



Урок 12. Python: Рекурсия и базовый случай

Часть 1. Материалы ученика

Программирование на Python и C++ · 9 класс

Урок 12 из 34 · 45 минут · Модуль «Функции»

Уровень: начинающий программист; Python 3.

Программа: <https://schoolotzyv.ru/blog/python-cpp-9-klass-programma>

Чтобы сложить цифры числа, можно отделить последнюю цифру и поручить той же функции меньшую задачу. Но без остановки цепочка вызовов не завершится.

Результат урока: Построить рекурсивную сумму цифр, объяснить базовый случай и уменьшение аргумента, проверить границы контракта.

01. Проблемная ситуация

0–5 минут. Разложите 507 на последнюю цифру и оставшуюся часть, затем повторите для 50 и 5. Запишите цепочку до нуля и предположите сумму цифр.

02. Понятия и разобранный пример

5–14 минут: 4 минуты на понятия, 3 на пример, 2 на объяснение способа партнёру.

Рекурсивный вызов получает меньшую задачу

Рекурсия — вызов функции из её собственного определения. Базовый случай даёт ответ без нового вызова. Для суммы цифр неотрицательного n база $n==0$ возвращает 0. Шаг $n\%10 + \text{digit_sum}(n//10)$ отделяет последнюю цифру и решает задачу для оставшихся.

При $n>0$ целое $n//10$ строго меньше n и неотрицательно. Поэтому за конечное число шагов достигается ноль. Спуск создаёт ожидающие сложения, возврат завершает их в обратном порядке. return обязателен и в базе, и в рекурсивной ветке.

Контракт: целое n от 0 до 1000000 включительно. Отрицательные и большие значения отклоняются ValueError. Эта учебная граница даёт не более восьми одновременно активных вызовов для суммы цифр; рекурсия не предназначена здесь для произвольно длинных последовательностей.

Отступы набирайте обычными пробелами: 4 на уровень. При копировании с веб-страницы замените неразрывные пробелы обычными; полный код с обычными пробелами есть в PDF.

$\text{digit_sum}(507)=7+\text{digit_sum}(50)=7+0+\text{digit_sum}(5)=7+0+5+\text{digit_sum}(0)=12$. На спуске аргументы 507,50,5,0; на возврате результаты 0,5,5,12.

```
def digit_sum(n):
    if not 0 <= n <= 1000000:
        raise ValueError("range")
    if n == 0:
        return 0
    return n % 10 + digit_sum(n // 10)
```

```
for n in [507,0,9,1000,1000000]:
    print(n, digit_sum(n))
```

Спуск	$507 \rightarrow 50 \rightarrow 5 \rightarrow 0$
База	$\text{digit_sum}(0) = 0$
Возврат	$0 \rightarrow 5 \rightarrow 5 \rightarrow 12$
Уменьшение	$n//10 < n$ при $n>0$

Стек вызовов: сложение завершается после возврата ответа меньшей задачи.



03. Практика 1. Проверяем материал

Трассируем спуск и возврат

14–25 минут. Для каждого числа выпишите аргументы вызовов и результаты возврата; затем запустите пример.

n	Сумма цифр; число вызовов
507	12; 4
0; 9	0; 1 и 9; 2 соответственно
1000	1; 5
1000000	1; 8

2 минуты — прогноз; 4 — трассировка и запуск; 3 — тесты -1 и 1000001, каждый отдельным запуском; 2 — вывод об остановке. Критерии: верные суммы, база, уменьшение, отказ вне диапазона.

04. Практика 2. Перенос способа

Считаем число цифр

25–33 минуты. Напишите `digits(n)`, возвращающую число десятичных цифр. Ноль имеет одну цифру. За 2 минуты выберите базу, за 4 реализуйте, за 2 проверьте 0,9,10,507,1000000. Критерии: база $n < 10$, шаг $1 + \text{digits}(n//10)$, прежний диапазон.

