

Задача № 1. Пятница 13–е.

Компьютерный вирус «Пятница 13–е» может повредить информацию только в те дни, когда 13–е попадает на пятницу. Определить все месяцы 2010 года, в которых 13–е число было пятницей. Учесть, что 1 января 2010 года – пятница. В качестве ответа распечатать номера месяцев.

Задача № 2. Танки

Танки шли М колоннами по N танков в каждой. Два из них вышли из строя, преградив путь идущим за ними. Определить количество танков, продолживших движение.

Выходные данные: в первой строке, через пробел, указывается количество колонн и количество танков в колонне, во второй строке через пробел указываются номер колонны и порядковый номер в колонне первого сломавшегося танка, в третьей строке через пробел указываются номер колонны и порядковый номер в колонне второго сломавшегося танка.

Выходные данные: программа на выходе должна выдавать количество танков, продолживших движение.

Пример входных данных:

3	4
1	2
2	4

Пример выходных данных:

8

Задача № 3. Забавная игра

Легендарный учитель математики Юрий Петрович придумал забавную игру с числами. А именно, взяв произвольное целое число, он переводит его в двоичную систему счисления, получая некоторую последовательность из нулей и единиц, начинающуюся с единицы. (Например, десятичное число $19_{10} = 1 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 10011_2$) Затем учитель начинает сдвигать цифры полученного двоичного числа по циклу (так, что последняя цифра становится первой, а все остальные сдвигаются на одну позицию вправо), выписывая образующиеся при этом последовательности из нулей и единиц в столбик, он подметил, что независимо от выбора исходного числа получающиеся последовательности начинают с некоторого момента повторяться. И наконец, Юрий Петрович отыскивает максимальное из выписанных чисел и переводит его обратно в десятичную систему счисления, считая это число результатом проделанных манипуляций.

Так, для числа 19 список последовательности будет таким:

10011
11001
11100
...
01110
00111
10011
...

И результатом игры, следовательно, окажется число $1 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 = 28$.

Поскольку придуманная игра с числами всё больше занимает воображение учителя, отвлекая тем самым его от работы с ну очень одарёнными школьниками, Вас просят написать программу, которая бы помогла Юрию Петровичу получать результат игры без утомительных вычислений вручную.

Формат входных данных:

На вход подается одно целое число N ($0 < N < 32767$).

Формат выходных данных:

Ваша программа должна вывести одно целое число, равное результату игры.

Пример

<i>Вход</i>	<i>Выход</i>
<i>19</i>	<i>28</i>

Задача № 4. Гостеприимные жители планеты Земля Кузя, Мила, и Пчелёнок решили научить Лунтика земной игре в анаграммы. Чтобы ему было проще, и он быстрее научился этой замечательной игре, помимо самой анаграммы друзья предлагали Лунтику ещё и различные отгадки, среди которых нужно выбрать правильную.

Требуется написать программу, автоматизирующую процесс решения анаграммы. Условия задачи не предполагают существование более одного решения.

Входной файл: input.txt

Входные данные: первая строка – строка-анаграмма, состоящая из заглавных букв латиницы. Вторая строка – натуральное число $N < 32767$ – количество предполагаемых вариантов решения. Далее N строк – варианты решения анаграммы (символы латиницы).

Выходной файл: output.txt

Выходные данные: одна строка – слово-решение анаграммы, либо NO в случае, если решения нет.

Пример:

<i>input.txt</i>	<i>output.txt</i>
MOSKOS	KOSMOS
3	
KOKOS	
KOSMOS	
MOKOKO	

Задача № 5. Оставшиеся на луне родственники Лунтика очень переживали за то, как проходит его визит на планету Земля. Так как Лунтик был очень ответственным, он постоянно посылал весточки своим родным. При этом, будучи грамотным, он знал, что любые его сообщения могут содержать персональные данные, которые обязательно надо зашифровать. Поэтому он использовал специальный шифр.

Шифр состоял из двух частей: зашифрованного сообщения и ключа к его расшифровке. Зашифрованное сообщение и ключ объединялись в одно сообщение, которое передавалось на Луну: сначала матрица сообщения, затем ключ. Они имели размер $N \times N$ и расположены друг за другом без пробелов.

Ключ состоит из цифр [0...9], а затем букв латиницы ['A' ... 'Z'], что соответствует лексикографическому порядку.

Требуется: определить способ расшифровки и написать программу, позволяющую по заданному сообщению и ключу осуществить расшифровку.

Входной файл: input.txt

Входные данные: Первая строка – одно натуральное число $N < 7$ – размер матрицы. Далее N строк по N символов – матрица сообщения (заглавные символы латиницы). Следующие N строк N символов – матрица ключа расшифровки.

Выходной файл: output.txt

Выходные данные: одна строка – расшифрованный текст.

