



Введение, сортировки, бинарный поиск



Как будет устроен курс?



Еженедельные лекции, семинары
и домашние работы



Посещаемость лекций
не учитывается в основной оценке



Посещаемость семинаров поможет получить
бонусы



На протяжении всего курса будут даваться
домашние работы — контесты на 5-10 задач



В конце курса пройдет экзамен в который
войдут все темы пройденные на курсе

Ссылки

Тестирующая система



Ссылка на телеграм



Частые вопросы



Формула оценивания

$$G_{final} = \min(0.7 \cdot G_{cumulative} + 0.3 \cdot G_{exam} + G_{bonus}, 10)$$

- **G_{exam}** — это оценка за экзамен,
- **G_{cumulative}** — это оценка за решение домашних заданий,
- **G_{bonus}** — оценка, которую можно получить за дополнительную активность

Все оценки подставляются в формулу с максимальной точностью, затем финальная оценка **G_{final}** округляется до целого числа.

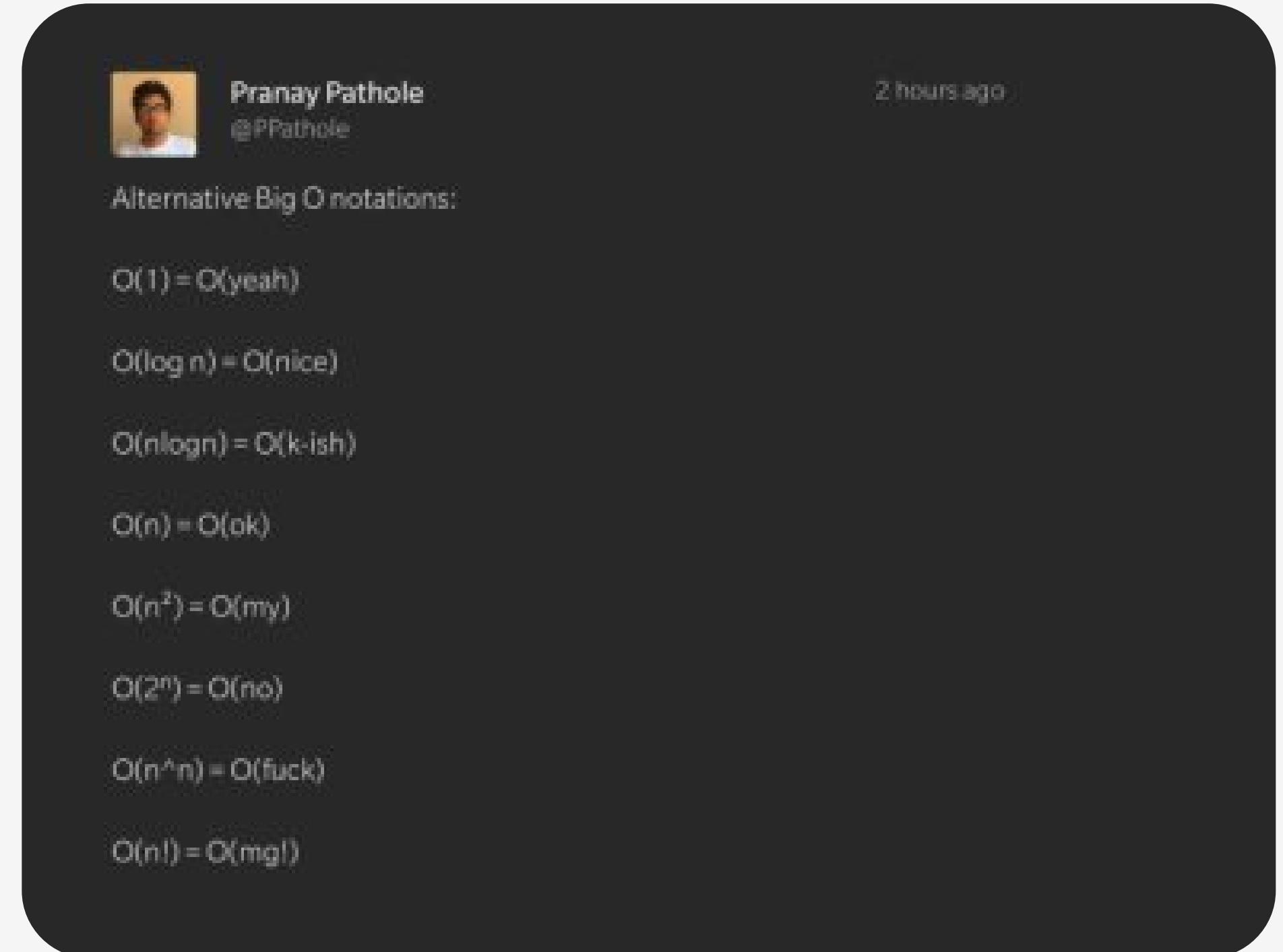
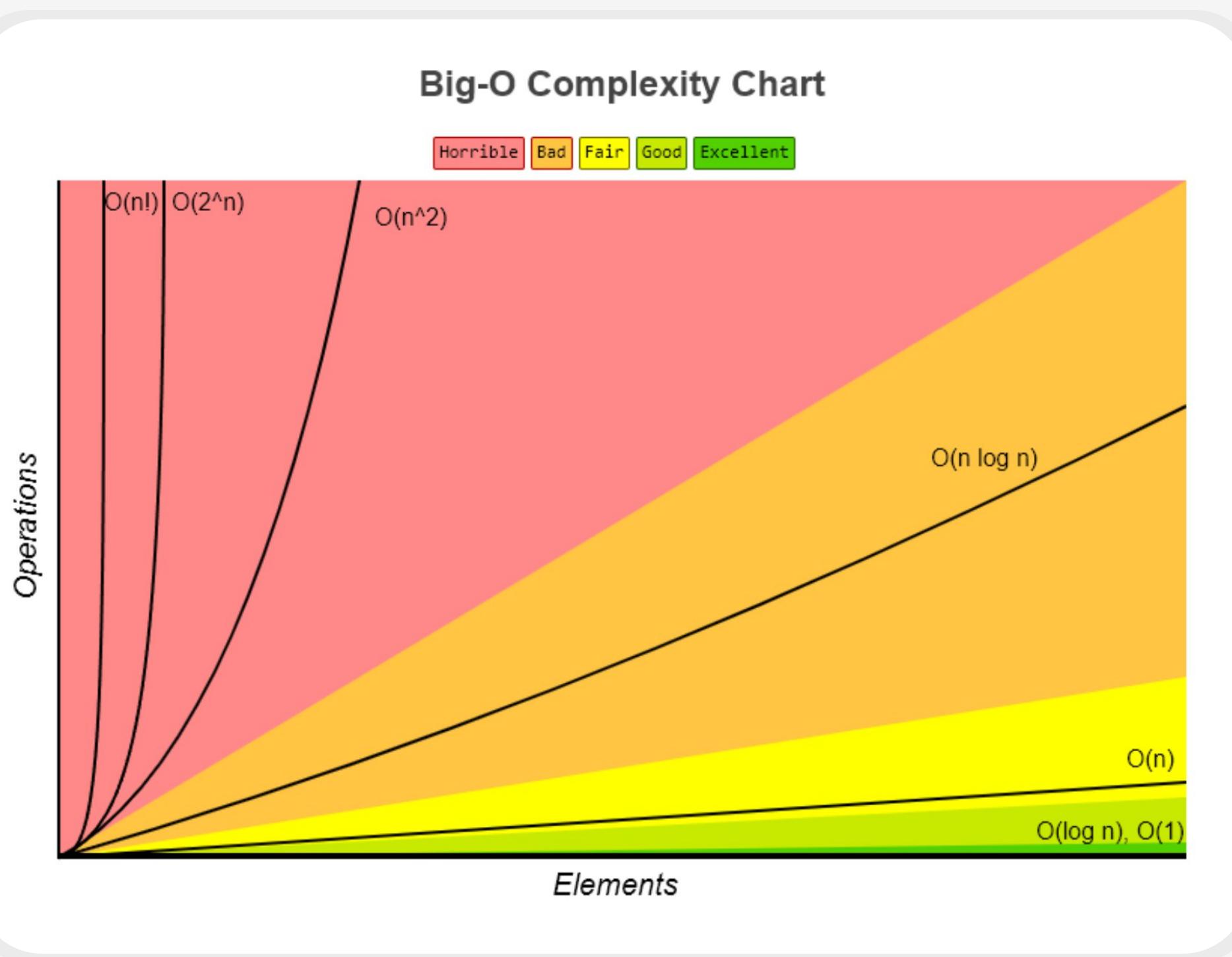
$$G_{cumulative} = \frac{\sum_{i=1}^n G_i \text{ Contest}}{n}$$

$$G_i \text{ Contest} = \sqrt{\frac{S_i}{N_i}}$$



Сложность алгоритмов

О-большое — максимальное количество операций, которые нужны для завершения алгоритма



