Перечислите основные простейшие типы данных в Python (минимум 4), укажите для каждого пример литерала и результат вызова `type()` для него. Объясните, почему тип переменной в Python считается динамическим.
3. Как работает функция `print()`? Приведите 3 разных примера её использования: вывод одной переменной, нескольких значений через запятую, форматированного сообщения (через f-строку или конкатенацию). Поясните разницу в выводе.
4. Что делает функция `type()`? Напишите небольшой фрагмент кода, где для переменных разных типов (число, строка, булево значение) выводится их тип, и объясните, как интерпретировать результат.

5. Какие арифметические операции поддерживаются в Python? Приведите примеры для `+`, `-`, `*`, `/`, `//`, `%`, `**` с числами разных типов и поясните, какой тип результата получается в каждом случае (например, деление всегда даёт **float**).

6. В чём разница между обычным делением (`/`) и целочисленным делением (`//`)? Приведите по 2 примера для положительных и отрицательных чисел и объясните поведение оператора в каждом случае.

7. Как в Python происходит преобразование типов? Покажите примеры использования функций **int()**, **float()**, **str()** для конвертации между типами и укажите, в каких ситуациях может возникнуть ошибка (например, при попытке преобразовать нечисловую строку в число).

8. Что означает выражение «вычисление в Python – это выражение, возвращающее значение»? Приведите 3 примера выражений (арифметическое, сравнение, логическое) и покажите, как результат каждого из них можно присвоить переменной или передать в **print()**.

9. Как работают операторы сравнения (`==`, `!=`, `<`, `>`, `<=`, `>=`) и логические операторы (**and**, **or**, **not**)? Напишите 2-3 составных выражения и предскажите их результат, затем объясните порядок вычисления (приоритет операторов).

10. Что происходит при выполнении операции над операндами разных типов (например, **int** и **float**, **str** и **int**)? Приведите примеры неявного приведения типов и случаи, когда Python не выполняет автоматическое приведение и требует явной конвертации.

11. Как вывести на экран несколько значений одновременно с помощью **print()**? Сравните способы: перечисление через запятую, f-строки, метод **.format()**, конкатенация строк. Укажите, когда каждый способ удобнее, и приведите по одному примеру.

12. Что такое порядок выполнения операций в Python и как его можно изменить? Напишите выражение с несколькими операциями и скобками, поясните, в каком порядке будут выполнены действия, и покажите результат через **print()**.

13. Как устроено множественное присваивание в Python? Приведите примеры: одновременное присваивание нескольких значений нескольким переменным, обмен значений двух переменных без временной переменной, распаковка кортежа/списка.

14. Что такое изменяемые и неизменяемые типы данных? На примере простейших типов (числа, строки) объясните, почему они считаются неизменяемыми, и продемонстрируйте, что попытка «изменить» строку приводит к созданию нового объекта (можно использовать `id()` для наглядности).

15. Составьте небольшой фрагмент кода (5-7 строк), где создаются переменные разных типов, выполняются вычисления, выводятся результаты через `print()`, а типы переменных проверяются через `type()`. Поясните каждую строку: что происходит, какой тип у результата и почему.

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ 2. ЦИКЛЫ, УСЛОВНЫЙ ОПЕРАТОР

Цель и содержание

сформировать у студентов практические навыки алгоритмического мышления и грамотного применения базовых конструкций языка Python для решения вычислительных задач. Обучающиеся должны научиться проектировать логику программы через комбинации условных операторов и циклов, корректно использовать встроенные и математические функции, выполнять вычисления с контролем типов данных и обрабатывать разные сценарии входных значений.

В ходе выполнения задания студенты должны:

1. Освоить применение условных конструкций (**if / elif / else**) для реализации разветвлений: научиться формулировать логические условия, комбинировать их через **and**, **or**, **not**, обрабатывать исключительные и граничные случаи (например, деление на ноль, отрицательные значения для корня).
2. Научиться использовать циклы **for** и **while** для итераций: понимать различия в сценариях их применения (фиксированное количество итераций или условие продолжения), отрабатывать использование **break**, **continue**, **else**, **finally** в циклах, избегать бесконечных циклов и корректно задавать условия выхода.
3. Отработать навыки вычислений с разными типами данных (**int**, **float**): применять арифметические операторы, учитывать приоритет операций, использовать скобки для управления порядком вычислений, контролировать точность результатов и типы промежуточных значений.