Пример:

<i>input.txt</i>	<i>output.txt</i>
4 KT_S OORA INRE _!BO 1D80 92A7 C63E 5FB4	SKORO_NA_ORBITE!

Задача № 6. Вупсень и Пупсень были те ещё электрики. Им нравилось собирать различные схемы. Причём не нравилось разнообразие деталей. Они предпочитали только резисторы и провода. Вупсень и Пупсень прекрасно знали, что резисторы можно соединить последовательно и параллельно. Однажды Вупсень и Пупсень собрали красивую, на их взгляд, схему. Проходящему мимо Лунтику она тоже понравилась, и он решил оставить её себе на память, но так как у него была склонность к шифрованию, то он зашифровал её.

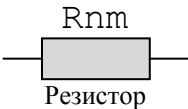
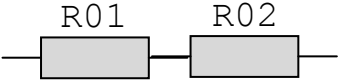
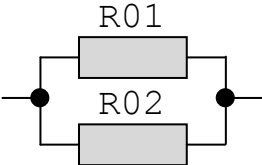
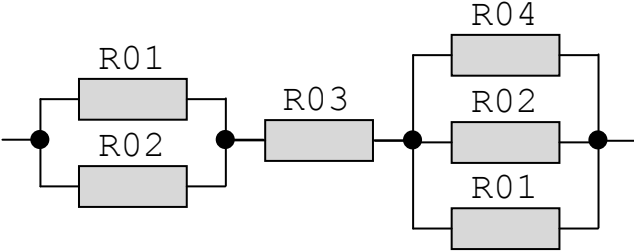
В шифровке он использовал только следующие обозначения:

1) каждый резистор в схеме обозначается номером, состоящим из символа R, затем двух цифр n и m ($0 < n, m < 10$,) – Rnm;

2) последовательное соединение обозначается символом «-»

3) параллельное соединение обозначается символом «=».

Никакие иные обозначения в зашифрованной схеме он не использовал.

Схема	Обозначение
 <p style="text-align: center;">Резистор</p>	R_{nm}
 <p style="text-align: center;">Последовательное соединение двух резисторов</p>	$R_{01}-R_{02}$
 <p style="text-align: center;">Параллельное соединение двух резисторов</p>	$R_{01}=R_{02}$
 <p style="text-align: center;">Произвольная схема</p>	$R_{01}=R_{02}-R_{03}-R_{04}=R_{02}=R_{01}$

Требуется: написать программу, позволяющую по заданной шифровке схемы и значениям сопротивлений определить общее сопротивление схемы.

Входной файл: INPUT.TXT

Входные данные: Первая строка – одно натуральное число N ($N < 100$) – количество видов резисторов в схеме. Вторая строка – символы R_{nm} , « \langle », « $=$ » – схема, зашифрованная по описанным выше правилам. Далее N строк вида $R_{nm}=Z_{nm}$, где R_{nm} – обозначение резистора в схеме, а Z_{nm} ($0 < Z_{nm} < 1000$) – вещественное значение сопротивления.

Выходной файл: OUTPUT.TXT

Выходные данные: одно число R – сопротивление всей схемы с точностью до трёх знаков в десятичной части.

Пример входных и выходных файлов:

INPUT.TXT	OUTPUT.TXT
4 R01=R02-R03-R04=R02=R01 R01=2.5 R02=4 R03=1 R04=6	3.763

Задача № 7. Ваня наблюдает за лягушкой. Изначально она сидит в точке 0 числовой прямой. Каждую секунду она прыгает на 1 вправо, пока не достигнет точки К. Затем она начинает каждую секунду прыгать на 1 влево, пока не вернётся в точку 0, затем опять вправо и так далее. Требуется определить, где окажется лягушка через Т секунд.

Формат входных данных: Во входном файле input.txt в двух строках находятся два числа К и Т. Оба натуральных и не превосходят 1 000 000 000.

Формат выходных данных: Вывести в выходной файл output.txt одно число – координату лягушки в момент времени Т.

Пример:

Пример входного файла input.txt	Пример выходного файла output.txt
5 8	2

Примечание: Программа не должна выводить никаких дополнительных сообщений, а также не может содержать операторов, вызывающих задержку выполнения программы (например, readln в конце программы).

Задача № 8. Юный математик Матвей интересуется теорией вероятностей, и по этой причине у него всегда есть с собой несколько стандартных шестигранных игральных кубиков. Стандартный шестигранный кубик имеет три противоположащих пары граней, которые размечены таким образом, что напротив грани с числом 1 находится грань с числом 6, напротив грани с числом 2 – грань с числом 5 и напротив грани с числом 3 – грань с числом 4.

