Разбор задач
Олимпиада СПбГУ, заочный тур 2024–2025 года

Иван Казменко

Санкт-Петербургский Государственный Университет

Пятница, 24 января 2025 года

Введение

В соревновании десять задач на различные темы
(и в разных форматах):

A. Арифметика
B. Длинная арифметика (открытые тесты)
C. Игра (интерактивная)
D. Кодирование и декодирование (двойной запуск)
E. Неточная задача (открытые тесты)
F. Комбинаторика
G. Математическая игра
H. Математический фокус (двойной запуск)
I. Задача со случайностью в данных
J. Задача со случайностью в решении (двойной запуск)

Условие

Задача A. Три блошки

- На числовой прямой выбраны три целых точки: a, b, c.
- За один шаг самая левая точка отражается симметрично относительно средней точки.
- Где будут точки через t шагов? Порядок точек важен.

Условие

Задача A. Три блошки

- На числовой прямой выбраны три целых точки: a, b, c.
- За один шаг самая левая точка отражается симметрично относительно средней точки.
- Где будут точки через t шагов? Порядок точек важен.
- Пример: $a = 1$, $b = 5$, $c = 4$, $t = 0$.

a	b	c	t
1	5	4	0
7	5	4	1
7	5	6	2
7	7	6	3
7	7	8	4
7	7	8	5

Условие

Задача A. Три блошки

- На числовой прямой выбраны три целых точки: a, b, c.
- За один шаг самая левая точка отражается симметрично относительно средней точки.
- Где будут точки через t шагов? Порядок точек важен.
- Пример: $a = 1$, $b = 5$, $c = 4$, $t = 1$.

a	b	c	t
1	5	4	0
7	5	4	1
7	5	6	2
7	7	6	3
7	7	8	4
7	7	8	5

Условие

Задача А. Три блошки

- На числовой прямой выбраны три целых точки: a, b, c.
- За один шаг самая левая точка отражается симметрично относительно средней точки.
- Где будут точки через t шагов? Порядок точек важен.
- Пример: $a = 1$, $b = 5$, $c = 4$, $t = 2$.

a	b	c	t
1	5	4	0
7	5	4	1
7	5	6	2
7	7	6	3
7	7	8	4
7	7	8	5

Условие

Задача A. Три блошки

- На числовой прямой выбраны три целых точки: a, b, c.
- За один шаг самая левая точка отражается симметрично относительно средней точки.
- Где будут точки через t шагов? Порядок точек важен.
- Пример: $a = 1$, $b = 5$, $c = 4$, $t = 3$.

a	b	c	t
1	5	4	0
7	5	4	1
7	5	6	2
7	7	6	3
7	7	8	4
7	7	8	5

Условие

Задача А. Три блошки

- На числовой прямой выбраны три целых точки: a, b, c.
- За один шаг самая левая точка отражается симметрично относительно средней точки.
- Где будут точки через t шагов? Порядок точек важен.
- Пример: $a = 1$, $b = 5$, $c = 4$, $t = 4$.

a	b	c	t
1	5	4	0
7	5	4	1
7	5	6	2
7	7	6	3
7	7	8	4
7	7	8	5

Условие

Задача A. Три блошки

- На числовой прямой выбраны три целых точки: a, b, c.
- За один шаг самая левая точка отражается симметрично относительно средней точки.
- Где будут точки через t шагов? Порядок точек важен.
- Пример: $a = 1$, $b = 5$, $c = 4$, $t = 5$.

a	b	c	t
1	5	4	0
7	5	4	1
7	5	6	2
7	7	6	3
7	7	8	4
7	7	8	5

Условие

Задача A. Три блошки

- На числовой прямой выбраны три целых точки: a, b, c.
- За один шаг самая левая точка отражается симметрично относительно средней точки.
- Где будут точки через t шагов? Порядок точек важен.
- Подзадачи:
  - все числа от 0 до 100,
  - все числа от 0 до 10^{12}.

Симуляция

- Симуляция: сделаем что сказано.
- Общее время работы: $\mathcal{O}(t)$.
- Напишем симуляцию одного шага. Как не писать её для шести случаев: какая точка самая левая, а какая — средняя?

Симуляция

- Симуляция: сделаем что сказано.
- Общее время работы: $\mathcal{O}(t)$.
- Напишем симуляцию одного шага. Как не писать её для шести случаев: какая точка самая левая, а какая — средняя?
  - Напишем функцию, симулирующую один шаг и принимающую $a \leqslant b \leqslant c$ как ссылки. При вызове функции расставим аргументы в правильном порядке — в зависимости от случая.
  - Или: будем хранить точки как пары — число и имя точки.
    В решении будем сортировать пары по числам.
    При выводе отсортируем пары по именам.

Пример посложнее

- Решение часто можно придумать, разбирая задачу на примерах.
- Рассмотрим пример посложнее: $a = 0$, $b = 10$, $c = 48$.

a	b	c	t
0	10	48	0
20	10	48	1
20	30	48	2
40	30	48	3
40	50	48	4
56	50	48	5
56	50	52	6
56	54	52	7
56	54	56	8
56	58	56	9
56	58	56	10

Пример посложнее

- Решение часто можно придумать, разбирая задачу на примерах.
- Рассмотрим пример посложнее: $a = 0$, $b = 10$, $c = 48$.

a	b	c	t
0	10	48	0
20	10	48	1
20	30	48	2
40	30	48	3
40	50	48	4
56	50	48	5
56	50	52	6
56	54	52	7
56	54	56	8
56	58	56	9
56	58	56	10

Пример посложнее

- Решение часто можно придумать, разбирая задачу на примерах.
- Рассмотрим пример посложнее: $a = 0$, $b = 10$, $c = 48$.

a	b	c	t
0	10	48	0
20	10	48	1
20	30	48	2
40	30	48	3
40	50	48	4
56	50	48	5
56	50	52	6
56	54	52	7
56	54	56	8
56	58	56	9
56	58	56	10

Пример посложнее

- Решение часто можно придумать, разбирая задачу на примерах.
- Рассмотрим пример посложнее: $a = 0$, $b = 10$, $c = 48$.

a	b	c	t
0	10	48	0
20	10	48	1
20	30	48	2
40	30	48	3
40	50	48	4
56	50	48	5
56	50	52	6
56	54	52	7
56	54	56	8
56	58	56	9
56	58	56	10

Пример посложнее

- Решение часто можно придумать, разбирая задачу на примерах.
- Рассмотрим пример посложнее: $a = 0$, $b = 10$, $c = 48$.

a	b	c	t
0	10	48	0
20	10	48	1
20	30	48	2
40	30	48	3
40	50	48	4
56	50	48	5
56	50	52	6
56	54	52	7
56	54	56	8
56	58	56	9
56	58	56	10

Пример посложнее

- Решение часто можно придумать, разбирая задачу на примерах.
- Рассмотрим пример посложнее: $a = 0$, $b = 10$, $c = 48$.

a	b	c	t
0	10	48	0
20	10	48	1
20	30	48	2
40	30	48	3
40	50	48	4
56	50	48	5
56	50	52	6
56	54	52	7
56	54	56	8
56	58	56	9
56	58	56	10

Пример посложнее

- Решение часто можно придумать, разбирая задачу на примерах.
- Рассмотрим пример посложнее: $a = 0$, $b = 10$, $c = 48$.

a	b	c	t
0	10	48	0
20	10	48	1
20	30	48	2
40	30	48	3
40	50	48	4
56	50	48	5
56	50	52	6
56	54	52	7
56	54	56	8
56	58	56	9
56	58	56	10