Задание для подготовки к практическому занятию и вопросы для обсуждения в рамках практического занятия

Повторите теорию:

1. синтаксис и логику работы **if / elif / else**;
2. циклы **for** и **while**, их отличия и типичные сценарии использования;

3. конструкцию `else` в циклах (когда выполняется, когда — нет), примеры с `break`;
4. встроенные функции Python: `min`, `max`, `sum`, `abs`, `round`, `len`, `range`, `input`, `print`;
5. модуль `math`: как импортировать, какие функции пригодятся (`sqrt`, `pow`, `factorial`, `sin`, `cos`, `pi`, `e` и т. д.).

Проанализируйте и запустите в среде разработки (IDLE, PyCharm, VS Code, Jupyter) следующие фрагменты кода. Для каждого фрагмента сформулируйте, что он выведет, и объясните, почему:

1. простой **`if-else`** с проверкой диапазона;
2. цикл **`for`** по **`range`** с вычислением суммы;
3. цикл **`while`** с условием и явным изменением счётчика;
4. цикл с **`break`** и блоком **`else`**;
5. пример с **`math.sqrt`** и обработкой случая отрицательного числа через **`if`**.

Вопросы для обсуждения на практическом занятии

1. В каких случаях лучше использовать **`for`**, а в каких — **`while`**? Приведите по одному реальному примеру задачи для каждого цикла и объясните, почему именно этот цикл подходит.
2. Как работает блок **`else`** в цикле? Почему он может запутывать новичков? Обсудим на примерах: когда **`else`** выполняется, а когда — нет, и как это связано с **`break`**.
3. Какие ошибки чаще всего возникают при работе с условиями и циклами (бесконечный цикл, неверный диапазон **`range`**, неправильное условие выхода и т. п.)? Приведите 1–2 типичные ошибки из своего опыта и способы их исправить.
4. Чем отличаются встроенные функции **`sum`**, **`min`**, **`max`** от ручного подсчёта в цикле? В каких ситуациях выгоднее использовать встроенные функции, а в каких — цикл?

5. Как корректно использовать функции из модуля **math**? Почему нельзя просто писать **sqrt(x)** без импорта, и какие ошибки возникают при передаче неподходящих аргументов (например, отрицательное число в **sqrt**).
6. Как правильно обрабатывать «проблемные» случаи в вычислениях (деление на ноль, отрицательные значения под корнем и т. п.) с помощью условных операторов? Приведите пример структуры **if-elif-else** для такой обработки.
7. Как контролировать типы данных при вычислениях и вводе от пользователя? Обсудите типичную ошибку: **input()** всегда возвращает строку, и что будет, если сразу использовать её в арифметике. Как это исправить?
8. Как выбрать нужную математическую функцию для задачи? Например: когда использовать **pow(x, y)** из **math**, а когда – оператор ******; когда нужен **math.factorial**, а когда проще написать цикл?
9. Как проверить корректность результата приближённых вычислений (например, корня методом итераций)? Проанализируйте критерии остановки (разница между итерациями, количество шагов, сравнение с **math.sqrt**) и как их реализовать в коде.
10. Как сделать код более читаемым и понятным при использовании вложенных условий и циклов? Проанализируйте простые приёмы: понятные имена переменных, комментарии, разбиение на логические блоки.

Задания (варианты) для практического занятия

Представлены задания трех уровней. Каждый студент выбирает по одному заданию каждого уровня: **базового**, **среднего** и **продвинутого** в соответствии со своим вариантом.

Уровень 1 – Базовый

Вариант 1. Пользователь вводит целое число. Программа определяет, является ли оно чётным или нечётным, и выводит соответствующее сообщение. Используйте условный оператор и операцию **%**.

Вариант 2. Пользователь вводит два числа. Программа выводит большее из них. Если числа равны — выводит «Числа равны». Используйте if/elif/else.