Анализируя различные игры с шестигранными кубиками, Матвей придумал новую игру. В эту игру играют два игрока, и проходит она следующим образом: первый игрок бросает один или несколько стандартных кубиков (количество кубиков он определяет сам). После этого первому игроку начисляется количество очков, равное сумме чисел, оказавшихся на верхних гранях всех кубиков, а второму игроку – сумма чисел, оказавшихся на нижних гранях этих кубиков. Побеждает тот, кто набрал больше очков.

Например, если был брошен один кубик, и на верхней его грани выпало число два, то первый игрок получает два очка, а второй – пять. В свою очередь, если было брошено два кубика и на их верхних гранях выпало по единице, то первый игрок получает также два очка, а второй игрок – двенадцать очков, так как на нижних гранях этих кубиков оказались шестёрки.

Матвей рассказал об этой игре своему другу, юному информатику Фоме, и они начали играть в неё через Интернет. Поскольку Фома не видит результат броска и не

знает, сколько кубиков бросает Матвей как первый игрок, то о набранных каждым игроком очках он узнает только от Матвея. Чтобы проверить достоверность этой информации, Фома решил узнать, какое минимальное и максимальное количество очков мог получить он, как второй игрок, если известно, сколько очков набрал Матвей.

Требуется написать программу, которая по количеству очко, набранных первым игроком после броска, определяет наименьшее и наибольшее количество очков, которое может получить второй игрок за этот бросок.

Формат входных данных: Первая строка входного файла содержит одно целое положительное число n – количество очков, которые получил первый игрок ($1 \leq n \leq 10^{10}$).

Формат выходного файла: Выходной файл должен содержать два разделённых пробелом целых числа: минимальное и максимальное количество очков соответственно, которые мог набрать второй игрок при таком броске кубиков.

Примеры:

<i>dices.in</i>	<i>dices.out</i>
2	5 12
36	6 216

Задача № 9. По территории компьютерного лагеря проложен маршрут для поездок на электрокарах. Поскольку на электрокаре можно добраться до ИКТ-центра, то школьник Пахом решил воспользоваться им. Следуя по маршруту, электрокар проехал с постоянной скоростью один за другим два светофора с зеленым светом. Пахому известно, что оба светофора находятся на расстоянии x метров друг от друга и переключаются абсолютно синхронно: зеленый свет горит a минут, потом включается красный свет и горит в течение b минут, после чего светофор переключается опять на зеленый свет и он горит также в течение a минут, и так далее. Переключений на желтый свет у светофоров нет. Скорость движения электрокара по маршруту не превышает 1000 м/мин. Электрокар может проехать на светофоре в тот момент, когда светофор переключается с одного света на другой.

Приехав в ИКТ-центр, Пахом заинтересовался, с какой максимальной постоянной скоростью он мог ехать на электрокаре между двумя светофорами.

Требуется написать программу, которая позволит Пахому выяснить это.

Формат входного файла

Первая строка входного файла содержит три целых числа: a , b и x ($1 \leq a \leq 100$, $1 \leq b \leq 100$, $1 \leq x \leq 10^5$).

Формат выходного файла

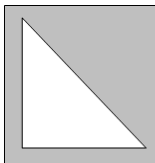
Выходной файл должен содержать одно число – максимальную возможную скорость электрокара между двумя светофорами. Ответ должен отличаться от правильного не более чем на 10^{-9} .

Примеры

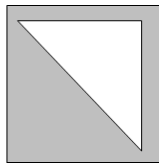
Входные данные	Выходные данные
3 5 4000	800.000000000

Входные данные	Выходные данные
5 10 21010	840.400000000

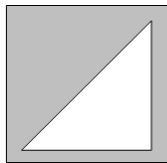
Задача № 10. Как известно, современные видеокарты умеют формировать изображения с использованием только треугольников. Видеокарта POBEDA-2014 не отстает от современных тенденций. Известно, что она умеет отображать только прямоугольные равнобедренные треугольники четырех типов ориентации, представленные на рисунках ниже. Изменять ориентацию этих треугольников видеокарта не может.



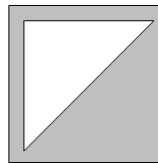
1 тип



2 тип



3 тип



4 тип

Длина катета каждого из представленных выше треугольников равна одному сантиметру. За один такт видеокарта не может отобразить более чем a_i треугольников i -того типа.

Необходимо определить максимально возможную длину стороны квадрата, который может быть изображен видеокарткой на экране монитора за один такт. При этом квадрат должен быть расположен так, чтобы его стороны были параллельны краям монитора.

Требуется написать программу, которая решает поставленную задачу.