Немного практики



...

```
for i in range(n):  
    for j in range(n):  
        print(i)
```

...

```
for i in range(4n):  
    for j in range(2int(math.sqrt(n))):  
        print(i)
```

...

```
for i in range(n):  
    for j in range(n/2):  
        print(i)
```

...

```
def fibonacci(n):  
    if n <= 1:  
        return n  
    else:  
        return fibonacci(n-1) + fibonacci(n-2)
```

Асимптотика	Примерное ограничение
$O(1)$	10^{18} (обычно ограничено 64 битной арифметикой)
$O(\log n)$	10^{18} (обычно ограничено 64 битной арифметикой)
$O(\sqrt{n})$	10^{10}
$O(n)$	10^6
$O(n \log n)$	10^5
$O(n^2)$	2000
$O(2n)$	17
$O(n!)$	7



Сортировки Полезные сортировки



Пузырьком (база)



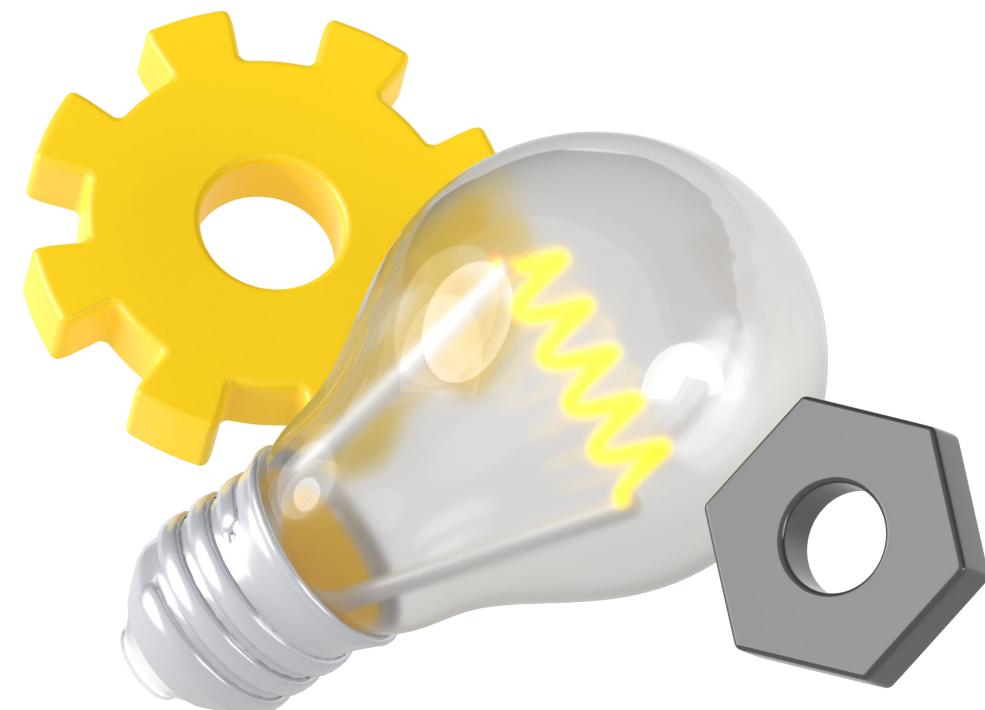
Слиянием



Подсчетом

Пузырьком

Идем по массиву и меняем соседние элементы местами. Повторяем такой проход n раз и получаем отсортированный массив.