Вариант 3. С помощью цикла for и функции range выведите все числа от 1 до 20, каждое на новой строке.

Вариант 4. С помощью цикла for вычислите и выведите сумму всех чисел от 1 до 100. Используйте встроенную функцию sum или накопление в переменной.

Вариант 5. Пользователь вводит число n. С помощью цикла for выведите таблицу умножения этого числа от 1 до 10 в формате $n \times i = \text{результат}$.

Вариант 6. Пользователь вводит число. Программа определяет, положительное оно, отрицательное или равно нулю, и выводит результат. Используйте условный оператор.

Вариант 7. С помощью цикла while выведите все числа от 10 до 1 (обратный порядок), каждое на новой строке.

Вариант 8. Пользователь вводит целое число n. Вычислите его факториал с помощью цикла for (без использования math.factorial). Выведите результат.

Вариант 9. Пользователь вводит число. Программа определяет, делится ли оно на 3 и на 5 одновременно, только на 3, только на 5, или ни на одно из них. Используйте if/elif/else.

Вариант 10. С помощью цикла for и функции range с шагом 2 выведите все чётные числа от 2 до 50.

Вариант 11. Пользователь вводит количество чисел n, затем n чисел по одному. С помощью встроенных функций min и max определите и выведите наименьшее и наибольшее из введённых чисел.

Вариант 12. Пользователь вводит число n. С помощью цикла while вычислите сумму чисел от 1 до n. Выведите результат.

Вариант 13. Пользователь вводит число. С помощью встроенной функции abs выведите его модуль. Дополнительно определите и выведите, положительное оно или отрицательное.

Вариант 14. Пользователь вводит три числа. Программа определяет и выводит наибольшее из трёх, используя только условные операторы (без max).

Вариант 15. С помощью цикла `for` выведите квадраты чисел от 1 до 15 в формате число — квадрат. Для вычисления квадрата используйте оператор `**`.

Вариант 16. Пользователь вводит число `n`. С помощью цикла `for` вычислите и выведите среднее арифметическое всех чисел от 1 до `n`. Используйте встроенную функцию `sum` или накопление.

Вариант 17. Пользователь вводит число. Программа определяет, кратно ли оно 10. Если да — выводит «кратно 10», если нет — выводит остаток от деления на 10.

Вариант 18. Импортируйте модуль `math`. Пользователь вводит положительное число. Программа вычисляет и выводит его квадратный корень через `math.sqrt`. Если число отрицательное — выводит сообщение об ошибке.

Вариант 19. Пользователь вводит число `n`. С помощью цикла `while` и счётчика выведите все числа от `n` до 1, которые делятся на 3.

Вариант 20. Пользователь вводит число `n`. С помощью цикла `for` выведите сумму его цифр. Подсказка: используйте `n % 10` для получения последней цифры и `n // 10` для удаления.

Уровень 2 – Средний

Вариант 1. Пользователь вводит число `n`. Программа определяет, является ли оно простым (делится только на 1 и на себя). Используйте цикл `for` или `while` и условный оператор. Выведите «простое» или «составное». Числа 0 и 1 обработайте отдельно.

Вариант 2. Пользователь вводит число `n`. Программа выводит все делители этого числа (кроме 1 и самого числа). Используйте цикл **`for`** и условный оператор. Если делителей нет – выведите «делителей нет».

Вариант 3. Пользователь вводит число `n`. Программа определяет, является ли оно простым, используя цикл **`for`** с блоком **`else`**. Если цикл завершился без **`break`** – число простое, если **`break`** сработал – составное.

Вариант 4. Пользователь вводит положительное число x и точность eps (например, 0.0001). С помощью цикла **while** вычислите квадратный корень из x методом Герона: $y_{new} = (y + x/y)/2$, начиная с $y = x$. Повторяйте, пока $|y_{new} - y| < eps$. Выведите результат и сравните с **math.sqrt(x)**.

Пользователь вводит число n . С помощью цикла **for** вычислите сумму ряда $1 + 1/2 + 1/3 + \dots + 1/n$. Выведите результат с округлением до 4 знаков после запятой через **round()**.

