Задача A. Интерференция

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Великий физик современности Берлштейн занимается изучением световых явлений. И прямо сейчас он готовит экспериментальную установку для изучения интерференции (явления наложения двух и более световых волн).

Установка должна состоять из N прожекторов, расположенных в одну линию. Каждый прожектор излучает свет определённой длины волны, от которой зависит цвет света. Когда прожектор включается, на экране, который расположен вдоль линии, параллельной линии прожекторов, появляется полоса определённого цвета. Зона экрана, освещаемая i-м прожектором, определяется целочисленными координатами L_i и R_i.

Зоны экрана, освещаемые различными прожекторами, могут перекрываться. Вследствие этого световые волны, излучаемые прожекторами, могут интерферировать между собой, образуя на экране полосу нового цвета. Эксперимент проводится в специальной комнате, в которой интерференция происходит по следующим правилам:

1. Интерферировать могут только исходные световые волны прожекторов;
2. Интерферировать могут только определённые совместимые наборы волн. Если в некоторый момент времени набор *всех* световых волн, которыми освещена определённая точка экрана, не является совместимым, интерференция в данной точке не наблюдается.

В ходе своего эксперимента Берлштейн включает прожекторы в определённые моменты времени T_i. В результате к концу эксперимента на экране наблюдаются полосы различных цветов. Для корректного выполнения расчётов Берлштейну необходимо знать, наблюдается ли интерференционная картина в точках экрана X_j в моменты времени T_j. А если не наблюдается, то по какой причине (интерференции в принципе не может быть или же точку освещают несовместимые световые волны).

Формат входных данных

Первая строка содержит целое число N ($1 \leqslant N \leqslant 10^5$) — количество прожекторов.

Следующие N строк описывают прожекторы. Каждая из них содержит целые числа L_i, R_i и T_i ($0 \leqslant L_i \leqslant R_i \leqslant 10^5$, $0 \leqslant T_i \leqslant 10^5$), а также строчную латинскую букву C_i — соответственно границы зоны экрана, освещаемой i-м прожектором, момент времени, в который включается i-й прожектор, и цвет света, излучаемого i-м прожектором.

Следующая строка содержит целое число M ($0 \leqslant M \leqslant 1000$) — количество моментов времени, интересующих Берлштейна.

Следующие M строк описывают моменты времени и точки, в которых Берлштейн выясняет наличие интерференции. Каждая из них содержит целые числа T_j и X_j ($0 \leqslant T_j \leqslant 10^5$, $0 \leqslant X_j \leqslant 10^5$) — соответственно момент времени и координату точки экрана, в которой проверяется существование интерференционной картины.

Следующие строки (не более 1000) содержат описание совместимых наборов цветов. Каждая из них содержит последовательность строчных латинских букв (не менее 2 и не более 100), обозначающих совместимые цвета, и строчную латинскую букву, обозначающую результирующий цвет. Гарантируется, что каждый набор цветов встречается во входных данных не более одного раза.

Формат выходных данных

Для каждого момента времени T_j и точки экрана X_j, расположенных в порядке возрастания момента времени, в отдельную строку выходного файла поместите «YES», если интерференционная картина в данный момент времени в данной точке наблюдается, «NO», если точку освещают несовместимые световые волны, и «NO INTERFERENCE», если интерференции вообще быть не может.

Примеры

input.txt	output.txt
3 0 2 5 y 1 4 0 b 1 7 8 r 6 0 0 1 1 5 2 7 3 8 1 10 3 yb g	NO INTERFERENCE NO INTERFERENCE YES NO INTERFERENCE NO NO
4 1 4 1 a 2 4 2 b 3 4 3 b 4 4 4 b 4 1 1 2 2 3 3 4 4 ab q abbb y	NO INTERFERENCE YES NO YES

Задача B. Космическая навигация

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

В плоском королевстве заканчивается постройка новейшего космического корабля «Энтерпрайз» для полётов в плоском космосе на суб- и сверхсветовых скоростях (при этом корабль подвергается действию ворп-фактора и гравитационных полей плоских черных дыр, что иногда выражается в изменении привычных нам математических правил). Сейчас ведутся активные работы по завершению написания программных модулей для сверхразума корабля.