Пример посложнее

- Решение часто можно придумать, разбирая задачу на примерах.
- Рассмотрим пример посложнее: $a = 0$, $b = 10$, $c = 48$.

a	b	c	t
0	10	48	0
20	10	48	1
20	30	48	2
40	30	48	3
40	50	48	4
56	50	48	5
56	50	52	6
56	54	52	7
56	54	56	8
56	58	56	9
56	58	56	10

Пример посложнее

- Решение часто можно придумать, разбирая задачу на примерах.
- Рассмотрим пример посложнее: $a = 0$, $b = 10$, $c = 48$.

a	b	c	t
0	10	48	0
20	10	48	1
20	30	48	2
40	30	48	3
40	50	48	4
56	50	48	5
56	50	52	6
56	54	52	7
56	54	56	8
56	58	56	9
56	58	56	10

Пример посложнее

- Решение часто можно придумать, разбирая задачу на примерах.
- Рассмотрим пример посложнее: $a = 0$, $b = 10$, $c = 48$.

a	b	c	t
0	10	48	0
20	10	48	1
20	30	48	2
40	30	48	3
40	50	48	4
56	50	48	5
56	50	52	6
56	54	52	7
56	54	56	8
56	58	56	9
56	58	56	10

Пример посложнее

- Решение часто можно придумать, разбирая задачу на примерах.
- Рассмотрим пример посложнее: $a = 0$, $b = 10$, $c = 48$.

a	b	c	t
0	10	48	0
20	10	48	1
20	30	48	2
40	30	48	3
40	50	48	4
56	50	48	5
56	50	52	6
56	54	52	7
56	54	56	8
56	58	56	9
56	58	56	10

Пример посложнее

- Рассмотрим пример посложнее: $a = 0$, $b = 10$, $c = 48$.
- Выпишем наши три числа в порядке возрастания: $lo \leqslant me \leqslant hi$.

a	b	c	t
0	10	48	0
20	10	48	1
20	30	48	2
40	30	48	3
40	50	48	4
56	50	48	5
56	50	52	6
56	54	52	7
56	54	56	8
56	58	56	9
56	58	56	10

lo	x	me	y	hi	t
0	10	10	38	48	0
10	10	20	28	48	1
20	10	30	18	48	2
30	10	40	8	48	3
40	8	48	2	50	4
48	2	50	6	56	5
50	2	52	4	56	6
52	2	54	2	56	7
54	2	56	0	56	8
56	0	56	2	58	9
56	0	56	2	58	10

Пример посложнее

- Выпишем наши три числа в порядке возрастания: $lo \leqslant me \leqslant hi$.
- Рассмотрим разности соседей: $x = me - lo$ и $y = hi - me$.

a	b	c	t
0	10	48	0
20	10	48	1
20	30	48	2
40	30	48	3
40	50	48	4
56	50	48	5
56	50	52	6
56	54	52	7
56	54	56	8
56	58	56	9
56	58	56	10

lo	x	me	y	hi	t
0	10	10	38	48	0
10	10	20	28	48	1
20	10	30	18	48	2
30	10	40	8	48	3
40	8	48	2	50	4
48	2	50	6	56	5
50	2	52	4	56	6
52	2	54	2	56	7
54	2	56	0	56	8
56	0	56	2	58	9
56	0	56	2	58	10

Пример посложнее

- Рассмотрим разности соседей: $x = me - lo$ и $y = hi - me$.
- На каждом шаге в паре x и y из большего вычитается меньшее.

a	b	c	t
0	10	48	0
20	10	48	1
20	30	48	2
40	30	48	3
40	50	48	4
56	50	48	5
56	50	52	6
56	54	52	7
56	54	56	8
56	58	56	9
56	58	56	10

lo	x	me	y	hi	t
0	10	10	38	48	0
10	10	20	28	48	1
20	10	30	18	48	2
30	10	40	8	48	3
40	8	48	2	50	4
48	2	50	6	56	5
50	2	52	4	56	6
52	2	54	2	56	7
54	2	56	0	56	8
56	0	56	2	58	9
56	0	56	2	58	10

Пример посложнее

- На каждом шаге в паре x и y из большего вычитается меньшее.
- Что это за процесс? Поиск наибольшего общего делителя x и y.

a	b	c	t
0	10	48	0
20	10	48	1
20	30	48	2
40	30	48	3
40	50	48	4
56	50	48	5
56	50	52	6
56	54	52	7
56	54	56	8
56	58	56	9
56	58	56	10

lo	x	me	y	hi	t
0	10	10	38	48	0
10	10	20	28	48	1
20	10	30	18	48	2
30	10	40	8	48	3
40	8	48	2	50	4
48	2	50	6	56	5
50	2	52	4	56	6
52	2	54	2	56	7
54	2	56	0	56	8
56	0	56	2	58	9
56	0	56	2	58	10

Пример посложнее

- Что это за процесс? Поиск наибольшего общего делителя x и y.
- Как ускорить поиск НОД? Применить алгоритм Евклида!

a	b	c	t
0	10	48	0
20	10	48	1
20	30	48	2
40	30	48	3
40	50	48	4
56	50	48	5
56	50	52	6
56	54	52	7
56	54	56	8
56	58	56	9
56	58	56	10

lo	x	me	y	hi	t
0	10	10	38	48	0
10	10	20	28	48	1
20	10	30	18	48	2
30	10	40	8	48	3
40	8	48	2	50	4
48	2	50	6	56	5
50	2	52	4	56	6
52	2	54	2	56	7
54	2	56	0	56	8
56	0	56	2	58	9
56	0	56	2	58	10

Решение

Решение похоже на алгоритм Евклида — но нам нужно поддерживать не два числа x и y, а координаты трёх точек.

- Будем рассматривать пары шагов.

Решение

Решение похоже на алгоритм Евклида — но нам нужно поддерживать не два числа x и y, а координаты трёх точек.

- Будем рассматривать пары шагов.
- Пусть для определённости $a \leqslant b \leqslant c$, после первого шага $b' \leqslant a' \leqslant c$, а после второго шага опять $a'' \leqslant b'' \leqslant c$.
  Введём обозначение: $\Delta = b - a$. Тогда $a'' = a + 2\Delta$ и $b'' = b + 2\Delta$.

Решение

Решение похоже на алгоритм Евклида — но нам нужно поддерживать не два числа x и y, а координаты трёх точек.

- Будем рассматривать пары шагов.
- Пусть для определённости $a \leqslant b \leqslant c$, после первого шага $b' \leqslant a' \leqslant c$, а после второго шага опять $a'' \leqslant b'' \leqslant c$. Введём обозначение: $\Delta = b - a$. Тогда $a'' = a + 2\Delta$ и $b'' = b + 2\Delta$.
- Сколько таких пар шагов мы сделаем, прежде чем a или b перепрыгнет через c? Количество пар равно $s = \lfloor (c - a)/(2\Delta) \rfloor$.
- Пусть осталось сделать t шагов. Тогда $s := \min(s, \lfloor t/2 \rfloor)$.

Решение

Решение похоже на алгоритм Евклида — но нам нужно поддерживать не два числа x и y, а координаты трёх точек.

