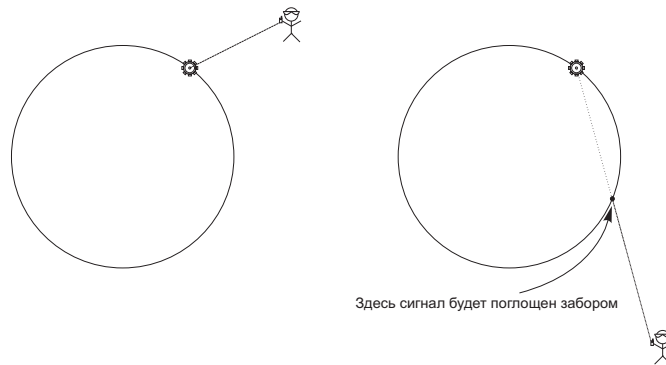


Задача А. Шпион

Имя входного файла: `spy.in`
Имя выходного файла: `spy.out`
Ограничение по времени: 2 секунды
Ограничение по памяти: 64 Мб

Секретный объект обнесен забором, выполненным в форме окружности радиуса r с центром в точке $(0, 0)$. Шпион, пытающийся разведать точное местоположение секретного объекта, для преодоления забора установил бомбу точно на заборе в точке (a_x, a_y) . Бомба активизируется дистанционным взрывателем, сигнал от которого распространяется по прямой, причём качество его приёма не зависит от пройденного расстояния. Из секретных источников известно, что забор полностью поглощает проходящий сквозь него сигнал (на сигнал, проходящий по касательной, забор не влияет). Шпион, находясь вне области, окруженной забором, в точке (b_x, b_y) , решил взорвать бомбу. Помогите ему узнать, дойдет ли сигнал до бомбы или будет поглощен.



Формат входных данных

В первой строке входного файла записано число r ($r > 0$). Во второй строке входного файла содержатся два числа (a_x, a_y) . В третьей строке входного файла содержатся два числа (b_x, b_y) . Все числа во входном файле целые и не превышают по модулю 10^3 .

Формат выходных данных

В единственной строке выходного файла выведите слово YES, если сигнал дойдет до бомбы, и NO в противном случае.

Примеры

| <code>spy.in</code> | <code>spy.out</code> |
|---------------------|----------------------|
| 1 1 0 2 0 | YES |
| 5 3 4 5 -1 | NO |
| 5 -3 -4 0 -10 | YES |

Задача В. Крестики-нолики

Имя входного файла: `xo.in`
Имя выходного файла: `xo.out`
Ограничение по времени: 2 секунды
Ограничение по памяти: 64 Мб

Крестики-нолики на бесконечном поле — это игра для двух человек. Поле представляет собой бесконечный клетчатый лист бумаги. Игроки ходят по очереди. За один ход игрок ставит в какую-нибудь свободную клетку игрового поля свой символ. Символ первого игрока — крестик, а символ второго игрока — нолик.

Если после очередного хода игрока на поле появляются **пять** стоящих в ряд (по горизонтали, вертикали или диагонали) символов этого игрока, то он объявляется победителем партии, и игра заканчивается.

Однажды одаренный школьник Антон обнаружил у себя в тетради некоторую картинку из крестиков и ноликов. Он не может вспомнить, что это такое: поле его поединка с другом Лёшей или просто произвольная картинка из крестов и нулей.

Напишите программу, которая по заданной картине из крестиков и ноликов определяет, могла ли такая позиция возникнуть на поле в результате игры — либо законченной, либо незаконченной. Предполагается, что игроки всегда соблюдают все правила.

Формат входных данных

Во входном файле находится картина из тетради Антона. Пустые клетки обозначаются символом `'.'` (точка). Символы игроков обозначаются символами `'X'` и `'O'` (заглавные латинские буквы `'икс'` и `'о'`).

