

Алгоритмы и структуры данных

Узнаем, что такое алгоритм. Изучим его свойства.
Познакомимся с понятием алгоритмической сложности,
сортировками, рекурсией и структурами данных





Регламент урока



Длительность



Регламент сдачи практических заданий



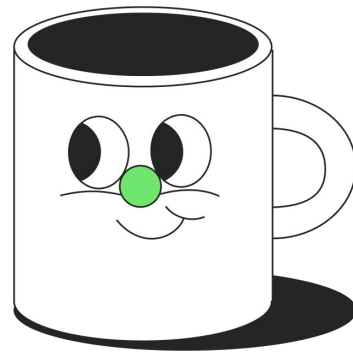
Когда и где будет доступна запись



Каналы коммуникации



Делаем ли перерыв?





Получите максимум от лекции



Задавай вопросы.
Для этого подними руку



Отключи микрофон



Делай заметки



Будь с нами! Отложи телефон, домашние дела, мессенджеры и возьми максимум для себя



Будь готов отвечать на вопросы и выполнять задания



Запаркуем



Что будет на уроке сегодня

- 📌 Что такое алгоритм?
- 📌 Свойства алгоритмов
- 📌 Алгоритмическая сложность
- 📌 Сортировки
- 📌 Рекурсия
- 📌 Структуры данных
- 📌 Практическая часть





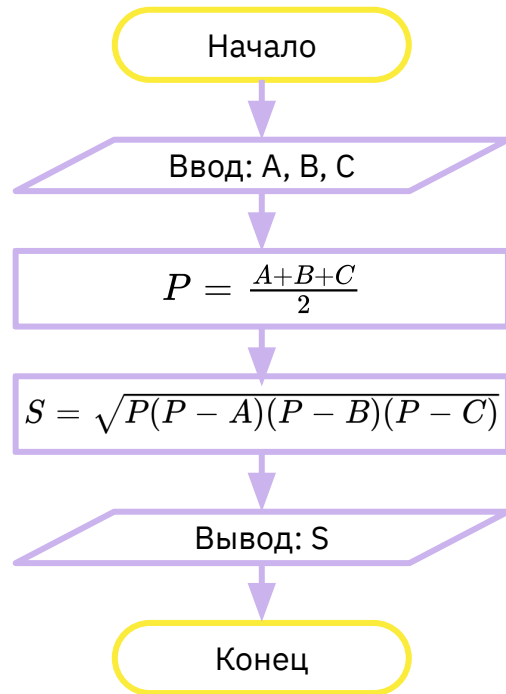
Что такое алгоритм?

Алгоритм — это заданный набор инструкций (шагов) для решения поставленной задачи.

Шаг алгоритма — это каждое отдельное действие алгоритма.

Применение:

Везде где требуется выполнять определённую последовательность действий (сортировка клиентов онлайн-сервиса по сумме покупок за заданный период; расчёт траектории движения околоземного спутника и т. д.)





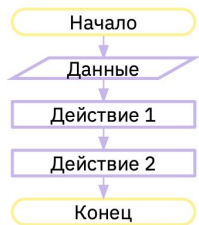
Свойства алгоритмов

- Дискретность
- Результативность
- Детерминированность
- Массовость
- Понятность
- Конечность