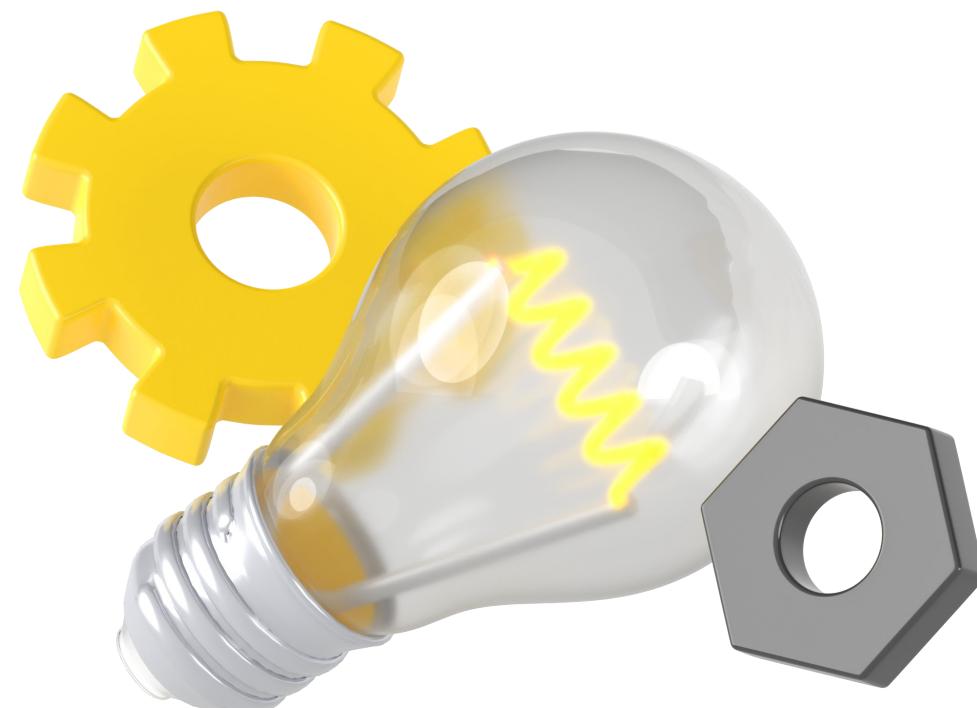
Какая сложность у такого алгоритма?

Когда выгодно использовать пузырек?

6 5 3 1 8 7 2 4

Слиянием

Рекурсивно разбиваем массив на две части, пока не получим массив из одного элемента, затем также сливаем их в один массив, используя два указателя



Какая сложность у такого алгоритма?

Когда выгодно использовать merge-sort?

6 5 3 1 8 7 2 4

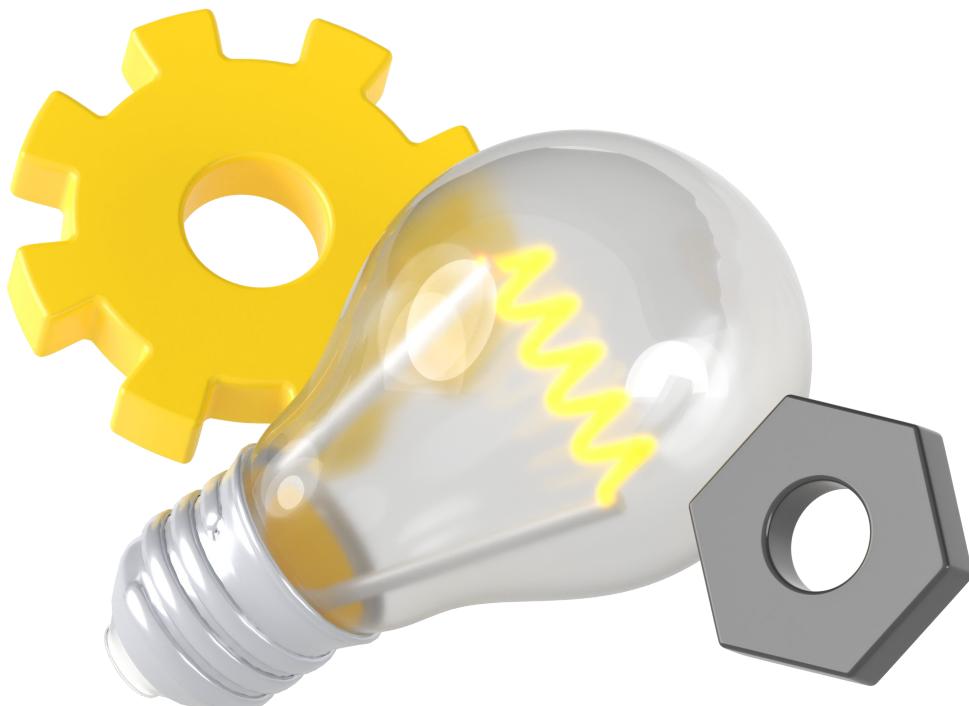
Подсчетом

Считаем количество каждого элемента в массиве, затем выписываем их в порядке возрастания

Какая сложность у такого алгоритма?

