Содержание

У вас не получается читать/выводить данные?
Воспользуйтесь примерами (c++) (python).

Обратите внимание, входные данные лежат в **стандартном потоке ввода** (он же stdin), вывести ответ нужно в **стандартный поток вывода** (он же stdout).

Возможность добрать баллов

Задача -1A. Наберите сумму [0.2 (1.0), 256]

Вам дан массив целых чисел a. Вам необходимо ответить на q запросов. Каждый запрос задаётся двумя целыми числами pos_i, s_i. В качестве ответа на запрос выведите сколько подряд чисел, начиная с позиции pos_i в массиве a и правее, необходимо взять, чтобы их сумма была хотя бы s_i. Если сколько бы чисел, начиная с позиции pos_i, вы не взяли, их сумма будет меньше s_i, ответом на запрос является -1.

Формат входных данных

На первой строке длина массива n ($1 \leqslant n \leqslant 10^5$). На второй строке массив a: n целых чисел от 1 до 10^9. На третьей строке количество запросов q ($1 \leqslant q \leqslant 10^5$). На следующих q строках по два целых числа pos_i, s_i, которые задают i-й запрос ($1 \leqslant pos_i \leqslant n,\ 1 \leqslant s_i \leqslant 10^{18}$).

Формат выходных данных

Выведите q чисел — ответы на запросы.

Примеры

stdin	stdout
5 1 7 9 5 7 4 1 6 3 14 5 7 1 99	2 2 1 -1
2 2000000000 2000000000 2 1 2000000001 1 1000000000000000000	2 -1

Замечание

Пояснение к первому примеру:

Для первых трёх запросов жирным выделены те числа, сумма которых не менее s_i.

$[\mathbf{1}, \mathbf{7}, 9, 5, 7]$
$[1, 7, \mathbf{9}, \mathbf{5}, 7]$
$[1, 7, 9, 5, \mathbf{7}]$

На последний запрос ответ -1, так как даже если взять все числа, начиная с первого, сумма всё равно будет меньше 99.

Задача -1B. Рекуррентное соотношение [0.2 (1.0), 256]

Рассмотрим рекурсивную функцию $T(n)$

```
def T(n):
    if n <= max(a,b,c):
        return 1
    return T(n-a) + T(n-b) + T(n-c)
```

Здесь n, a, b, c – целые положительные числа. $T(n)$ возвращает ровно время своей работы.

Известно, что время работы таких функций экспоненциально, а именно равно $\Theta(\alpha^n)$ для некоторого вещественного α. Такое α легко найти, решив уравнение $\alpha^n = \alpha^{n-a} + \alpha^{n-b} + \alpha^{n-c}$.

Даны a, b, c, найдите α.

Формат входных данных

Три целых положительных числа a, b, c ($1 \leqslant a, b, c, \leqslant 100$).

Формат выходных данных

Выведите число α с точностью не менее 9 верных знаков после запятой.

Примеры

stdin	stdout
1 1 1	2.99999999999999911182
1 2 3	1.83928675521416096217

Замечание

Пояснение ко второму примеру:

Варианты верного ответа, все получат OK:
1.83928675521
1.839286755
1.839286756

Варианты неверного ответа:
1.83928675
1.839286754

Подсказка по решению

Запишите уравнение в таком виде, чтобы было хорошо видно, что из монотонности у него ровно один корень.

Задача -1C. Различные разбиения [0.2 (1.0), 256]

Найдите количество различных разбиений натурального числа n на натуральные слагаемые таких, что для любых двух различных чисел $a \neq b$, входящих в разбиение, верно, что количества чисел a и b в разбиении различны. Разбиения, отличающиеся только порядком слагаемых, различными не считаются.

Например, если $n = 4$, то из пяти возможных разбиений этому условию удовлетворяют все, кроме разбиения на слагаемые 1 и 3: в этом разбиении количество единиц равно количеству троек.

$$\begin{array}{lcll} 4 & = & 1+1+1+1 & \text{4 единицы} \\ 4 & = & 1+1+2 & \text{2 единицы, 1 двойка} \\ 4 & = & 1+3 & \text{1 единица и 1 тройка!} \\ 4 & = & 2+2 & \text{2 двойки} \\ 4 & = & 4 & \text{1 четвёрка} \end{array}$$

Формат входных данных

В первой строке входного файла записано натуральное число n ($1 \leqslant n \leqslant 100$).

