Контест 20.03.22

A. Ленивый контролер

1 секунда, 256 мегабайт

В одном из цехов N-ского машиностроительного завода выпускаются детали. В целях контроля качества, вес каждой детали должен контролироваться, для чего существует контрольный пост. К сожалению, контролер на этом посту довольно ленив, и вместо того чтобы отдельно взвешивать каждую деталь, он решил немного «оптимизировать» процесс.

В конце рабочего дня, на проверку поступает ящик с множеством однотипных деталей. Контролер знает, что качественная деталь должна весить целое количество грамм, но слишком ленив чтобы заглянуть в документы и узнать сколько именно. Также, все однотипные детали должны весить одинаково.

«Оптимизированный» процесс контроля заключается в следующем:

1. Контролер выполняет k взвешиваний, и нумерует их от 1 до k.
2. Для i-го взвешивания случайно выбирается число a_i — количество деталей, это число записывается в журнал взвешиваний.
3. Выбранное количество произвольно взятых деталей помещается на весы, и их суммарная масса w_i также записывается в журнал.
4. По выполнению k взвешиваний, журнал анализируется, и делается вывод о наличии или отсутствии брака.

Естественно, что такой «оптимизированный» процесс выявляет наличие брака далеко не всегда, да и не указывает на конкретную бракованную деталь. Контролер настолько ленив, что в случае наличия признаков брака бракует всю партию, в противном случае партия получает знак качества.
Вашей задачей будет написать программу, которая по журналу взвешиваний определит, получит ли партия знак качества или окажется забракованной.

Входные данные
Первая строка входного файла содержит целое число k $(1 \le k \le 1000)$. Далее следуют k строк, на каждой из которых записаны целые числа a_i $(0 < a_i \le 1000)$ и w_i $(0 < w_i < 1000000)$, разделенные пробелом.

Выходные данные
Выходная строка должна содержать слово «DEFECT» если в партии точно есть бракованные детали, или «QUALITY», если признаков брака не обнаружено.

входные данные
3 10 30 5 15 2 6
выходные данные
QUALITY

входные данные
2 2 5 4 10
выходные данные
DEFECT

Участник должен выводить «QUALITY» в том случае, если нет однозначных признаков брака, даже если для однозначного вывода о качественности деталей не хватает данных.

B. Гремлины атакуют!

4 секунды, 64 мегабайта

В уездном американском городе чрезвычайная ситуация. По городу разбежались гремлины и остальному миру угрожает опасность, так как как минимум один из них покинул город. Специальные службы восстанавливают картину событий и пытаются обнаружить самый ранний момент, когда это могло случиться.

Город представлен в виде квадратной клетчатой доски размера $N \times N$. Каждая клетка представляет один дом. Гремлины боятся яркого света и поэтому прячутся в темноте. Далее, по мере того как люди ложатся спать, свет в домах гаснет. Гремлины могут перемещаться по горизонтали и по вертикали от одного дома, где не горит свет, к другому. Как только они достигают дома, который находится на границе города, они сбегают из города.

У вас есть достоверные данные о том в каких домах гремлины находились в начале, а также история выключения света в домах. Вам надо вывести порядковый номер дома в истории, после выключения света в котором хотя бы один гремлин может сбежать из города. Если гремлины сразу же могут сбежать из города, вывести 0. Нумерация домов в истории начинается с единицы.

Входные данные
В первой строке через пробел вводятся три целых числа $1 < N <= 500$, $1 <= M <= N^2$, $1 <= K <= N^2$, где N определяет размер города, M - количество домов, в которых в самом начале находятся гремлины, K - размер истории выключений света в домах.

Далее следуют M строк, каждая из которых содержит пару чисел $0 <= x_i, y_i < N$, задающих координаты домов, где находятся гремлины в начале.

После этого следуют K строк, каждая из которых содержит пару чисел $0 <= x_j, y_j < N$, задающих координаты домов, в которых выключается свет.

Выходные данные
Единственное число от 0 до K, которое задает номер дома, после выключения света в котором хотя бы один гремлин имел возможность сбежать из города.