Формат входного файла

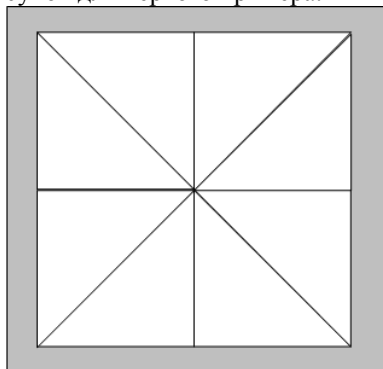
Первая строка входного файла содержит разделенные пробелами четыре целых числа: a_1, a_2, a_3, a_4 .

Формат выходного файла

Выходной файл должен содержать одно число — максимально возможную длину стороны квадрата.

Пояснения к примеру

Далее приведен рисунок для первого примера.



Примеры:

Входные данные	Выходные данные
2 2 2 2	2

Входные данные	Выходные данные
10 10 0 0	3

Задача № 11. Вася готовит инвентарь для ролевой игры. В игре должны принять участие n игроков, каждый из которых будет изображать персонажа фантастического мира. В процессе игры каждый персонаж будет обладать некоторым уровнем x , который представляет собой целое число от 1 до m .

Для обозначения уровня планируется использовать специальные значки двух цветов. Белый значок обозначает один уровень, а красный значок — k уровней. Игрок, изображающий персонажа с уровнем x , должен иметь a белых значков и b красных значков, чтобы сумма $(a + bk)$ была равна x . При этом персонажу не разрешается иметь более чем $(k - 1)$ белых значков.

Значки для игры готовятся заранее, однако уровни персонажей заранее неизвестны. Для успешного проведения игры всем персонажам необходимо выдать соответствующее их уровням количество значков. Возникает вопрос: какое минимальное суммарное количество значков необходимо подготовить для успешного проведения игры при любых уровнях участвующих персонажей.

Требуется написать программу, которая по заданным числам n , m и k вычисляет минимальное количество значков, которое необходимо подготовить для успешного проведения игры.

Формат входного файла

Входной файл содержит расположенные в одной строке три целых числа: n , m и k ($1 \leq n \leq 10^4$, $1 \leq m \leq 10^5$, $1 \leq k \leq 10^5$).

Формат выходного файла

В выходном файле должно содержаться одно целое число — минимальное количество значков, которое требуется подготовить.

Пример входных и выходных данных

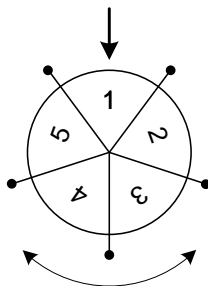
game.in	game.out
3 4 2	9

Пояснения к примеру

В приведенном примере необходимо подготовить 6 красных и 3 белых значка.

Задача № 12. Развлекательный телеканал транслирует шоу «Колесо Фортуны». В процессе игры участники шоу крутят большое колесо, разделенное на сектора. В каждом секторе этого колеса записано число. После того как колесо останавливается, специальная стрелка указывает на один из секторов. Число в этом секторе определяет выигрыш игрока.

Юный участник шоу заметил, что колесо в процессе вращения замедляется из-за того, что стрелка задевает за выступы на колесе, находящиеся между секторами. Если колесо вращается с угловой скоростью v градусов в секунду, и стрелка, переходя из сектора X к следующему сектору, задевает за очередной выступ, то текущая угловая скорость движения колеса уменьшается на k градусов в секунду. При этом если $v \leq k$, то колесо не может преодолеть препятствие и останавливается. Стрелка в этом случае будет указывать на сектор X .



Юный участник шоу собирается вращать колесо. Зная порядок секторов на колесе, он хочет заставить колесо вращаться с такой начальной скоростью, чтобы после остановки колеса стрелка указала на как можно большее число. Колесо можно вращать в любом направлении и придавать ему начальную угловую скорость от a до b градусов в секунду.

Требуется написать программу, которая по заданному расположению чисел в секторах, минимальной и максимальной начальной угловой скорости вращения колеса и величине замедления колеса при переходе через границу секторов вычисляет максимальный выигрыш.

Формат входного файла

Первая строка входного файла содержит целое число n — количество секторов колеса ($3 \leq n \leq 100$).

Вторая строка входного файла содержит n положительных целых чисел, каждое из которых не превышает 1000 — числа, записанные в секторах колеса. Числа приведены в порядке следования секторов по часовой стрелке. Изначально стрелка указывает на первое число.

Третья строка содержит три целых числа: a , b и k ($1 \leq a \leq b \leq 10^9$, $1 \leq k \leq 10^9$).

Формат выходного файла

В выходном файле должно содержаться одно целое число — максимальный выигрыш.

