

Задача А. Складирование ноутбуков

Имя входного файла: a.in
 Имя выходного файла: a.out
 Максимальное время работы на одном тесте: 1 секунда
 Максимальный объем используемой памяти: 64 мегабайта

На склад, который имеет форму прямоугольного параллелепипеда, привезли ноутбуки, упакованные в коробки. Каждая коробка также имеет форму прямоугольного параллелепипеда.

По правилам хранения коробки с ноутбуками должны быть размещены на складе с выполнением следующих двух условий:

- Стороны коробок должны быть параллельны сторонам склада
- Коробку при помещении на склад разрешается расположить где угодно (с выполнением предыдущего условия), в том числе на другой коробке, но все коробки должны быть ориентированы одинаково (т.е. нельзя одну коробку расположить «стоя», а другую – «лежа»)

Напишите программу, которая по размерам склада и размерам коробки с ноутбуком определит максимальное количество ноутбуков, которое может быть размещено на складе.

Формат входных данных

Вводится шесть натуральных чисел. Первые три задают длину, высоту и ширину склада. Следующие три задают соответственно длину, высоту и ширину коробки с ноутбуком. Каждое из чисел не превышает 1000.

Формат выходных данных

Выведите одно число — максимальное количество ноутбуков, которое может быть размещено на складе.

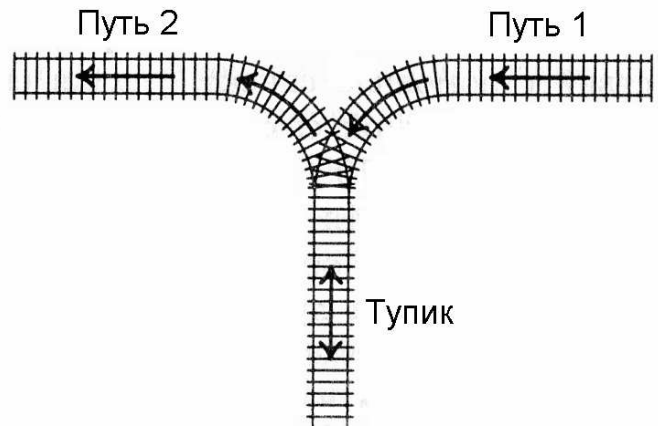
Примеры

a.in	a.out
100 200 300 1 2 3	1000000
100 200 300 3 2 1	1000000
100 100 1 2 2 2	0
7 7 7 3 3 3	8

Задача В. Сортировка вагонов

Имя входного файла: b.in
 Имя выходного файла: b.out
 Максимальное время работы на одном тесте: 1 секунда
 Максимальный объем используемой памяти: 64 мегабайта

К тупику со стороны пути 1 (см. рисунок) подъехал поезд. Разрешается отцепить от поезда один или сразу несколько первых вагонов и завезти их в тупик (при желании, можно даже завезти в тупик сразу весь поезд). После этого часть из этих вагонов вывезти в сторону пути 2. После этого можно завезти в тупик еще несколько вагонов и снова часть оказавшихся вагонов вывезти в сторону пути 2. И так далее



(так, что каждый вагон может лишь один раз заехать с пути 1 в тупик, а затем один раз выехать из тупика на путь 2). Заезжать в тупик с пути 2 или выезжать из тупика на путь 1 запрещается. Нельзя с пути 1 попасть на путь 2, не заезжая в тупик.

Известно, в каком порядке изначально идут вагоны поезда. Требуется с помощью указанных операций сделать так, чтобы вагоны поезда шли по порядку (сначала первый, потом второй и т.д., считая от головы поезда, едущего по пути 2 в сторону от тупика).

Формат входных данных

Вводится число N — количество вагонов в поезде ($1 \leq N \leq 2000$). Далее идут номера вагонов в порядке от головы поезда, едущего по пути 1 в сторону тупика. Вагоны пронумерованы натуральными числами от 1 до N , каждое из которых встречается ровно один раз.

Формат выходных данных

Если сделать так, чтобы вагоны шли в порядке от 1 до N , считая от головы поезда, когда поезд поедет по пути 2 из тупика, можно, выведите действия, которые нужно проделать с поездом. Каждое действие описывается двумя числами: типом и количеством вагонов:

- если нужно завезти с пути 1 в тупик K вагонов, должно быть выведено сначала число 1, а затем — число K ($K \geq 1$),
- если нужно вывезти из тупика на путь 2 K вагонов, должно быть выведено сначала число 2, а затем — число K ($K \geq 1$).

Если возможно несколько последовательностей действий, приводящих к нужному результату, выведите любую из них.

Если выстроить вагоны по порядку невозможно, выведите одно число 0.

