

Задача А. Rope Folding

Имя входного файла: `folding.in`
Имя выходного файла: `folding.out`
Ограничение по времени: 2 секунды
Ограничение по памяти: 256 мегабайт

У Фермера Джона есть веревка длины L ($1 \leq L \leq 10\,000$), которую он использует на ферме. Вдоль веревки завязаны узлы в N ($2 \leq N \leq 100$) различных местах, в том числе на обоих её концах.

ФД заметил, что в некоторых точках можно перегнуть веревку так, что после складывания узлы на обеих сторонах веревки встанут точно друг напротив друга:



Помогите ФД посчитать количество точек, в которых можно перегнуть и сложить веревку указанным образом. Допускается перегибание веревки непосредственно по узлу, за исключением узлов в начале и конце веревки. Лишние узлы, оставшиеся на более длинной части, не учитываются — требуется лишь выравнивание узлов в той части, где веревка находится с обеих сторон. Делать можно лишь одно складывание за раз.

Формат входных данных

- Строка 1: Два разделенных пробелом целых числа, N и L .
- Строки 2... $N+1$: Каждая строка содержит единственное целое число от 0 до L — расстояние от начала веревки до данного узла. Две из этих строк будут 0 и L .

Формат выходных данных

- Строка 1: Количество корректных точек перегиба веревки.

Пример

<code>folding.in</code>	<code>folding.out</code>
5 10 0 10 6 2 4	4

Замечание

В примере на веревке длины 10 завязаны 5 узлов на расстояниях 0, 2, 4, 6, 10 от начала веревки. Корректные точки перегиба находятся на расстояниях 1, 2, 3 и 8 от начала веревки.

Задача В. Overplanting

Имя входного файла: `planting.in`
Имя выходного файла: `planting.out`
Ограничение по времени: 2 секунды
Ограничение по памяти: 256 мегабайт

Фермер Джон купил новую машину, которая умеет садить траву на прямоугольных участках со сторонами, параллельными осям координат. К несчастью, однажды машина сломалась и посадила траву не на одном, а сразу на N ($1 \leq N \leq 10$) прямоугольных участках, некоторые из которых могут даже накладываться друг на друга.

По заданным прямоугольным участкам, засаженным травой, помогите ФД определить общую площадь, покрытую травой.

Формат входных данных

- Строка 1: Целое число N .
- Строки 2... $N+1$: В каждой строке через пробел записано четыре целых числа x_1, y_1, x_2, y_2 , описывающих прямоугольный участок, засаженный травой, с левым-верхним углом (x_1, y_1) и правым-нижним (x_2, y_2) . Все координаты — целые числа от $-10\,000$ до $10\,000$.

Формат выходных данных

- Строка 1: Общая площадь, покрытая травой.

Пример

<code>planting.in</code>	<code>planting.out</code>
2 0 5 4 1 2 4 6 2	20

Задача С. Моо

Имя входного файла: moo.in
Имя выходного файла: moo.out
Ограничение по времени: 2 секунды
Ограничение по памяти: 256 мегабайт

Коровы чрезвычайно увлечены новой игрой, которая называется «Моо». В ходе этой игры коровы выстраиваются в ряд и друг за другом быстро произносят каждая свою букву из специальной последовательности. Первая из них, кто допустит ошибку — проигрывает.

Последовательность букв для этой игры бесконечна. Ее начало выглядит как:

m o o m o o o m o o m o o o o m o o m o o o m o o m o o o o o . . .

Проще всего эту последовательность описать рекурсивно. Пусть S_0 — последовательность из трех символов «m o o». Последовательность S_k получается в результате записывания друг за другом: копии последовательности S_{k-1} , затем «m o . . . o» с $k+2$ символами «o», а затем еще одной копии последовательности S_{k-1} . Например:

$S_0 = m o o$

$S_1 = m o o m o o o m o o$

$S_2 = m o o m o o o m o o m o o o o m o o m o o o m o o$

Строка, получающаяся в результате бесконечного применения этого правила, и используется для игры.

Бесси считает себя умной коровой и хочет предсказать каким будет символ, находящийся на позиции N в этой строке — «m» или «o». Помогите ей!

Формат входных данных

- Строка 1: Единственное целое число N ($1 \leq N \leq 10^9$).

Формат выходных данных

- Строка 1: Единственная строка вывода должна содержать один символ, либо «m», либо «o».

Пример

moo.in	moo.out
11	m

Замечание

В примере Бесси хочет предсказать 11-й символ строки.

Задача D. Times17

Имя входного файла: `times17.in`
Имя выходного файла: `times17.out`
Ограничение по времени: 2 секунды
Ограничение по памяти: 256 мегабайт

Фермер Джон понял, что разработка программного обеспечения — прибыльный бизнес, и теперь пишет небольшие программы для местных фермеров.