Вариант 6. Пользователь вводит число n . Программа определяет, является ли оно совершенным (равно сумме всех своих делителей, кроме самого себя). Используйте цикл **for** и условный оператор. Пример: $6 = 1 + 2 + 3$.

Вариант 7. Пользователь вводит количество чисел n , затем n чисел. С помощью цикла **for** и условного оператора выведите: количество положительных, количество отрицательных и количество нулей среди введённых чисел.

Вариант 8. Пользователь вводит число n . С помощью цикла **while** и операции **%** определите, является ли запись этого числа в двоичной системе палиндромом. Выведите «да» или «нет».

Вариант 9. Импортируйте **math**. Пользователь вводит число x . Программа вычисляет и выводит: **math.sin(x)**, **math.cos(x)**, **math.tan(x)** с округлением до 3 знаков. Если x кратно $\pi/2$ – выведите предупреждение о том, что тангенс не определён, и проверьте это через **math.isclose**.

Вариант 10. Пользователь вводит два числа a и b . С помощью цикла **for** выведите все числа в диапазоне от a до b , которые делятся на 7, но не делятся на 3. Если таких чисел нет – выведите «нет таких чисел» (используйте флаг или **else** в цикле).

Вариант 11. Пользователь вводит число n . С помощью цикла **for** выведите первые n чисел Фибоначчи (1, 1, 2, 3, 5, 8, ...). Каждое число — на новой строке.

Вариант 12. Пользователь вводит число n . Программа вычисляет и выводит его факториал двумя способами: через цикл **for** и через **math.factorial**. Сравните результаты и выведите, совпадают ли они.

Вариант 13. Пользователь вводит число n . С помощью цикла **while** разложите число на простые множители и выведите их в виде **множитель1 * множитель2 ***
Пример: **60 = 2 * 2 * 3 * 5.**

Вариант 14. Пользователь вводит число n . Программа определяет, является ли число палиндромом (читается одинаково слева направо и справа налево). Используйте цикл **while** и операции **//** и **%** для переворота числа. Выведите результат.

Вариант 15. Пользователь вводит число n . С помощью цикла **for** вычислите сумму квадратов всех чисел от 1 до n : $1^2 + 2^2 + \dots + n^2$. Сравните результат с формулой $n(n + 1)(2n + 1)/6$. Выведите оба значения.

Вариант 16. Пользователь вводит число n . С помощью цикла **while** и условного оператора определите, является ли число степенью двойки (1, 2, 4, 8, 16, ...). Выведите «да» или «нет» и, если да, – какую степень.

Вариант 17. Пользователь вводит число n . С помощью цикла **for** выведите «пирамиду» из звёздочек высотой n :

```
*
**
***
****
*****
```

Вариант 18. Пользователь вводит число n . С помощью цикла **for** вычислите приближённое значение π по формуле Лейбница: $\pi/4 = 1 - 1/3 + 1/5 - 1/7 + \dots$ (первые n членов). Выведите результат и сравните с **math.pi**.

Вариант 19. Пользователь вводит число n . Программа находит все числа Армстронга (числа, равные сумме своих цифр в степени количества цифр) от 1 до n . Пример: $153 = 1^3 + 5^3 + 3^3$. Используйте вложенные циклы или **for** с **str**.

Вариант 20. Пользователь вводит число n . С помощью цикла **for** и встроенной функции **pow** (или оператора ******) определите, является ли число точным квадратом целого числа. Выведите корень, если да, или «не является точным квадратом».

Уровень 3 – Продвинутый

Вариант 1. Пользователь вводит число n . Программа определяет, является ли оно простым, за $O(\sqrt{n})$ — цикл **for** от 2 до `int(math.sqrt(n)) + 1` с **break** и **else**. Дополнительно замерьте время выполнения через `time.perf_counter` для $n = 10^9 + 7$ и $n = 10^9 + 9$ и сравните.

Вариант 2. Пользователь вводит число n (большое). С помощью решета Эратосфена и цикла **for** найдите все простые числа до n . Выведите их количество и последние 5 найденных чисел. Используйте список булевых значений для отметки составных чисел.