Примеры

b.in	b.out
3 3 2 1	1 3 2 3
4 4 1 3 2	1 2 2 1 1 2 2 3
3 2 3 1	0

Задача С. Остаток от деления на цифру

Имя входного файла: `c.in`
 Имя выходного файла: `c.out`
 Максимальное время работы на одном тесте: 1 секунда
 Максимальный объем используемой памяти: 64 мегабайта

Напишите программу, вычисляющую остаток от деления заданного «длинного» числа на заданную цифру.

Формат входных данных

В первой строке задана цифра K ($1 \leq K \leq 9$). Во второй строке задано натуральное число N , состоящее из не более чем 100000 цифр.

Формат выходных данных

Выведите остаток от деления N на K .

Примеры

<code>c.in</code>	<code>c.out</code>
5 123456789	4
1 123	0

Задача D. Контрольная по ударениям

Имя входного файла:	d.in
Имя выходного файла:	d.out
Максимальное время работы на одном тесте:	1 секунда
Максимальный объем используемой памяти:	64 мегабайта

Учительница задала Пете домашнее задание — в заданном тексте расставить ударения в словах, после чего поручила Васе проверить это домашнее задание. Вася очень плохо знаком с данной темой, поэтому он нашел словарь, в котором указано, как ставятся ударения в словах. К сожалению, в этом словаре присутствуют не все слова. Вася решил, что в словах, которых нет в словаре, он будет считать, что Петя поставил ударения правильно, если в этом слове Петей поставлено ровно одно ударение.

Оказалось, что в некоторых словах ударение может быть поставлено больше, чем одним способом. Вася решил, что в этом случае если то, как Петя поставил ударение, соответствует одному из приведенных в словаре вариантов, он будет засчитывать это как правильную расстановку ударения, а если не соответствует, то как ошибку.

Вам дан словарь, которым пользовался Вася и домашнее задание, сданное Петей. Ваша задача — определить количество ошибок, которое в этом задании насчитает Вася.

Формат входных данных

Вводится сначала число N — количество слов в словаре ($0 \leq N \leq 20000$).

Далее идет N строк со словами из словаря. Каждое слово состоит не более чем из 30 символов. Все слова состоят из маленьких и заглавных латинских букв. В каждом слове заглавная ровно одна буква — та, на которую попадает ударение. Слова в словаре расположены в алфавитном порядке. Если есть несколько возможностей расстановки ударения в одном и том же слове, то эти варианты в словаре идут в произвольном порядке.

Далее идет упражнение, выполненное Петей. Упражнение представляет собой строку текста, суммарным объемом не более 300000 символов. Строка состоит из слов, которые разделяются между собой ровно одним пробелом. Длина каждого слова не превышает 30 символов. Все слова состоят из маленьких и заглавных латинских букв (заглавными обозначены те буквы, над которыми Петя поставил ударение). Петя мог по ошибке в каком-то слове поставить более одного ударения или не поставить ударения вовсе.

Формат выходных данных

Выведите количество ошибок в Петинем тексте, которые найдет Вася.

Примеры

d.in	d.out	Комментарии
<p>4 cAnnot cannOt fOund pAge thE pAge cAnnot be fouNd</p>	<p>2</p>	<p>В слове cannot, согласно словарю возможно два варианта расстановки ударения. Эти варианты в словаре могут быть перечислены в любом порядке (т.е. как сначала cAnnot, а потом cannOt, так и наоборот). Две ошибки, совершенные Петей — это слова be (ударение вообще не поставлено) и fouNd (ударение поставлено неверно). Слово thE отсутствует в словаре, но поскольку в нем Петя поставил ровно одно ударение, признается верным.</p>
<p>4 cAnnot cannOt fOund pAge The PAGE cannot be found</p>	<p>4</p>	<p>Неверно расставлены ударения во всех словах, кроме The (оно отсутствует в словаре, в нем поставлено ровно одно ударение). В остальных словах либо ударные все буквы (в слове PAGE), либо не поставлено ни одного ударения.</p>

Задача Е. Свинки-копилки

Имя входного файла: e.in
 Имя выходного файла: e.out
 Максимальное время работы на одном тесте: 1 секунда
 Максимальный объем используемой памяти: 64 мегабайта

У Васи есть N свинок-копилков, свинки занумерованы числами от 1 до N . Каждая копилка может быть открыта единственным соответствующим ей ключом или разбита.

Вася положил ключи в некоторые из копилков (он помнит, какой ключ лежит в какой из копилков). Теперь Вася собрался купить машину, а для этого ему нужно достать деньги из всех копилков. При этом он хочет разбить как можно меньшее количество копилков (ведь ему еще нужно копить деньги на квартиру, дачу, вертолет...). Помогите Васе определить, какое минимальное количество копилков нужно разбить.

Формат входных данных

В первой строке содержится число N — количество свинок-копилков ($1 \leq N \leq 100000$). Далее идет N строк с описанием того, где лежит ключ от какой копилки: в i -ой из этих строк записан номер копилки, в которой находится ключ от i -ой копилки.

