



Урок 7. Python: Вложенные ветвления

Часть 1. Материалы ученика

Программирование на Python и C++ · 8 класс

Урок 7 из 33 · 45 минут · Модуль «Ветвления. Вложенные условные операторы»

Уровень: начинающий программист; Python 3.

Программа: <https://schoolotzyv.ru/blog/python-cpp-8-klass-programma>

Точка находится справа от вертикальной оси. Этого мало, чтобы назвать четверть: нужно ещё узнать, выше она или ниже горизонтальной оси. Построим дерево из двух проверок.

Результат урока: Прочитать вложенный if, проследить единственный путь решения и проверить все ветви, включая особый случай.

01. Проблемная ситуация

0–5 минут. Отметьте точки $(2,3)$, $(2,-3)$, $(0,4)$. Можно ли всем назначить четверть по одному условию $x > 0$? Запишите дополнительный вопрос и выделите точку, для которой слово «четверть» не подходит. Координаты условные, ось y здесь математическая — вверх.

02. Понятия и разобранный пример

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

Внешний выбор задаёт контекст внутреннего

Вложенный if выполняется только внутри выбранной внешней ветви. После проверки $x > 0$ внутренний $y > 0$ различает I и IV четверти; при $x < 0$ — II и III. Отступы в Python определяют принадлежность else. Один сдвиг может изменить смысл при синтаксически корректной программе.

Особые случаи полезно отделить до основного дерева. Если $x=0$ или $y=0$, точка на оси и возвращаем axis. После этого известно, что оба числа ненулевые. Внутри дерева не приходится заново проверять равенство нулю.

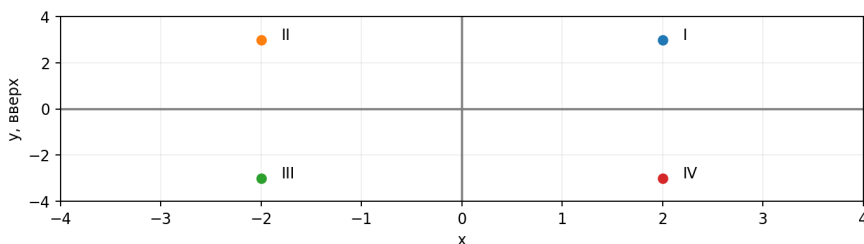
Для проверки дерева нужен пример каждого конечного результата. Пять одинаковых положительных точек не проверяют отрицательные ветви. Трассировка записывает ответы на вопросы, а не только итог: $(-2,3) \rightarrow$ не ось $\rightarrow x > 0$ ложно $\rightarrow y > 0$ истинно $\rightarrow$ II.

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

Программа выводит I,II,III,IV,axis на пяти строках. Для $(2,-3)$ внешняя положительная ветвь выбрана, внутренняя $y > 0$ ложна, поэтому IV. Для $(0,4)$ срабатывает ранний return и остальные сравнения уже не выполняются.

```
def quadrant(x, y):
    """Вернуть I/II/III/IV или axis для целых x,y."""
    if x == 0 or y == 0:
        return "axis"
    if x > 0:
        if y > 0:
            return "I"
        else:
            return "IV"
    else:
        if y > 0:
            return "II"
        else:
            return "III"

for x, y in [(2,3), (-2,3), (-2,-3), (2,-3), (0,4)]:
    print(quadrant(x, y))
```



Четверти математической плоскости: I (+,+), II (-,+), III (-,-), IV (+,-); нулевые координаты — оси.

03. Практика 1. Проверяем материал

Покрываем все листья дерева

14–25 минут. По исходнику из примера составьте протокол семи точек таблицы. Затем временно замените ответ IV на II и найдите минимальный тест, обнаруживающий ошибку.

Точка	Ожидаемая ветвь
(1,1)	I
(-1,1)	II
(-1,-1)	III
(1,-1)	IV
(0,0); (3,0); (0,-3)	axis

2 минуты — прогноз путей; 4 — ввод программы; 3 — семь запусков и поиск ошибки; 2 — восстановление и повторная проверка. Продукт: таблица «точка, ответы проверок, результат». Критерии: все четыре четверти, оси, контрпример изменённой ветви, правильные отступы.

04. Практика 2. Перенос способа

Дерево для сравнения двух чисел

25–33 минуты. Постройте вложенное решение `compare(a,b)`: при равенстве вернуть `equal`; иначе, если $a > b$, вернуть `first`, иначе `second`. Сначала внешний вопрос $a == b$, затем внутренний $a > b$ в ветви неравенства. Целые от -100 до 100 . За 2 минуты дерево, за 4 код, за 2 тесты $(4,4), (5,2), (-3,1), (0,-1)$. Критерии: три результата, верная вложенность, отрицательные входы.

05. Обсуждение и границы вывода

33–40 минут. Как выбрать между вложенным `if` и цепочкой `elif`? Сравните четверти и три результата сравнения. Одна пара защищает дерево, другая — плоскую цепочку. Согласуйте, какая запись легче показывает зависимость проверок и не теряет случай равенства.

06. Индивидуальная проверка

40–45 минут. Назовите результат `quadrant(-5,-2)`, последовательность двух проверок знаков после исключения осей, результат `compare(-4,-1)` и один тест для оси. По баллу за четыре элемента.



Часть 2. Ответы и материалы учителя

Практика 1: разбор и критерии

Для точек с нулём достаточно первой проверки. Ошибочную замену обнаруживает (1,−1): ожидается IV, получается II. (−1,1) не обнаруживает ошибку, потому что проходит другую ветвь. После восстановления все семь тестов соответствуют таблице.

Практика 2: разбор и критерии

equal,first,second,first. Внутреннее сравнение выполняется только после установленного неравенства. Можно записать это плоской цепочкой elif, но здесь намеренно тренируем путь вложенного дерева.

```
def compare(a, b):
    """Сравнить целые a,b в −100...100."""
    if a == b:
        return "equal"
    else:
        if a > b:
            return "first"
        else:
            return "second"

for a, b in [(4,4), (5,2), (-3,1), (0,-1)]:
    print(compare(a, b))
```

Обсуждение: возможный вывод

Вложенность естественна, когда второй вопрос зависит от первого, как знак y после знака x . Для независимого перечисления результатов часто короче elif. Оба варианта могут быть верны; критерий — ясность путей и полное покрытие случаев, а не максимальная глубина дерева.

Индивидуальная проверка: ответ и критерии

III; $x > 0$ False, затем $y > 0$ False; second; например (2,0) → axis. 4 балла. Координаты рисунка SVG направлены иначе по y , но здесь используется обычная математическая плоскость.

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, 8 класс. Предпосылки: if/else, отступы, знаки чисел и координатная плоскость. 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
Ветвления. Вложенные условные операторы	Разбирает пример и создаёт две подпрограммы	Исходники, прогнозы и протокол проверки
Составление и отладка программ	Сравнивает фактический результат с контрактом	Объяснение границ и самостоятельный ответ

Источники и права на изображения

Исходная программа курса

<https://schoolotzyv.ru/blog/python-cpp-8-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.