входные данные
3 1 3 1 1 0 0 0 1 0 2
выходные данные
2

входные данные
5 2 5 0 1 4 1 0 0 1 1 2 2 3 3 4 4
выходные данные
0

входные данные
4 2 3 1 1 1 2 2 0 3 1 1 3
выходные данные
3

входные данные
5 2 6 1 1 3 3 1 2 1 3 2 3 3 0 3 1 2 1
выходные данные
6

входные данные
7 6 7 1 4 1 1 2 3 3 1 4 4 5 2 0 4 2 4 3 4 1 0 2 1 5 1 5 0
выходные данные
1

Входные данные гарантируют, что побег мог быть совершен.

C. Размерности

1 секунда, 256 мегабайт

При выполнении различных физических расчетов часто приходится иметь дело не только с числами, но и с размерностями величин. При этом размерности могут получаться весьма сложными и их требуется сокращать: например, размерность кг/(кг/м/с) это просто м/с.

Напишите программу, которая максимально сократит размерность и запишет все встречающиеся величины слева направо в алфавитном порядке (если размерность является дробью, то в алфавитном порядке записываются отдельно числитель и знаменатель).

Входные данные

Первая строка входного файла содержит единственную строку символов — размерность, подлежащую сокращению.

Размерности величин обозначаются латинскими строчными и заглавными буквами.

Кроме латинских букв в выражении могут присутствовать знаки умножения «*», деления «/» и круглых скобок. Степени величин выражаются кратным повторением величины. Порядок выполнения операций соответствует общепринятому. Величины, отличающиеся регистром букв считаются различными.

Длина сокращаемой размерности не превосходит 1000 символов.

Выходные данные

Выходные данные содержат сокращенную размерность. Первая строка содержит числитель размерности, вторая — знаменатель. Допустимо в виде числителя и/или знаменателя указывать единицу.

входные данные
kg/(kg/(m/s))
выходные данные
m s

входные данные
kg*a/(Fs*B)*A*Kt
выходные данные
A*a*Kt*kg B*Fs

Сортировка в алфавитном порядке подразумевает следующий порядок букв: «AaBbCbDdEeFfGgHhIiJjKkLlMmNnOoPpQqRrSsTtUuVvWwXxYyZz». Более короткие строки стоят раньше более длинных.

D. Мы делили апельсин...

1 секунда, 256 мегабайт

Как известно апельсины состоят из долек, и их довольно удобно делить на несколько человек. Очевидно, что сколько бы ни было долек в апельсине его всегда можно съесть в одиночку, или поделить не компанию в которой столько же людей, сколько долек в самом апельсине.

Если дольки апельсина можно поровну разделить среди n человек, то скажем что он подходит для такой компании. А какое минимальное количество долек m должно быть в апельсине, чтобы он подходил k различным компаниям (компании считаются различными, если они состоят из разного числа человек)?

Напишите программу, которая по числу компаний k найдет минимальное число долек апельсина, подходящее ровно k различным компаниям, или сообщит, что такого числа не существует.

Входные данные

Входной файл содержит единственное целое число k $(1 \le k \le 1000)$.

Выходные данные

Выходной файл содержит единственное целое число m, или -1 если такого числа не существует.

входные данные
4
выходные данные
6

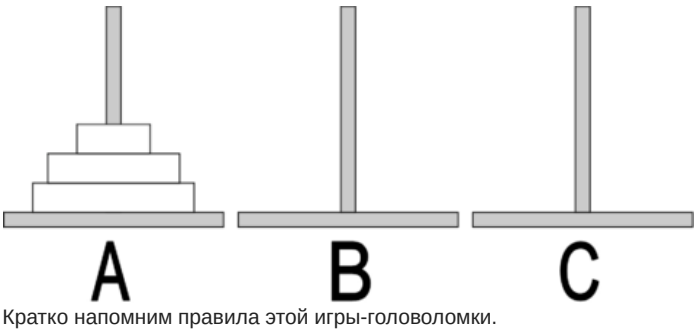
входные данные
1
выходные данные
1

Апелисин из 6 долек можно поровну разедить на 1, 2, 3 или 6 человек.

E. Ханойская башня

1 секунда, 256 мегабайт

Стало доброй традицией на соревнованиях по программированию решать задачу про Ханойскую башню.



Кратко напомним правила этой игры-головоломки.