- Будем рассматривать пары шагов.
- Пусть для определённости $a \leqslant b \leqslant c$, после первого шага $b' \leqslant a' \leqslant c$, а после второго шага опять $a'' \leqslant b'' \leqslant c$. Введём обозначение: $\Delta = b - a$. Тогда $a'' = a + 2\Delta$ и $b'' = b + 2\Delta$.
- Сколько таких пар шагов мы сделаем, прежде чем a или b перепрыгнет через c? Количество пар равно $s = \lfloor (c - a)/(2\Delta) \rfloor$.
- Пусть осталось сделать t шагов. Тогда $s := \min(s, \lfloor t/2 \rfloor)$.
- Сделаем все s пар шагов за одно действие: $t := t - 2s$, $a := a + 2s\Delta$, $b := b + 2s\Delta$.
- После этого, если $t > 0$, просто сделаем ещё один шаг.

Время работы

Как оценить время работы алгоритма?

- Когда из $a \leqslant b \leqslant c$ мы получили $a \geqslant c$ или $b \geqslant c$, это соответствует одному шагу алгоритма Евклида.
- Для этого в нашем алгоритме мы делаем одно действие с s парами шагов, а потом ещё один шаг обычной симуляцией.
- Значит, общее количество действий — такого же порядка, как в алгоритме Евклида.
- Общее время работы: $\mathcal{O}(\log t)$.

Частичная идея

Можно не придумывать доказательство, а просто делать много однотипных шагов за одно действие и надеяться, что этого достаточно.

- Например, если $a \leqslant b \leqslant c$ и через $\sqrt{limit} = 10^6$ шагов число c не поменяется, можно сделать все эти шаги за одно действие, как в предыдущем решении.
- Иначе — делать шаги по одному, обычной симуляцией.
- Такое решение работает за время $\mathcal{O}(\sqrt{limit})$. Почему? Как мы знаем из предыдущего решения, большее число меняется $\mathcal{O}(\log t)$ раз, а $\log t < \sqrt{limit}$.

Условие

Задача B. Кубический корень

- Это задача с открытыми тестами.
- Даны n (от 1 до 100) и s (от 10 до 100 000).
- Вычислите $\sqrt[3]{n}$ и выведите как десятичную дробь, ровно с s цифрами после точки, округлив вниз.
- Пример: $n = 2$, $s = 10$. Ответ: 1.2599210498.

Открытые тесты

Открытые тесты — часть условия задачи.

- Нужно скачать и посмотреть в них. Что мы видим?

Открытые тесты

Открытые тесты — часть условия задачи.

- Нужно скачать и посмотреть в них. Что мы видим?

n	s
2	10
3	15
4	50
10	123
11	1234
12	12 345
23	60 000
39	70 000
41	80 000
48	90 000
56	100 000

Варианты решения

Как решать задачу с открытыми тестами?

1. Быстрое решение.

- Можно придумать и написать быстрое решение, которое успеет посчитать ответ на всех тестах.
- Для этого придётся придумать по-настоящему быстрое решение.

Варианты решения

Как решать задачу с открытыми тестами?

2. Предподсчёт.

- Можно вычислить ответы локально, а на проверку послать программу со строками-ответами.
- Но: по условию максимальный размер программы, которую можно послать на проверку, равен 262 144 байтам.
- Все ответы не поместятся. Можно послать программу со сжатыми строками. Например, использовать zip и base64, если они есть в стандартной библиотеке языка. Или написать своё сжатие.

Варианты решения

Как решать задачу с открытыми тестами?

3. Комбинированное решение.

- Наконец, можно комбинировать эти подходы: часть тестов решить заранее и записать как строки, другую часть решать на проверяющем сервере.
- Или в программу записать подсказки, ускоряющие решение. Например, начальное приближение. Или первую половину цифр ответа.

Тип данных

- Точности обычных вещественных чисел — примерно 16 знаков — хватит только на пример и первый тест.
- Нужна длинная вещественная арифметика (например, `Decimal` в Python, `BigDecimal` в Java, ...).
- Или длинная арифметика в целых числах: домножим на 10^s (например, обычные числа в Python, `BigInteger` в Java, ...).
- Если в языке (C или C++) нет длинной арифметики — можно реализовать её самостоятельно.
- Если задача решается предподсчётом — локально можно установить и использовать нестандартные библиотеки.

Способы вычисления

Как вычислять корень?

1. Встроенные функции.

- Возможно, в библиотеке есть функция `pow` для длинных чисел.
- Обычно получается медленно.

Способы вычисления

Как вычислять корень?

2. Вручную, как считают корень «в столбик».

- Будем вычислять цифры ответа одну за другой.
- Пусть $x = \lfloor \sqrt[3]{n} \rfloor$. Ещё будем хранить числа $x_2 = x^2$, $x_3 = x^3$ и n.
- Как получить следующую цифру?
- Сначала добавим в конец цифру 0. Выполним $x := x \cdot 10$, $x_2 := x_2 \cdot 100$, $x_3 := x_3 \cdot 1000$ и $n := n \cdot 1000$.
- Пока можно, будем прибавлять к x единицу: $x_3 := x_3 + 3x_2 + 3x + 1$, $x_2 := x_2 + 2x + 1$, $x := x + 1$.
- Будем так делать, пока новое значение $x_3 \leqslant n$. Когда мы остановимся, последняя цифра x будет посчитана верно.
- Такое решение работает за $\mathcal{O}(s^2)$.

Способы вычисления

Как вычислять корень?

3. Метод Ньютона.

- Мы хотим решить уравнение $f(x) = 0$, где $f(x) = x^3 - n$.
- Найдём производную: $f'(x) = 3x^2$.
- Установим начальное приближение: $x_0 = n$.
- Теперь будем считать $x_{k+1} = x_k - f(x_k)/f'(x_k) =$
  $= x_k - (x_k^3 - n)/(3x_k^2) = (2x_k + n/x_k^2)/3$.
- Когда у x_{k+1} и x_k совпадёт нужное количество знаков (чуть больше s), закончим работу.
- На каждой итерации количество точных знаков примерно удваивается. Поэтому понадобится порядка $\log s$ итераций.
- Если умножение и деление s-значных чисел производятся за $\mathcal{O}(s \log s)$, общее время работы можно оценить как $\mathcal{O}(s \log^2 s)$.

Условие

Задача C. Встретиться или разминуться?

- Это интерактивная задача.
- Рассмотрим прямоугольник из $r \times c$ клеток ($r + c$ чётно).
- Поставим фишки двух игроков — Марины и Насти — в противоположные клетки: $(r, 1)$ и $(1, c)$.
- На каждом ходу фишка двигается на соседнюю клетку.
- Можно двигаться в две стороны — к противоположному углу, но из прямоугольника выходить нельзя.
- Жюри делает ходы за первого игрока, решение — за второго.
- Сделайте так, чтобы фишки встретились.

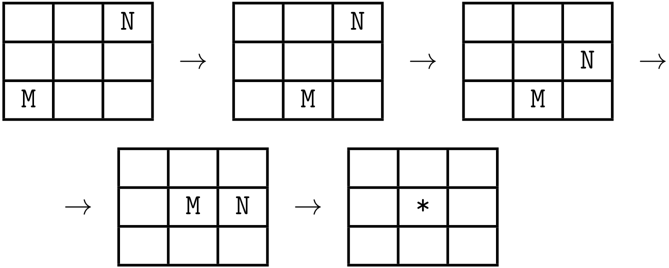
Условие

Задача C. Встретиться или разминуться?