Когда выгодно использовать count-sort?

5	6	1	8	8	7	1	0	4	8	3	2	1	3	4	6	1	1
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18



00:28
Easy GIF Animator

Бинарный поиск

Есть массив из 0 и 1 вида 0000000111111111...

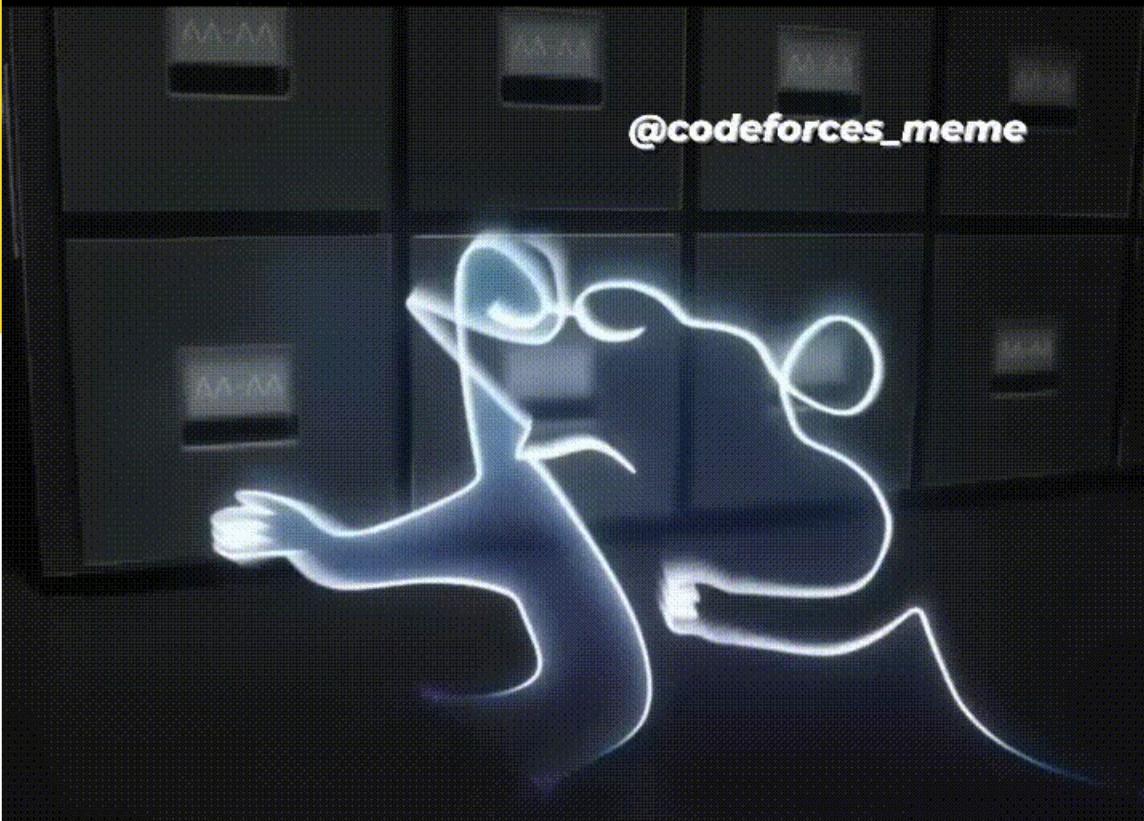
Нужно найти место разделения 0 / 1



Просто пройдем по массиву с начала
и когда встретим единичку остановимся

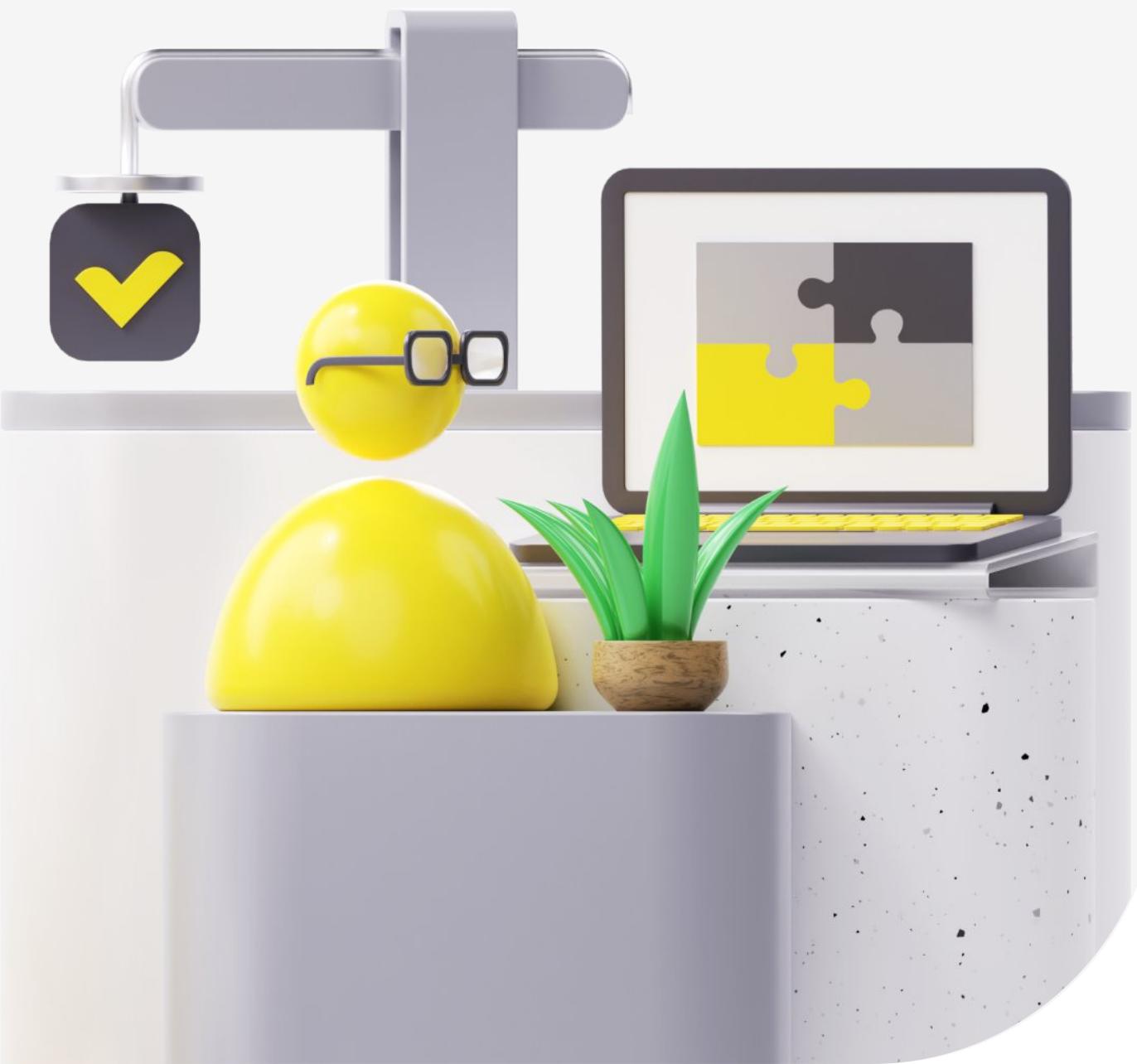
***Linear Search
Algorithm***

@codeforces_meme



***Time
Complexity: O(N)***

Немного другая задача



Если заменить 0 и 1 на значение $f(idx) \rightarrow \text{bool}$,
то можно решать мощные задачи

```
def f(idx):  
    return a[idx] >= K
```

```
def f(idx):  
    return sum(a[0:idx]) >= K
```

Алгоритм: ключевая идея

Представьте себе, что вы ищете слово «ТБанк» в словаре. Вы открыли словарь на случайной странице и нашли там слово Компьютер.

«К» < «Т»

Значит, искать нужно точно после буквы «К».

Если повторить много раз и делить примерно пополам, то мы быстро окажемся в ситуации, где разделение между 0 и 1 находится в очень маленьком промежутке.