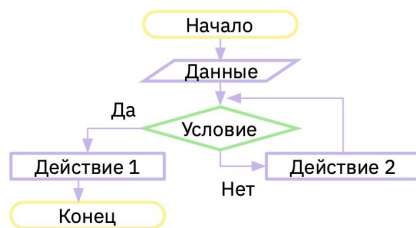


Виды алгоритмов

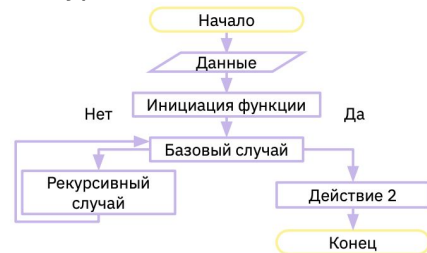
Линейный



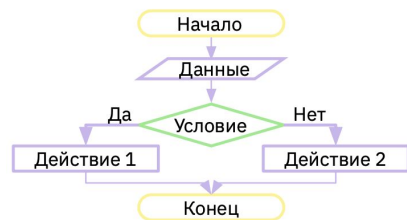
Циклический



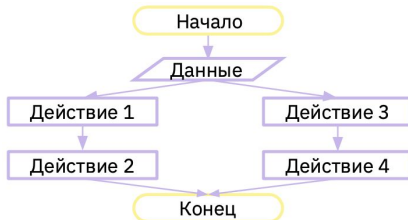
Рекурсивный*



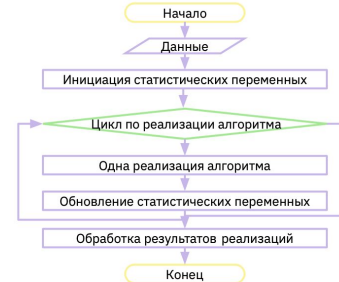
Разветвляющийся



Распараллеливающийся



Вероятностный**

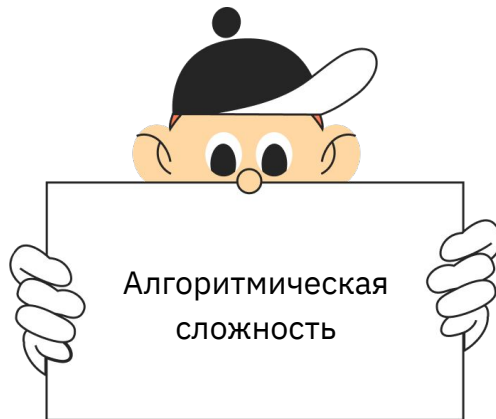


* расчёт числа Фибоначчи, расчёт факториала числа, ** алгоритм Монте-Карло



Алгоритмическая сложность

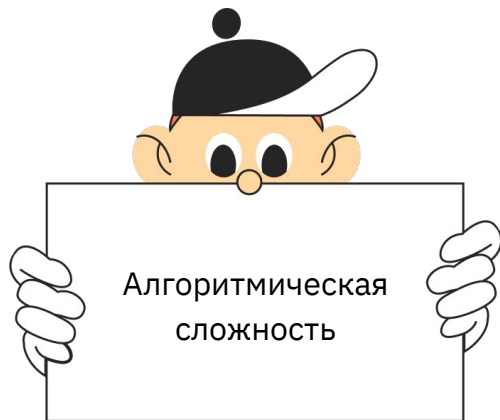
Сложность алгоритмов обычно оценивают **по времени выполнения...**





Алгоритмическая сложность

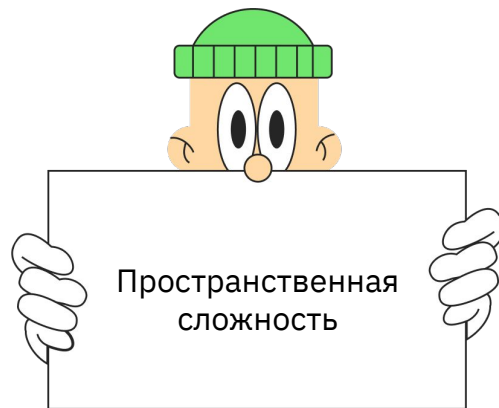
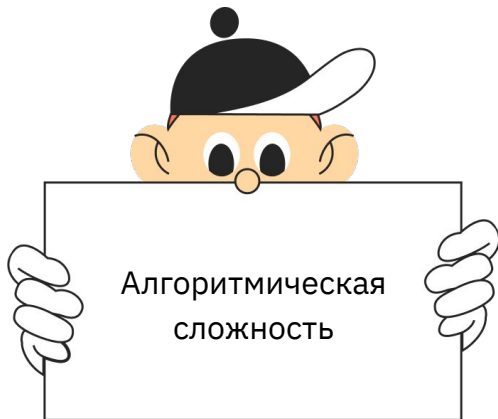
Сложность алгоритмов обычно оценивают **по времени выполнения** или **по используемой памяти**.





Алгоритмическая сложность

Сложность алгоритмов обычно оценивают **по времени выполнения** или **по используемой памяти**.