- Это интерактивная задача.
- Рассмотрим прямоугольник из $r \times c$ клеток ($r + c$ чётно).
- Поставим фишки двух игроков — Марины и Насти — в противоположные клетки: $(r, 1)$ и $(1, c)$.
- На каждом ходу фишка двигается на соседнюю клетку.
- Можно двигаться в две стороны — к противоположному углу, но из прямоугольника выходить нельзя.
- Жюри делает ходы за первого игрока, решение — за второго.
- Сделайте так, чтобы фишки встретились.
- Подзадачи:
    - $r = c$,
    - $r + c$ чётно.

Пример

Пример: $r = 3$, $c = 3$.



Решение для $r = c$

Сначала решим задачу для $r = c$.

- Инвариант: расстояния по обеим координатам равны.
- Противник уменьшает одно из расстояний на 1.
- Своим ходом мы всегда можем уменьшить другое расстояние на 1, и тем самым опять сделать их равными.

Общее решение

Теперь решим задачу целиком.

- Зачем нужно условие «$r + c$ чётно»? Чтобы встреча происходила после хода второго игрока, за которого нам и нужно играть.

Общее решение

Теперь решим задачу целиком.

- Зачем нужно условие «$r + c$ чётно»? Чтобы встреча происходила после хода второго игрока, за которого нам и нужно играть.
- Будем стремиться к тому, чтобы расстояние по вертикали стало равно расстоянию по горизонтали. То есть уменьшать то расстояние, которое больше. Почему этого достаточно?

Общее решение

Теперь решим задачу целиком.

- Зачем нужно условие «$r + c$ чётно»? Чтобы встреча происходила после хода второго игрока, за которого нам и нужно играть.
- Будем стремиться к тому, чтобы расстояние по вертикали стало равно расстоянию по горизонтали. То есть уменьшать то расстояние, которое больше. Почему этого достаточно?
- А как вообще можно проиграть? Только если первый игрок попадёт в клетку, куда нам уже не попасть. Сделает какое-то из расстояний отрицательным.
- Пока одно из расстояний больше, мы будем идти вдоль границы прямоугольника.
- Противник может сделать другое расстояние равным нулю, но не может выйти за границу прямоугольника. Значит, отрицательным расстояние не станет.

Условие

Задача D. В полтора раза больше

- Это задача с двойным запуском.
- Дана двоичная матрица размера $2n \times 2n$.
- Мы можем передать любую двоичную матрицу размера $3n \times 3n$.
- При передаче каждая пара элементов, симметричная относительно главной диагонали, может поменяться местами.
- По полученной после этого матрице $3n \times 3n$ нужно восстановить исходную матрицу $2n \times 2n$.

Пример

Пример: $n = 2$.

$$
\begin{array}{|c|c|c|c|}\hline 1&0&0&1\\\hline 0&1&1&0\\\hline 0&0&1&1\\\hline 1&0&0&0\\\hline\end{array}
\xrightarrow{?}
\begin{array}{|c|c|c|c|c|c|}\hline 0&0&1&0&1&1\\\hline 1&0&1&1&1&1\\\hline 1&0&1&1&0&0\\\hline 0&1&0&0&0&0\\\hline 0&0&0&1&1&1\\\hline 0&1&0&1&0&1\\\hline\end{array}
\begin{array}{c}\color{red}{\to}\\ \color{blue}{\to}\\ \color{orange}{\to}\end{array}
\begin{array}{|c|c|c|c|c|c|}\hline 0&0&1&0&1&\color{red}{0}\\\hline 1&0&1&1&\color{blue}{0}&1\\\hline 1&0&1&1&0&0\\\hline 0&1&0&0&0&\color{orange}{1}\\\hline 0&\color{blue}{1}&0&1&1&1\\\hline \color{red}{1}&1&0&\color{orange}{0}&0&1\\\hline\end{array}
\xrightarrow{?}
\begin{array}{|c|c|c|c|}\hline 1&0&0&1\\\hline 0&1&1&0\\\hline 0&0&1&1\\\hline 1&0&0&0\\\hline\end{array}
$$

Передаваемая матрица может быть любой.

Ей не обязательно быть такой же, как в примере.

А вот итоговая матрица должна быть равна исходной.

Решение

Какую матрицу передавать?

- Элементы, симметричные относительно главной диагонали, могут поменяться местами.

Решение

Какую матрицу передавать?

- Элементы, симметричные относительно главной диагонали, могут поменяться местами.
- Что, если просто сделать их равными, чтобы это ни на что не влияло?

Решение

Какую матрицу передавать?

- Элементы, симметричные относительно главной диагонали, могут поменяться местами.
- Что, если просто сделать их равными, чтобы это ни на что не влияло?
- В исходной матрице $2n \cdot 2n = 4n^2$ двоичных цифр.
- В передаваемой матрице мы будем нижний треугольник копировать из верхнего. Тогда мы можем записать $3n \cdot (3n-1)/2 = 4.5n^2 - 1.5n$ различных цифр.
- Можно убедиться, что для $n \geqslant 1$ нам хватит места, чтобы просто записать все данные цифры.
- При декодировании просто рассматриваем верхний треугольник передаваемой матрицы.

Решение

Пример:

1	0	0	1
0	1	1	0
0	0	1	1
1	0	0	0

$\rightarrow$

1	0	0	1	0	1
	1	0	0	0	1
		1	1	0	0
			0	.	.
				.	.
					.

$\rightarrow$

1	0	0	1	0	1
0	1	0	0	0	1
0	0	1	1	0	0
1	0	1	0	0	0
0	0	0	0	0	0
1	1	0	0	0	0

Условие

Задача E. Отрезок с k простыми числами

- Это задача с оценкой частичных решений и открытыми тестами.
- Зафиксируем число k. В тестах $k = 10, 20, 30, 40, 50$.
- Найдите как можно более короткий отрезок $[l, r]$, содержащий ровно k простых чисел.
- Ограничения: $10^9 \leqslant l < r \leqslant 10^{18}$.
- Чем короче отрезок, тем больше баллов за тест:
- Если длина отрезка жюри равна $jury$, а участника $cont$, участник получает $20 \cdot jury^3 / cont^3$ баллов.
- Отрезок короче, чем у жюри, получает 20 баллов за тест.
- Пример: $k = 2$, один из оптимальных ответов: $[10^9 + 7, 10^9 + 9]$.

Решение

Что поможет нам решать?

- Открытые тесты — часть условия задачи.
  В этот раз они написаны прямо в условии.
- Вероятно, у задачи нет быстрого решения.
  Зато можно найти хорошие ответы предподсчётом.
- Обший вид решения: проверяем числа подряд на простоту, отслеживаем последние k виденных чисел.
- Можно решать одновременно для всех нужных k: если следить за последними 50 простыми числами, можно заодно следить и за последними 40 из них.

Решение

Как искать простые числа подряд?

- Решето Эратосфена.
- Зафиксируем большой отрезок $[a, a + d]$. Например, пусть $d = 10^7$.
- Заранее найдём все простые числа до $\sqrt{a + d}$.
- Для каждого простого числа вычеркнем на отрезке $[a, a + d]$ то, что делится на это число.
- Числа, которые остались не вычеркнуты — простые.
- Затем сделаем то же самое на отрезке $[a + d, a + 2d]$, и так далее.

Решение

Как ещё можно искать простые числа подряд?

- Проверка тестом на простоту.
- Например, хорошо подходит тест Миллера–Рабина: для чисел до 10^{18} достаточно семи проверок.
- Для ускорения можно сначала выкинуть из рассмотрения числа, которые делятся на небольшие простые — скажем, на одно из первых восьми.
- Программа жюри, реализующая это решение, проходит порядка 10^7 чисел в секунду.

Решение

Что дальше?