Примеры входных и выходных данных

wheel.in	wheel.out
5 1 2 3 4 5 3 5 2	5
5 1 2 3 4 5 15 15 2	4
5 5 4 3 2 1 2 5 2	5

Пояснения к примерам

В первом примере возможны следующие варианты: можно придать начальную скорость колесу равную 3 или 4, что приведет к тому, что стрелка преодолеет одну границу между секторами, или придать начальную скорость равную 5, что позволит стрелке преодолеть 2 границы между секторами. В первом варианте, если закрутить колесо в одну сторону, то выигрыш получится равным 2, а если закрутить его в противоположную сторону, то — 5. Во втором варианте, если закрутить колесо в одну сторону, то выигрыш будет равным 3, а если в другую сторону, то — 4.

Во втором примере возможна только одна начальная скорость вращения колеса — 15 градусов в секунду. В этом случае при вращении колеса стрелка преодолеет семь границ между секторами. Тогда если его закрутить в одном направлении, то выигрыш составит 4, а если в противоположном направлении, то — 3.

Наконец, в третьем примере оптимальная начальная скорость вращения колеса равна 2 градусам в секунду. В этом случае стрелка вообще не сможет преодолеть границу между секторами, и выигрыш будет равен 5.

Задача № 13. На вход программе подаётся файл с именем `input.txt`, содержащий имена других файлов, располагающихся в том же каталоге, что и программа.

Пример файла `input.txt`:

<code>input.txt</code>
<code>file1.txt</code>
<code>file3.txt</code>
<code>file2.txt</code>
<code>...</code>
<code>text.txt</code>
<code>file.txt</code>
<code>...</code>

Входных файлов может быть очень много. Внутри этих файлов содержатся тексты, оформленные неправильно. В связи с тем, что исправление ошибок оформления вручную занимает много времени, потребовалось написать программу, автоматизирующую данный процесс.

Во всех входных файлах в первой строке содержится одно число – номер типа оформления текста. Существует три типа оформления, которые должна выполнять написанная программа:

1) Программа должна убрать из текста все знаки препинания кроме пробелов. При этом не допускается, чтобы в тексте шли два пробела подряд. В результате данного оформления текста, должны остаться только слова на русском или английском языке, разделённые одним пробелом.

После окончания исправления текста, результат должен быть выгружен в файл с именем *type1-номер-data.txt*, где *номер* – число от 1 до 999, порядковый номер обработанного файла. При этом длина имени файла фиксирована, таким образом, если будет 200 файлов, которые должны быть отформатированы по первому типу, в папке с программой, после её выполнения, должны находиться 200 файлов, начиная с *type1-001-data.txt* и заканчивая *type1-200-data.txt*.

Пример:

<i>file1.txt</i>	<i>type1-001-data.txt</i>
1 asdff , sxzcv,, asdff,,f ,,f aw ddcv z d asdff , sxzcv,, asdff,,f ,,f aw ddcv z d xxcvzcc,,...sdf,,a ss df "" aa d w wwq z	asdff sxzcv asdff f f aw ddcv z d asdff sxzcv asdff f f aw ddcv z d xxcvzcc sdf a ss df aa d w wwq z

2) Программа должна отформатировать текст так, чтобы не было нескольких идущих подряд одних и тех же знаков препинания: «.», «,», «!», «?», «:», «-».

Например, после форматирования строка «abcde,,, fghi...» должна принять вид «abcde, fgi.». Также программа должна исправлять текст так, чтобы перед этими знаками препинания не оказывалось пробелов, но после знака препинания должен быть один пробел, например, строка «abcc ,ae» должна принять вид «abcc, ae».

Пример:

<i>file2.txt</i>	<i>type2-001-data.txt</i>
2 asdf , zz ,, a ss ... dd f,,z aas w q df zxcvvv d,, , , , ,	asdf, zz, a ss. dd f, z aas w q df zxcvvv d,

3) В результате данного форматирования каждое предложение полученного текста должно начинаться с новой строки. Считается, что предложение окончено, если оно заканчивается на знак «!», «,», «?».

Пример:

<i>file3.txt</i>	<i>type3-001-data.txt</i>
3 pred 1. proba 222. prof 3. ap!	pred 1. proba 222. prof 3. ap!

Задача № 14. Списки школ.

Имя входного файла:	schools.in
Имя выходного файла:	schools.out

При регистрации на портале интернет–олимпиады все участники заполняют регистрационную форму, где они указывают название школы, в которой они учатся. Разные участники могут по-разному писать название школы, например, «Физико–математическая школа №18», «ФМШ № 18».

Организаторам олимпиады предоставлена информация о названиях школ, которые написали регистрируемые участники олимпиады. Точно известно, что цифры в названии школы встречаются только в номере школы, а число в записи названия школы встречается ровно один раз и однозначно определяет номер школы. Номер школы является положительным целым числом и не может начинаться с нуля.

Требуется написать программу для сайта интернет–олимпиады, которая поможет организаторам олимпиады получить следующую информацию: количество школ и номера школ, из которых зарегистрировалось не более пяти участников.

