Задача A. Муравьи на кубической Земле

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

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 доползает 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 входного файла

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

В каждой из следующий 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.

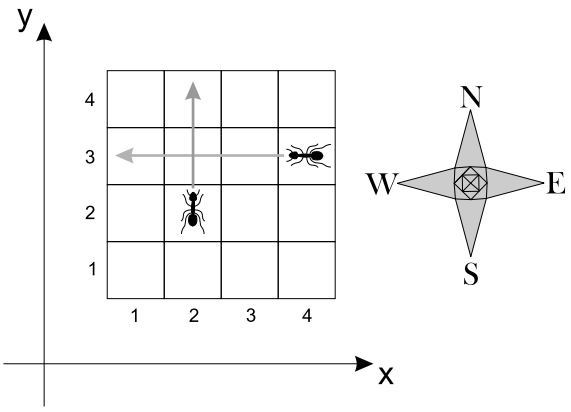
Несколько муравьев как в начальный момент времени, так и в любой другой, могут оказаться на одной клетке. Это никак не влияет на траектории их двежения.

Формат выходного файла

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 B. Наилучшая расчёска

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

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

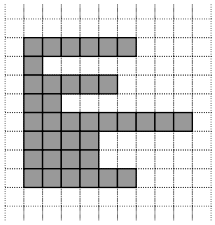


рис. 1

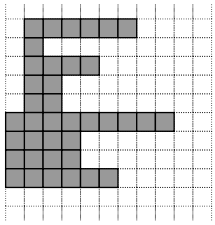


рис. 2

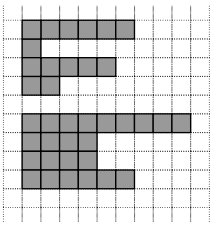


рис. 3

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

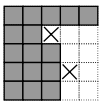


рис. 4

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

Формат входного файла

На первой строке входного файла даны два целых числа — высота ($1 \leq m \leq 500$) и ширина ($1 \leq n \leq 500$) матрицы соответственно. Далее, на m строках дано по n чисел через пробел — исходная матрица.

Формат выходного файла

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

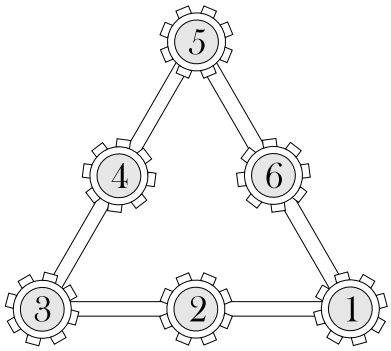
Примеры

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 C. Крепость

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

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.



Формат входного файла

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 выходного файла

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 с двумя знаками после точки.

Примеры

castle.in	castle.out
0 0 0 1 ? ? ? ? 2 2 ? ?	POSSIBLE 0.00 0.00 0.00 1.00 0.00 2.00 1.00 2.00 2.00 2.00 1.00 1.00
0 1 ? ? ? ? ? ? -2 -1 -1 0	IMPOSSIBLE

Задача D. Цифры

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

Найдите сумму цифр, которыми записаны все числа данной последовательности

$$1,\ 2,\ \ldots,\ 10^n - 1$$

.

Формат входного файла

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

Формат выходного файла

Выведите в выходной файл одно число — сумму цифр, которыми записаны все числа данной последовательности.

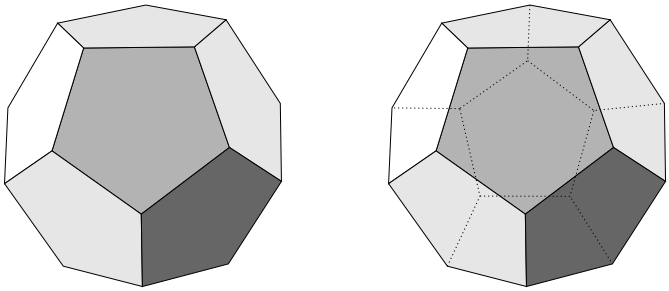
Пример

digits.in	digits.out
1	45

Задача E. Додекаэдр

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

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 расположился на одной из граней додекаэдра. Полицейские и З. Хуссейн двигаются по очереди — 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. Оставаться на месте в свой ход нельзя. Перемещения повторяются до тех пор, пока Хуссейн не будет пойман. Если злодей в свой ход встаёт на грань, где находится полицейский, то он будет пойман в следующий же ход полицейского. Если полицейский в свой ход встаёт на грань, где находится злодей — то злодей сразу считается пойманным.



Формат входного файла

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 выходного файла

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

dodeca.in	dodeca.out
1	5

Задача F. Эндшпиль

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

Партия в шахматы подходила к концу, и на поле осталось всего две фигуры: белый и чёрный 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 бьёт эти же клетки. Вам необходимо сказать, чем закончится партия при оптимальной игре обоих соперников.

Формат входного файла

Во входном файле дано через пробел расположение охранников (сначала координаты белого, затем чёрного), а затем информация о том, кто будет ходить первым (символ `'B'` — если первым будет ходить чёрный 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 выходного файла

Необходимо вывести, чем закончится партия при оптимальной игре обоих соперников: `White wins` — если выиграют белые, `Black wins` — если победят чёрные, 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

endspiel.in	endspiel.out
g6 g7 B	Black wins

Задача G. Маленький шахматный Ним

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

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 входного файла

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

Формат выходного файла

Если белый король может выиграть при наилучшей игре черного короля, то в первую строку выходного файла выведите слово `YES`, а во вторую строчу 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 строчу файла выведите `NO`.

Примеры

smallnim.in	smallnim.out
2 2 3	YES 1
1 1	NO

Задача H. Вычитание

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

Сотрудник одного известного банка неожиданно обнаружил, что в некоторых программах, обслуживающих работу банка, процедура вычитания одного числа из другого иногда выдаёт 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 счёт почему-то оказался пустым, хотя на нём 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 входного файла

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

Формат выходного файла

Необходимо вывести результат вычитания второго числа из первого.

Примеры

subtract.in	subtract.out
296 125	171

Задача I. Сколько же переменных?

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

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 входного файла

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) 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 выходного файла

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

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