- Запустим написанную программу, чтобы она шла по простым числам и время от времени выдавала наилучший найденный ответ.
- Оставим её работать на минуту, час, день, неделю...
- Соберём найденные наилучшие ответы и пошлём на проверку.

Статистика

Наиболее осмысленно искать в начале отрезка $[10^9, 10^{18}]$.

- Известно, что k-е простое число имеет порядок $k \log k$. Поэтому, например, среди миллиона 10-значных чисел подряд простых будет больше, чем среди миллиона 11-значных чисел подряд.
- Статистика для сравнения:
- Ответы среди чисел от 10^9 до 10^{12}:

k	10	20	30	40	50
длина отрезка	32	120	240	366	522

- Ответы среди чисел от 10^{12} до $\approx 2.6 \cdot 10^{13}$ (их в 25 раз больше!):

k	10	20	30	40	50
длина отрезка	32	122	242	390	560

- Вполне возможно, что найденные ответы — оптимальные.

Условие

Задача F. Символ повтора

- Рассмотрим строку из n десятичных цифр.
- Пройдём по ней слева направо и избавимся от повторов: если две соседние цифры равны, заменим правую на символ повтора «R».
- Например, 12341 $\rightarrow$ 12341, 99999 $\rightarrow$ 9R9R9, 00022 $\rightarrow$ 0R02R.
- Теперь рассмотрим все 10^n получившихся записей.
- Отсортируем их.
- Дано k. Найдите k-ю в лексикографическом порядке запись.

Условие

Задача F. Символ повтора

- Рассмотрим строку из n десятичных цифр.
- Пройдём по ней слева направо и избавимся от повторов: если две соседние цифры равны, заменим правую на символ повтора «R».
- Например, $12341 \to 12341$, $99999 \to 9R9R9$, $00022 \to 0R02R$.
- Теперь рассмотрим все 10^n получившихся записей.
- Отсортируем их.
- Дано k. Найдите k-ю в лексикографическом порядке запись.
- Подзадачи:
  - длина строки $n \leqslant 5$,
  - длина строки $n \leqslant 10\,000$.

Примеры

Пусть $n = 2$. Вот все 10^2 строк в лексикографическом порядке:

00: 01	01: 02	02: 03	03: 04	04: 05	05: 06	06: 07	07: 08	08: 09	09: 0R
10: 10	11: 12	12: 13	13: 14	14: 15	15: 16	16: 17	17: 18	18: 19	19: 1R
20: 20	21: 21	22: 23	23: 24	24: 25	25: 26	26: 27	27: 28	28: 29	29: 2R
30: 30	31: 31	32: 32	33: 34	34: 35	35: 36	36: 37	37: 38	38: 39	39: 3R
40: 40	41: 41	42: 42	43: 43	44: 45	45: 46	46: 47	47: 48	48: 49	49: 4R
50: 50	51: 51	52: 52	53: 53	54: 54	55: 56	56: 57	57: 58	58: 59	59: 5R
60: 60	61: 61	62: 62	63: 63	64: 64	65: 65	66: 67	67: 68	68: 69	69: 6R
70: 70	71: 71	72: 72	73: 73	74: 74	75: 75	76: 76	77: 78	78: 79	79: 7R
80: 80	81: 81	82: 82	83: 83	84: 84	85: 85	86: 86	87: 87	88: 89	89: 8R
90: 90	91: 91	92: 92	93: 93	94: 94	95: 95	96: 96	97: 97	98: 98	99: 9R

Примеры

Пусть $n = 2$. Вот все 10^2 строк в лексикографическом порядке:

00: 01	01: 02	02: 03	03: 04	04: 05	05: 06	06: 07	07: 08	08: 09	09: 0R
10: 10	11: 12	12: 13	13: 14	14: 15	15: 16	16: 17	17: 18	18: 19	19: 1R
20: 20	21: 21	22: 23	23: 24	24: 25	25: 26	26: 27	27: 28	28: 29	29: 2R
30: 30	31: 31	32: 32	33: 34	34: 35	35: 36	36: 37	37: 38	38: 39	39: 3R
40: 40	41: 41	42: 42	43: 43	44: 45	45: 46	46: 47	47: 48	48: 49	49: 4R
50: 50	51: 51	52: 52	53: 53	54: 54	55: 56	56: 57	57: 58	58: 59	59: 5R
60: 60	61: 61	62: 62	63: 63	64: 64	65: 65	66: 67	67: 68	68: 69	69: 6R
70: 70	71: 71	72: 72	73: 73	74: 74	75: 75	76: 76	77: 78	78: 79	79: 7R
80: 80	81: 81	82: 82	83: 83	84: 84	85: 85	86: 86	87: 87	88: 89	89: 8R
90: 90	91: 91	92: 92	93: 93	94: 94	95: 95	96: 96	97: 97	98: 98	99: 9R

- Наблюдения:
  - чёрные номера k не изменились,
  - красные номера k увеличились,
  - для синих номеров k последний символ равен «R».

Решение

Наблюдения:

- Наш алфавит состоит из 11 символов. Вот они по порядку: `0123456789R`.
- В каждой позиции есть ровно 10 символов, которые могут встретиться:
  - в первой позиции и в позиции после символа «`R`» это цифры,
  - в позиции после цифры d это все цифры по порядку, кроме d, а за ними символ «`R`».
- Например, после цифры 4 алфавит для следующей позиции такой: `012356789R`.

Решение

Решение:

- Будем строить ответ слева направо.
- Пусть осталось поставить m цифр.
- Значит, всего есть 10^m вариантов достроить строку.
- Пусть алфавит для текущей позиции равен $(a_0, a_1, \ldots, a_9)$.
- Тогда:
  - в первых 10^{m-1} строках в текущей позиции стоит a_0,
  - в следующих 10^{m-1} строках в текущей позиции стоит a_1,
  - ...
  - в последних 10^{m-1} строках в текущей позиции стоит a_9.
- Чтобы выбрать один из десяти вариантов — смотрим, какая из десяти цифр стоит в нашем номере k на текущей позиции.

Решение

Пример работы решения:

- Пусть $n = 10$ и $k = 9900534448$.

позиция	цифра числа k	алфавит	символ в ответе
0	9	0123456789	9
1	9	012345678R	R
2	0	0123456789	0
3	0	123456789R	1
4	5	023456789R	6
5	3	012345789R	3
6	4	012456789R	5
7	4	012346789R	4
8	4	012356789R	5
9	8	012346789R	9

- Получился ответ 9R01635459.

Условие

Задача G. Чисел всё больше

- На доске написано целое число n. Алиса и Боб играют в игру, делая ходы по очереди. Начинает Алиса.
- За один ход можно стереть с доски одно любое число x. После этого, если x чётно, ход завершается. Если же x нечётно, нужно написать на доске все числа от 1 до $\lfloor x/2 \rfloor$ по одному разу, а затем ход также завершается.
- Чтобы выиграть, нужно оставить противнику пустую доску. Кто выиграет при правильной игре?
- Пример: $n = 11$.
$\underline{11} \xrightarrow{Alice} \underline{5}, 4, 3, 2, 1 \xrightarrow{Bob} 4, 3, 2, 1, \underline{2}, 1 \xrightarrow{Alice} 4, 3, 2, \underline{1}, 1 \xrightarrow{Bob}$
$4, 3, 2, \underline{1} \xrightarrow{Alice} \underline{4}, 3, 2 \xrightarrow{Bob} \underline{3}, 2 \xrightarrow{Alice} 2, \underline{1} \xrightarrow{Bob} \underline{2} \xrightarrow{Alice} .$

Условие

Задача G. Чисел всё больше

- На доске написано целое число n. Алиса и Боб играют в игру, делая ходы по очереди. Начинает Алиса.
- За один ход можно стереть с доски одно любое число x. После этого, если x чётно, ход завершается. Если же x нечётно, нужно написать на доске все числа от 1 до $\lfloor x/2 \rfloor$ по одному разу, а затем ход также завершается.
- Чтобы выиграть, нужно оставить противнику пустую доску. Кто выиграет при правильной игре?
- Подзадачи:
  - $n \leqslant 10^2$,
  - $n \leqslant 10^4$,
  - $n \leqslant 10^6$,
  - $n \leqslant 10^{12}$.