Вариант 3. Пользователь вводит число x и точность ϵ . Вычислите $\sin(x)$ через разложение в ряд Тейлора: $\sin(x) = x - x^3/3! + x^5/5! - \dots$, пока модуль очередного члена не станет меньше ϵ . Используйте **while**, `math.factorial` или накопление факториала в цикле. Сравните с `math.sin(x)`.

Вариант 4. Пользователь вводит число n . С помощью цикла **while** реализуйте алгоритм Евклида для нахождения НОД двух чисел a и b (их тоже вводит пользователь). Дополнительно найдите НОК через формулу $a * b // \text{НОД}$. Выведите оба результата.

Вариант 5. Пользователь вводит число n . Программа вычисляет число e (основание натурального логарифма) через ряд $e = 1 + 1/1! + 1/2! + 1/3! + \dots$, пока следующий член не станет меньше 10^{-10} . Используйте цикл **while** и накопление факториала. Выведите результат с 15 знаками и сравните с `math.e`.

Вариант 6. Пользователь вводит число n . Программа вычисляет золотое сечение $\phi = (1 + \sqrt{5})/2$ двумя способами: через `math.sqrt` и через предел последовательности Фибоначчи F_{n+1}/F_n . Выведите оба результата с 10 знаками и разность между ними.

Вариант 7. Пользователь вводит число n . Программа находит все пары дружественных чисел (каждое равно сумме делителей другого) от 1 до n . Используйте функцию для суммы делителей и вложенные циклы. Пример: 220 и 284. Оптимизируйте перебор.

Вариант 8. Пользователь вводит число n . Программа вычисляет $\ln(2)$ через ряд $\ln(2) = 1 - 1/2 + 1/3 - 1/4 + \dots$ (первый способ) и через ряд $\ln((1+x)/(1-x)) = 2(x + x^3/3 + x^5/5 + \dots)$ при $x = 1/3$ (второй способ). Сравните скорость сходимости обоих рядов для одного и того же количества членов n и выведите оба результата и `math.log(2)`.

Вариант 9. Пользователь вводит число n (до 100). Программа строит треугольник Паскаля до n -й строки, используя только списки и циклы `for`. Выведите треугольник выровненным по центру. Каждое число вычислите через комбинацию $C_n^k = n!/(k! \cdot (n-k)!)$ через `math.comb` или вручную.

Вариант 10. Пользователь вводит число n . С помощью цикла `for` и условных операторов реализуйте игру «FizzBuzz» с тремя условиями: кратно 3 — «Fizz», кратно 5 — «Buzz», кратно обоим — «FizzBuzz», иначе — само число. Дополнительно: вместо «Fizz» и «Buzz» выводите делитель с наибольшей суммой цифр (если 3 и 5 оба делят n , выведите тот, чья сумма цифр больше — как мета-правило).

Вариант 11. Пользователь вводит число n . Программа вычисляет $\sqrt{n}$ методом бинарного поиска (дихотомии) с точностью $\text{eps} = 1\text{e-}10$. Используйте цикл `while` с границами `low` и `high`. Выведите результат, количество итераций и сравните с `math.sqrt(n)`.

Вариант 12. Пользователь вводит число n . Программа находит количество способов разложения n на сумму натуральных чисел (число разбиений), используя динамическое программирование и циклы `for`. Выведите результат. Пример: для 4 ответ — 5 (4, 3+1, 2+2, 2+1+1, 1+1+1+1).

Вариант 13. Пользователь вводит число n . Программа реализует угадывание числа от 1 до n методом бинарного поиска: компьютер загадывает, пользователь отвечает «больше»/«меньше»/«равно». Используйте `while` и `input`. Выведите количество попыток и сравните с теоретическим максимумом $\lceil \log_2 n \rceil$ (используйте `math.log2` и `math.ceil`).

Вариант 14. Пользователь вводит число n . Программа находит все простые числа-палиндромы до n (числа, которые одновременно простые и палиндромы). Используйте циклы **for** и **while**, функцию для проверки простоты и функцию для проверки палиндрома. Выведите их количество и топ-5 наибольших.

