Разбор задач

Открытый турнир по спортивному программированию на призы Главы Республики Мордовия 2023

Открытый турнир

по спортивному программированию на призы Главы Республики Мордовия

24 марта 2023

Задача А. Плейлист

Необходимо проверить, что ни в одном блоке подряд идущих m чисел нет повторяющихся.

Для этого достаточно для каждого числа сохранить все его позиции и проверить, что разница между позициями >=m.

Так как ограничения на сами числа 1’000’000, можно использовать обычный массив, который будет хранить последнюю позицию числа. Когда мы обрабатываем текущее число, находим разницу между текущей позицией и последней встреченной.

Итоговая сложность O(n).

Задача B. Красивые числа

Жадный алгоритм

Сгенерируем все числа вида 3^a*5^b, не превосходящие n. Таких даже при n = 10^18 примерно 500

Отсортируем по убыванию. Жадно будем добавлять очередное слагаемое, если это возможно

Доказательство

База: на практике проверим верность для n <= 81

Если верно для чисел от 1 до x и x вида 3^a*5^b, то верно и для чисел от 1 до x * 2 - 1

Осталось доказать, что среди чисел от x + 1 до x * 2 - 1 есть хотя бы одно вида 3^a*5^b и тогда утверждение верно для всех чисел

Если x делится на 5, то можно взять число x * 9 / 5. Если нет, то число x / 81 * 125

Чтд.

Задача C. Три товарища

Если общее количество конфет S не делится на 3, то ответ “NO”.
Пусть $S1 = S / 3$, $k2$ – количество всех коробок с 2 конфетами, а $k3$ - количество всех коробок с 3 конфетами.
Мы можем найти все решения уравнения:

$$2x + 3y = S1$$

Получили некоторое множество пар чисел M.
Переберем всевозможные тройки пар из M.
Если мы нашли из этого множества три пары $M1(x1, y1)$, $M2(x2, y2)$, $M3(x3, y3)$, такие, что $x1 + x2 + x3 = k2$ и $y1 + y2 + y3 = k3$, то ответ “YES”, в противном случае ответ “NO”.

Задача D. Дедушка и внук

То, что среди этих чисел есть хотя бы две пары одинаковых – это очень сильно облегчает решение задачи: нужно найти первые два одинаковых числа и, естественно, запомнить это значение. Тогда первое число, отличное от этих двух одинаковых и даст нам первый ответ: наименьший период **а**. Теперь «двигаемся» дальше по массиву и «пропускаем» числа, которые отличаются от предыдущих на только что найденное значение **а**. А как только встретится число, у которого отличие от предыдущего будет иное, то разность между ним и тем, которое мы запомнили как два одинаковых, даст нам второй ответ.

Задача E. Проще простого

«Прямое» моделирование процесса поиска количества простых для каждого натурального i на отрезке [1; i] слишком долго. Даже сохранение текущего количества простых чисел на таком отрезке даст нам сложность $n\sqrt{n}$. Задача решается с помощью Решета Эратосфена и, при данных ограничениях, в несложном классическом варианте: составим список всех чисел от 1 до n и будем изначально считать их все простыми; далее будем брать очередное число из такого списка простым (на первом шаге – это 2) и вычеркивать все последующие числа, кратные ему; когда мы дойдем до $\sqrt{n}$, список будет содержать только простые числа (на отрезке [1; i]). После этого «зайдет» любое решение поиска простых в полученном списке. Можно ускорить такое решение сделав поиск бинарным.

Задача F. Номер цифры

То, что число **n** не очень большое (всего каких-то 59 тысяч с «хвостиком») должно было натолкнуть на мысль: просто вычислить явно эту строку, а потом просто напечатать символ с нужным номером (правда, в Паскале нумерация начинается с 1, а в Питоне и C++ с 0). Текст программы на Питоне:

```
x = '00'
s = '123456789'
n = int(input())
for i in s:
    y = i*2
    for j in range(len(x)):
        y = y + x[j] + i*2
    x = y
print(x[n-1])
```

Задача G. Лазерное обнаружение

Заметим, что любая точка (x, y) преобразуется в множество точек:

- (x + 2n * i, y + 2m * j)
- (-x + 2n * i, y + 2m * j)
- (x + 2n * i, -y + 2m * j)
- (-x + 2n * i, -y + 2m * j)