По проекту навигатор вводит описание траектории полёта в сверхразум корабля на специальном языке программирования 2NAV. Затем сверхразум должен рассчитать все параметры полёта и по специальной команде выполнить его.

В полёте корабль посылает через равные интервалы времени сигнал, содержащий координаты его текущего местоположения. Также сигнал всегда отправляется из точек отправления и прибытия. Выполнив кусочно-линейную интерполяцию по этим координатам, можно оценить траекторию корабля и пройденный им путь. Мы просим вас помочь в разработке навигационного модуля и написать модуль для вычисления длины пройденного пути.

Путь задаётся в параметрическом виде $x = x(t)$ и $y = y(t)$ на языке 2NAV. Этот язык программирования разработан на основе LISP специально для навигации в плоском космосе. Язык оперирует только вещественными числами двойной точности, записанными в формате с фиксированной точкой без знака, предопределёнными константами **PI** (число π) и **E** (число e), предопределённой переменной **t**. Выражением на этом языке может быть число, предопределённая переменная или константа, либо конструкция вида ($op\ arg_1\ arg_2\ ...\ arg_n$), где op — предопределённый в языке оператор или функция, а arg_i — аргумент, который может быть любым допустимым в языке выражением. Поддерживаемые операторы и функции приведены в таблице ниже.

op	Количество аргументов	Результат
+	переменное	Сумма аргументов
−	переменное	Разность первого аргумента и суммы всех остальных аргументов (сначала вычисляется сумма)
*	переменное	Произведение аргументов
/	переменное	Частное от деления первого аргумента на произведение всех остальных аргументов (сначала вычисляется произведение), либо 0, если делитель равен 0
sqrt	1	Квадратный корень аргумента, либо 0, если аргумент отрицателен
sin	1	Синус аргумента, заданного в радианах
cos	1	Косинус аргумента, заданного в радианах
exp	1	Число e в степени аргумента
ln	1	Натуральный логарифм аргумента, либо 0, если аргумент отрицателен или равен 0

Если у оператора с переменным числом аргументов задан только один аргумент, то остальные аргументы полагаются нейтральными относительно соответствующей операции.

Если абсолютная величина результата любого математического действия (включая промежуточные вычисления) оказывается больше 10^6, то результат округляется до -10^6 или 10^6 в зависимости от того, был результат отрицательным или положительным. Если абсолютная величина результата любого математического действия оказывается меньше 10^{-6}, то результат округляется до 0.

Формат входных данных

Первая строка содержит целые числа T и N ($1 \leqslant T \leqslant 1000$, $1 \leqslant N \leqslant T$) — соответственно время, в течение которого корабль находился в пути, и интервал отправки сигналов с координатами.

Следующие строки содержат два корректных выражения на языке 2NAV для $x(t)$ и $y(t)$ соответственно. Числа, константы, переменные, операторы и функции в выражениях могут быть разделены

произвольным количеством символов пробела, табуляции и перевода строки. Количество математических действий в каждом из выражений, включая неявные действия в операторах с переменным числом аргументов, не превосходит 10^4.

Формат выходных данных

Выведите одно вещественное число — длину пути, пройденного кораблём за время T, определённое на основе кусочно-линейной интерполяции его траектории по переданным координатам. Точность ответа должна составлять не менее 2 знаков после десятичной точки.

Примеры

input.txt	output.txt
1 1 t t	1.41
10 1 (cos (/ (* t PI) 10.0)) (sin (/ (* t PI) 10.0))	3.13

Задача C. Сейф

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Сокровища короны скрыты в сейфе, который находится под наблюдением охраны. Однако охранник время от времени отходит на перерыв, и у грабителя Буратино появляется шанс взломать сейф.