Помогите ФД написать одну из таких программ. Программа должна считывать число N , записанное в двоичной системе счисления с количеством разрядов, не превосходящих 1000. И в ответ печатать число $17 \cdot N$, также в двоичной системе счисления.

Формат входных данных

- Строка 1: Двоичная запись числа N (не более 1000 цифр).

Формат выходных данных

- Строка 1: Двоичная запись числа $17 \cdot N$.

Пример

<code>times17.in</code>	<code>times17.out</code>
10110111	110000100111

Замечание

Число 10110111 в двоичной системе равно числу 183 в десятичной. $183 \cdot 17 = 3111$, в двоичной записи — 110000100111.

Задача E. Connect the Cows

Имя входного файла: `connect.in`
Имя выходного файла: `connect.out`
Ограничение по времени: 2 секунды
Ограничение по памяти: 256 мегабайт

Каждый день Фермер Джон обходит свою ферму, чтобы проведать N ($1 \leq N \leq 10$) своих коров.

Коровы находятся в точках на координатной плоскости, а сам ФД начинает в точке $(0, 0)$. Чтобы сделать маршрут более интересным, ФД ходит только параллельно осям координат (на север, юг, восток и запад). Кроме того, он может менять направление своего движения только в тех точках, где находятся коровы (а может и не менять, если захочет). Изменяя направление, можно повернуть либо на 90 , либо на 180 градусов. После посещения всех коров ФД должен вернуться обратно в точку начала координат.

Найдите количество различных маршрутов, которыми ФД может посетить всех своих коров, при условии, что он изменит направление ровно по одному разу около каждой коровы. При этом, не изменяя направление движения, он может проходить мимо коровы любое количество раз. Один и тот же геометрический путь, пройденный в прямом и обратном направлениях, считается двумя различными маршрутами.

Формат входных данных

- Строка 1: Целое число N .
- Строки $2 \dots N + 1$: Строка $i + 1$ содержит x и y координаты i -й точки, разделенные пробелом (все числа от -1000 до 1000).

Формат выходных данных

- Строка 1: Количество различных маршрутов ФД (0, если их нет).

Пример

<code>connect.in</code>	<code>connect.out</code>
4 0 1 2 1 2 0 2 -5	2

Замечание

4 коровы находятся в точках $(0, 1)$, $(2, 1)$, $(2, 0)$, $(2, -5)$.

Два различных маршрута: ФД может посетить коров в порядке $1 - 2 - 4 - 3$ или $3 - 4 - 2 - 1$ до возвращения в точку $(0, 0)$.

Задача F. Wrong Directions

Имя входного файла: `wrongdir.in`
Имя выходного файла: `wrongdir.out`
Ограничение по времени: 2 секунды
Ограничение по памяти: 256 мегабайт

Фермер Джон купил программируемый трактор. Задавая ему программу, ФД вводит строку длиной N ($1 \leq N \leq 100\,000$), состоящую из символов «F», «L» и «R». Символ «F» перемещает трактор на один шаг вперед, а символы «L» и «R» поворачивают его на 90 градусов влево или вправо, соответственно. Изначально трактор стоит в точке $(0, 0)$ и смотрит на север.

Вводя программу, ФД опечатался ровно в одном символе. Но в каком именно неизвестно. Зная исходную программу без ошибок, определите в скольких различных точках на плоскости может оказаться трактор из-за ошибки ФД (направление, в котором трактор смотрит после выполнения программы роли не играет).

Формат входных данных

- Строка 1: Исходная программа ФД, не содержащая ошибок.

Формат выходных данных

- Строка 1: Количество различных позиций, в которых может оказаться трактор из-за ошибки в каком-то одном символе.

Пример

<code>wrongdir.in</code>	<code>wrongdir.out</code>
FF	3

Замечание

ФД хотел, чтобы его трактор продвинулся на два шага вперед и оказался в точке $(0, 2)$.

В результате одной опечатки могло получиться 4 возможных последовательности команд: «FL», «FR», «LF», «RF». После их выполнения трактор оказывается в точках $(0, 1)$, $(0, 1)$, $(-1, 0)$, $(1, 0)$ соответственно. Итого 3 различных точки.

Задача G. Find the Cow!

Имя входного файла: cowfind.in
Имя выходного файла: cowfind.out
Ограничение по времени: 2 секунды
Ограничение по памяти: 256 мегабайт

Корова Бесси сбежала и прячется на холме, покрытом высокой травой. Фермер Джон, пытаясь остаться незамеченным, ползет в этой траве.