Бинарный поиск (метод деления пополам, дихотомия) — классический алгоритм поиска элемента в отсортированном массиве (векторе), использующий дробление массива на половины.

Алгоритм сортировки — это алгоритм для упорядочивания элементов в массиве.

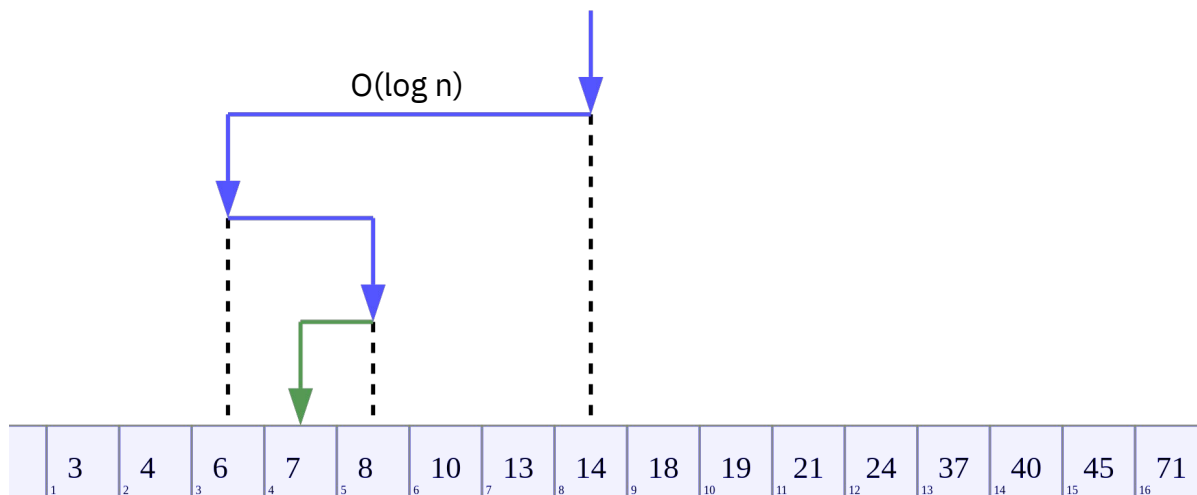


Бинарный поиск

Бинарный поиск — это поиск в структурах данных.

Используется при нахождении приближённого решения уравнений (метод бисекции).

Применяется при нахождении экстремума функции.



[Источник](#)



Алгоритмическая сложность сортировок

Алгоритм	Структура данных	Временная сложность			Вспомогательные данные
		Лучшее	В среднем	В худшем	В худшем
Быстрая сортировка	Массив	$O(n \log(n))$	$O(n \log(n))$	$O(n^2)$	$O(n)$
Сортировка слиянием	Массив	$O(n \log(n))$	$O(n \log(n))$	$O(n \log(n))$	$O(n)$
Пирамидальная сортировка	Массив	$O(n \log(n))$	$O(n \log(n))$	$O(n \log(n))$	$O(1)$
Пузырьковая сортировка	Массив	$O(n)$	$O(n^2)$	$O(n^2)$	$O(1)$
Сортировка вставками	Массив	$O(n)$	$O(n^2)$	$O(n^2)$	$O(1)$
Сортировка выбором	Массив	$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(1)$
Блочная сортировка	Массив	$O(n+k)$	$O(n+k)$	$O(n^2)$	$O(nk)$
Поразрядная сортировка	Массив	$O(nk)$	$O(nk)$	$O(nk)$	$O(n+k)$



Рекурсия

Рекурсия — определение, описание, изображение какого-либо объекта или процесса внутри самого этого объекта или процесса, то есть ситуация, когда объект является частью самого себя.

