



# Урок 27. Python: Двоичный поиск по ответу

## Часть 1. Материалы ученика

Программирование на Python и C++ · 9 класс

Урок 27 из 34 · 45 минут · Модуль «Поиск»

Уровень: начинающий программист; Python 3.

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

Два учебных станка делают детали параллельно. Нужно найти первый момент, когда готово достаточно деталей, не перебирая каждую секунду.

Результат урока: Найти минимальное целое время по монотонной проверке и подтвердить ответ двумя соседними значениями.

### 01. Проблемная ситуация

0–5 минут. Станок А выпускает деталь каждые 3 секунды, Б — каждые 5 секунд. Оба начинают в момент 0; готовой детали в этот момент нет. Можно ли получить 5 деталей к 9-й секунде? К 10-й? Запишите два расчёта.

### 02. Понятия и разобранный пример

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

#### От ответа к проверке

Пусть  $ok(t)$  означает  $t//a+t//b \geq need$ . Это число готовых деталей к моменту  $t$  включительно. Незавершённая деталь не считается. Если  $ok(t)$  истинно, оно останется истинным для любого большего  $t$ : счётчик не уменьшается. Ищем первую истину, а не элемент заранее записанного массива.

Контракт:  $a$  и  $b$  — целые от 1 до 1000,  $need$  — целое от 0 до 1000000. Пустой заказ  $need=0$  требует 0 секунд. Иначе  $lo=0$  заведомо недостаточно,  $hi=\min(a,b)*need$  достаточно даже одному быстрому станку. В цикле сохраняем  $ok(lo)=False$ ,  $ok(hi)=True$ . Когда  $hi-lo=1$ ,  $hi$  — первый подходящий момент.

Если середина подходит, заменяем  $hi$ ; иначе  $lo$ . Расстояние уменьшается. Число проверок растёт логарифмически от исходной верхней границы; каждая проверка здесь выполняет два деления. Метод неприменим без доказательства монотонности. Это модель станков без задержек, поломок и совместных ресурсов.

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

$a=3$ ,  $b=5$ ,  $need=5$ :  $lo=0$ ,  $hi=15$ . Середина 7 даёт  $2+1=3$ ,  $lo=7$ ; 11 даёт  $3+2=5$ ,  $hi=11$ ; 9 даёт  $3+1=4$ ,  $lo=9$ ; 10 даёт  $3+2=5$ ,  $hi=10$ . На 9-й секунде мало, на 10-й достаточно: ответ 10.

```
def first_time(a, b, need):
    if not (1 <= a <= 1000 and 1 <= b <= 1000
            and 0 <= need <= 1000000):
        raise ValueError("contract")
    if need == 0:
        return 0
    lo, hi = 0, min(a, b) * need
    while hi - lo > 1:
        mid = (lo + hi) // 2
        if mid // a + mid // b >= need:
            hi = mid
        else:
            lo = mid
    return hi

print(first_time(3, 5, 5))
```



|                 |                             |
|-----------------|-----------------------------|
| t, секунды      | 7   11   9   10             |
| Готово, деталей | 3   5   4   5               |
| После проверки  | lo=7   hi=11   lo=9   hi=10 |
| Граница         | 9: мало; 10: достаточно     |

Трасса поиска минимального времени для заказа из пяти деталей.

### 03. Практика 1. Проверяем материал

#### Проверяем минимальность

14–25 минут. Учитель выдаёт файл примера. Все параметры — учебная модель.

| a,b,need       | Что проверить           |
|----------------|-------------------------|
| 3,5,5          | Основной заказ          |
| 2,2,3          | Одновременный выпуск    |
| 7,9,1          | Первая деталь           |
| 3,5,0          | Пустой заказ            |
| 1,1000,1000000 | Большой допустимый вход |

2 минуты — прогноз первых четырёх ответов; 4 — пять запусков; 3 — для каждого положительного ответа T вычислите число деталей при T и T-1; 2 — оформите журнал. Критерии: точное время, достаточность, минимальность и отдельный случай пустого заказа.

### 04. Практика 2. Перенос способа

#### Меняем производительность

25–33 минуты. Третий станок выпускает деталь каждые 4 секунды, первые два имеют периоды 3 и 5. Нужно 6 деталей. За 2 минуты выпишите новый  $ok(t)$ , за 4 найдите ответ на бумаге и измените копию функции, за 2 объясните, почему прежняя верхняя граница остаётся достаточной. В изменённой функции с положительно и не больше 1000.

### 05. Обсуждение и границы вывода

33–40 минут. Сравните требования «не менее 5» и «ровно 5» для двух станков с периодом 2. За 2 минуты выпишите счётчик при  $t=0\dots6$ , за 3 оцените монотонность условий, за 2 сформулируйте ограничение метода.

### 06. Индивидуальная проверка

40–45 минут. Для  $a=4, b=6, need=4$  найдите T, счётчики при T и T-1 и объясните два свойства границ lo/hi. Четыре балла.



## Часть 2. Ответы и материалы учителя

### Практика 1: разбор и критерии

Ответы: 10, 4, 7, 0, 999001. Для последнего  $T=999001$ :  $999001+999=1000000$ ; при 999000:  $999000+999=999999$ . Для 2,2,3 при 4 готовы 4 детали, при 3 только 2; лишняя деталь допустима. При  $need=0$  отрицательное время не проверяем.

### Практика 2: разбор и критерии

Используйте  $t//a+t//b+t//c \geq need$ ; в заголовок добавьте  $c$ , в проверку контракта  $1 \leq c \leq 1000$ . Для 3,5,4,6 ответ 9:  $3+1+2=6$ ; при 8:  $2+1+2=5$ . Верхняя граница  $\min(a,b)*need$  всё ещё обеспечивается одним из первых двух станков; можно уменьшить её до  $\min(a,b,c)*need$ . Монотонность сохраняется.

### Обсуждение: возможный вывод

Счётчик: 0,0,2,2,4,4,6. Ровно 5 не возникает вообще; для других целей равенство может стать истинным и затем ложным. Условие  $\geq 5$  меняется только от лжи к истине. Двоичный поиск требует такой границы, а не произвольного условия.

### Индивидуальная проверка: ответ и критерии

$T=12$ ; при 12:  $3+2=5$ , при 11:  $2+1=3$ .  $lo$  недостаточно,  $hi$  достаточно; при расстоянии 1 другого целого кандидата нет. По баллу за  $T$ , два счётчика вместе, смысл  $lo/hi$ , условие остановки.

### 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: целочисленное деление, функции и двоичный поиск границы. 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                                                                                |
|-------------------------------------|---------------------------|------------------------------------------------------------------------------------------------------|
| Алгоритмы поиска                    | Проверяем минимальность   | Найти минимальное целое время по монотонной проверке и подтвердить ответ двумя соседними значениями. |
| Самостоятельная проверка результата | Меняем производительность | Обоснованный ответ и исправление ошибки                                                              |

### Источники и права на изображения

Исходная программа курса

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

Python: принцип поиска границы

<https://docs.python.org/3/library/bisect.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.