Задача A. Смена стиля

Автор задачи: Николай Будин, разработчик: Ильдар Загретдинов

Одно из решений представляет из себя построчное считывание данных, то есть считываем очередную строку и изменяем ее по следующим правилам: если есть заглавная буква, которая находится в начале строки, то заменяем ее на последовательность из символа «_» и соответствующей строчной буквы, если же она в начале строки, то только на соответствующую строчную букву. После этой операции выводим измененную строку.

Задача B. Результаты контеста

Автор задачи: Николай Будин, разработчик: Григорий Хлытин

Заметим, что мы можем игнорировать посылки, которые содержат вердикт `CE` или посылки по задаче, которая уже была сдана. Давайте для каждой задачи хранить количество неудачных попыток и была ли она сдана к текущему моменту. Таким образом, если мы видим первую успешную посылку по задаче, мы прибавим к ответу штраф за эту задачу, вычисленный по заданной формуле $t + 20 \cdot k$, где t — время первой успешной сдачи задачи в минутах, k — количество неправильных попыток перед первой успешной сдачей.

Задача C. Собака, предатель и кабеля

Автор задачи: Владислав Кузнецов, разработчик: Дмитрий Гнатюк

Решим задачу методом динамического программирования. Посчитаем для каждой клетки следующее значение — сколько проводов может максимум съесть собака на пути до этой клетки. Обозначим это значение за $\mathtt{dp}[i][j] = \max(\mathtt{dp}[i-1][j] + x, \mathtt{dp}[i][j-1] + y)$, где x и y обозначают наличие или отсутствие провода между соответствующими клетками.

Теперь для каждого расположения человека посчитаем сколько ходов сделает собака. Это можно сделать простой формулой: если человек был в клетке x, y, то собака сделает $\left\lfloor \frac{x+y+1}{2} \right\rfloor$ ходов.

Осталось только посчитать максимум динамики на соответствующей диагонали, не забывая, что собака не может пройти выше или правее изначального положения человека. Ограничения в задаче позволяют считать этот самый максимум за линейное время, что дает нам полное решение задачи.

Задача E. Настройка коммуникаций

Автор задачи и разработчик: Даниил Орешников

Сразу стоит отметить, что эту задачу достаточно трудно решить, находя приблизительные значения a, b и c как $\frac{xy}{z}$, $\frac{xz}{y}$ и $\frac{yz}{x}$, а затем делая локальный поиск вокруг полученных значений (границы могут быть слишком широкие).

Давайте, не теряя общности, считать, что $x \leqslant y \leqslant z$, иначе их можно переставить и вернуть обратно перед выводом ответа. Выпишем неравенства, которые задаются взятием целой части: $\sqrt{ab} \in [x, x+1)$, $\sqrt{ac} \in [y, y+1)$, $\sqrt{bc} \in [z, z+1)$. Теперь, выражая a как $\frac{\sqrt{ab}\sqrt{ac}}{\sqrt{bc}}$, мы можем получить следующее утверждение:

$$a \in \left(\frac{xy}{z+1}, \frac{(x+1)(y+1)}{z}\right)$$

Следующее наблюдение позволяет понять, что перебор значений a занимает константное время работы:

$$\frac{(x+1)(y+1)}{z} - \frac{xy}{z+1} = \frac{(x+1)(y+1)(z+1) - xyz}{z(z+1)} =$$
$$= \frac{xy + xz + yz + x + y + z + 1}{z(z+1)} = \frac{xy}{z(z+1)} + \frac{x+y}{z} + \frac{1}{z} \leqslant 4$$

Давайте переберем значения a в этом интервале и проверим для каждого, существуют ли подходящие b и c. Для того, чтобы перебор значений b и c занимал меньше времени, уточним границы для них. Для заданного a_0:

- $b \in \left[\frac{x^2}{a_0}, \frac{(x+1)^2}{a_0}\right)$
- $c \in \left[\frac{y^2}{a_0}, \frac{(y+1)^2}{a_0}\right)$
- и при этом $bc \in [z^2, (z+1)^2)$