Формат входного файла. Первая строка входного файла содержит одно целое число n ($1 \leq n \leq 1000$) – количество названий школ, указанных всеми участниками при регистрации.

Последующие n строк содержат названия школ, указанные всеми участниками. Название школы содержит только заглавные и строчные буквы латинского алфавита, цифры и пробелы, длина названия не превышает 100 символов.

Формат выходного файла. Первая строка выходного файла должна содержать одно целое число m – количество школ, от которых на олимпиаду зарегистрировалось от одного до пяти участников. Последующие m строк должны содержать только номера таких школ, при этом номера должны располагаться по одному в строке в произвольном порядке.

Примеры входных и выходных файлов.

schools.in	schools.out
9	2
Physics and Mathematics School 18	42
9ya shkola imeni Pushkina	18
Lyceum 9	
PaMS 18	
Gemnasium 42	
School 9	
Shkola nomer 9	
High School 9	
School N 9	

Пояснения к примерам. В приведённом примере для участия в интернет–олимпиаде зарегистрировались: два ученика из школы с номером 18, один ученик из школы с номером 42 и шесть учеников из школы с номером 9. Таким образом, от 1 до 5 участников зарегистрировалось от школ с номерами 18 и 42.

Задача № 15. Многие системы форматирования текста, например TEX или Wiki, используют для разбиения текста на абзацы пустые строки. Текст представляет собой последовательность слов, разделенных пробелами, символами перевода строк и следующими знаками препинания: «,», «.», «?», «!», «-», «:» и «'» (ASCII коды 44, 46, 63, 33, 45, 58, 39). Каждое слово в тексте состоит из заглавных и прописных букв латинского алфавита и цифр. Текст может состоять из нескольких абзацев. В этом случае соседние абзацы разделяются одной или несколькими пустыми строками. Перед первым абзацем и после последнего абзаца также могут идти одна или несколько пустых строк.

Дальнейшее использование исходного текста предполагает его форматирование, которое осуществляется следующим образом. Каждый абзац должен быть разбит на строки, каждая из которых имеет длину не больше w . Первая строка каждого абзаца должна начинаться с отступа, состоящего из b пробелов. Слова внутри одной строки должны быть разделены ровно одним пробелом. Если после слова идет один или несколько знаков препинания, они должны следовать сразу после слова без дополнительных пробелов. Если очередное слово вместе со следующими за ним знаками препинания помещается на текущую строку, оно размещается на текущей строке. В противном случае, с этого слова начинается новая строка. В отформатированном тексте абзацы не должны разделяться пустыми строками. В конце строк не должно быть пробелов.

Требуется написать программу, которая по заданным числам w и b и заданному тексту выводит текст, отформатированный описанным выше образом.

Формат входного файла:

Первая строка входного файла содержит два целых числа: w и b ($5 \leq w \leq 100$, $1 \leq b \leq 8$, $b < w$).

Затем следует одна или более строк, содержащих заданный текст. Длина слова в тексте вместе со следующими за ними знаками препинания не превышает w , а длина первого слова любого абзаца вместе со следующими за ним знаками препинания не превышает $(w - b)$.

Размер входного файла не превышает 100 Кбайт. Длина каждой строки во входном файле не превышает 250.

Формат выходного файла:

Выходной файл должен содержать заданный текст, отформатированный в соответствии с приведенными в условии задачи правилами.

Примеры входных и выходных данных

<i>Входные данные</i>	<i>Выходные данные</i>
20 4 Yesterday, All my troubles seemed so far away, Now it looks as though they're here to stay, Oh, I believe in yesterday. Suddenly, I'm not half the man I used to be,	Yesterday, All my troubles seemed so far away, Now it looks as though they' re here to stay, Oh, I believe in yesterday. Suddenly, I' m

There's a shadow hanging over me, Oh, yesterday came suddenly...	not half the man I used to be, There's a shadow hanging over me, Oh, yesterday came suddenly...
---	--

Задача № 16. Реализовать функции, позволяющие производить побитовый сдвиг числа на указанное число влево и вправо. В функции должны передавать следующие параметры: *ch* – исходное число, *bk* – количество бит числа, *count* – смещение.

Функция `mov_left(ch, bk, count: integer): integer` производит побитовое смещение влево.

Функция `mov_right(ch, bk, count: integer): integer` производит побитовое смещение вправо.

Данные функции должны выполнять сдвиг без использования массивов или строковых переменных для хранения промежуточных значений битов числа. Смещения должны выполняться только с использованием математических операций. Внутри этих функций разрешается использовать переменные целого и вещественных типов.