Формат выходных данных

В первой строке выходного файла выведите количество разбиений числа n, удовлетворяющих заданным ограничениям.

Примеры

stdin	stdout
4	4
6	7

Задача -1D. Количество циклов [0.2 (1.0), 256]

Формально, *путь* в графе — это чередующаяся последовательность вершин и рёбер $u_1, e_1, u_2, e_2, u_3, \ldots, u_k$, начинающаяся и заканчивающаяся вершиной и такая, что любые соседние вершина и ребро в ней инцидентны.

Цикл — это путь, начальная и конечная вершины которого совпадают. В цикле должно быть хотя бы одно ребро.

Простой путь отличается от обычного пути тем, что в нём не может быть повторяющихся вершин.

Простой цикл — это цикл, в котором нет повторяющихся вершин и рёбер.

Дан неориентированный граф. Посчитайте, сколько в нём различных простых циклов. Заметим, что циклы считаются одинаковыми, если они обходят одно и то же множество вершин в одном и том же порядке, возможно, начиная при этом из другой вершины, или если порядок обхода противоположный. Например, циклы с порядком обхода вершин $1, 2, 3, 1$, $2, 3, 1, 2$ и $1, 3, 2, 1$ считаются одинаковыми, а циклы $1, 2, 3, 4, 1$ и $1, 3, 4, 2, 1$ — нет, поскольку порядок обхода вершин различен.

Формат входных данных

В первой строке входного файла заданы числа N и M через пробел — количество вершин и рёбер в графе, соответственно ($1 \leqslant N \leqslant 10$). Следующие M строк содержат по два числа u_i и v_i через пробел ($1 \leqslant u_i, v_i \leqslant N$, $u_i \neq v_i$); каждая такая строка означает, что в графе существует ребро между вершинами u_i и v_i. В графе нет кратных рёбер.

Формат выходных данных

Выведите одно число — количество простых циклов в заданном графе.

Примеры

stdin	stdout
3 2 1 2 2 3	0
4 5 1 2 2 3 3 4 4 1 1 3	3

Задача -1E. Шаблоны [0.4 (2.0), 256]

Многие операционные системы используют шаблоны для ссылки на группы объектов: файлов, пользователей, и т. д. Ваша задача — реализовать простейший алгоритм проверки шаблонов для имен файлов.

В этой задаче алфавит состоит из маленьких букв английского алфавита и точки (‘.’). Шаблоны могут содержать произвольные символы алфавита, а также два специальных символа: ‘?’ и ‘*’. Знак вопроса (‘?’) соответствует ровно одному произвольному символу. Звездочка ‘*’ соответствует подстроке произвольной длины (возможно, нулевой). Символы алфавита, встречающиеся в шаблоне, отображаются на ровно один такой же символ в проверяемой строчке. Строка считается подходящей под шаблон, если символы шаблона можно последовательно отобразить на символы строки таким образом, как описано выше. Например, строчки “ab”, “aab” и “beda. подходят под шаблон “*a?”, а строчки “bebe”, “a” и “ba” — нет.

Формат входных данных

Первая строка входного файла определяет шаблон P. Вторая строка S состоит только из символов алфавита. Ее необходимо проверить на соответствие шаблону. Длины обеих строк не превосходят 10 000. Строки могут быть пустыми — будьте внимательны!

Формат выходных данных

Если данная строка подходит под шаблон, выведите YES. Иначе выведите NO.

Примеры

stdin	stdout
k?t*n kitten	YES
k?t?n kitten	NO

Задача -1F. Гиперкуб [0.2 (1.0), 256]

Гиперкуб — это обобщение понятия трёхмерного куба на N измерений. Нуль-мерным гиперкубом является точка, одномерным — отрезок, двумерным — квадрат. В общем же случае N-мерный гиперкуб — это правильный N-мерный многогранник, каждая из $2 \cdot N$ граней которого является $(N-1)$-мерным гиперкубом. Например, для $N = 2$ квадрат — это правильный многоугольник, каждая из $2 \cdot 2 = 4$ сторон которого — отрезок, то есть одномерный гиперкуб. Отметим, что N-мерный гиперкуб имеет 2^N вершин.