Количество строчек во входном файле не превосходит 100. Количество символов в каждой строке также не превосходит 100. Пустых строчек в файле нет. Гарантируется, что во входном файле будет хотя бы один крестик или нолик.

Формат выходных данных

В выходной файл выведите слово `CORRECT`, если данная позиция могла возникнуть в результате игры в крестики-нолики на бесконечном поле (в конце игры или в середине). В противном случае выведите слово `INCORRECT`.

Примеры

| <code>xo.in</code> | <code>xo.out</code> |
|---|------------------------|
| <code>.X ..X ..OXO ...XO</code> | <code>CORRECT</code> |
| <code>.... .XO... .XO. .XO... .XO. .XO...</code> | <code>INCORRECT</code> |

Задача С. Муравьи на кубической Земле

Имя входного файла: ants.in
Имя выходного файла: ants.out
Ограничение по времени: 2 секунды
Ограничение по памяти: 64 Мб

В условиях данной задачи будем пользоваться предположением, что Земля имеет форму куба, и каждая грань — квадрат $m \times m$, расчерченный на клетки размером 1×1 .

В начальный момент времени n муравьев стоят на верхней грани этого куба. Каждый муравей направлен в одну из четырех сторон — на север, на юг, на запад или на восток.

В определенный момент муравьи начинают двигаться по прямой, каждый в своем направлении. Когда муравей доходит до ребра куба, он переползает через него и продолжает движение по следующей грани. При этом он все время движется перпендикулярно тому ребру, которое переполз.

Такое движение продолжается бесконечно долго. Выясните, сколько есть клеток на кубе, на которых ни разу во время этого процесса не побывает ни один муравей.

Формат входных данных

В первой строчке входного файла содержатся два натуральных числа — n и m — количество муравьев на земле и длина стороны планеты ($1 \leq n \leq 100000$; $1 \leq m \leq 15000$).

В каждой из следующих n строчек находится описание начального положения очередного муравья. Сначала идут два натуральных числа x и y — координаты муравья на верхней грани, а затем символ, задающий направление муравья — 'N', 'S', 'W' или 'E'. Числа и символ разделяются ровно одним пробелом.

Оси координат и направления сторон света приведены на рисунке. Все координаты лежат в интервале от 1 до m включительно.

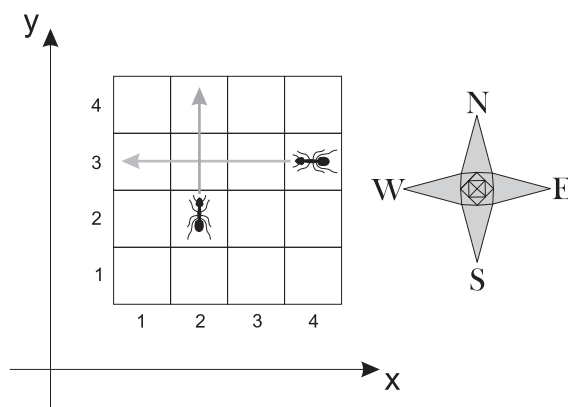
Несколько муравьев как в начальный момент времени, так и в любой другой, могут оказаться на одной клетке. Это никак не влияет на траектории их движения.

Формат выходных данных

В выходной файл выведите одно число — количество клеток, никогда не посещаемых муравьями.

Пример

| ants.in | ants.out |
|-----------------------|----------|
| 2 4 2 2 N 4 3 W | 66 |



Верхняя грань куба, вид сверху

Задача D. Наилучшая расческа

Имя входного файла: `bestcomb.in`
Имя выходного файла: `bestcomb.out`
Ограничение по времени: 5 секунд
Ограничение по памяти: 64 Мб