Имеются три стержня: A, B, C. Первоначально на исходном стержне A размещены N дисков различного диаметра: самый маленький диск наверху, и далее – по возрастанию диаметра. Второй и третий стержни пока пусты.

Требуется переместить все диски с исходного стержня A на результирующий стержень B, используя третий стержень C как вспомогательный. За один ход разрешается взять один (обязательно верхний) диск и надеть его на другой стержень, возможно пустой. Причем, диск большего диаметра запрещается класть на диск меньшего размера. Во многих учебниках по программированию приводится процедура решения этой головоломки.

```
Procedure Hanoi (X, Y, Z: char; N: integer);
Begin
 If N>0 then
    Begin
       Hanoi (X, Z, Y, N-1);
       Writeln('Disk ', N, ' from ', X, ' to ', Y);
       Hanoi (Z, Y, X, N-1)
    End
End;
```

Доцент П. из города Р., готовясь к чемпионату по программированию, решил размяться и принялся вручную перекладывать диски Ханойской башни. (Для этого он использовал кольца от детской пирамидки и три стержня, отобранные у внучки.)

В какой-то момент внучка, обидевшись на деда, отобрала часть колец, не тронув только исходный стержень с нанизанными дисками-кольцами.

Помогите доценту и рассчитайте, какое минимально возможное число ходов он успел сделать, если известно текущее расположение дисков на исходном стержне.

Перекладывания дисков выполняются по стандартному алгоритму, описанному в процедуре выше. Комбинация дисков, заданная во входном файле, гарантированно достижима.

Входные данные

В первой строке записано одно целое число N – исходное количество дисков ($1 \le N \le 100$). Во второй строке файла заданы N символов '0' и '1'. Символ в i-й позиции означает, присутствует (задается знаком '1') или нет (задается знаком '0') диск с номером i на исходном стержне в момент прерывания игры.

Выходные данные

Выходной файл должен содержать одно целое число в двоичном виде без ведущих нулей – минимальный из возможных номеров ходов, после выполнения которого диски на исходном стержне размещены заданным образом.

входные данные
3 111
выходные данные
0

входные данные
3 001
выходные данные
10

входные данные
5 00000
выходные данные
10000

F. Камешки

1 секунда, 256 мегабайт

На экзамене по информатике Вася решал следующую задачу:

«Два игрока, Петя и Ваня, играют в следующую игру. Перед игроками лежит куча камней. Игроки ходят по очереди, первый ход делает Петя. За один ход игрок может добавить в кучу a каменей или увеличить количество камней в куче в два раза. Например, если $a = 2$, имея кучу из 10 камней, за один ход можно получить кучу из 12 или 20 камней. У каждого игрока, чтобы делать ходы, есть неограниченное количество камней.

Игра завершается в тот момент, когда количество камней в куче становится не менее n. Проигравшим считается игрок, сделавший последний ход, то есть получивший кучу, в которой будет n или больше камней. В начальный момент в куче было s камней. Говорят, что игрок имеет выигрышную стратегию, если он может выиграть при любых ходах противника.»

К сожалению Вася не помнит значения a и n, которые были на экзамене, но помнит при каких начальных s выигрывали Петя или Ваня. Также он помнит что a, n и s не превосходили 10^3. Напишите программу, которая по различным значениям s и информации о выигрышах для этих s найдет минимальные подходящие a и n.

Входные данные

Первая строка входного файла содержит число k — количество позиций ($1 \le k \le 1000$), для которых Вася помнит исходы.

Далее следуют k строк, каждая из которых содержит два целых числа. Первое из этих чисел — начальное количество камней s_i ($0 < s_i \le 1000$), второе равно 1 если при количестве камней s_i выигрывает Петя, или 2 если при этом количестве камней выигрывает Ваня. Все начальные позиции s_i различны.

Выходные данные

Выходной файл должен содержать два целых числа n и a, разделенных пробелом ($0 < n, a \le 1000$).

Если задача допускает несколько решений то должно быть выведено решение с минимальным n, если для минимального n существует несколько решений, то должно быть выведено решение с минимальными n и a.

Если решения в заданных ограничениях не существует то должны быть выведены два нуля.