Медленные решения

Общие соображения:

- Каждое число, которое есть на доске, рано или поздно будет стёрто.
- В момент стирания однозначно определено, что появится.
- Получается, что выбор игроков не влияет на результат игры.
- Результат зависит только от чётности общего количества ходов в игре.
- Эту чётность мы и будем считать.

Медленные решения

Решение симуляцией:

- Пусть x_n — результат игры для числа n: $x_n = 1$, если количество ходов в этой игре нечётно (выигрывает Алиса), и $x_n = 0$, если оно чётно (выигрывает Боб).
- Напишем рекуррентную формулу:
  $x_{2k} = 1$,
  $x_{2k+1} = 1 \oplus x_k \oplus x_{k-1} \oplus \ldots \oplus x_1$.
  Здесь $\oplus$ — сложение по модулю 2.
  База: $x_1 = 1$.
- Вычисление x_n можно оформить в виде рекурсивной функции.

Медленные решения

Ускорение симуляции:

- Чтобы ускорить решение, можно заменить обычную рекурсию на рекурсию с запоминанием.
- Или завести массив для хранения $\{x_k\}$ и считать значения в нём динамическим программированием.
- Ещё идея ускорения: будем отдельно хранить префиксные суммы.
  Пусть $y_n = x_1 \oplus x_2 \oplus \ldots \oplus x_n$.
  Тогда $x_{2k+1} = 1 \oplus y_k$ и $y_k = x_k \oplus y_{k-1}$.
- Такое решение уже может посчитать ответ, используя $\mathcal{O}(n)$ времени и памяти.

Быстрое решение

Как получить решение за $\mathcal{O}(\mathrm{Poly}(\log n))$ (многочлен от логарифма n)? Вот один из способов.

- 1. Давайте внимательно посмотрим на x_{17}:
  $x_{17} = 1 \oplus x_8 \oplus x_7 \oplus x_6 \oplus x_5 \oplus x_4 \oplus x_3 \oplus x_2 \oplus x_1$.
- Распишем отдельные слагаемые:
  $$\begin{aligned} x_7 &= 1 \oplus x_3 \oplus x_2 \oplus x_1 \\ x_5 &= 1 \oplus x_2 \oplus x_1 \\ x_3 &= 1 \oplus x_1 \\ x_1 &= 1 \end{aligned}$$
- Их сумма равна $x_3 \oplus x_1$. В левом (синем) столбике — чётное количество единиц. Элементы с чётными индексами встретились чётное количество раз, то есть сократились. Элементы с нечётными индексами встретились нечётное количество раз, то есть по одному разу (красные) входят в результат.

Быстрое решение

Как получить решение за $\mathcal{O}(\mathrm{Poly}(\log n))$ (многочлен от логарифма n)? Вот один из способов.

- 1. Давайте внимательно посмотрим на x_{17}:
  $x_{17} = 1 \oplus x_8 \oplus x_7 \oplus x_6 \oplus x_5 \oplus x_4 \oplus x_3 \oplus x_2 \oplus x_1$.
- Запишем сумму $x_3 \oplus x_1$ как $x_4 \oplus x_3 \oplus x_2 \oplus x_1$.
  Действительно, мы добавили чётное количество слагаемых, равных единице.
- Заметим, что эта сумма, в свою очередь, равна $x_9 \oplus 1$.
  Действительно, ведь $x_9 = 1 \oplus x_4 \oplus x_3 \oplus x_2 \oplus x_1$.
- Получилось, что $x_{17} = 1 \oplus x_9 \oplus 1 = x_9$.

Быстрое решение

Как получить решение за $\mathcal{O}(\mathrm{Poly}(\log n))$ (многочлен от логарифма n)? Вот один из способов.

- 2. Давайте внимательно посмотрим на x_{19}:
  $x_{19} = 1 \oplus x_9 \oplus x_8 \oplus x_7 \oplus x_6 \oplus x_5 \oplus x_4 \oplus x_3 \oplus x_2 \oplus x_1$.
- Распишем отдельные слагаемые:

$$\begin{aligned}
x_9 &= {\color{blue}1} \oplus {\color{red}x_4} \oplus x_3 \oplus x_2 \oplus x_1 \\
x_7 &= {\color{blue}1} \qquad\;\; \oplus x_3 \oplus x_2 \oplus x_1 \\
x_5 &= {\color{blue}1} \qquad\qquad\quad\; \oplus {\color{red}x_2} \oplus x_1 \\
x_3 &= {\color{blue}1} \qquad\qquad\qquad\qquad\;\; \oplus x_1 \\
x_1 &= {\color{blue}1}
\end{aligned}$$

- Их сумма равна $1 \oplus x_4 \oplus x_2$. В левом (синем) столбике — нечётное количество единиц. Элементы с нечётными индексами встретились чётное количество раз, то есть сократились. Элементы с чётными индексами встретились нечётное количество раз, то есть по одному разу (красные) входят в результат.

Быстрое решение

Как получить решение за $\mathcal{O}(\mathrm{Poly}(\log n))$ (многочлен от логарифма n)? Вот один из способов.

- 2. Давайте внимательно посмотрим на x_{19}:
  $x_{19} = 1 \oplus x_9 \oplus x_8 \oplus x_7 \oplus x_6 \oplus x_5 \oplus x_4 \oplus x_3 \oplus x_2 \oplus x_1$.
- Заметим теперь, что $1 \oplus x_4 \oplus x_2 = 1$: это нечётное количество слагаемых, равных единице.
- Получилось, что $x_{19} = 1 \oplus 1 = 0$.

Быстрое решение

Как получить решение за $\mathcal{O}(\mathrm{Poly}(\log n))$ (многочлен от логарифма n)? Вот один из способов.

- 1'. В общем случае $x_{16k+1} = x_{8k+1}$. Например, $x_{33} = x_{17}$. Действительно: $x_{33} = 1 \oplus x_{16} \oplus x_{15} \oplus x_{14} \oplus x_{13} \oplus x_{12} \oplus x_{11} \oplus x_{10} \oplus x_9 \oplus x_8 \oplus x_7 \oplus x_6 \oplus x_5 \oplus x_4 \oplus x_3 \oplus x_2 \oplus x_1$.
- Распишем отдельные слагаемые:

$$
\begin{aligned}
x_{15} &= 1 \oplus x_7 \oplus x_6 \oplus x_5 \oplus x_4 \oplus x_3 \oplus x_2 \oplus x_1 \\
x_{13} &= 1 \oplus x_6 \oplus x_5 \oplus x_4 \oplus x_3 \oplus x_2 \oplus x_1 \\
x_{11} &= 1 \oplus x_5 \oplus x_4 \oplus x_3 \oplus x_2 \oplus x_1 \\
x_9 &= 1 \oplus x_4 \oplus x_3 \oplus x_2 \oplus x_1 \\
x_7 &= 1 \oplus x_3 \oplus x_2 \oplus x_1 \\
x_5 &= 1 \oplus x_2 \oplus x_1 \\
x_3 &= 1 \oplus x_1 \\
x_1 &= 1
\end{aligned}
$$

Быстрое решение

Как получить решение за $\mathcal{O}(\mathrm{Poly}(\log n))$ (многочлен от логарифма n)? Вот один из способов.