Формат выходных данных

Выведите единственное число: минимальное количество копилков, которые необходимо разбить.

Пример

e.in	e.out	Комментарий
4 2 1 2 4	2	Ключи от первой и третьей копилки лежат в копилке 2, ключ от второй — в первой, а от четвертой — в ней самой. Чтобы открыть все копилки, достаточно разбить, например, копилки с номерами 1 и 4.

Задача F. Трехцветные таблицы

Имя входного файла: f.in
 Имя выходного файла: f.out
 Максимальное время работы на одном тесте: 1 секунда
 Максимальный объем используемой памяти: 64 мегабайта

Прямоугольную таблицу, состоящую из N строк и M столбцов, раскрашивают следующим образом. Каждый столбец таблицы и каждую строку красят либо в синий, либо в желтый цвет. В итоге клетки, оказавшиеся на пересечении синего столбца и синей строки оказываются синими, желтого столбца и желтой строки — желтыми, а клетки на пересечении синего столбца и желтой строки, или, наоборот, желтого столбца и синей строки — зелеными.

Раскраска всех клеток таблицы (или просто сама таблица) называется правильной, если она может быть получена описанным выше способом.

Вам дана прямоугольная таблица, которую нужно раскрасить таким образом. Про некоторые клетки известно, какого цвета они должны быть, а остальные клетки могут в итоге быть любого цвета. Определите, сколько существует различных правильных таблиц, в которых нужные клетки покрашены в нужный цвет.

Формат входных данных

Вводятся числа N и M — количество строк и столбцов таблицы ($1 \leq N \leq 30$, $1 \leq M \leq 30$). Далее записано N строк по M чисел в каждой, задающие цвета, в которые должны быть окрашены клетки:

- 0 — клетка может в итоге быть любого цвета
- 1 — клетка должна быть синей
- 2 — клетка должна быть желтой
- 3 — клетка должна быть зеленой

Формат выходных данных

Выведите одно число — количество различных правильных таблиц, в которых нужные клетки покрашены в нужный цвет. Обратите внимание, что если два или более способов раскраски столбцов и строк таблицы приводят к одинаковой раскраске самой таблицы, то это нужно считать как *один* вариант раскраски таблицы (см. пример 2).

Примеры

f.in	f.out
3 4 1 0 0 0 3 0 0 0 0 0 0 0	16
2 2 3 3 3 3	1
2 2 2 2 2 3	0

Задача G. Путь через горы

Имя входного файла: g.in
 Имя выходного файла: g.out
 Максимальное время работы на одном тесте: 1 секунда
 Максимальный объем используемой памяти: 64 мегабайта

Поверхность Земли в горной местности можно представить в виде ломаной линии. Вершины ломаной расположены в точках $(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)$, при этом $x_i < x_{i+1}$.

Обычный горный маг находится в точке (x_1, y_1) и хочет попасть в точку (x_N, y_N) . При этом он может перемещаться только пешком. Он может ходить по поверхности Земли (т.е. вдоль ломаной). А может сотворить в воздухе мост и пройти по нему. Мост может соединять две вершины ломаной: мост не может начинаться и заканчиваться не в вершине ломаной, и мост не может проходить под землей (в том числе не может быть туннелем в горе), но мост может каким-то своим участком проходить по поверхности земли. Длина моста не может быть больше R . Суммарно маг может построить не более K мостов.

Какое наименьшее расстояние придется пройти магу, чтобы оказаться в точке (x_N, y_N) .

Формат входных данных

Вводится сначала натуральное число N ($2 \leq N \leq 100$). Затем водится натуральное число K ($1 \leq K \leq 100$) — максимальное количество мостов. Далее вводится целое число R ($0 \leq R \leq 10000$) — максимальная возможная длина моста. Далее вводятся координаты $(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)$. Все координаты — целые числа, не превышающие по модулю 10000, для всех i от 1 до $N-1$: $x_i < x_{i+1}$.

Формат выходных данных

Выведите одно число — минимальную длину пути, которую придется пройти магу (как по земле, так и по мостам). Ответ выведите не менее чем с 5 цифрами после десятичной точки.

Примеры

g.in	g.out
5 2 5 0 0 2 2 3 -1 4 1 5 0	6.47871
9 2 3 1 2 2 1 3 3 5 -1 6 2 7 0 8 1 9 0 10 1	14.93498

Задача Н. Медианы объединений

Имя входного файла: h.in
 Имя выходного файла: h.out
 Максимальное время работы на одном тесте: 2 секунды
 Максимальный объем используемой памяти: 64 мегабайта

Дано N упорядоченных по неубыванию последовательностей целых чисел (т.е. каждый следующий элемент больше либо равен предыдущему), в каждой из последовательностей ровно L элементов. Для каждой двух последовательностей выполняют следующую операцию: объединяют их элементы (в объединенной последовательности каждое число будет идти столько раз, сколько раз оно встречалось суммарно в объединяемых последовательностях), упорядочивают их по неубыванию и смотрят, какой элемент в этой последовательности из $2L$ элементов окажется на месте номер L (этот элемент называют левой медианой).

