



Урок 25. Python: Линейный и двоичный поиск

Часть 1. Материалы ученика

Программирование на Python и C++ · 9 класс

Урок 25 из 34 · 45 минут · Модуль «Поиск»

Уровень: начинающий программист; Python 3.

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

Двоичный поиск может исключить половину ряда только тогда, когда значения упорядочены. Быстрый алгоритм с нарушенной предпосылкой может уверенно пропустить существующий элемент.

Результат урока: Реализовать линейный и двоичный поиск первого вхождения, проверить отсутствие и повторы и объяснить цену предварительной сортировки.

01. Проблемная ситуация

0–5 минут. В ряду [8,1,5] ищем 8. Средний элемент 1 меньше 8. Можно ли отбросить левую половину? Укажите, где находится ответ, и какое условие для двоичного поиска нарушено.

02. Понятия и разобранный пример

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

Одинаковый контракт, разные предпосылки

Линейный поиск проверяет элементы слева направо и возвращает первый подходящий индекс или -1. Он работает на произвольном порядке. Двоичный вариант здесь ищет левую границу значений не меньше x , затем проверяет равенство. Его предпосылка — список по неубыванию.

В полуинтервале $[lo, hi)$ остаются кандидаты на границу. При $a[mid] < x$ ответ правее mid , поэтому $lo = mid + 1$; иначе $hi = mid$. После $lo == hi$ это позиция первого элемента $\geq x$ или n . Проверка $lo < len(a)$ выполняется раньше чтения $a[lo]$, что защищает пустоту и запрос выше максимума.

Линейный поиск требует до n проверок значений; двоичный — $O(\log n)$ итераций на уже отсортированном списке. Проверка упорядоченности сама стоит $O(n)$, поэтому она не включена внутрь функции: предпосылку обеспечивает подготовка данных. Для одного запроса сортировка может стоить дороже простого просмотра. Контракт урока: до 100 целых и целый запрос; индексы начинаются с 0.

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

Для $[1, 3, 3, 7, 9]$, $x=3$: $[lo, hi)=[0, 5)$, $mid=2$, $a[2]=3 \Rightarrow hi=2$; $mid=1$, $a[1]=3 \Rightarrow hi=1$; $mid=0$, $a[0]=1 \Rightarrow lo=1$. Граница 1, равенство подтверждено. Линейный поиск тоже даёт 1. Для $x=4$ граница 3, но $a[3]=7$, поэтому ответ -1.

```
def linear(a, x):
    for i, value in enumerate(a):
        if value == x:
            return i
    return -1

def binary(a, x):
    lo, hi = 0, len(a)
    while lo < hi:
        mid = (lo + hi) // 2
        if a[mid] < x:
            lo = mid + 1
        else:
            hi = mid
    if lo < len(a) and a[lo] == x:
        return lo
    return -1

a = [1, 3, 3, 7, 9]
for x in [3, 4, 1, 9, 10]:
    print(x, linear(a,x), binary(a,x))
```



Ищем 3	[1,3,3,7,9]; [0,5)
mid=2, значение 3	hi=2 → [0,2)
mid=1, значение 3	hi=1 → [0,1)
mid=0, значение 1	lo=1; ответ 1

Поиск левой границы возвращает первое, а не произвольное вхождение.

03. Практика 1. Проверяем материал

Сравниваем результаты и пробы

14–25 минут. Для каждой строки сначала предскажите индекс; затем выполните обе функции. Все двоичные входы таблицы отсортированы.

Вход и запрос	Оба результата
[1,3,3,7,9]; x=3,4	1; -1
Тот же ряд; x=1,9,10	0;4;-1
[]; x=0	-1
[5]; x=5,4	0;-1

2 минуты — прогноз, 4 — восемь запросов, 3 — трасса lo,hi,mid для x=10, 2 — журнал. Критерии: первый индекс при повторях, -1 при отсутствии, нет a[n], явно записана предпосылка.

04. Практика 2. Перенос способа

Сохраняем исходные номера

25–33 минуты. Исходные номера относятся к [8,1,5,1]. Нужно найти первое исходное вхождение 1, не подменяя индекс отсортированным. За 2 минуты подпишите исходные номера, за 4 запустите готовый вариант ниже, за 2 проверьте запросы 8 и 6. Критерии: ответы 1,0,-1 и объяснение хранения номера.

```
from bisect import bisect_left

a = [8, 1, 5, 1]
pairs = sorted((value, i) for i, value in enumerate(a))
values = [value for value, i in pairs]
for x in [1, 8, 6]:
    pos = bisect_left(values, x)
    answer = pairs[pos][1] if pos < len(pairs) and values[pos] == x else -1
    print(x, answer)
```

05. Обсуждение и границы вывода

33–40 минут. «Двоичный поиск всегда быстрее, потому что логарифм меньше n». За 2 минуты назовите скрытую подготовку, за 3 сравните один и тысячу запросов к неизменному списку, за 2 сформулируйте выбор.

06. Индивидуальная проверка

40–45 минут. В [2,2,5,8] найдите первые индексы для 2 и 8, ответ для 3 и объясните, почему [8,2,5] не является допустимым входом binary. Четыре балла.



Часть 2. Ответы и материалы учителя

Практика 1: разбор и критерии

Для $x=10$ интервалы $[0,5) \rightarrow [3,5) \rightarrow [5,5)$, затем $lo==len(a)$, чтения $a[5]$ нет. Для $[]$ цикл не запускается и короткое замыкание не даёт читать $a[0]$. На неупорядоченном $[8,1,5]$ линейный поиск 8 вернёт 0, двоичный может вернуть -1; это нарушение предпосылки, не доказательство отсутствия.

Практика 2: разбор и критерии

Пары после сортировки: $(1,1),(1,3),(5,2),(8,0)$. Для 1 первая пара несёт исходный индекс 1; для 8 индекс 0; 6 отсутствует. Сортировка пар при равных значениях упорядочивает исходные индексы, поэтому возвращается минимальный исходный номер. Для одного запроса проще $linear(a,x)$; подготовка полезна для многих запросов.

Обсуждение: возможный вывод

На неупорядоченных данных сначала нужна сортировка и, возможно, сохранение исходных индексов. Для одного запроса линейный просмотр не требует этой работы. Для многих запросов подготовленная отсортированная копия может окупиться. После изменения данных надо восстановить предпосылку; одну оценку цикла нельзя выдавать за стоимость всего процесса.

Индивидуальная проверка: ответ и критерии

0;3;-1; последний ряд не отсортирован по неубыванию, поэтому исключение половины не обосновано. По одному баллу.

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: списки, индексы, `while` и упорядоченность. 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	Сравниваем результаты и пробы	Реализовать линейный и двоичный поиск первого вхождения, проверить отсутствие и повторы и объяснить цену предварительной сортировки.
Структурирование решения и отладка	Сохраняем исходные номера	Исходник, журнал обычных и граничных тестов, самостоятельное объяснение

Источники и права на изображения

Исходная программа курса

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

Python: поиск позиции в отсортированной последовательности

<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.