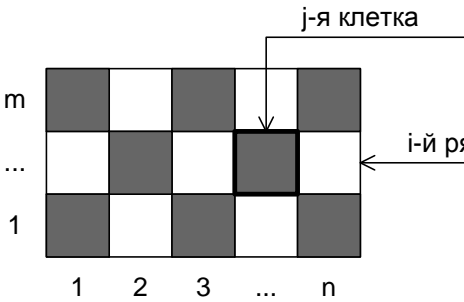
На вход программе подаётся исходное число *ch*, а также два числа *l* и *r*, указывающие, на сколько бит надо сместить указанное число влево и вправо соответственно. На выходе программа должна вывести два числа после выполнения побитового сдвига влево и вправо.

Примеры входных и выходных данных:

№	Входные данные	Выходные данные
1	13 3 3	14 11
2	25 2 4	7 19
3	100 3 5	38 19

Задача № 17. Аня разделила доску размера $m \times n$ на клетки размера 1×1 и раскрасила их в черный и белый цвет в шахматном порядке. Васю заинтересовал вопрос: клеток какого цвета получилось больше — черного или белого.

Для того, чтобы выяснить это, он спросил у Ани, в какой цвет она раскрасила *j*-ю клетку в *i*-м ряду доски. По этой информации Вася попытался определить, клеток какого цвета на доске больше.



Требуется написать программу, которая по размерам доски и цвету *j*-й клетки в *i*-м ряду определит, клеток какого цвета на доске больше — черного или белого.

Формат входного файла

Входной файл содержит пять целых чисел: m, n, i, j и c ($1 \leq m, n \leq 10^9, 1 \leq i \leq m, 1 \leq j \leq n, c = 0$ или $c = 1$). Значение $c = 0$ означает, что j -я клетка в i -м ряду доски раскрашена в черный цвет, а значение $c = 1$ — в белый цвет.

Формат выходного файла

Выходной файл должен содержать одно из трех слов:

- black, если черных клеток на доске больше,
- white, если белых клеток на доске больше,
- equal, если черных и белых клеток на доске поровну.

Примеры входных и выходных данных

chess.in	chess.out
3 5 1 1 0	black
3 5 2 1 0	white
4 4 1 1 1	equal

Задача № 18. Чемпионат по стрельбе

Победитель школьного этапа олимпиады по информатике нашел дома в старых бумагах результаты чемпионата страны по стрельбе из лука, в котором участвовал его папа. К сожалению, листок с результатами сильно пострадал от времени, и разобрать фамилии участников было невозможно. Остались только набранные каждым участником очки, причем расположились они в том порядке, в котором участники чемпионата выполняли стрельбу.

Расспросив папу, школьник выяснил, что количество очков, которое набрал папа, заканчивается на 5, один из победителей чемпионата стрелял раньше, а папин друг, который стрелял сразу после папы, набрал меньше очков. Теперь он заинтересовался, какое самое высокое место мог занять его папа на том чемпионате.

Будем считать, что участник соревнования занял k -е место, если ровно $(k - 1)$ участников чемпионата набрали строго больше очков, чем он. При этом победителями считались все участники чемпионата, занявшие первое место.

Требуется написать программу, которая по заданным результатам чемпионата определяет, какое самое высокое место на чемпионате мог занять папа победителя школьного этапа олимпиады по информатике.

Формат входного файла

Первая строка входного файла содержит целое число n — количество участников чемпионата страны по стрельбе ($3 \leq n \leq 10^5$).

Вторая строка входного файла содержит n положительных целых чисел, каждое из которых не превышает 1000, — очки участников чемпионата, приведенные в том порядке, в котором они выполняли стрельбу.

Формат выходного файла

В выходном файле должно содержаться одно целое число — самое высокое место, которое мог занять папа школьника. Если не существует ни одного участника чемпионата, который удовлетворяет, описанным выше условиям, выведите в выходной файл число 0.

Примеры входных и выходных данных

shooting.in	shooting.out
7 10 20 15 10 30 5 1	6
3 15 15 10	1
3 10 15 20	0

Задача № 19. Натуральное число a называется делителем натурального числа b , если $b = ac$ для некоторого натурального числа c . Например, делителями числа 6 являются числа 1, 2, 3 и 6. Два числа называются *взаимно простыми*, если у них нет общих делителей кроме 1. Например, 16 и 27 взаимно просты, а 18 и 24 — нет.

Будем называть *нормальным* набор из k чисел (a_1, a_2, \dots, a_k) , если выполнены следующие условия:

- 1) каждое из чисел a_i является делителем числа n ;
- 2) выполняется неравенство $a_1 < a_2 < \dots < a_k$;
- 3) числа a_i и a_{i+1} для всех i от 1 до $k - 1$ являются взаимно простыми;
- 4) произведение $a_1 a_2 \dots a_k$ не превышает n .

Например, набор (2, 9, 10) является нормальным набором из 3 делителей числа 360.

Требуется написать программу, которая по заданным значениям n и k определяет количество нормальных наборов из k делителей числа n .