После того, как задачи нахождения в числовой матрице наибольших квадратов и прямоугольников из нулей стали стандартными, появилась новая задача, обобщающая все предыдущие. Дана матрица из нулей и единиц. Необходимо найти наилучшую “расческу”, которую можно построить на нулевых ячейках этой матрицы (наилучшей называется расческа, которая занимает наибольшее количество ячеек данной матрицы). Звеном расчески называется непустая прямоугольная область из нулей в матрице, то есть некоторая подматрица ненулевой площади, состоящая только из нулей. k -расческой называется фигура, получаемая соединением по вертикали k ($0 \leq k \leq m$) звеньев. У всех звеньев должна совпадать x -координата левой стороны, соседние звенья должны иметь разную ширину, а все звенья вместе должны образовывать цельную фигуру, то есть должны быть связаны. При этом 0-расческа существует всегда и имеет размер 0.

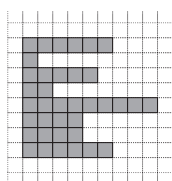


рис. 1

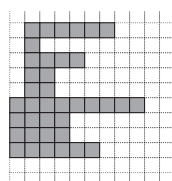


рис. 2

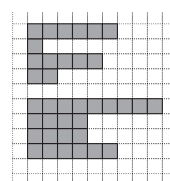


рис. 3

Фигура на рисунке 1 является правильной 7-расческой. Хотя фигура занимает 8 строк, она является 7-расческой, так как на второй и третьей снизу строках находится одно и то же звено (соседние звенья обязательно должны быть разной ширины, но звено может быть высотой более 1 строки). Фигура на рисунке 2 не является правильной расческой, так как левая x -координата звеньев не совпадает. Фигура на рисунке 3 также не является правильной расческой, так как не является связной фигурой. Можно заметить, что наилучший прямоугольник это то же самое что наилучшая 1-расческа (расческа высоты 1 звено). Очевидно, что для каждой длины k в матрице можно либо найти наилучшую расческу, либо заявить, что ее нет (то есть наилучшая расческа имеет размер 0). Например, на рисунке 4 обозначена лучшая 3-расческа (знаком X обозначены ненулевые элементы в матрице).

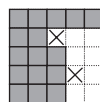


рис. 4

Ваша задача — найти в данной матрице наилучшую расческу среди всех k -расчесок матрицы (для всех k от 0 до m).

Формат входных данных

В первой строке входного файла даны два целых числа — высота ($1 \leq m \leq 2000$) и ширина ($1 \leq n \leq 2000$) матрицы соответственно. Далее, в m строках дано по n чисел через пробел — исходная матрица.

Формат выходных данных

В выходной файл необходимо вывести размер наибольшей расчески среди всех k -расчесок, которую можно построить на нулевых ячейках входной матрицы.

Примеры

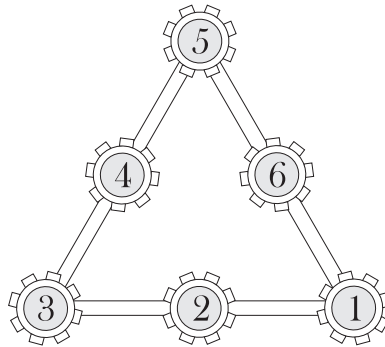
| bestcomb.in | bestcomb.out |
|--|--------------|
| 3 4 0 0 1 0 0 1 0 0 0 0 0 0 | 7 |
| 5 5 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 | 20 |

Задача Е. Крепость

Имя входного файла: `castle.in`
Имя выходного файла: `castle.out`
Ограничение по времени: 2 секунды
Ограничение по памяти: 64 Мб

В 2123 году на территории Ленинградской Области археологи обнаружили останки старинной крепости. К сожалению, некоторые фрагменты крепости не сохранились. Археологи точно знают, что крепость содержала шесть башен, три из которых являются вершинами треугольника, а остальные три — серединами сторон этого треугольника. Известно точное расположение только некоторых башен. Ваша задача определить расположение всех башен.