Старшеклассник Петя долго разбирался, что же такое гиперкуб, но наконец понял, как этот объект устроен, и ему настолько понравилось, что он даже придумал свою собственную игру на гиперкубе. Игра заключается в следующем.

Рассмотрим N-мерный единичный гиперкуб. Расположим его таким образом, чтобы одна из вершин находилась в начале координат — точке $(0, 0, \ldots, 0)$ в N-мерном пространстве, а для любой из остальных вершин каждая координата равнялась бы нулю или единице. В каждой из 2^N вершин запишем по целому неотрицательному числу. Игрок начинает свой путь в начале координат. За один ход можно переместиться из текущей вершины по любому ребру при условии, что сумма координат новой вершины строго больше суммы координат старой. Игра заканчивается, когда игрок попадает в вершину $(1, 1, \ldots, 1)$, имеющую максимальную сумму координат — N. Результат игры — сумма чисел во всех посещённых игроком вершинах. Цель игры — пройти по гиперкубу таким образом, чтобы эта сумма (количество очков за игру) оказалась как можно больше.

Петя довольно быстро понял, что между двумя вершинами гиперкуба A и B ребро есть тогда и только тогда, когда все координаты этих вершин $(A_1, A_2, \ldots, A_N)$ и $(B_1, B_2, \ldots, B_N)$ совпадают, кроме одной, которая равна нулю у одной из вершин (скажем, A) и единице у другой (B). Поскольку при этом $A_1 + A_2 + \ldots + A_N + 1 = B_1 + B_2 + \ldots + B_N$, то по такому ребру можно перемещаться из A в B, но не наоборот. Однако, сыграв в свою игру, Петя не может с уверенностью сказать, является ли полученная им сумма максимальной или можно на данном гиперкубе сыграть по-другому и набрать больше очков.

Напишите программу, которая по данному гиперкубу находит максимальную сумму, которую можно получить, сыграв в эту игру.

Формат входных данных

В первой строке входного файла записано число N ($1 \leqslant N \leqslant 10$) — размерность гиперкуба. В следующих 2^N строках содержится по одному числу в каждой; в $(k+2)$-ой строке записано C_k ($0 \leqslant C_k \leqslant 1000$) — число в вершине с номером k.

Номер вершины вычисляется так: вершина A с координатами $(A_1, A_2, \ldots, A_N)$ имеет номер, равный $A_1 \cdot 2^{N-1} + A_2 \cdot 2^{N-2} + \ldots + A_{N-1} \cdot 2 + A_N$, то есть координаты просто интерпретируются как двоичная запись номера вершины. В этой нумерации начальная вершина имеет номер 0, а конечная — номер $2^N - 1$.

Формат выходных данных

В выходной файл выведите одно число — максимальную сумму, которую можно получить при игре на данном гиперкубе.

Пример

stdin	stdout
3 1 2 3 4 5 6 7 8	21

Пояснение к примеру

Наш маршрут таков:

- вершина 0 (число 1, координаты (0, 0, 0)) — начальная
- вершина 4 (число 5, координаты (1, 0, 0))
- вершина 6 (число 7, координаты (1, 1, 0))
- вершина 7 (число 8, координаты (1, 1, 1)) — конечная

Наше количество очков: $1 + 5 + 7 + 8 = 21$.
Любой другой маршрут с соблюдением правил игры даёт меньшее количество очков.

Задача -1G. Площадь комнаты [0.2 (1.0), 256]

Требуется вычислить площадь комнаты в квадратном лабиринте.

Формат входных данных

В первой строке вводится число N —- размер лабиринта ($3 \leqslant N \leqslant 10$). В следующих N строках задан лабиринт (‘.’ —- пустая клетка, ‘*’ —- стенка). И наконец, последняя строка содержит два числа —- номер строки и столбца клетки, находящейся в комнате, площадь которой необходимо вычислить. Гарантируется, что эта клетка пустая и что лабиринт окружен стенками со всех сторон.

Формат выходных данных

Требуется вывести единственное число —- количество пустых клеток в данной комнате.