Давайте представим, что мы выбрали b_0 из данного интервала и хотим проверить, существует ли подходящее c. У нас есть два условия на c:

- $c \in \left[\frac{y^2}{a_0}, \frac{(y+1)^2}{a_0}\right)$
- $c \in \left[\frac{z^2}{b_0}, \frac{(z+1)^2}{b_0}\right)$

Необходимым и достаточным условием существования подходящего c является факт того, что эти два полуинтервала пересекаются, и их пересечение содержит хотя бы одно целое число. Давайте выпишем условие на пересечение полуинтервалов, оно заключается в неравенствах на начало одного и конец другого интервала:

- $\frac{y^2}{a_0} \leqslant \frac{(z+1)^2}{b_0} \implies b_0 \leqslant \frac{a_0(z+1)^2}{y^2}$
- $\frac{z^2}{b_0} \leqslant \frac{(y+1)^2}{a_0} \implies b_0 \geqslant \frac{a_0 z^2}{(y+1)^2}$

Получаем следующее ограничение на b_0:

$$b_0 \in \left[\mathtt{max}\left(\frac{a_0 z^2}{(y+1)^2}, \frac{x^2}{a_0}\right), \mathtt{min}\left(\frac{a_0(z+1)^2}{y^2}, \frac{(x+1)^2}{a_0}\right)\right]$$

Перебор значений b_0 в данном интервале также за константное время дает возможность восстановить подходящий c, если он существует. Константа получается достаточно большая, но 10^5 тестов можно пройти за секунду с лишним. Для большей уверенности можно было применить оптимизации вида «перебирать интервал не от края до края, а начиная с середины».

Следует также отметить, что с некоторого момента все вычисления надо производить в `long double` (если вы не используете Python), потому что целочисленные типы данных начинают переполняться. Это добавляет некоторые неточности в вычисления, поэтому имело смысл сдвигать границы на ± 1, чтобы не пропустить подходящее значение из-за погрешностей.

Задача F. Электронный замок

Автор задачи: Николай Будин, разработчик: Арсений Кириллов

Заметим, что мы всегда хотим получить более длинное число, так как длинное число больше короткого. А это значит, что на каждую цифру мы хотим включать как можно меньше сегментов. Меньше всего сегментов у цифры 1, их всего два. А значит длина итого числа будет $\lfloor \frac{n}{2} \rfloor$. Однако если число n нечётное, то у можно включить ещё один сегмент. Тогда надо превратить одну из единиц в семёрку, и разумеется, чтобы получить максимальное число, мы превратим первую единицу в семёрку. И того для чётных n мы получим число $111\ldots111$, а для нечётных число $7111\ldots111$.

Задача G. Место преступления

Автор задачи: Степанов Семен, разработчик: Николай Будин

Для решения этой задачи, нужно разобрать несколько случаев.

Если $n = 3$, то у итогового многоугольника тоже будет 3 вершины.

Если $n = 4$, то всегда тоже можно построить многоугольник с 3 вершинами. В качестве одной из вершин можно выбрать одну из вершин четырехугольника, а две другие будут лежать на лучах, образуемых сторонами четырехугольника. См. рис. 1.

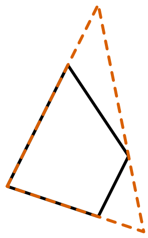
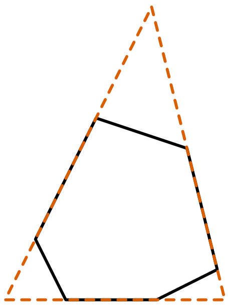
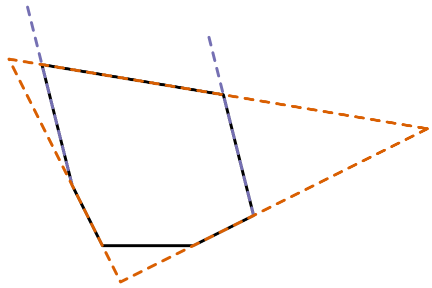


Рис. 1: Пример решения для $n = 4$.