- 2'. В общем случае $x_{16k+3} = 0$. Например: $x_{35} = 0$. Действительно: $x_{35} = 1 \oplus x_{17} \oplus x_{16} \oplus x_{15} \oplus x_{14} \oplus x_{13} \oplus x_{12} \oplus x_{11} \oplus x_{10} \oplus x_9 \oplus x_8 \oplus x_7 \oplus x_6 \oplus x_5 \oplus x_4 \oplus x_3 \oplus x_2 \oplus x_1$.
- Распишем отдельные слагаемые:

$$
\begin{aligned}
x_{17} &= 1 \oplus x_8 \oplus x_7 \oplus x_6 \oplus x_5 \oplus x_4 \oplus x_3 \oplus x_2 \oplus x_1 \\
x_{15} &= 1 \oplus x_7 \oplus x_6 \oplus x_5 \oplus x_4 \oplus x_3 \oplus x_2 \oplus x_1 \\
x_{13} &= 1 \oplus x_6 \oplus x_5 \oplus x_4 \oplus x_3 \oplus x_2 \oplus x_1 \\
x_{11} &= 1 \oplus x_5 \oplus x_4 \oplus x_3 \oplus x_2 \oplus x_1 \\
x_9 &= 1 \oplus x_4 \oplus x_3 \oplus x_2 \oplus x_1 \\
x_7 &= 1 \oplus x_3 \oplus x_2 \oplus x_1 \\
x_5 &= 1 \oplus x_2 \oplus x_1 \\
x_3 &= 1 \oplus x_1 \\
x_1 &= 1
\end{aligned}
$$

Быстрое решение

Как получить решение за $\mathcal{O}(\mathrm{Poly}(\log n))$ (многочлен от логарифма n)? Вот один из способов.

- Можно аналогично расписать x_{16k+5}, x_{16k+7}, x_{16k+9}, x_{16k+11}, x_{16k+13} и x_{16k+15}. Тогда получится решение, работающее за $\mathcal{O}(\log n)$.
- Вместо этого можно ограничиться следующими формулами:
  $x_1 = 1$,
  $x_{2k} = 1$,
  $x_{16k+1} = x_{8k+1}$,
  $x_{16k+3} = 0$,
  $x_{16k+2d+1} = x_{8k+d} \oplus x_{16k+2d-1}$ при $d \geqslant 2$.
- Такое решение уже работает за $\mathcal{O}(\mathrm{Poly}(\log n))$, если реализовать его в виде рекурсии с запоминанием.

Условие

Задача H. Фокус с цифрами

- Это задача с двойным запуском.
- Фокусник и ассистент показывают фокус.
- Ассистент получает строку из 10 десятичных цифр.
- Ассистент заменяет одну цифру на вопросительный знак.
- Фокусник получает строку с заменённой цифрой.
- Фокусник должен сказать сумму всех цифр в исходной строке.

Условие

Задача H. Фокус с цифрами

- Это задача с двойным запуском.
- Фокусник и ассистент показывают фокус.
- Ассистент получает строку из 10 десятичных цифр.
- Ассистент заменяет одну цифру на вопросительный знак.
- Фокусник получает строку с заменённой цифрой.
- Фокусник должен сказать сумму всех цифр в исходной строке.
- Подзадачи:
  - Исходные цифры — только 0 и 1.
  - Исходные цифры любые.

Условие

Задача H. Фокус с цифрами

- Это задача с двойным запуском.
- Фокусник и ассистент показывают фокус.
- Ассистент получает строку из 10 десятичных цифр.
- Ассистент заменяет одну цифру на вопросительный знак.
- Фокусник получает строку с заменённой цифрой.
- Фокусник должен сказать сумму всех цифр в исходной строке.
- Подзадачи:
  - Исходные цифры — только 0 и 1.
  - Исходные цифры любые.
- Пример 1: $0000011111 \xrightarrow{?} 0000011?11 \xrightarrow{?} 5$.
- Пример 2: $3791355645 \xrightarrow{?} 379135?645 \xrightarrow{?} 48$.

Двоичные цифры

Как решить задачу с двоичными цифрами?

- Пусть позиция для замены — это сумма всех цифр.
  Будем нумеровать позиции с нуля.
- Но позиций 10, а возможных сумм 11: от 0 до 10 включительно.
- Можно, например, закодировать сумму 10 позицией 0.
  Тогда легко отличить случаи «все нули, сумма 0» и «все единицы, сумма 10» по оставшимся цифрам.
- Пример: 0000011111 $\to$ 00000?1111 $\to$ 5.

Десятичные цифры

Как решить задачу в общем виде?

- Попробуем реализовать ту же идею: позиция для замены — сумма всех цифр.
- Но позиций 10, а возможных сумм 91: от 0 до 90 включительно.
- Какую часть суммы мы можем передать номером позиции? Передадим остаток от деления суммы на 10.
- Оказывается, этого достаточно: все 10 возможных цифр приводят к различным остаткам от деления общей суммы на 10. Мы знаем, какой остаток правильный. Из десяти возможных сумм выберем нужную.
- Пример: пусть дана строка 3791355645.
  - Первый запуск: сумма цифр равна 48, позиция равна $48 \bmod 10 = 8$. Передаём строку 37913556?5.
  - Второй запуск: сумма оставшихся цифр равна 44, позиция равна 8. Мы ищем такое s, что $44 \leqslant s < 54$ и $s \bmod 10 = 8$. Это число 48.

Условие

Задача I. Прямоугольник из хаоса

- Рассмотрим точки с целыми координатами от 0 до 10^5 включительно.
- Даны n (от 1 до 300 000) таких точек.
- Точки выбраны случайно, равномерно и независимо.
- Найдите прямоугольник максимальной площади, стороны которого параллельны осям координат, а все четыре вершины лежат в выбранных точках.

Условие

Задача I. Прямоугольник из хаоса

- Рассмотрим точки с целыми координатами от 0 до 10^5 включительно.
- Даны n (от 1 до 300 000) таких точек.
- Точки выбраны случайно, равномерно и независимо.
- Найдите прямоугольник максимальной площади, стороны которого параллельны осям координат, а все четыре вершины лежат в выбранных точках.
- Подзадачи:
    - $n \leqslant 100$,
    - $n \leqslant 300\,000$.

Наивные решения

Важное свойство задачи: заданные точки выбраны случайно.

- Наивное решение:
    - переберём четвёрки точек,
    - проверим на прямоугольность,
    - найдём максимум.
- Время работы: $\mathcal{O}(n^4)$.

Наивные решения

Важное свойство задачи: заданные точки выбраны случайно.

- Ещё более наивное решение: выведем 0.
- Почему при $n \leqslant 100$ это, скорее всего, правда?
- Чтобы прикинуть вероятность, разберём другое решение.

Полное решение

Важное свойство задачи: заданные точки выбраны случайно.