Пример

stdin	stdout
5 ***** **..* *.*.* *..** ***** 2 4	3

Задача -1H. Грязь [0.2 (1.0), 256]

— Здравствуйте! Могу я поговорить с Петровым? Алё, милый, привет... ты знаешь, у нас дома небольшая авария произошла... Но твой компьютер не пострадал, не волнуйся. Но теперь там немного грязно. Ну, то есть очень грязно. Но ты не волнуйся, я приготовила тебе твои болотные сапоги, у входа стоят. А грязь я уберу, как будет свободное время. Когда? Ну, наверное, когда в отпуск пойду. А, ну когда вернёмся из Турции. А, ну значит в следующий отпуск, но обязательно уберу. А пока я у мамы поживу. И ты, кстати, тоже можешь. Ну, как хочешь, я же не заставляю... Только пока я не убрала, ты там грязь не разводи, сильно сапогами по грязи не шлёпай и когда по чистому ходишь, сапоги снимай и тапочки обувай, я их тоже возле входа поставила, ты их бери с собой, когда идёшь по грязи и переобувай. А когда по чистому идёшь, бери сапоги, там грязь в разных местах. Программисты, как известно, не самые трудолюбивые люди, поэтому убирать грязь не станут. Но переобувать болотные сапоги каждый раз, когда переходишь от грязного пола к чистому и наоборот — удовольствие ниже среднего, уж лучше пройти лишние несколько метров. Чтобы прожить время до следующего отпуска с комфортом, надо срочно выработать способ добираться из одной точки квартиры с минимальным количеством переобуваний по пути, ну а уж среди них, конечно, выбрать самый короткий.

Формат входных данных

В первой строке даны два целых числа M и N — размеры квартиры (в у.е.). $1 \leqslant N, M \leqslant 500$. Два целых числа во второй строке — координаты компьютера (в у.е.), а два целых числа в третьей строке — координаты холодильника (тоже в у.е.). Далее идут M строк по N символов в каждой — план квартиры. На плане 1 означает чистое место, 2 — грязное, 0 — стена или непроходимая грязь. Переходить можно только на клетки, имеющие общую вершину с данной, при переходе с чистой на грязную и наоборот надо переобуваться. Холодильник и компьютер находятся не в клетках, помеченных нулём. Левая верхняя клетка плана имеет координаты $(1, 1)$.

Формат выходных данных

Длина кратчайшего пути (количество преодоленных квадратиков, включая начальный и конечный) с минимальным количеством переобуваний, и, через пробел, количество переобуваний (переобувание проходит при переходе с грязного на чистое и наоборот). Если пройти к холодильнику невозможно, вывести числа 0 0.

Пример

stdin	stdout
3 7 1 1 3 7 1200121 1212020 1112021	8 4

Задача -1I. Ребра добавляются, граф растет [0.2 (1.0), 256]

В неориентированный граф последовательно добавляются новые ребра. Изначально граф пустой. После каждого добавления нужно говорить, является ли текущий граф двудольным.

Формат входных данных

На первой строке n — количество вершин, m — количество операций «добавить ребро». Следующие m строк содержат пары чисел от 1 до n — описание добавляемых ребер.

Формат выходных данных

Выведите в строчку m нулей и единиц. i-й символ должен быть равен единице, если граф, состоящий из первых i ребер, является двудольным.

Система оценки

Подзадача 1 (25 баллов) $1 \leqslant n, m \leqslant 1\,000$.
Подзадача 2 (50 баллов) $1 \leqslant n, m \leqslant 50\,000$.
Подзадача 3 (25 баллов) $1 \leqslant n, m \leqslant 300\,000$.

Примеры

stdin	stdout
3 3 1 2 2 3 3 1	110

Задача -1J. Range Minimum Query [0.5 (2.5), 256]

Компания *Giggle* открывает свой новый офис в Судиславле, и вы приглашены на собеседование. Ваша задача — решить поставленную задачу.

Вам нужно создать структуру данных, которая представляет из себя массив целых чисел. Изначально массив пуст. Вам нужно поддерживать две операции:

- запрос: «`? i j`» — возвращает минимальный элемент между i-ым и j-м, включительно;
- изменение: «`+ i x`» — добавить элемент x после i-го элемента списка. Если $i = 0$, то элемент добавляется в начало массива.