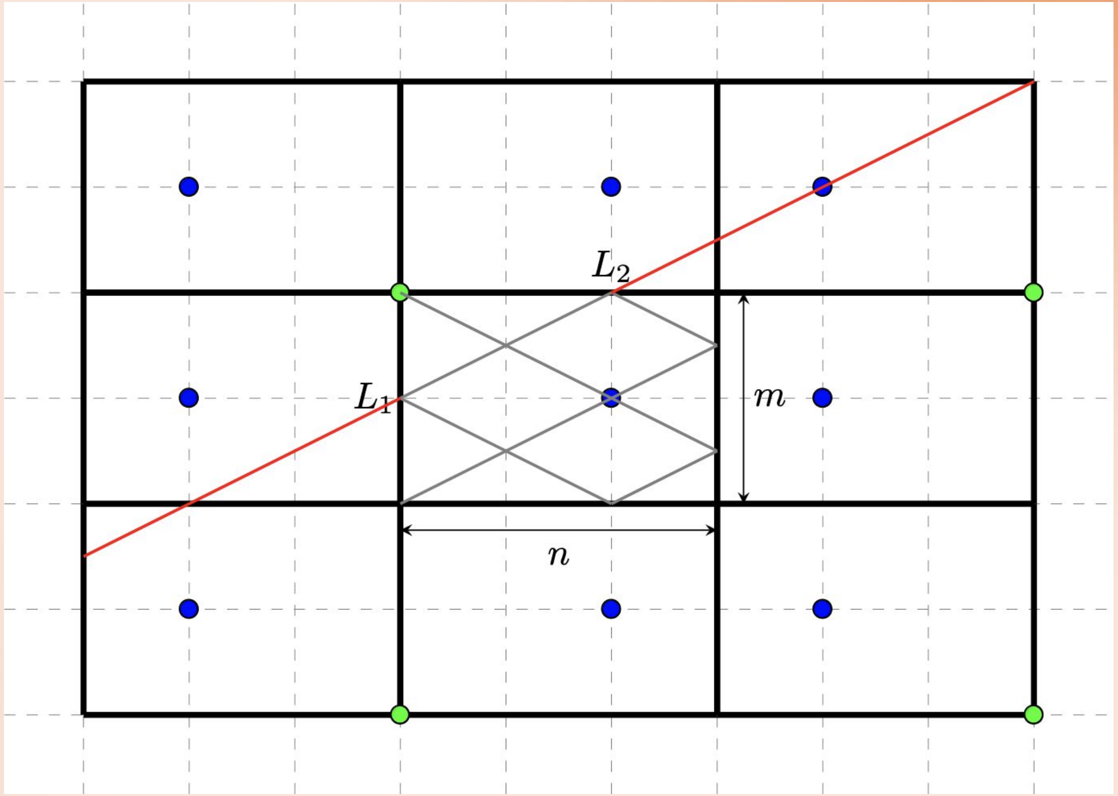
где i, j – любые целые числа.

При помощи точек L1 и L2 найдем коэффициенты A, B, C в уравнении прямой лазера Ax + By + C = 0.

Осталось проверить, есть ли решения у уравнения:

- A(x + 2n * i) + B(y + 2m * j) + C = 0

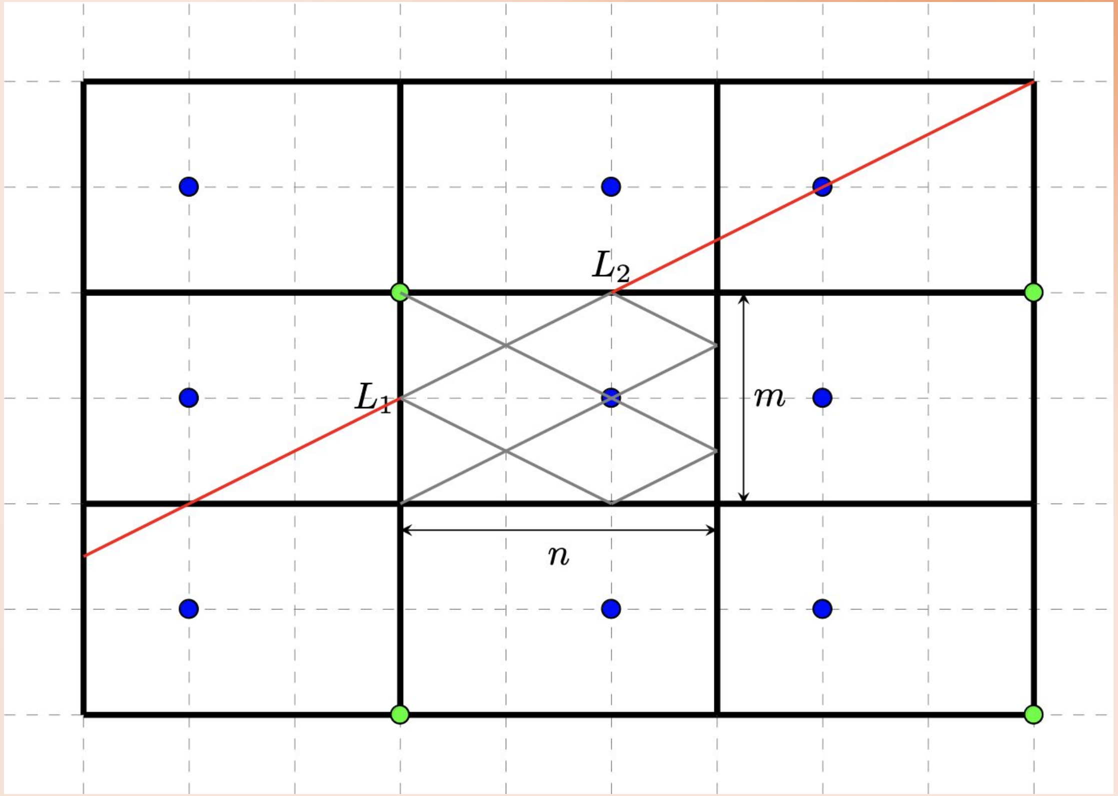
при каких-то целых числах i, j.



Задача G. Лазерное обнаружение

Перепишем уравнение:

i * (2 * A * n) + **j** * (2 * B * m) +
 \+ (C + A * x + B * y) = 0



Обозначим:

- Q = (2 * A * n)
- P = (2 * B * m)
- T = (C + A * x + B * y)

Получили:

i * Q + **j** * P + T = 0

Уравнение имеет решение тогда и только тогда, когда:

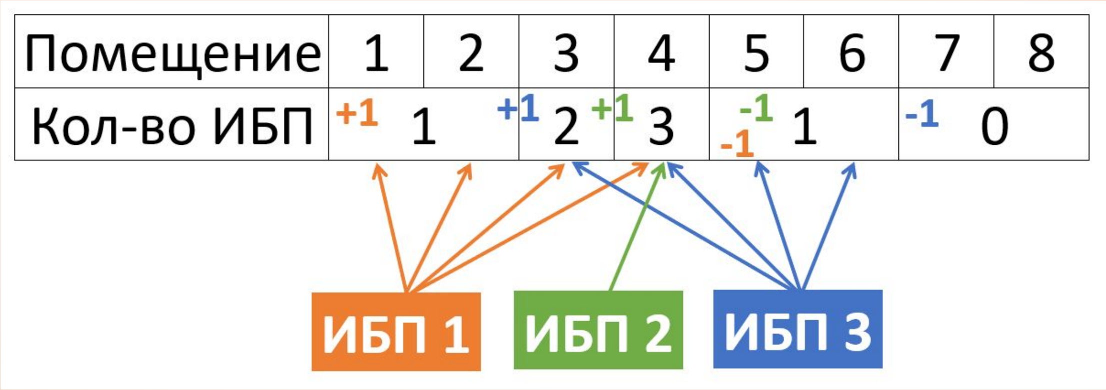
- |T| делится на НОД(|Q|, |P|).

Задача H. Камеры

Разобьём все помещения в здании на не более чем m * 2 групп и про каждую скажем, сколько ИБП могут обеспечить работу камер в этой группе помещений.
Используем метод сканирующей прямой – будем рассматривать события в отсортированном по связанному с этим событием номеру помещения порядке.
Поясним на следующем примере.

Задача H. Камеры

Пример для одного здания с 7-ю помещениями. В здании ИБП 1 может обеспечить помещения с 1-го по 4-е, ИБП 2 – с 3-го по 3-е, ИБП 3 – с 3-го по 6-е.