Вариант 15. Пользователь вводит число n . Программа вычисляет постоянную Каталана G через ряд $G = 1 - 1/9 + 1/25 - 1/49 + \dots = \sum_{k=0}^{\infty} (-1)^k / (2k + 1)^2$, пока член суммы не станет меньше $\text{eps} = 1\text{e-}8$. Выведите результат с 10 знаками. Дополнительно вычислите числа Каталана $C_n = (2n)! / ((n + 1)! \cdot n!)$ для n от 0 до введённого значения и выведите их.

Вариант 16. Пользователь вводит число n . Программа реализует задачу Коллатца: если n чётное – делим на 2, если нечётное – умножаем на 3 и прибавляем 1. Цикл **while** продолжается, пока n не станет равным 1. Выведите всю последовательность и её длину. Дополнительно найдите число до 1000 с самой длинной последовательностью Коллатца.

Вариант 17. Пользователь вводит число n – количество шагов. Программа генерирует случайное блуждание на числовой прямой: на каждом шаге +1 или -1 с равной вероятностью (используйте `random.choice([-1, 1])`). Выведите позицию после n шагов, максимум и минимум траектории. Повторите 1000 раз и выведите среднюю итоговую позицию и средний максимум. Используйте `math.fabs` для анализа.

Вариант 18. Пользователь вводит число n . Программа вычисляет интеграл функции $f(x) = \sin(x)$ на отрезке $[0, \pi]$ методом прямоугольников с n разбиениями. Используйте цикл **for**, `math.sin` и `math.pi`. Выведите результат с 6 знаками и сравните с точным значением 2. Исследуйте, при каком n ошибка станет меньше 10^{-4} .

Вариант 19. Пользователь вводит число n . Программа находит все пары простых чисел-близнецов (разность равна 2) до n . Используйте решето Эратосфена для генерации простых чисел, затем цикл **for** для поиска пар. Выведите все пары и их количество. Пример: (3, 5), (5, 7), (11, 13).

Вариант 20. Пользователь вводит число n и систему счисления $base$ (от 2 до 16). Программа переводит n в систему $base$ с помощью цикла `while` и условных операторов (цифры выше 9 обозначайте буквами A–F). Дополнительно: программа переводит дробную часть (если есть) с точностью до 10 знаков после точки, используя умножение на $base$ в цикле. Выведите результат.

Контрольные вопросы

1. Ответы на контрольные вопросы также необходимо предоставить в блокноте Python Notebook:
2. В чём принципиальное отличие циклов `for` и `while` в Python? Приведите по одному примеру задачи, где каждый из них будет наиболее уместен, и кратко обоснуйте выбор.
3. Как работает блок `else` внутри цикла (`for/while`)? Опишите, при каких условиях он выполняется, а при каких – нет. Приведите два коротких примера кода: в одном `else` срабатывает, в другом – нет (из за `break`).
4. Объясните разницу между операторами `break` и `continue`. Напишите фрагмент кода с циклом `while`, где используются оба оператора, и прокомментируйте, что произойдёт на каждой итерации.
5. Какие типичные ошибки возникают при организации циклов в Python? Перечислите минимум 3 ошибки (например, бесконечный цикл, неверный диапазон `range`, неправильное условие выхода) и для каждой приведите короткий пример ошибочного кода и его исправленную версию.
6. Как устроена условная конструкция `if/elif/else`? Напишите пример, где проверяется принадлежность числа диапазонам (например, «меньше 0», «от 0 до 10», «больше 10»), и объясните, почему порядок условий важен.
7. Что такое логические операторы `and`, `or`, `not` и как они влияют на работу условий? Придумайте условие, которое должно быть истинным только тогда, когда число чётное и находится в диапазоне от 10 до 20, либо нечётное и больше 100. Запишите его в виде одного выражения.

8. Как в Python происходит вычисление арифметических выражений? Объясните приоритет операций и покажите, как скобки меняют порядок вычислений. Приведите 2–3 примера выражений с разными комбинациями операторов (+, -, *, /, //, %, **) и предскажите результат.

9. Чем отличаются операторы деления / и //? Напишите небольшой фрагмент кода, где для одного и того же набора чисел выводятся результаты обоих операторов, и поясните типы получаемых значений.