На рисунке приведен возможный вид крепости сверху:



Формат входных данных

Входной файл содержит шесть строк, каждая строка представляет собой описание башни. Если расположение башни известно, то строка содержит два целых числа, разделенных одним пробелом, в противном случае два знака вопроса ('?'), разделенных одним пробелом. Башни даны в порядке обхода, начиная с любой угловой башни.

Формат выходных данных

В выходной файл необходимо вывести `IMPOSSIBLE`, если однозначно восстановить расположение всех башен невозможно, в противном случае вывести в первой строке `POSSIBLE`, а в следующих шести строках расположение башен, в таком же порядке как во входном файле. Числа необходимо вывести по крайней мере с двумя знаками после точки.

Примеры

| <code>castle.in</code> | <code>castle.out</code> |
|------------------------|-------------------------|
| 0 0 | POSSIBLE |
| 0 1 | 0.00 0.00 |
| ? ? | 0.00 1.00 |
| ? ? | 0.00 2.00 |
| 2 2 | 1.00 2.00 |
| ? ? | 2.00 2.00 |
| | 1.00 1.00 |
| 0 1 | IMPOSSIBLE |
| ? ? | |
| ? ? | |
| ? ? | |
| -2 -1 | |
| -1 0 | |

Задача F. Цифры

Имя входного файла: `digits.in`
Имя выходного файла: `digits.out`
Ограничение по времени: 2 секунды
Ограничение по памяти: 64 Мб

Назовем *суммой цифр числовой последовательности* сумму цифр всех ее чисел. Например, для последовательности чисел 14, 22, 239 сумма цифр будет равна $(1 + 4) + (2 + 2) + (2 + 3 + 9) = 23$.

Ваша задача — при данном n найти сумму цифр следующей числовой последовательности:

$$1, 2, 3, \dots, 10^n - 1$$

Формат входных данных

На первой строке входного файла находится целое число n ($1 \leq n \leq 100000$).

Формат выходных данных

Выведите в выходной файл одно число — искомую сумму цифр числовой последовательности.

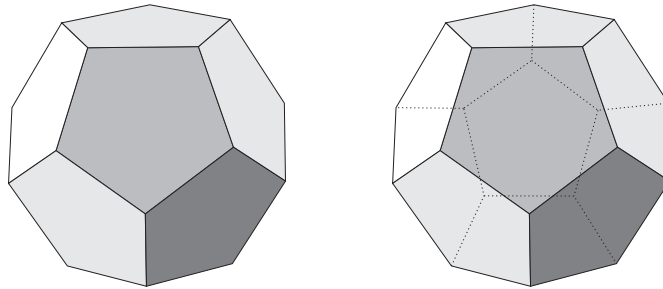
Пример

| | <code>digits.in</code> | <code>digits.out</code> |
|---|------------------------|-------------------------|
| 1 | 1 | 45 |

Задача G. Додекаэдр

Имя входного файла: `dodeca.in`
Имя выходного файла: `dodeca.out`
Ограничение по времени: 2 секунды
Ограничение по памяти: 64 Мб

Додекаэдром называется правильный многогранник, состоящий из 12 граней. Каждая его грань является правильным пятиугольником. Двое полицейских, находящихся на некоторых гранях додекаэдра (возможно, на одной и той же), гоняются за З. Хуссейном, мировым террористом номер один, который также расположен на одной из граней додекаэдра. Полицейские и З. Хуссейн ходят последовательно — сначала двигается один из полицейских (любой), затем Хуссейн. Каждый ход заключается в перемещении на соседнюю грань, а соседней называется грань, имеющая с данной гранью общее ребро. Остаться на месте в свой ход нельзя. Перемещения повторяются до тех пор, пока Хуссейн не будет пойман. Если злодей в свой ход встает на грань, где находится полицейский, то он будет пойман в следующий же ход полицейского. Если полицейский в свой ход встает на грань, где находится злодей — то злодей сразу считается пойманным.



Формат входных данных

Дано расстояние n между полицейскими на додекаэдре. Расстоянием называется минимальное количество ходов, которое потребуется одному из них, чтобы оказаться на одной грани с другим.