- Полное решение:
- • Для каждого x заведём список точек с таким x.
- • Для каждого y заведём список точек с таким y.
- • Положим все точки в `set` S.
- • Для каждой точки (x, y) — первой вершины прямоугольника:
- • • Для каждой второй вершины вида (x, y'):
- • • • Для каждой третьей вершины вида (x', y):
- • • • • Если четвёртая вершина (x', y') есть в S:
- • • • • • Вычислим площадь $|x' - x| \cdot |y' - y|$, обновим максимум.

Полное решение

Важное свойство задачи: заданные точки выбраны случайно.

- Анализ. Пусть $C = 100\,001$ — диапазон возможных координат.
- • Для каждой точки (x, y) — первой вершины прямоугольника:
  • (n вариантов)
- • • Для каждой второй вершины вида (x, y'):
  • • (в среднем будет $(n-1)/C$ таких точек)
- • • • Для каждой третьей вершины вида (x', y):
  • • • (в среднем будет $(n-1)/C$ таких точек)
- • • • • Если четвёртая вершина (x', y') есть в S:
  • • • • (проверка за $\mathcal{O}(\log n)$ или $\mathcal{O}(1)$)
- • • • • • Вычислим площадь $|x' - x| \cdot |y' - y|$, обновим максимум.
  • • • • • (вычисление за константу)
- Порядок можно грубо оценить, перемножив эти числа (хотя для точной оценки это некорректно). Получаем, что мы проверим существование $\approx n^3/C^2$ четвёртых вершин прямоугольника.

Полное решение

Важное свойство задачи: заданные точки выбраны случайно.

- Итак, грубая оценка: мы проверим существование $\approx n^3/C^2$ четвёртых вершин прямоугольника.
- Когда $n = 100$ и $C = 100\,001$, получаем $n^3/C^2 \approx 10^{-6}$.
  И каждая такая вершина есть с вероятностью $n/C^2 \approx 10^{-8}$.
  Значит, ответ будет больше нуля с вероятностью 10^{-14}.
  Действительно, ответ почти всегда равен нулю.
- Когда $n = 300\,000$ и $C = 100\,001$, получаем $n^3/C^2 \approx 27 \cdot 10^5$.
  И каждая такая вершина есть с вероятностью $n/C^2 \approx 3 \cdot 10^{-5}$.
  Значит, в среднем (по нашей грубой оценке) найдётся 81 прямоугольник.
- Время работы: когда $n = 300\,000$ и $C = 100\,001$, получаем $n^3/C^2 \approx 27 \cdot 10^5$ обращений к `set`.

Условие

Задача J. Кольцевой маршрут

- Это задача с двойным запуском.
- Дано $n \leqslant 50\,000$. У жюри есть секретный ориентированный цикл на числах $1, 2, \ldots, n$. Цикл зафиксирован заранее.
- Решение может задать числа $k \leqslant 1000$ и $l \leqslant 1000$, а также k начальных позиций.
- После этого жюри от каждой начальной позиции выдаст l следующих вершин цикла.
- Наконец, решение должно восстановить весь цикл.
- Пример: $n = 5$, цикл $1 \to 3 \to 5 \to 4 \to 2 \to 1$.
  - Участник задаёт $k = 2$, $l = 3$ и позиции 1 и 2.
  - Жюри выдаёт части цикла: $1 \to 3 \to 5$ и $2 \to 1 \to 3$.
  - Участник по ним восстанавливает весь цикл.

Условие

Задача J. Кольцевой маршрут

- Это задача с двойным запуском.
- Дано $n \leqslant 50\,000$. У жюри есть секретный ориентированный цикл на числах $1, 2, \ldots, n$. Цикл зафиксирован заранее.
- Решение может задать числа $k \leqslant 1000$ и $l \leqslant 1000$, а также k начальных позиций.
- После этого жюри от каждой начальной позиции выдаст l следующих вершин цикла.
- Наконец, решение должно восстановить весь цикл.
- Подзадачи:
    - $n \leqslant 1000$,
    - $n \leqslant 50\,000$.

Нерешаемая задача

Очевидно, задача не имеет решения.

- Пусть $n = 50\,000$, а решение спросит какие-то 1000 позиций.
- Может оказаться, что они все расположены подряд в цикле.
- Тогда решение узнает порядок не более чем для 2000 вершин, и не узнает, в каком порядке расположены остальные 48 000.
- Что же делать?

Вероятностные решения

Чтобы справиться с задачей, чуть понизим требования к тому, что такое «решение».

- Пусть нашему решению можно работать не всегда, а почти всегда — с какой-то вероятностью, близкой к единице.
- Обычно в задаче десятки тестов, максимум — сотни.
- Если вероятность успеха велика — неважно, что существуют тесты, на которых решение не работает. Важно, чтобы на любом зафиксированном заранее тесте оно работало с хорошей вероятностью.

Решение

Зафиксируем $k = 1000$ и $l = 1000$. Выберем k случайных позиций. Будем надеяться, что k полученных путей содержат каждое ребро цикла хотя бы по разу.

- Как восстанавливать цикл?
- Начнём с какой-нибудь выбранной позиции p.
- Будем идти по полученному пути из p, пока не придём в другую выбранную позицию q.
- После этого будем идти по пути из q, пока не придём в очередную выбранную позицию r.
- И так далее, пока не вернёмся в p.

Решение

Зафиксируем $k = 1000$ и $l = 1000$. Выберем k случайных позиций. Будем надеяться, что k полученных путей содержат каждое ребро цикла хотя бы по разу.

- Почему это вообще может сработать?
- Представим себе цикл как окружность с n точками.
- Мы выбрали на этой окружности k случайных точек.
- Мы не знаем порядок точек, но знаем, что выбирали каждую точку C равномерно: вероятность того, что C равна какой-то конкретной точке A, не зависит от A и равна $1/n$.
- Чтобы успешно восстановить цикл, нужно, чтобы все расстояния по циклу между соседними выбранными точками были меньше l.

Решение

Зафиксируем $k = 1000$ и $l = 1000$. Выберем k случайных позиций. Будем надеяться, что k полученных путей содержат каждое ребро цикла хотя бы по разу.

- Предположим, что наше решение не работает: на окружности есть «плешь» размера хотя бы l, в которую не попала ни одна из наших точек.
- Значит, каждая из наших точек попала в одну из $(n - l)/n$ оставшихся позиций. Если мы выбирали точки случайно и равномерно — вероятность этого равна $(n - l)/n$ для каждой из k выбранных точек.
- Вероятность того, что все точки одновременно не попали в «плешь», равна произведению отдельных вероятностей: $((n - l)/n)^k$.

Решение

Зафиксируем $k = 1000$ и $l = 1000$. Выберем k случайных позиций. Будем надеяться, что k полученных путей содержат каждое ребро цикла хотя бы по разу.

- В выражение $((n - l)/n)^k$ подставим конкретные числа: $((50\,000 - 1000)/50\,000)^{1000} = 0.98^{1000} \approx 1.683 \cdot 10^{-9}$.
- Даже если, грубо оценивая, считать вероятность каждой из n возможных «плешей» отдельно, получаем $\approx 1.683 \cdot 10^{-9} \cdot 50\,000 = 8.415 \cdot 10^{-5}$.
- Итак, вероятность того, что наше решение не работает, на каждом тесте точно меньше 10^{-4}. Для задачи, по которой есть 99 попыток, это вполне приемлемо.

Автор задач и разбора:

- Иван Казменко

Помощь в подготовке:

- Наталья Гинзбург
- Константин Дроздов
- Владислав Макаров

Видеоразбор:

- Александр Савченко

Решения и проверяющие программы для этой олимпиады написаны на языке D (`https://dlang.org`).