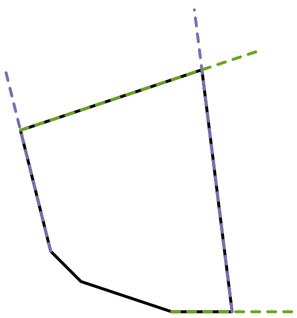
Если n четное. Рассмотрим стороны многоугольника через один. Попробуем пересечь соседние пары выбранных. Если получилось, это ответ (см. рис. 2a). Нельзя получить меньше, потому что на каждой стороне итогового многоугольника не может лежать больше 2 вершин исходного. Если не получилось, попробуем рассмотреть стороны с номерами другой четности (см. рис. 2b). Если снова не получилось, в многоугольнике есть четыре последовательные стороны, такие что первая и третья не пересекаются, и вторая и четвертая не пересекаются (см. рис. 2c). В таком случае, можно построить ответ, содержащий $\frac{n}{2} + 1$ вершин (см. рис. 2d).



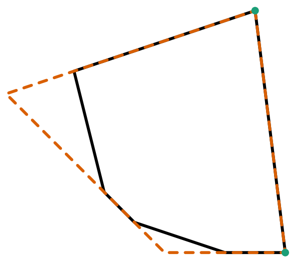
(a) Пример решения для четного n, если стороны через один пересеклись.



(b) Пример решения для четного n, если стороны через один не пересеклись, но стороны с номерами другой четности пересеклись. На рисунке сиреневые лучи не пересекаются.



(c) Случай, при котором не существует ответа $\frac{n}{2}$. Сиреневые лучи не пересекаются. Зеленые лучи не пересекаются



(d) Решение с $\frac{n}{2}+1$ вершиной. Нужно взять две вершины на многоугольнике в том месте, где лежат четыре плохих стороны, и далее взять стороны через одну.

Рис. 2: Случаи с четным n.

Если n нечетное, всегда можно построить ответ, содержащий $\lceil \frac{n}{2} \rceil$ вершин. Даже если есть четыре плохие последовательные стороны, как на рисунке 4 (см. рис. 3).

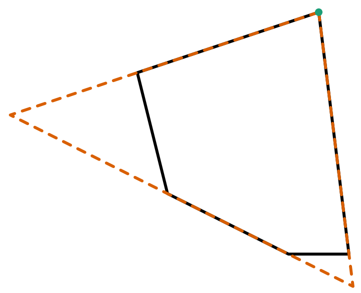


Рис. 3: Пример решения для нечетного n, когда есть четыре плохие последовательные стороны.

Задача H. Похожие имена

Автор задачи: Николай Будин, разработчик: Даниил Орешников

Первым действием стоит применить стандартный трюк, который используется, когда надо работать с циклическими сдвигами строки s — взять конкатенацию строки с самой собой (ss) и рассматривать все ее подстроки длины $|s|$. В частности, если нас интересует префикс некоторого циклического сдвига, то в строке ss нас просто интересуют любые подстроки длины не больше $|s|$.

С меньшими ограничениями эту задачу можно было бы решить следующим образом: перебрать циклический сдвиг самой короткой строки и применить префикс-функцию. С имеющимися ограничениями задачу можно было решить, применив хеши или суффиксный массив и алгоритм Касаи. Далее будет приведено решение с суффиксным массивом.

Давайте сконкатенируем вместе продублированные строки из ввода, разделив их специальным символом, не встречающимся в алфавите (например, `'$'`). Заведем также массив `from`, отвечающий за номер исходной строки, которой соответствует каждый символ конечной строки, полученной в результате конкатенации.

Построим суффиксный массив. Заметим, что если ответ на задачу — a, то в полученной строке должны быть одинаковые подстроки длины a. Переформулируем это как «в строке есть суффиксы с общим префиксом длины a, с началами в каждой из исходных строк». Если применить алгоритм Касаи, мы сможем посчитать `lcp` — наибольший общий префикс соседних суффиксов в суффиксном массиве. Получается, что ответ — это такое a, что в массиве `lcp` есть отрезок лежащих подряд (потому что суффиксный массив лексикографически отсортирован) значений $\geqslant a$, соответствующие которым индексы из суфф. массива покрывают все множество исходных строк.