10. Как работают встроенные функции **min**, **max**, **sum**, **abs**, **round**? Для каждой функции приведите по одному примеру использования и укажите, какие типы аргументов она принимает и что возвращает.

11. Как правильно использовать функцию **range()** в цикле **for**? Опишите параметры **start**, **stop**, **step** и приведите 3 разных примера: диапазон с шагом 1, с отрицательным шагом и с произвольным шагом. Поясните, включается ли верхняя граница в диапазон.

12. Как импортировать и использовать модуль **math**? Перечислите 5 функций из этого модуля, которые чаще всего нужны для вычислений, и для каждой укажите, в какой ситуации она полезна. Приведите пример корректного вызова одной из них и типичную ошибку (например, отсутствие импорта).

13. Как вычислить квадратный корень в Python? Сравните три способа: **math.sqrt()**, возведение в степень ****0.5** и приближённые методы (например, метод Герона). Укажите плюсы и минусы каждого подхода.

14. Как обрабатывать «проблемные» случаи в вычислениях (деление на ноль, корень из отрицательного числа и т. п.)? Напишите фрагмент кода, который запрашивает у пользователя число и безопасно вычисляет его квадратный корень: если число отрицательное – выводит понятное сообщение, иначе – результат. Используйте условные операторы.

15. Почему результат **input()** нужно преобразовывать к нужному типу данных перед вычислениями? Приведите пример ошибки, которая возникает, если использовать строку в арифметическом выражении, и покажите правильный способ преобразования (через **int()**, **float()**) с обработкой возможной ошибки.

16. Как выбрать между ручным подсчётом в цикле и использованием встроенных функций? Придумайте задачу (например, сумма элементов, поиск минимума), напишите два варианта решения: один с циклом и аккумуляцией, другой – с `sum()`, `min()` и т. д. Сравните их по читаемости, длине кода и надёжности.

СПИСОК ЛИТЕРАТУРЫ

Книги

1. Лутц М. Изучаем Python, 4-е издание. – Пер. с англ. – СПб.: Символ-Плюс, 2011. – 1280 с.
 2. Златопольский Д.М. Основы программирования на языке Python. – М.: ДМК Пресс, 2017. – 284 с.
 3. Лутц М. Программирование на Python, том I, 4-е издание. – Пер. с англ. – СПб.: Символ-Плюс, 2011. – 992 с.
 4. Лутц М. Программирование на Python, том II, 4-е издание. – Пер. с англ. – СПб.: Символ-Плюс, 2011. – 992 с.
 5. Лучано Рамальо Python. К вершинам мастерства. – М.: ДМК Пресс, 2016. – 768 с.
 6. Свейгарт, Эл. Автоматизация рутинных задач с помощью Python: практическое руководство для начинающих. Пер. с англ. — М.: Вильямс, 2016. – 592 с.
 7. Федоров, Д. Ю. Программирование на языке высокого уровня Python : учебное пособие для прикладного бакалавриата / Д. Ю. Федоров. – 2-е изд., перераб. и доп. – Москва : Издательство Юрайт, 2019. – 161 с. – (Бакалавр. Прикладной курс). – ISBN 978-5-534-10971-9. – Текст: электронный // ЭБС Юрайт [сайт]. – URL: <https://urait.ru/bcode/437489> (дата обращения: 13.02.2020).
- Шелудько, В. М. Язык программирования высокого уровня Python. Функции, структуры данных, дополнительные модули: учебное пособие / В. М. Шелудько. – Ростов-на-Дону, Таганрог: Издательство Южного федерального университета, 2017. – 107 с. – ISBN 978-5-9275-2648-2. – Текст: электронный // Электронно-библиотечная система IPR BOOKS: [сайт]. – URL: <http://www.iprbookshop.ru/87530.html> (дата обращения: 13.02.2020). – Режим доступа: для авторизир. пользователей

Перечень Интернет-ресурсов

1. <https://github.com/enikolaev/IT-and-Programming> - репозиторий с исходными кодами для дисциплины.
2. <https://www.w3schools.com/python/default.asp> - W3C Tutorials.