Полистав «Справочник юного взломщика», Буратино узнал, что код к сейфам подобной конструкции обязательно удовлетворяет грамматике, определяемой следующей формой Бэкуса-Наура:

$$<код> ::= \mathrm{AX} \mid \mathrm{BY} \mid \mathrm{A}<код>\mathrm{X} \mid \mathrm{B}<код>\mathrm{Y} \mid <код><код>$$

Другими словами, если c — произвольный корректный код, то AcX и BcY — также корректный код; кроме того, если c_1 и c_2 — произвольные корректные коды, то c_1c_2 — также корректный код.

Также Буратино выяснил, что код данного конкретного сейфа должен состоять ровно из N символов. В момент, когда в сейф положили сокровища и закрыли его дверцу, значение кода было следующим:

$$\overbrace{\mathrm{AAA...A}}^{N/2}\overbrace{\mathrm{XXX...X}}^{N/2}$$

Затем каждую секунду код изменялся на лексикографически минимальный корректный код, больший текущего, а в том случае, если текущий код являлся максимально возможным, — на показанный выше стартовый код.

Буратино тщательно замерил время, прошедшее с момента закрытия дверцы сейфа. Помогите ему подобрать код к сейфу.

Формат входных данных

Ввод содержит целое чётное число N и целое число T ($2 \leqslant N \leqslant 30$, $0 \leqslant T \leqslant 10^{18}$) — соответственно длину кода и количество секунд, прошедших с момента закрытия сейфа.

Формат выходных данных

Выведите строку, состоящую из N символов 'A', 'X', 'B', 'Y', — код к сейфу в момент времени T.

Примеры

input.txt	output.txt
6 0	AAAXXX
6 1	AABYXX
6 2	AAXAXX
6 3	AAXBYX
6 4	AAXXAX
6 10	ABYXAX

Задача D. Искусственный интеллект

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Студенческий совет Берляндского государственного университета собирается устроить Чемпионат по Пятнашкам. Эту идею активно поддержало руководство университета, ведь ректор — большой фанат Пятнашек (он даже хочет все головоломки для Чемпионата подготовить самостоятельно). А ещё он — большой фанат программирования, со всеми вытекающими отсюда последствиями.

Ректор хочет, чтобы организаторы устроили соревнования не просто между участниками, а между программами, написанными участниками. Победителем станет участник, программа которого сможет решить наибольшее количество головоломок, предложенных ректором. А вот приз, который получит победитель, должен зависеть от того, насколько его результат отличается от результата эталонного искусственного интеллекта, написанного организаторами. Если количество головоломок, решённых победителем, больше или равно количеству головоломок, решённых эталонным ИИ, то победитель получает 1 миллион рублей. Если же нет, то победитель получает 100 тысяч рублей.

У ректора совершенно чёткие представления о скорости, с которой должна работать программа организаторов, — решение одной головоломки не должно занимать больше 2 секунд. Ректор просит вас помочь организаторам и написать такой эталонный искусственный интеллект, который пройдёт все имеющиеся головоломки и уложится в указанный лимит времени.

Формат входных данных

Первая строка содержит целое число N ($2 \leqslant N \leqslant 4$) — размер игрового поля Пятнашек.

Следующие N строк описывают игровое поле. Каждая из них содержит N целых чисел A_{ij} ($0 \leqslant A_{ij} \leqslant N^2 - 1$). Все числа A_{ij} различны, число 0 означает пустую ячейку игрового поля.

Формат выходных данных

Выведите строку, состоящую из заглавных латинских букв 'L', 'R', 'U', 'D', — решение головоломки. Каждый символ строки должен означать направление изменения положения пустой ячейки на игровом поле: 'L' — влево, 'R' — вправо, 'U' — вверх, 'D' — вниз. Гарантируется, что решение головоломки существует, и количество ходов в нем не превышает 100. Если подходящих решений несколько, выведите любое из них.

Примеры

input.txt	output.txt
2 1 2 0 3	RULRD
3 0 1 3 5 2 6 4 7 8	RDLDRR

Задача E. Видеоклип

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Команда Макса затащила важный матч в Доту! Макс рад вдвойне, потому что он не только успешно играл, но и попутно записывал процесс игры на видео. Теперь Макс хочет увековечить свою былинную победу, смонтировав видеоклип из самых важных моментов матча.