Для поиска максимального подходящего a можно воспользоваться двоичным поиском. Внутри цикла для конкретного значения a_0 достаточно пройтись по массиву `lcp` и для каждого отрезка из значений не менее a_0 проверить, что значения `from` на нем покрывают все множество $1 \ldots n$. Также стоит искусственно ограничить ответ сверху минимальной из длин исходных строк, чтобы избежать ситуации, когда трюк с конатенацией увеличил потенциальный ответ.

Время работы решения — $\mathcal{O}(n \log n)$ на построение суффиксного массива и столько же на проход по `lcp` с бинпоиском.

Задача I. Теория Рамсея

Автор задачи и разработчик: Михаил Иванов

Теория Рамсея говорит, что для любых двух чисел L и K существует такое число R, что в любом графе на R вершинах есть либо L-клика, либо K-антиклика. Наименьшее такое число называется *числом Рамсея* и обозначается $R(L, K)$.

Мы предлагаем такое решение: взять любые $R(L, K)$ вершин графа (а если в графе меньше вершин — то взять все вершины), оставить только рёбра, проведённые между выбранным подмножеством, и в полученном графе перебрать все L-множества и K-множества вершин и проверить, что они подходят. Этого достаточно, так как, если $N < R(L, K)$, то мы переберём все нужные множества, а если $N \geqslant R(L, K)$, то мы в любом случае найдём требуемое множество.

Для всех $1 \leqslant K, L \leqslant 5$ выполняется $R(K, L) \leqslant 48$, поэтому перебрать достаточно C_{48}^5 множеств, что меньше двух миллионов и запросто можно успеть сделать за отведённое программе время.

Более того, если просто поверить, что $R(5, 5)$ достаточно мало, то можно не знать, насколько именно оно мало. Достаточно было поверить, что ограничения допускают решение с перебором всех множеств вершин до какого-то предела R, и написать программу, которая в бесконечном цикле перебирает подмножества всё большего набора вершин, каждый раз проверяя, что у неё ещё есть, скажем, полсекунды до истечения ограничения по времени.

Задача J. Супер-счастливые билетики

Автор задачи: Орешников Даниил, разработчик: Николай Будин

В этой задаче есть два случая.

Если $\frac{n}{2}$ — четное. Тогда обозначим за a сумму цифр на нечетных позициях в первой половине числа. За b обозначим сумму цифр на четных позициях в первой половине числа. За c – на нечетных позициях во второй половине. За d — на четных позициях во второй половине. Мы знаем, что $a + b = c + d$ и $a + c = b + d$. Преобразованиями можно получить, что $b = c$ и $a = d$. Заметим, что эти равенства необходимые и достаточные для того, чтобы билетик был супер-счастливым.

Обозначим за $f(l, s)$ количество последовательностей из l цифр с суммой s. Тогда ответом является:

$$\sum_{a,b} f^2(\frac{n}{4}, a) \cdot f^2(\frac{n}{4}, b) = (\sum_a f^2(\frac{n}{4}, a))^2$$

Значит, нужно научиться вычислять значения $f(\frac{n}{4}, a)$ $(0 \leqslant a \leqslant 10 \cdot \frac{n}{4})$. Для этого можно воспользоваться быстрым преобразованием Фурье для перемножения многочленов и вычислить:

$$(1 + x + \cdots + x^8 + x^9)^l$$

Тогда коэффициент при x^s будет равняться количеству способов выбрать последовательность из l цифр так, чтобы их сумма равнялась s.

Если же $\frac{n}{2}$ нечетно, то по аналогичным соображениям ответ равняется:

$$(\sum_a f^2(\lfloor\frac{n}{4}\rfloor, a)) \cdot (\sum_a f^2(\lceil\frac{n}{4}\rceil, a))$$