Формат выходных данных

Выведите, какое максимальное количество ходов потребуется, чтобы гарантированно поймать З. Хуссейна, где бы на додекаэдре он ни находился. Координаты Хуссейна всегда известны полицейским, и наоборот. Полицейские также всегда знают координаты друг друга.

Пример

| | <code>dodeca.in</code> | <code>dodeca.out</code> |
|---|------------------------|-------------------------|
| 1 | | 5 |

Задача Н. Эндшпиль

Имя входного файла: `endspiel.in`
Имя выходного файла: `endspiel.out`
Ограничение по времени: 2 секунды
Ограничение по памяти: 64 Мб

Партия в шахматы подходила к концу, и на поле оставалось всего две фигуры: белый и черный короли. Однако, вопреки шахматным правилам, игроки решили доиграть эту партию до самого конца, руководствуясь девизом “остаться должен только один!”. Но короли, в целях государственной безопасности, были заменены “охранниками”. Охранник — это фигура, которая в каждый свой ход перемещается ровно на одну клетку по горизонтали или вертикали, и бьет эти же клетки. Вам необходимо сказать, чем закончится партия при оптимальной игре обоих соперников.

Формат входных данных

Во входном файле дано через пробел расположение охранников (сначала координаты белого, затем черного), а затем информация о том, кто будет ходить первым (символ ‘B’ — если первым будет ходить черный охранник, и символ ‘W’ в противном случае). Расположение фигур дано в стандартной шахматной записи — сначала указывается символ столбца шахматного поля (строчная латинская буква от `a` до `h`), затем указывается номер строки шахматного поля (цифра от 1 до 8).

Формат выходных данных

Необходимо вывести, чем закончится партия при оптимальной игре обоих соперников: `White wins` — если выиграют белые, `Black wins` — если победят черные, и `Draw` — если будет ничья. Партия заканчивается ничьей, если ни один из соперников не может победить. Партия закончится победой одной из сторон, когда охранник “съедает” охранника соперника. Изначально охранники могут занимать абсолютно любую позицию на поле, за исключением того, что не могут находиться на одной клетке.

Пример

| <code>endspiel.in</code> | <code>endspiel.out</code> |
|--------------------------|---------------------------|
| <code>g6 g7 B</code> | <code>Black wins</code> |

Задача I. Маленький шахматный Ним

Имя входного файла: `smallnim.in`
Имя выходного файла: `smallnim.out`
Ограничение по времени: 2 секунды
Ограничение по памяти: 64 Мб

В шахматной стране в последнее время стала очень популярна игра Ним. Правила игры просты. Перед началом игры на стол выкладываются несколько кучек камней. Два игрока ходят по очереди и за каждый ход берут из одной любой кучки произвольное число камней. Игрок, который берет последний камень из последней оставшейся кучки — проигрывает.

Черный и белый короли тоже решили сыграть в Ним, но игра оказалась слишком сложной, поэтому они решили немного изменить правила: камни теперь можно брать не из любой кучки, а только из такой, в которой содержится минимальное число камней.

После нескольких партий обнаружилось, что черный король очень хорошо освоил эту игру и каждый раз ходит наилучшим образом, то есть если у черного короля есть хотя бы один ход, ведущий к победе, то он его и делает. Таким образом, белый король стал подозревать, что исход каждой партии можно определить по начальной позиции в игре.

Теперь он хочет, чтобы вы, как главный мудрец шахматной страны, помогли ему определить по количеству камней в каждой кучке, может ли он выиграть, и если может, то сколько камней ему нужно взять из минимальной кучки для того, чтобы сохранить возможность победы.

Формат входных данных

В первой строке входного файла записано целое число n — количество кучек ($1 \leq n \leq 100$). Во второй строке входного файла записано n целых чисел b_i ($1 \leq b_i \leq 1000$) — количество камней в i -ой кучке.