Конечно, эта структура должна быть достаточно хорошей.

Формат входных данных

Первая строка входного файла содержит единственное целое число n — число операций над массивом ($1 \leqslant n \leqslant 200\,000$). Следующие n строк описывают сами операции. Все операции добавления являются корректными. Все числа, хранящиеся в массиве, по модулю не превосходят 10^9.

Формат выходных данных

Для каждой операции в отдельной строке выведите её результат.

Примеры

stdin	stdout
8 + 0 5 + 1 3 + 1 4 ? 1 2 + 0 2 ? 2 4 + 4 1 ? 3 5	4 3 1

Задача -1K. Range Variation Query [0.2 (1.0), 256]

В начальный момент времени последовательность a_n задана следующей формулой: $a_n = n^2 \bmod 12345 + n^3 \bmod 23456$. Требуется много раз отвечать на запросы следующего вида:

- Найти разность между максимальным и минимальным значениями среди элементов $a_i, a_{i+1}, \ldots, a_j$.
- Присвоить элементу a_i значение j.

Формат входных данных

Первая строка входного файла содержит натуральное число k — количество запросов ($1 \leqslant k \leqslant 100\,000$). Следующие k строк содержат запросы, по одному на строке. Запрос номер i описывается двумя целыми числами x_i, y_i.

Если $x_i > 0$, то требуется найти разность между максимальным и минимальным значениями среди элементов $a_{x_i}, \ldots, a_{y_i}$. При этом $1 \leqslant x_i \leqslant y_i \leqslant 100\,000$.

Если $x_i < 0$, то требуется присвоить элементу $a_{|x_i|}$ значение y_i. В этом случае $-100\,000 \leqslant x_i \leqslant -1$ и $|y_i| \leqslant 100\,000$.

Формат выходных данных

Для каждого запроса первого типа в выходной файл требуется вывести одну строку, содержащую разность между максимальным и минимальным значениями на соответствующем отрезке.

Примеры

stdin	stdout
7 1 3 2 4 -2 -100 1 5 8 9 -3 -101 2 3	34 68 250 234 1

Задача -1L. Хорошие дни [0.2 (1.0), 256]

Билл разрабатывает новую математическую теорию, описывающую человеческие эмоции. Его последние исследования посвящены изучению того, насколько хорошие и плохие дни влияют на воспоминания людей о различных периодах жизни.

Недавно Билл придумал методику, которая описывает, насколько хорошим или плохим был день человеческой жизни с помощью сопоставления дню некоторого неотрицательного целого числа. Билл называет это число *эмоциональной значимостью* этого дня. Чем больше это число, тем лучше этот день. Билл полагает, что значимость некоторого периода человеческой жизни равна сумме эмоциональных значимостей каждого из дней периода, помноженной на минимум эмоциональных значимостей дней этого периода. Эта методика отражает то, что период, который в среднем может быть весьма неплох, бывает испорчен одним плохим днем.

Теперь Билл хочет проанализировать свою собственную жизнь и найти в ней период максимальной значимости. Помогите ему это сделать.

Формат входных данных

Первая строка входного файла содержит число n — количество дней в жизни Билла, которые он хочет исследовать ($1 \leqslant n \leqslant 100\,000$). Оставшаяся часть файла содержит n целых чисел $a_1, a_2, \ldots, a_n$, все в пределах от 0 до 10^6 — эмоциональные значимости дней. Числа во входном файле разделяются пробелами и переводами строки.

Формат выходных данных

В первой строке выходного файла выведите максимальную значимость периода жизни Билла. Во второй строке выведите два числа l и r, означающие, что значимость периода с l-го по r-й день (включительно) в жизни Билла была максимально возможной.

Примеры

stdin	stdout
6 3 1 6 4 5 2	60 3 5

Задача -1M. Основание строки [0.2 (1.0), 256]

Строка S была записана много раз подряд, после чего из получившейся строки взяли подстроку и дали вам. Ваша задача определить минимально возможную длину исходной строки S.

Формат входных данных

В первой и единственной строке входного файла записана строка, которая содержит только латинские буквы, длина строки не превышает 50 000 символов.

Формат выходных данных

В выходной файл выведите ответ на задачу.

Пример

stdin	stdout
zzz	1
bcabcab	3