Какая будет сложность?



Если мы проходимся по всему массиву



Если мы идем по массиву в обратном порядке



Если мы выбираем место для разделения в массиве случайно



Если мы делим массив пополам

Бинарный поиск по ответу

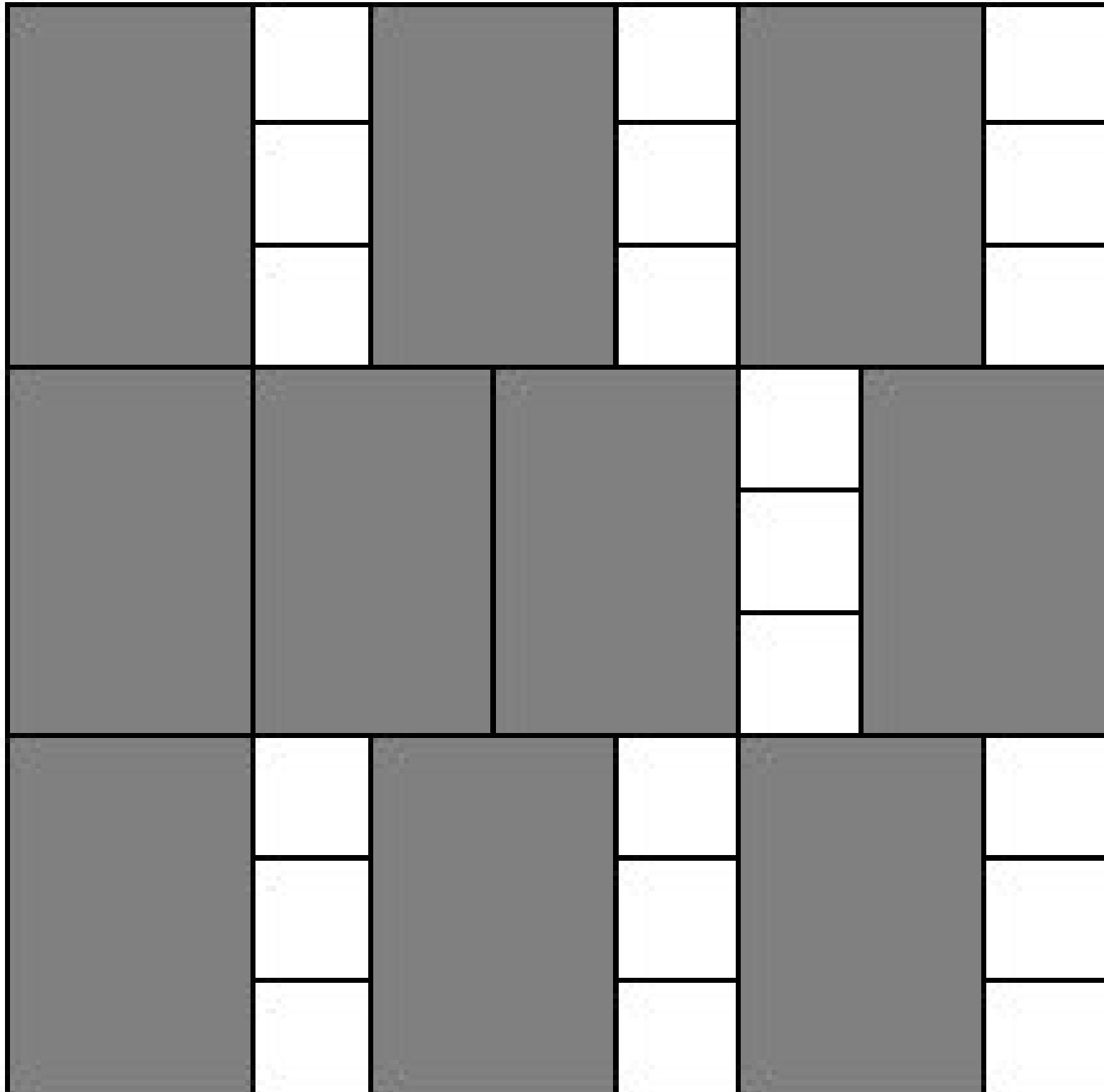
Задача про дипломы.

Есть n дипломов со своей высотой и шириной. Нужно их поместить в квадрат минимального размера.