Формат выходных данных

Если белый король может выиграть при наилучшей игре черного короля, то в первую строку выходного файла выведите слово YES, а во вторую строку — целое число s , которые определяет, сколько камней необходимо взять белому королю из минимальной кучки на первом ходе.

Если же белый король не может выиграть, то в первую строку файла выведите NO.

Примеры

| <code>smallnim.in</code> | <code>smallnim.out</code> |
|--------------------------|---------------------------|
| 2 | YES |
| 2 3 | 1 |
| 1 | NO |
| 1 | |

Задача J. Вычитание

Имя входного файла: `subtract.in`
Имя выходного файла: `subtract.out`
Ограничение по времени: 2 секунды
Ограничение по памяти: 64 Мб

Сотрудник одного известного банка неожиданно обнаружил, что в некоторых программах, обслуживающих работу банка, процедура вычитания одного числа из другого иногда выдает ошибочный результат, что, естественно, угрожает безопасности банка. Например, один клиент заявил, что, когда на его счету было \$296, и он снял с него \$125, его счет почему-то оказался пустым, хотя на нем должен был остаться \$171. Вам нужно срочно написать простую, но очень ответственную программу, которая правильно реализует вычитание одного числа из другого, пока электронные воры не воспользовались дыркой в системе безопасности.

Формат входных данных

Во входном файле даны два целых числа, каждое из которых лежит в пределах от -2^{63} до $2^{63} - 1$.

Формат выходных данных

Необходимо вывести результат вычитания второго числа из первого.

Примеры

| <code>subtract.in</code> | <code>subtract.out</code> |
|--------------------------|---------------------------|
| 296 125 | 171 |

Задача К. Сколько же переменных?

Имя входного файла: vars.in
Имя выходного файла: vars.out
Ограничение по времени: 2 секунды
Ограничение по памяти: 64 Мб

Маленький мальчик Петя очень любит программировать на языке Паскаль. К сожалению, в последнее время он стал замечать, что использует слишком много переменных. Так много, что сам не может толком сосчитать, сколько же в каждой программе их используется.

Поэтому он просит вас написать программу, которая поможет ему в этом нелегком деле.

Так как Петя еще не очень хорошо знает Паскаль, то использует он только четыре стандартных типа: `boolean`, `char`, `double`, `integer`.

Формат входных данных

Во входном файле содержится несколько строк с объявлениями переменных.

Каждая строка входного файла является объявлением переменных одного типа.

Все объявления являются корректными с точки зрения синтаксиса языка Паскаль, названия переменных не повторяются, при объявлении используются только стандартные типы: `boolean`, `char`, `double`, `integer`.

Формально каждое объявление имеет следующий вид:

```
<идентификатор> {',' <идентификатор>} ':' <тип> ';' ;
```

где <тип> — это одна из строк `boolean`, `char`, `double`, `integer`; <идентификатор> — строка, состоящая из букв, цифр или символов '_' и начинающаяся с буквы или символа '_'.

Строки <тип> и <идентификатор>, а также символы ':' и ';' и ',' являются неделимыми элементами, между которыми (а также до и после них) может находиться произвольное число пробелов.

Гарантируется, что во входном файле количество строк — не более 100, в каждой строке не более 20 переменных, и название каждой переменной не длинее 1000 символов.

Формат выходных данных

В выходной файл необходимо вывести четыре строки, в следующем формате:

```
boolean: <количество объявленных переменных типа boolean>  
char: <количество объявленных переменных типа char>  
double: <количество объявленных переменных типа double>  
integer: <количество объявленных переменных типа integer>
```

Обратите внимание, что между двоеточием и числом переменных должен быть выведен один пробел.

Пример

| vars.in | vars.out |
|---|--|
| i, j, k : integer ; flag: boolean; length: integer; c1, c2: char; c3: char; | boolean: 1 char: 3 double: 0 integer: 4 |