входные данные
4 1 2 2 1 3 1 4 2
выходные данные
5 1

входные данные
2 1 2 2 2
выходные данные
0 0

G. Неслучайные числа

1 секунда, 256 мегабайт

Для генерации псевдослучайных чисел используют различные методы. Одним из таких методов является использование булевых рекуррентных выражений. В таком случае, псевдослучайное число представляется в виде битовой последовательности, такой что каждый следующий бит вычисляется из предыдущих по некоторой формуле. Однако такое задание фактически определяет последовательность по первым ее членам.

Альтернативой может служить использование булевых уравнений. Будем говорить, что битовая последовательность $x_1, x_2, x_3, \ldots, x_n$ удовлетворяет некоторому уравнению

$$F(x_i, x_{i-1}, \ldots, x_{i-k}) = 1,$$

если это уравнение верно для любых $x_i, x_{i-1}, \ldots x_{i-k}$, при $i > k$. Напишите программу, которая по заданному уравнению найдет количество последовательностей длины n, удовлетворяющих этому уравнению.

Входные данные

Первая строка входного файла содержит целое число n $(2 \le n \le 10^3)$. Вторая строка содержит левую часть уравнения, закодированного следующим способом:

- бит с индексом x_{i-r} $(0 \le r \le 9, r < n)$ обозначается как r, то есть x_{i-5} будет обозначено как 5
- операция «и» (конъюнкция) обозначается как «&»
- операция «или» (дизъюнкция) обозначается как «|»
- операция «исключающее или» (сложение по модулю 2) обозначается как «+»
- операция «не» (отрицание) обозначается как «-» (знак минус)
- в уравнении допускаются скобки, операции в скобках выполняются раньше чем другие
- бинарные операции не стоящие в скобках считаются одноранговыми и выполняются строго слева направо
- пробелы внутри соотношения не допускаются

Длина левой части уравнения не превосходит 1000 символов.

Выходные данные

Выходной файл содержит единственное число — количество битовых последовательностей длинны n удовлетворяющих уравнению. Число должно быть выведено по модулю 2^{60}.

входные данные
3 (0+2)|(0&2)
выходные данные
6

входные данные
4 (0+2)|-(0&2)
выходные данные
9

Первый пример кодирует уравнение $(x_i + x_{i-2})|(x_i \& x_{i-2}) = 1$, которому удовлетворяют следующие последовательности: 001, 011, 100, 101, 110, 111. Второй пример кодирует уравнение $(x_i + x_{i-2})|(-(x_i \& x_{i-2})) = 1$, которому удовлетворяют последовательности: 0000, 0001, 0010, 0011, 0100, 0110, 1000, 1001, 1100.

H. Последовательность в себе

1 секунда, 256 мегабайт

Назовем последовательность $a_0, a_1, a_2, \ldots a_{k-1}$ «самоописывающейся», если значение a_i — это количество чисел i, встречающихся в этой последовательности.

Например, такова последовательность 1, 2, 1, 0 — «0» встречается 1 раз, «1» встречается 2 раза, «2» встречается 1 раз и «3» вообще не встречается.

Вашей задачей, будет написание программы, которая по заданной длине последовательности k, выведет ее элементы с указанными индексами, либо сообщит что такой последовательности не существует. Если таких последовательностей несколько, то правильной считается любая из них.

Входные данные

Первая строка входного файла содержит длину последовательности k $(1 \le k \le 2^{30})$. Вторая строка входного файла содержит число n $(0 < n \le 10^5)$ — количество искомых элементов последовательности. Третья строка строка содержит n целых чисел $r_1\ r_2\ r_3 \ldots r_n$ $(0 \le n_i < k)$, разделенных пробелами — индексы тех элементов последовательности, которые нужно вывести.

Выходные данные

Первая строка выходного файла содержит 0 если «самоописывающейся» последовательности длины k не существует. В противном случае в первой строке указывается число n, далее, во второй строке выходного файла, следуют n чисел, разделенным пробелами — элементы последовательности с индексами указанными во входном файле в порядке появления этих индексов.

входные данные
4 4 0 1 2 3
выходные данные
4 1 2 1 0

входные данные
4 4 3 2 1 0
выходные данные
4 0 1 2 1

входные данные
1 1 0
выходные данные
0

входные данные
5 3 4 4 4

выходные данные
3 0 0 0

I. Крестики и нолики

1 секунда, 256 мегабайт

Существует множество вариантов игры в крестики и нолики. Рассмотрим одну из них.