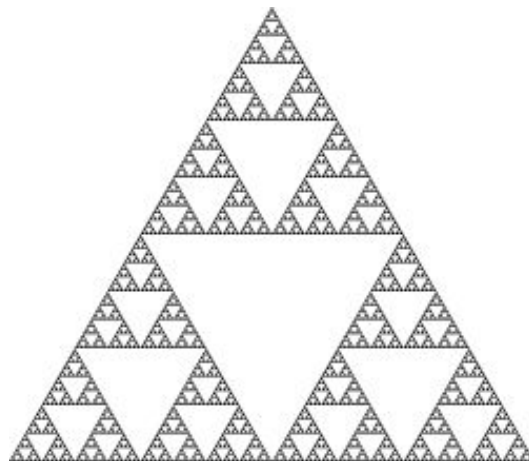
Примеры:

$$F_0 = 0, \quad F_1 = 1, \quad F_n = F_{n-1} + F_{n-2}$$

где $n \geq 2, n \in \mathbb{Z}$.

$$n! = \begin{cases} n \cdot (n-1)!, & n > 0 \\ 1, & n = 0 \end{cases}$$

$$e = 2 + \frac{2}{2 + \frac{3}{3 + \frac{4}{4 + \dots}}} = 2 + f(2), \text{ где } f(n) = \frac{n}{n + f(n+1)}$$





Структуры данных

В языке программирования Python всего 4 встроенных структуры данных:

- Список (list):
`list('abcd'), [1, 2, 'a', 'b'], [1, 23, 45, 6]`
- Кортеж (tuple):
`tuple(), (), ('s',), 's'`
- Словарь (dictionary):
`{}, {'a': 1, 'b': 2}, dict([(1, 1), (2, 4)])`
- Набор или множество (set):
`set(), set('hello'), set([1, 2, 2, 3, 4, 5])`



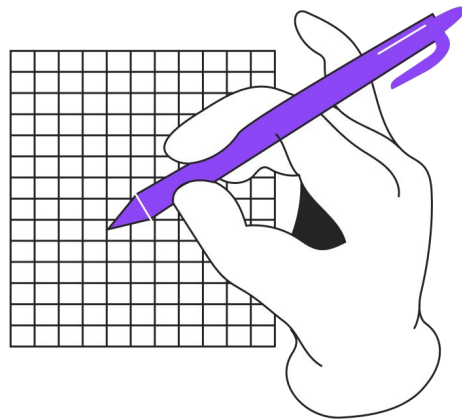
Алгоритмическая сложность операций

Структура данных	Временная сложность								Сложность по памяти
	В среднем				В худшем				В худшем
	Индексация	Поиск	Вставка	Удаление	Индексация	Поиск	Вставка	Удаление	
Обычный массив	$O(1)$	$O(n)$	-	-	$O(1)$	$O(n)$	-	-	$O(n)$
Динамический массив	$O(1)$	$O(n)$	$O(n)$	$O(n)$	$O(1)$	$O(n)$	$O(n)$	$O(n)$	$O(n)$
Односвязный список	$O(n)$	$O(n)$	$O(1)$	$O(1)$	$O(n)$	$O(n)$	$O(1)$	$O(1)$	$O(n)$
Двусвязный список	$O(n)$	$O(n)$	$O(1)$	$O(1)$	$O(n)$	$O(n)$	$O(1)$	$O(1)$	$O(n)$
Список с пропусками	$O(\log(n))$	$O(\log(n))$	$O(\log(n))$	$O(\log(n))$	$O(n)$	$O(n)$	$O(n)$	$O(n)$	$O(n \log(n))$
Хеш таблица	-	$O(1)$	$O(1)$	$O(1)$	-	$O(n)$	$O(n)$	$O(n)$	$O(n)$
Бинарное дерево поиска	$O(\log(n))$	$O(\log(n))$	$O(\log(n))$	$O(\log(n))$	$O(n)$	$O(n)$	$O(n)$	$O(n)$	$O(n)$



Практическое задание

1. Реализуйте бинарный поиск
2. Реализуйте несколько алгоритмов сортировки и сравните их время работы на одних и тех же входных данных





Полезные ссылки/дополнительные материалы



[Оценка сложности алгоритмов, или Что такое \$O\(\log n\)\$](#)



[Сложности алгоритмов](#)










[Асимптотика функций](#)



«Алгоритмы. Построение и анализ» Кормен Томас Х. и др.



Что мы сегодня узнали и чему научились

-  Что такое алгоритм
-  Свойства алгоритмов
-  Алгоритмическая сложность
-  Сортировки
-  Рекурсия
-  Структуры данных
-  Практическая часть



Вопросы?

Вопросы?



Вопросы?