Запись игры Макс разделил на несколько фрагментов. Теперь он должен выбрать некоторые из этих фрагментов, чтобы, склеив их, получить видеоклип. Каждый фрагмент Макс может использовать не более одного раза; кроме того, Максу важно сохранить порядок показанных событий в первозданном виде, поэтому выбранные фрагменты в клипе должны быть упорядочены так же, как и в исходной записи. Другими словами, в качестве материала для клипа Макс должен выбрать некоторую подпоследовательность (не обязательно непрерывную) фрагментов исходной записи.

Разумеется, неотъемлемой частью хорошего видеоклипа должна быть драматичная музыка. При воспроизведении клипа видеоряд и музыкальное сопровождение начинаются одновременно в момент времени 0. Максу особенно нравится, когда конец некоторого фрагмента клипа сопровождается эффектным моментом в музыкальной композиции. Макс уже выбрал подходящую фоновую музыку и составил список имеющихся в ней эффектных моментов.

Теперь Макс хочет смонтировать свой видеоклип так, чтобы как можно больше раз конец фрагмента совпадал с эффектным моментом фоновой музыки. Итоговая продолжительность клипа при этом не имеет значения.

Помогите Максу узнать, каким образом лучше всего смонтировать видеоклип.

Формат входных данных

Первая строка содержит целое число N ($1 \leqslant N \leqslant 3000$) — количество фрагментов исходной видеозаписи.

Вторая строка содержит N целых чисел L_i ($1 \leqslant L_i \leqslant 3000$) — продолжительность каждого из фрагментов в секундах.

Третья строка содержит целое число M ($1 \leqslant M \leqslant 3000$) — количество эффектных моментов в музыкальной композиции.

Четвёртая строка содержит M различных целых чисел T_j ($0 < T_j \leqslant 3000$), упорядоченных по возрастанию, — время в секундах, когда наступает каждый из эффектных моментов.

Формат выходных данных

Выведите одно целое число — максимально возможное количество моментов, когда конец фрагмента клипа совпадает с эффектным моментом в музыкальной композиции.

Примеры

input.txt	output.txt
8 1 2 5 5 1 6 4 2 6 2 4 6 7 9 10	3
8 2 6 1 8 2 5 4 1 7 2 5 6 7 8 9 10	4

Задача F. Кинотеатр для мизантропов

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Наконец-то в городе открылось заведение, предназначенное специально для тех, кто любит кино и не любит компанию. Добро пожаловать в кинотеатр для мизантропов!

Чтобы ничьи головы не заслоняли зрителям экран, кресла в кинотеатре расположены в один ряд и пронумерованы слева направо числами от 1 до N. Перед сеансом посетители заходят в зрительный зал по очереди, и каждый из них может выбрать любое свободное место по своему вкусу.

Разумеется, публика кинотеатра полностью оправдывает его название, поэтому каждый зритель, начиная со второго, будет выбирать себе место таким образом, чтобы оказаться как можно дальше от ближайшего соседа. Если подходящих мест несколько, зритель выберет кресло с наименьшим номером.

Вы знаете, сколько зрителей пришло на сеанс, и какое место выбрал первый из них. Сможете ли вы определить, какое место в итоге займёт каждый зритель?

Формат входных данных

Ввод содержит целые числа N, M и S_1 ($1 \leqslant N \leqslant 10^6$, $1 \leqslant M, S_1 \leqslant N$) — соответственно количество мест в кинотеатре, количество зрителей и номер места, которое занял первый зритель.

Формат выходных данных

Вычислите сумму $\sum_{i=1}^{M}(i \cdot S_i)$ (где S_i — номер места, занятого i-м зрителем) и выведите остаток от деления этой суммы на 1000000007.

Примеры

input.txt	output.txt
10 7 3	125
40 20 20	3421

Замечание

В первом примере зрители занимают следующие места: #3, #10, #6, #1, #8, #2, #4.

Задача G. Шарики и хомяки

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Для награждения участников Интеллектуальной олимпиады ПФО организаторы соревнования по программированию купили множество воздушных шариков.