Трава перед ФД выглядит как строка, состоящая из N круглых скобок ($1 \leq N \leq 50\,000$), например:

)(((())())

ФД знает, что задние ноги коровы выглядят как две подряд идущие открывающиеся скобки «(», а передние — как две как две подряд идущие закрывающиеся «)». Тем самым возможная позиция Бесси может быть описано парой индексов $x < y$ таких, что «(» находятся на позиции x , а «)» находятся на позиции y .

Найдите количество различных позиций, в которых может находиться Бесси.

Формат входных данных

- Строка 1: строка длины N , состоящая из круглых скобок.

Формат выходных данных

- Строка 1: Количество возможных позиций Бесси (то есть количество различных пар (x, y) таких, что $x < y$ и «(» стоят на позиции x , а «)» — на позиции y).

Пример

cowfind.in	cowfind.out
)(((())())	4

Замечание

Всего имеется четыре варианта расположения Бесси:

1.)((()))()
2.)(((()))()
3.)(((())))
4.)(((())))

Задача Н. Туро

Имя входного файла: `typo.in`
Имя выходного файла: `typo.out`
Ограничение по времени: 2 секунды
Ограничение по памяти: 256 мегабайт

Бесси только что купила новый ноутбук. Но её копыта слишком велики по сравнению с небольшими кнопками клавиатуры, поэтому печатать у неё получается неважно. Например, она только что попыталась напечатать свой любимый вид строк — правильную скобочную последовательность. Однако по вышеупомянутым причинам она могла ошибиться и какой-то символ в ней напечатать неправильно («(» вместо «)» или наоборот). Помогите Бесси посчитать количество позиций в этой строке, таких, что замена в этом месте одной скобки на противоположную превращает строку в правильную скобочную последовательность.

Есть несколько способов определить, что такое правильная скобочная последовательность. Пожалуй, один из самых простых: в строке должно быть поровну открывающихся и закрывающихся скобок, и на каждом из префиксов этой строки число открывающихся скобок должно быть не меньше числа закрывающихся.

Например, все следующие строки являются правильными скобочными последовательностями:

- `()`
- `(())`
- `()(())`

Тогда как все эти — нет:

- `)()`
- `()()`
- `((()))`

Формат входных данных

- Строка 1: строка, состоящая из круглых скобок, длины N ($1 \leq N \leq 100\,000$).

Формат выходных данных

- Строка 1: количество позиций в этой строке (если они вообще есть), таких, что при замене в ней скобки на противоположную строка становится правильной скобочной последовательностью.

Пример

<code>typo.in</code>	<code>typo.out</code>
<code>()((()))</code>	4

Замечание

В строке из примера к превращению в правильную скобочную последовательность ведет замена скобки на противоположную в позициях 2, 5, 6 и 7 (при нумерации позиций с единицы).

Задача I. Horseshoes

Имя входного файла: `hshoe.in`
Имя выходного файла: `hshoe.out`
Ограничение по времени: 2 секунды
Ограничение по памяти: 256 мегабайт

Хотя Бесси и нравятся все правильные скобочные последовательности, *идеальными* она считает строки, представляющие собой нескольких подряд идущих открывающихся скобок, за которыми идет ровно столько же закрывающихся. Например:

`((((()))`

Зайдя однажды в конюшню, Бесси увидела лежащие на полу в виде квадрата $N \times N$ подковы, каждая из которых выглядит либо как «(», либо как «)». Бесси хочет, начав с левого верхнего угла квадрата, пройти некоторый путь так, чтоб строка из подков, которые она поднимет во время прохода, была *идеальной*. Помогите ей найти максимальную длину *идеальной* строки, которую она сможет собрать.

За один шаг Бесси может сдвинуться на клетку вверх, вниз, влево или вправо. При этом перейти можно только на клетку, ещё содержащую подкову, и после перехода Бесси эту подкову поднимает (тем самым не получится посетить ни какую клетку более одного раза). В начале пути Бесси поднимает подкову, лежащую в левом верхнем углу. Не обязательно посещать все клетки квадрата.

Формат входных данных

- Строка 1: Целое число N ($2 \leq N \leq 5$).
- Строки 2... $N + 1$: Каждая строка состоит из N круглых скобок. Вместе все эти строки описывают квадрат $N \times N$.

Формат выходных данных

- Строка 1: Максимальная длина *идеальной* строки, которую Бесси сможет собрать. Если Бесси не сможет собрать ни какую *идеальную* строку (например, если в левом верхнем углу стоит «)»), выведите 0.

Пример

<code>hshoe.in</code>	<code>hshoe.out</code>
4 (()) ()((()))	8

Замечание

Последовательность шагов, которую нужно выполнить в пример, чтобы получить ответ 8 следующая:

1()
2)()
345(
876)