Формат входного файла

Первая строка входного файла содержит два целых числа: n и k ($2 \leq n \leq 10^8$, $2 \leq k \leq 10$).

Формат выходного файла

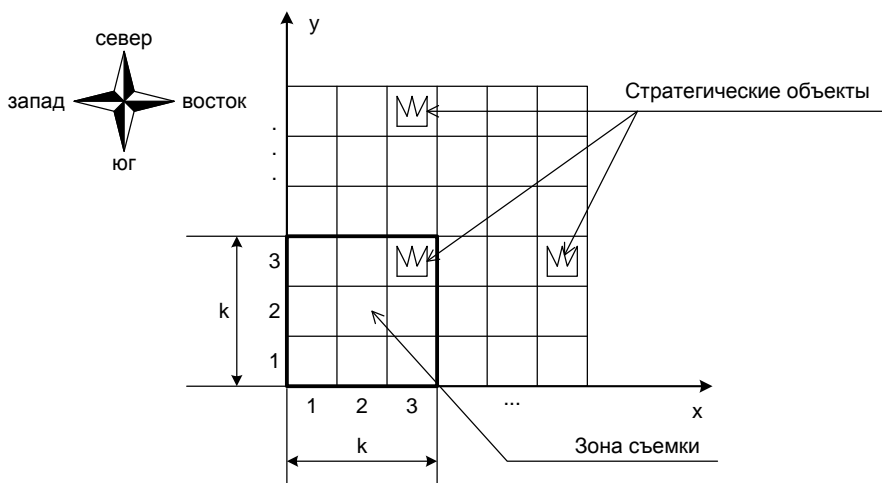
В выходном файле должно содержаться одно число — количество нормальных наборов из k делителей числа n .

Пример входных и выходных данных

divisors.in	divisors.out
90 3	16
10 2	4

Задача № 20. Отделу космических исследований поступило задание сфотографировать из космоса n объектов в заданной области. Область имеет форму квадрата размером 50×50 километров. Если разделить ее на квадраты размером 1×1 километр, то интересующие отдел объекты окажутся в центрах некоторых единичных квадратов.

Введем систему координат, направив ось OX с запада на восток и ось OY с юга на север. Тогда каждому единичному квадрату будут сопоставлены координаты в диапазоне от 1 до 50, как показано на рисунке ниже.



Для космической съемки используется специальный фотоаппарат высокого разрешения, установленный на космическом спутнике. Фотоаппарат может делать снимки квадратных участков земной поверхности размером $k \times k$ километров. Исходно аппарат наведен на юго-западный угол заданной области, то есть, если сделать снимок, на нем будут видны единичные квадраты с координатами x и y от 1 до k километров.

С помощью специальных двигателей можно изменять орбиту спутника, что приводит к изменению участка съемки. За один день орбиту спутника можно изменить таким образом, что участок съемки сместится либо на один километр на запад, либо на один километр на восток, либо на один километр на север. Переместить участок съемки на юг невозможно. Непосредственно между перемещениями спутника можно сделать снимок, временем съемки можно пренебречь.

Руководство отдела заинтересовалось вопросом: за какое минимальное количество дней можно сделать снимки всех объектов заданной области.

Требуется написать программу, которая по заданному расположению объектов и размеру снимка k определит минимальное время, за которое можно сделать снимки всех объектов заданной области.

Формат входного файла

Первая строка входного файла содержит два целых числа: n и k ($1 \leq n \leq 1000$, $1 \leq k \leq 5$).

Следующие n строк содержат по два целых числа: x_i и y_i — координаты объектов в заданной области ($1 \leq x_i, y_i \leq 50$).

Формат выходного файла

В выходном файле должно содержаться одно целое число: минимальное количество дней, которое требуется для получения снимков всех объектов в заданной области.

Примеры входных и выходных данных

space.in	space.out
4 1 1 1 10 2 1 3 10 4	30
4 2 1 1 10 2 1 3 10 4	10
1 1 1 1	0
3 3 3 3 3 6 6 3	7

Пояснения к примерам

В первом примере возможна следующая последовательность действий: сделать снимок, 9 раз сместиться на восток, сместиться на север, сделать снимок, 9 раз сместиться на запад, сместиться на север, сделать снимок, 9 раз сместиться на восток, сместиться на север, сделать снимок. Всего требуется 30 перемещений участка съемки.

Во втором примере объекты расположены там же, но размер снимка больше, поэтому можно действовать так: сделать снимок, сместиться на север, сделать снимок, 8 раз сместиться на восток, сделать снимок, сместиться на север, сделать снимок. Всего требуется лишь 10 перемещений участка съемки.

В третьем примере перемещать участок съемки не требуется, можно просто сделать снимок.

Четвертый пример соответствует приведенному выше рисунку.