Весьма удивившись их количеству, волонтёры зачем-то озадачились вопросом: сколько шариков понадобится, чтобы поднять в воздух, скажем, хомяка? Разумеется, Гугл знал ответ на этот вопрос.

Но теперь волонтёры хотят узнать другую вещь: сколько хомяков можно было бы поднять в воздух, используя ВСЕ купленные шарики? На этот вопрос Гугл ответа не знал. Поэтому удовлетворить волонтёрское любопытство придётся вам.

Формат входных данных

Ввод содержит целые числа N и M ($1 \leqslant N, M \leqslant 1000$) — соответственно количество шариков, купленных организаторами, и количество шариков, способных поднять в воздух одного хомяка.

Формат выходных данных

Выведите одно целое число — количество хомяков, которых можно поднять в воздух, воспользовавшись всеми купленными шариками.

Примеры

input.txt	output.txt
10 2	5
27 4	6

Замечание

При составлении задачи ни один хомяк не пострадал.

Задача H. Подземелье

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

После прохождения пещеры со сталактитами и сталагмитами дрона Петра принимают на работу в исторический музей, и его основной обязанностью становится поиск кладов. На очередное задание Петра отправляют в подземелье, созданное древней высокоразвитой цивилизацией.

Подземелье представляет собой прямоугольник из N строк и M столбцов, содержащий $N \times M$ ячеек. Каждая ячейка либо является свободным квадратом, в котором можно перемещаться, либо монолитным. В свободных ячейках могут находиться сундуки с определённым количеством золота, но не более одного сундука в ячейке. Также в свободных ячейках могут находиться порталы, но не более одного портала в ячейке. В ячейке могут находиться и сундук, и портал одновременно.

Из ячейки, в которой находится портал, можно перемещаться в любую другую ячейку подземелья, если она не является монолитной; пользоваться одним и тем же порталом можно несколько раз. Также дрон может свободно перемещаться по пустым ячейкам, если они являются соседними по горизонтали или вертикали.

Изначально Пётр находится в ячейке $(S_i;\ S_j)$, и затем он некоторое время путешествует, собирает деньги из сундуков, прыгает по порталам и вообще весело проводит время. Однако рано или поздно Пётр должен вернуться домой, а сделать это он может только с помощью специального портала, расположенного в точке $(E_i;\ E_j)$. Этот портал предназначен только для телепортации на поверхность земли. Точки входа $(S_i;\ S_j)$ и выхода $(E_i;\ E_j)$ различны, а также могут содержать сундук или обычный портал (или и то, и другое).

Ваша задача — выяснить, может ли Пётр выбраться, и если может, то какое максимальное количество золота он может забрать с собой.

Формат входных данных

Первая строка содержит целые числа N и M $(1 \leqslant N, M \leqslant 200)$ — размеры подземелья.

Вторая строка содержит целые числа S_i, S_j, E_i, E_j $(1 \leqslant S_i, E_i \leqslant N,\ 1 \leqslant S_j, E_j \leqslant M)$ — координаты начальной точки и точки с порталом наружу.

Следующие N строк описывают карту подземелья. Каждая из них содержит M символов '.' или '#', обозначающих соответственно свободные и монолитные ячейки.

Следующая строка содержит целое число P $(0 \leqslant P \leqslant N \cdot M)$ — количество порталов.

Следующие P строк описывают порталы. Каждая из них содержит целые числа P_i и P_j $(1 \leqslant P_i \leqslant N,\ 1 \leqslant P_j \leqslant M)$ — координаты соответствующего портала.

Следующая строка содержит целое число G $(0 \leqslant G \leqslant N \cdot M)$ — количество сундуков с золотом.

Следующие G строк описывают сундуки. Каждая из них содержит целые числа G_i, G_j, G_c $(1 \leqslant G_i \leqslant N,\ 1 \leqslant G_j \leqslant M,\ 1 \leqslant G_c \leqslant 10^9)$ — координаты соответствующего сундука, а также количество золота, находящегося в нём.

Гарантируется, что порталы и сундуки находятся только в свободных ячейках.