Напишите программу, которая для каждой пары последовательностей выведет левую медиану их объединения.

Формат входных данных

Сначала вводятся числа N и L ($2 \leq N \leq 200$, $1 \leq L \leq 50000$). В следующих N строках задаются параметры, определяющие последовательности.

Каждая последовательность определяется пятью целочисленными параметрами: x_1 , d_1 , a , c , m . Элементы последовательности вычисляются по следующим формулам: x_1 нам задано, а для всех i от 2 до L : $x_i = x_{i-1} + d_{i-1}$. Последовательность d_i определяется следующим образом: d_1 нам задано, а для $i \geq 2$ $d_i = ((a * d_{i-1} + c) \bmod m)$, где \bmod – операция получения остатка от деления $(a * d_{i-1} + c)$ на m .

Для всех последовательностей выполнены следующие ограничения: $1 \leq m \leq 40000$, $0 \leq a < m$, $0 \leq c < m$, $0 \leq d_1 < m$. Гарантируется, что все члены всех последовательностей по модулю не превышают 10^9 .

Формат выходных данных

В первой строке выведите медиану объединения 1-й и 2-й последовательностей, во второй строке — объединения 1-й и 3-й, и так далее, в $(N-1)$ -ой строке — объединения 1-й и N -ой последовательностей, далее медиану объединения 2-й и 3-й, 2-й и 4-й, и т.д. до 2-й и N -ой, затем 3-й и 4-й и так далее. В последней строке должна быть выведена медиана объединения $(N-1)$ -й и N -ой последовательностей.

Пример

h.in	h.out	Комментарии
3 6 1 3 1 0 5 0 2 1 1 100 1 6 8 5 11	7 10 9	Последовательности, объединения которых мы считаем, таковы: 1 4 7 10 13 16 0 2 5 9 14 20 1 7 16 16 21 22

Задача I. Заполните массив

Имя входного файла: `i.in`
 Имя выходного файла: `i.out`
 Максимальное время работы на одном тесте: 1 секунда
 Максимальный объем используемой памяти: 64 мегабайта

Требуется заполнить N элементов массива, пронумерованных числами от 1 до N ($A[1] \dots A[N]$), натуральными числами от 2 до $N+1$, используя каждое число ровно один раз, так, чтобы значение каждого элемента массива делилось бы нацело на его номер (т.е. для каждого i $A[i]$ делилось бы на i).

Напишите программу, которая для заданного N вычислит количество способов такого заполнения массива.

Формат входных данных

Вводится одно натуральное число N ($1 \leq N \leq 60000$).

Формат выходных данных

Выведите одно число — искомое количество способов заполнения массива.

Пример

<code>i.in</code>	<code>i.out</code>	Комментарии
2	1	Массив можно заполнить единственным способом: 3 2

Задача J. Перекраска N-угольника

Имя входного файла: j.in
 Имя выходного файла: j.out
 Максимальное время работы на одном тесте: 1 секунда
 Максимальный объем используемой памяти: 64 мегабайта

В правильном N -угольнике провели некоторые диагонали так, что он оказался разбит на треугольники. Изначально стороны N -угольника и все его диагонали черные.

Разрешается выбрать четырехугольник, в котором ровно одна диагональ, и при этом эта диагональ черного цвета (сам четырехугольник не обязан быть полностью черным) и проделать с ним следующее: заменить диагональ на противоположную (т.е. если сам четырехугольник был $ABCD$ и в нем была диагональ AC , то она меняется на диагональ BD), после чего перекрасить стороны этого четырехугольника и новую диагональ в красный цвет.

Требуется определить, можно ли с помощью таких операций сделать так, чтобы все отрезки (т.е. стороны N -угольника и изображенные диагонали) стали красными, и не осталось бы ни одного черного отрезка. А если это возможно, то какое минимальное количество операций для этого требуется.

Формат входных данных

Вводится сначала число N ($3 \leq N \leq 30000$). Далее идет описание $N-3$ проведенных диагоналей. Каждая диагональ описывается двумя натуральными числами — номерами вершин, которые она соединяет. Гарантируется, что проведенные диагонали внутри N -угольника не пересекаются.

Формат выходных данных

Выведите минимальное число действий, необходимое для того, чтобы перекрасить весь N -угольник и все его диагонали. Если перекрасить многоугольник указанным способом невозможно, выведите одно число -1 (минус один).

Примеры

j.in	j.out
3	-1
4	1
1 3	