Отсортированные события:

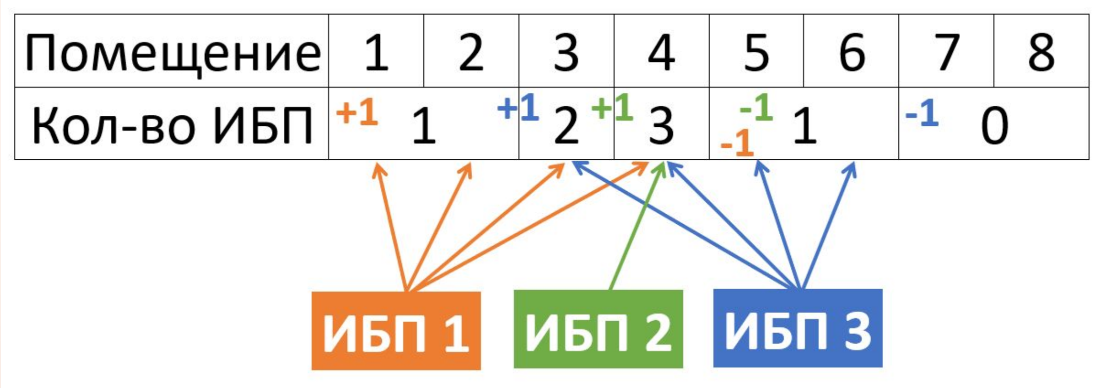
Начиная с 1-го помещения можно использовать ИБП 1 (+1 к количеству подключенных ИБП)

Начиная с 2-го помещения можно использовать ИБП 3 (+1 к количеству подключенных ИБП)

Начиная с 3-го помещения можно использовать ИБП 2 (+1 к количеству подключенных ИБП)

Начиная с 5-го помещения нельзя использовать ИБП 1 (-1 к количеству подключенных

Задача H. Камеры



Таким образом, все помещения в здании разбились на не более чем 2 * m + 1 групп, где m – количество ИБП в здании. К каждому помещению в рамках одной группы подключены одни и те же ИБП. Для каждой группы можно определить количество подключенных к помещениям внутри группы ИБП, используя количество подключенных ИБП к помещениям предыдущей группы и события, связанные с первым по номеру помещением в группе. Получившиеся данные можно использовать для определения того, возможно ли проникновения в помещения той или иной группы: если количество связанных с помещениями в группе ИБП больше максимального количества ИБП, при несрабатывании которых опасность не регистрируется, то проникновение в помещения этой группы невозможно, иначе – возможно.

Задача I. Бактеринно

Для каждой колонии l найдём max_r, при котором всё ещё можно перегородки снять.

Заметим, что max_r для l+1 больше или равен max_r для l. Следовательно можно применить метод двух указателей.

Для проверки можно ли увеличивать текущее значение max_r проще всего с использованием битовых операций. Но можно и перевести все числа в двоичную систему счисления и работать со строками/массивами из 0 и 1.

Задача J. Строченьки

Переберем все строки из набора и посмотрим, подходят ли они под критерии 1, 2, 3 условия задачи. Все подходящие строки запомним в формате: «a = какой символ стоит между s1 и s2; b = длина подстроки между s1 и s2». Заведем массив set'ов cnt[a] = {b1, b2, …, bk}. Для каждого символа c от 'a' до 'z' найдем минимальное число x, что строка вида s1+(c*x)+s2 не встречается в наборе (через + обозначена конкатенация строк, * обозначено повторение символа c x раз). Для этого пройдемся по cnt[c] и найдем первое число >=1, которое там не встречается. В ответ запишем минимальное из таких чисел. Случай с равенством можно отдельно не рассматривать, т.к. в порядке символов от 'a' до 'z' это условие и так будет соблюдено. Итоговая сложность – O(S+n), где S – суммарная длина строк в наборе, а n – количество строк в наборе.

Задача K. Алиса и медведь

Необходимо найти количество урона, которое нанесет медведь.

Так как ограничение на здоровье персонажа 4e18, простым перебором найти урон не получается.

Заметим, что количество урона, нанесенного медведем за n ударов, равно $n*(n + 1) / 2$ (сумма арифметической прогрессии).

Чтобы найти минимальное такое n, что урон будет не меньше здоровья персонажа, можно воспользоваться бинарным поиском.

Итоговая сложность $O(\log x)$

Задача L. Аэропорты

- Краткое условие: Дан граф, выделены 4 вершины. Необходимо покрасить некоторые ребра так, чтобы между любыми двумя выделенными вершинами был путь только по крашеным ребрам. Какое минимальное количество ребер нужно покрасить, чтобы выполнить условие?
- Идея: найти кратчайший путь, который будет соединять все выделенные 4-ре вершины.

Открытый турнир по спортивному программированию на призы Главы Республики Мордовия

Спасибо за внимание!