На клетчатой квадратной доске расставлено некоторое количество крестиков и ноликов, а также оставлены пустые клетки. Игрок должен заполнить пустые клетки так, чтобы ни крестики ни нолики не образовали линии более чем из трех одинаковых значков подряд. Линии учитываются по горизонтали, вертикали или любой диагонали.

Вашей задачей будет написать программу, которая по заданному начальному полю заполнит все пустые клетки крестиками и ноликами по описанным выше правилам. Если таких расстановок несколько, то правильной будет считаться любая из них.

Гарантируется, что для входных данных, существует хотя бы одно правильное решение.

Входные данные
Первая строка входного файла содержит число n — размер доски $(3 < n \leq 20)$. Далее следуют n строк по n символов в каждой — начальное состояние игрового поля. Крестик обозначается знаком «+» (плюс), нолик — символом «0» (нуль), пустое поле — символом «.» (точка).

Выходные данные
Формат выходных идентичен формату входных данных (включая размер доски). Решение не должно содержать пустых полей.

входные данные
4 0.0. 00+0 0+00 ++..
выходные данные
4 0+00 00+0 0+00 ++0+

J. R u really ready?

1 секунда, 256 мегабайт

У вас есть две строки, состоящие из строчных латинских букв: шаблон p и текст t. Кроме того шаблон может содержать символ $+$, который означает повторение предшествующего ему символа один или более раз, и символ $*$, означающий повторение предшествующего ему символа любое количество раз.

Вам необходимо ответить на вопрос - соответствует ли текст t шаблону p.

Например, текст**"*ebooooy*"** соответствует шаблонам **"*ebo+y*"**, **"*ebo*o*o+y+*"**, но не соответствует **"*ebb*oy*"** или **"*ebooo*"**.

Входные данные
Первая строка содержит шаблон p, а вторая текст t. Обе строки не пусты и содержат не более 1000 символов. Гарантируется, что p задает допустимый шаблон, то есть не содержит двух подряд идущих символов $+$ и $*$ и первым символом p является строчная латинская буква.

Выходные данные
Выведете *Yes* если текст соответствует шаблону и *No* в противном случае.

входные данные
pa*t+ern pattern
выходные данные
Yes

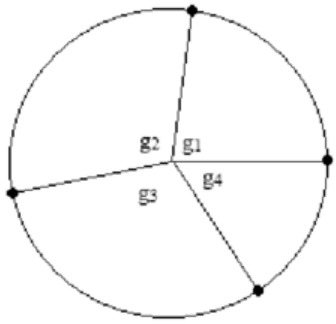
входные данные
c*cp+p cpp
выходные данные
Yes

входные данные
b+b b
выходные данные
No

K. Мезонный коллайдер

1 секунда, 256 мегабайт

Вдоль большого мезонного коллайдера установлены 4 искровые камеры, следящие за взаимодействиями двигающихся по кругу с большими скоростями мезонов.



Чтобы получать достоверные синхронизированные данные с искровых камер и быстро их обрабатывать, внутри круга требуется разместить 2 компьютера, соединить их кабелем друг с другом, а также с искровыми камерами. Каждая камера должна быть соединена хотя бы с одним компьютером. Для максимальной синхронизации поставлена задача минимизации общей длины используемого кабеля.

Напишите программу, которая по известному расположению искровых камер рассчитывает длину кабеля.

Входные данные
Первая строка содержит пять целых чисел, разделенных пробелами: R – радиус окружности, по которой двигаются мезоны ($1 \leq R \leq 100$), g_1, g_2, g_3, g_4– углы, под которыми из центра окружности видны искровые камеры ($60^\circ \leq g_i \leq 360^\circ$).

Выходные данные
Выведите одно вещественное число — минимальную общую длину необходимого кабеля. Ваш ответ будет засчитан, если абсолютная или относительная погрешность вашего ответа не превышает 10^{-6}.

Формально, пусть ваш ответ равен a, а ответ жюри равен b. Ваш ответ будет зачтен, если $\frac{|a-b|}{\max(1,|b|)} \leq 10^{-6}$.

входные данные
10 60 120 60 120
выходные данные
34.64101615

Иллюстрация для теста