Формат выходных данных

Если дрон может выбраться из пещеры живым, в первой строке выведите «possible», а во второй — максимальное количество золота, которое он может собрать.

Если же ему суждено вечно жить в пещере, выведите единственное слово «impossible».

Примеры

input.txt	output.txt
4 4 1 1 2 1 ...# .#.# .##. #... 2 1 1 4 4 4 1 1 10 1 2 14 4 3 6 4 4 7	possible 37
2 2 1 1 2 2 .# #. 0 0	impossible

Задача I. Симметричные биты

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Перед тем как стать исследователем пещер, дрон Пётр долгое время служил в Дронных Войсках Особого Назначения, где совершил очень много боевых вылетов. После окончания службы он решил сделать себе надпись на борту — общее число секунд, которые он провел в полёте на заданиях — 1274329 секунд. Поскольку Пётр — перфекционист, а число 1274329 далеко от идеала, он решил записать это число в двоичной системе счисления и получил 10011011100011011001. Пётр очень обрадовался, потому что получившаяся запись оказалась палиндромом, то есть одинаково читалась слева направо и справа налево. Такое совпадение глубоко засело в процессор нашего героя, и он решил выяснить, как много существует подобных чисел.

Он рассказал эту историю нам, а мы даём вам задачу. Чтобы она была не такой простой, вы должны найти количество чисел, битовая запись которых является палиндромом, на отрезке $[L;\ R]$.

Формат входных данных

Ввод содержит целые числа L и R ($1 \leqslant L \leqslant R \leqslant 10^{18}$).

Формат выходных данных

Выведите одно целое число — ответ на задачу.

Пример

input.txt	output.txt
2 8	3

Замечание

В первом тесте числами с двоичной записью-палиндромом являются 3, 5, 7.

Задача J. Теория больших шахмат

Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 секунды
Ограничение по памяти:	128 мегабайт

Шелдон и Леонард в очередной раз решили написать программу для игры в 3D-шахматы. Пока они сконцентрировали внимание на реализации стратегии самой игры. Но чтобы не терять время, они бросили клич к научной общественности — помочь реализовать простейшие подпрограммы, на которые им, естественно, жаль тратить свое драгоценное время.

Одна из задач — определить путь, посредством которого конь может попасть из одной клетки в другую за наименьшее число ходов, всё время оставаясь на одном и том же уровне (каждый уровень является плоской квадратной шахматной доской). В пределах одного уровня конь перемещается по стандартным шахматным правилам, но некоторые клетки доски перемещают оказавшиеся на них фигуры на другие уровни, поэтому коню запрещено оказываться в таких клетках в конце хода.

Формат входных данных

Первая строка содержит целое число N ($3 \leqslant N \leqslant 1000$) — размер шахматной доски.

Следующие N строк описывают шахматную доску. i-я из них содержит N символов, j-й из которых равен '.', если клетка $(i;\ j)$ является обычной клеткой доски, '*', если клетка является началом или концом искомого пути, либо '+', если в клетку запрещено попадать коню.

Гарантируется, что ввод содержит ровно два символа '*'.

Формат выходных данных

В первой строке выведите число M — количество клеток, входящих в искомый путь, включая начальные и конечные клетки.

Далее выведите M строк, каждая из которых содержит пару целых чисел — вертикальную и горизонтальную координаты клеток, которые посещает конь; строки должны быть упорядочены в порядке обхода. Строки доски нумеруются сверху вниз, начиная с 0. Столбцы доски нумеруются слева направо, начиная с 0.

Если искомого пути не существует, выведите единственное число 0.

Примеры

input.txt	output.txt
5 ..... .**..	4 1 1 2 3 0 4 1 2
5 *..*. ..++.	6 0 0 2 1 4 0 3 2 2 4 0 3
5 *.... ..+.. .+... ....*	0

Замечание

Шахматный конь может перемещаться на две клетки вверх, влево, вниз или вправо, а затем на одну клетку в любом перпендикулярном направлении. Конь «перепрыгивает» через клетки, то есть непосредственно посещает только начальную и конечную клетки хода.