



Урок 16. Python: Сортировка выбором

Часть 1. Материалы ученика

Программирование на Python и C++ · 9 класс

Урок 16 из 34 · 45 минут · Модуль «Сортировка»

Уровень: начинающий программист; Python 3.

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

Можно искать минимум оставшейся части и ставить его на очередное место. Такой способ делает мало обменов, но всё равно просматривает много пар.

Результат урока: Реализовать сортировку выбором, доказать правильность левого префикса, проверить число сравнений и показать ограничение устойчивости.

01. Проблемная ситуация

0–5 минут. В [4,2,3,1] найдите минимум всей серии, поменяйте его с первым элементом. Затем найдите минимум оставшейся части. Какие позиции уже можно не трогать?

02. Понятия и разобранный пример

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

Минимум неотсортированной части

На шаге i ищем индекс $best$ минимального элемента от i до конца. После поиска выполняем один обмен $a[i], a[best]$, если $best \neq i$. Первые $i+1$ элементов после этого шага — наименьшие элементы в нужном порядке. В отличие от пузырька, здесь строится левый готовый префикс.

Функция работает с копией, считает сравнения значений и реальные обмены разных позиций.

Контракт: список до 100 целых от -1000 до 1000. Для любой серии длины n число сравнений равно $n(n-1)/2$, даже если она уже отсортирована. Обменов не больше $n-1$.

Обычная сортировка выбором с обменом неустойчива: может переставить записи с равными ключами.

Пример [2A,2B,1C]: обмен первого элемента с минимумом даёт [1C,2B,2A]. Числа отсортированы, но относительный порядок A,B изменился.

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

[4,2,3,1]: шаг $i=0$ выбирает индекс 3 и получает [1,2,3,4]; $i=1$ и $i=2$ находят минимум уже на месте. Всего $3+2+1=6$ сравнений и один обмен.

```
def selection(data):
    a = data.copy()
    comparisons = swaps = 0
    for i in range(len(a)-1):
        best = i
        for j in range(i+1, len(a)):
            comparisons += 1
            if a[j] < a[best]:
                best = j
        if best != i:
            a[i], a[best] = a[best], a[i]
            swaps += 1
    return a, comparisons, swaps

for a in [[4,2,3,1],[1],[7],[1,2,3],[3,2,1]]:
    print(a, selection(a))
```



Начало	4 2 3 1
Шаг i=0	1 2 3 4; best=3
Шаг i=1	1 2 3 4; best=1
Шаг i=2	1 2 3 4; best=2

Левый готовый префикс растёт; минимум ищется только в оставшейся части.

03. Практика 1. Проверяем материал

Проверяем левый префикс

14–25 минут. Запишите best и готовый префикс после каждого внешнего шага для таблицы, затем запустите программу.

Вход	Результат; сравнения; обмены
[4,2,3,1]	[1,2,3,4]; 6; 1
[1]; [7]	[]; 0; 0 и [7]; 0; 0
[1,2,3]; [3,2,1]	[1,2,3]; 3; 0 и [1,2,3]; 3; 1
[2,2,1]	[1,2,2]; 3; 1

2 минуты — прогноз; 4 — запуск; 3 — сопоставление с sorted и формулой сравнений; 2 — объяснение неизменности входа. Критерии: порядок, кратности, исходник, различие сравнения и обмена.

04. Практика 2. Перенос способа

Исследуем порядок равных ключей

25–33 минуты. Даны записи (ключ,метка): [(2,"A"),(2,"B"),(1,"C")]. За 2 минуты предскажите порядок меток, за 4 запустите код ниже, за 2 сравните с исходным порядком A перед B. Критерии: сравнение только ключа [0], показ конкретного обмена, вывод об устойчивости.

```
def selection_records(data):
    a = data.copy()
    for i in range(len(a)-1):
        best = i
        for j in range(i+1, len(a)):
            if a[j][0] < a[best][0]:
                best = j
        a[i], a[best] = a[best], a[i]
    return a

print(selection_records([(2, "A"), (2, "B"), (1, "C")]))
```

05. Обсуждение и границы вывода

33–40 минут. За 2 минуты сопоставьте пузырьёк и выбор на [1,2,3], за 3 обсудите сравнения и обмены отдельно, за 2 выберите способ для записей, где надо сохранить порядок равных ключей.

06. Индивидуальная проверка

40–45 минут. Для [3,1,2] укажите результат selection, сравнения, обмены и максимальное число сравнений при n=100. Четыре балла.



Часть 2. Ответы и материалы учителя

Практика 1: разбор и критерии

Для отсортированного [1,2,3] всё равно 3 сравнения, хотя обменов 0. Для [3,2,1] один обмен крайних элементов завершает числовую перестановку, но оставшийся поиск всё равно выполняется. Счётчик comparisons увеличивается на каждой паре j,best, swaps — только при best!=i.

Практика 2: разбор и критерии

Результат [(1,"C"),(2,"B"),(2,"A")]. Обмен A с C переносит A за B, хотя два ключа равны. Упорядоченность по ключу соблюдена, устойчивость нарушена. Сравнение целых кортежей вместо [0] добавило бы сортировку по меткам и изменило контракт.

Обсуждение: возможный вывод

Пузырёк с ранним выходом: 2 сравнения, 0 обменов; выбор: 3 сравнения, 0 обменов. Малое число обменов не означает малое число сравнений. Для устойчивой сортировки записей используйте sorted(records,key=lambda p:p[0]); относительный порядок равных ключей сохраняется. Наш выбор с обменом этого не гарантирует.

Индивидуальная проверка: ответ и критерии

[1,2,3]; 3; 2; 4950. Каждый пункт — один балл; последняя величина одинакова для всех входов длины 100.

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: минимум списка, индексы, вложенные циклы и обмен. 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 сортировки	Проверяем левый префикс	Реализовать сортировку выбором, доказать правильность левого префикса, проверить число сравнений и показать ограничение устойчивости.
Структурирование решения и отладка	Исследуем порядок равных ключей	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.