05. Обсуждение и границы вывода

33–40 минут. За 2 минуты сравните рекурсию с циклом по цифрам, за 3 объясните, где хранится отложенная работа, за 2 выберите подход для очень длинного числа в строке.

06. Индивидуальная проверка

40–45 минут. Для 204 запишите спуск `digit_sum`, сумму, число вызовов и `digits(204)`. Четыре балла.



Часть 2. Ответы и материалы учителя

Практика 1: разбор и критерии

Для 1000: 1000→100→10→1→0, возврат 0→1→1→1→1. Для 0 функция вызывается один раз. -1 и 1000001 вызывают ValueError до рекурсивного шага. Замена $n//10$ на n сохраняет аргумент и не обеспечивает остановку для положительного n .

Практика 2: разбор и критерии

Ответы 1,1,2,3,7. База отличается от суммы: однозначное число уже содержит одну цифру. Полная программа:

```
def digits(n):
    if not 0 <= n <= 1000000:
        raise ValueError("range")
    if n < 10:
        return 1
    return 1 + digits(n // 10)
```

```
for n in [0,9,10,507,1000000]:
    print(n, digits(n))
```

Обсуждение: возможный вывод

Цикл может накапливать сумму без цепочки вызовов; рекурсия наглядно отражает разложение задачи, но использует стек. Для длинной строки цифр удобен цикл по символам с проверкой входа. Наш код имеет малую глубину благодаря контракту, а не благодаря бесконечному запасу стека.

Индивидуальная проверка: ответ и критерии

204→20→2→0; 6; 4 вызова; 3 цифры. По одному баллу за пункт.

Учителю: подготовка и план на 45 минут

Подготовьте компьютер, редактор, папку с правом записи, распечатку ученической части PDF, бумагу и ручку. Python 3.11+ и IDLE или редактор; запуск `python lesson.py` (либо `python3 lesson.py`). Внешние библиотеки не нужны. Установку и проверку запуска учитель выполняет заранее. Все программы, данные и ожидаемые ответы даны; регистрация детей и интернет на занятии не нужны.

Уровень — начинающий программист, 9 класс. Предпосылки: функции и `return`, целочисленное деление и остаток. В паре автор прогнозирует и вводит код, проверяющий ведёт журнал, затем роли меняются. Без компьютера выполните трассировку полного кода на бумаге и сопоставьте с готовыми результатами; учитель демонстрирует подготовленный запуск. Бумажный вариант не считается фактическим запуском ученика. Перед практикой учитель выдаёт готовый файл разобранного примера, чтобы дети тратили время на изменение и проверку, а не длинное перепечатывание. Матрица программы предназначена учителю.

Минуты	Работа	Результат
0–5	Проблема	Вопрос и первое решение
5–14	Понятия и пример	Способ проверки
14–25	Практика 1	Готовый продукт и проверка
25–33	Практика 2	Перенос способа
33–40	Обсуждение	Аргумент и ограничение
40–45	Индивидуальная проверка	Самостоятельный ответ



07. Соответствие исходной программе

Пункт программы	Действие ученика	Проверяемый результат
Функции. Логические функции. Рекурсивные функции.	Трассируем спуск и возврат	Построить рекурсивную сумму цифр, объяснить базовый случай и уменьшение аргумента, проверить границы контракта.
Структурирование решения и отладка	Считаем число цифр	Исходник, журнал обычных и граничных тестов; индивидуальный ответ

Источники и права на изображения

Исходная программа курса

<https://schoolotzyv.ru/blog/python-cpp-9-klass-programma>

Python: функции и управление выполнением

<https://docs.python.org/3/tutorial/controlflow.html>

Тексты, учебные данные, реплики, программы и схемы подготовлены для уроков портала «Школы России». Числовые серии — учебные модели, не результаты проведённых классом опытов. Схемы авторские, сторонние фотографии не используются. Методические ссылки приведены для учителя; ученик получает все материалы здесь. Проверка первичных источников: 18.10.2026.