Нужно найти сторону этого квадрата.

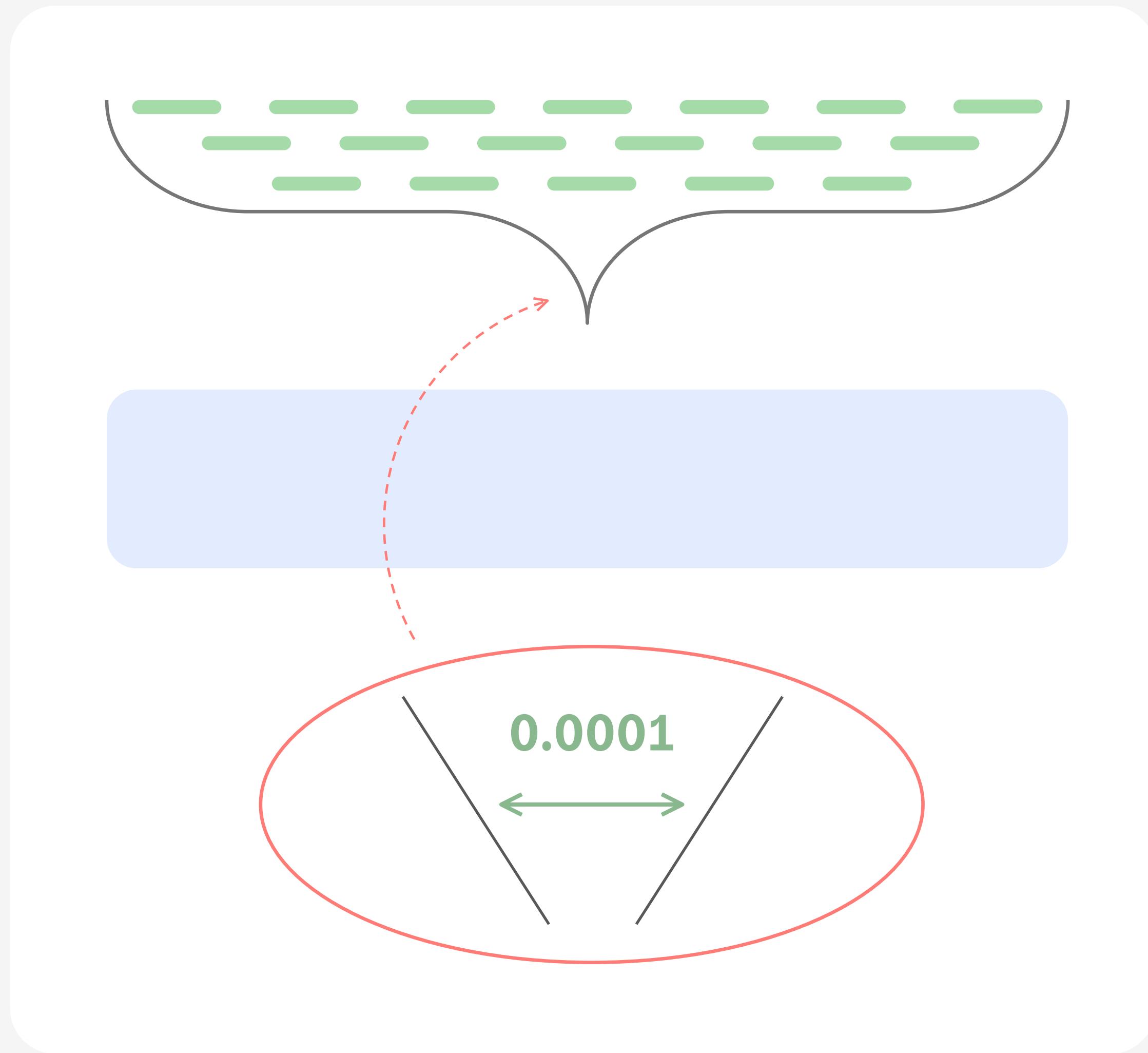


Можно ли, зная размер квадрата, понять помещаются ли в него все дипломы?



Вещественный двоичный поиск

Когда границы — это дробные числа, следует останавливать бинарный поиск, когда левая и правая границы находятся достаточно близко.



**ТИНЬКОФФ
ОБРАЗОВАНИЕ**



Спасибо!

