Общая информация по задачам первого тура

Доступ к результатам проверки решений задач во время тура

В течение тура можно не более 10 раз по каждой задаче запросить информацию о результатах оценивания решения на тестах жюри.

Ограничение на размер исходного кода программы-решения

Во всех задачах размер файла с исходным кодом решения не должен превышать 256 КБ.

Процесс тестирования

Во всех задачах, очередная подзадача будет тестироваться, только если пройдены все тесты всех предыдущих подзадач.

Ограничения

Задача	Ограничение по времени	Ограничение по памяти	Получение результатов во время тура
1. Автоматические друзья	2 секунды	512 МБ	Для каждой подзадачи сообщаются баллы за эту подзадачу и результат проверки программы на каждом тесте
2. Памятник	2 секунды	512 МБ	Для каждой подзадачи сообщаются баллы за эту подзадачу и результат проверки программы на каждом тесте
3. Вышивка жемчугом	2 секунды	512 МБ	Для каждой подзадачи сообщаются баллы за эту подзадачу и результат проверки программы на каждом тесте
4. Пингвиноведение	2 секунды	512 МБ	Для каждой подзадачи сообщаются баллы за эту подзадачу и результат проверки программы на каждом тесте

Задача 1. Автоматические друзья

Имя входного файла:	`onlyone.in`
Имя выходного файла:	`onlyone.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 МБ

Школа юных программистов решила разработать собственную социальную сеть, которая должна автоматически подбирать для каждого пользователя потенциальных друзей. При регистрации каждому пользователю сети предлагается пройти психологическое тестирование, по результатам которого определяются значения трёх психологических характеристик этого пользователя. Значение каждой характеристики — целое положительное число.

Считается, что если у двух пользователей различаются значения всех трёх психологических характеристик, то они будут постоянно ссориться, а если совпадают значения двух или трёх характеристик, то им будет скучно. Таким образом, потенциальными друзьями являются только такие пары пользователей, у которых совпадают значения ровно одной характеристики, а значения двух других — различаются.

Требуется написать программу, которая по данным n тройкам (a_i, b_i, c_i) значений характеристик каждого из пользователей определяет количество пар потенциальных друзей, то есть таких пар индексов $i < j$, что из трёх равенств $a_i = a_j$, $b_i = b_j$, $c_i = c_j$ выполняется ровно одно.

Формат входных данных

Первая строка входных данных содержит число n — количество пользователей. Каждая из последующих n строк содержит три целых положительных числа a_i, b_i и c_i — значения характеристик i-го пользователя

Формат выходных данных

Выходные данные должны содержать искомое количество пар потенциальных друзей.

Таблица системы оценивания

Номер подзадачи	Баллы	Ограничения	
		n	a_i, b_i, c_i
1	45	$1 \leqslant n \leqslant 100$	$1 \leqslant a_i, b_i, c_i \leqslant 50$
2	55	$1 \leqslant n \leqslant 100\,000$	$1 \leqslant a_i, b_i, c_i \leqslant 100$

Примеры

onlyone.in	onlyone.out
3 1 2 3 1 4 5 1 2 4	2
4 100 100 100 100 100 100 100 99 99 99 99 100	5

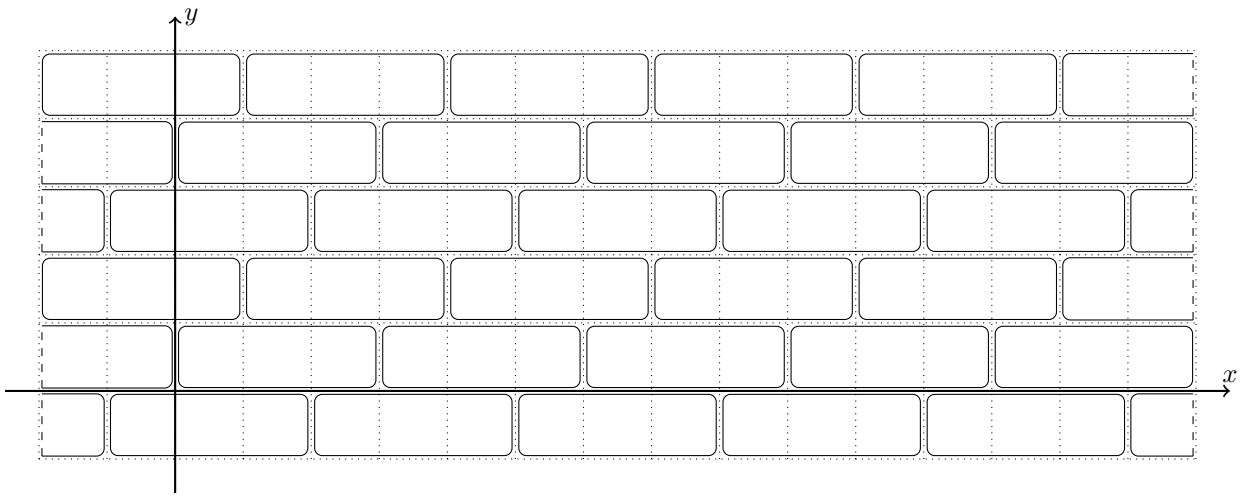
Пояснение к примеру

В первом примере потенциальную пару друзей образуют пользователи 1 и 2, а также 2 и 3. В обоих случаях у пользователей совпадает значение первой характеристики и различаются значения второй и третьей характеристик. Пользователи 1 и 3 имеют одинаковые значения первых двух характеристик, поэтому они не образуют пару потенциальных друзей.

Задача 2. Памятник

Имя входного файла:	`monument.in`
Имя выходного файла:	`monument.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 МБ

Одна из центральных площадей Архангельска замощена прямоугольными плитками размера $1 \times k$. Если ввести систему координат, так что левый нижний угол одной из плиток будет иметь координаты $(0, 0)$, то левые нижние углы плиток будут иметь координаты $(i \cdot k + j, j)$ для всех целых i и j.



На площади было решено установить памятник известному архангельскому писателю и художнику Степану Писахову. Для установки памятника необходимо удалить все плитки, полностью или частично попадающие под его основание. Основание памятника имеет форму многоугольника с целочисленными координатами вершин, все стороны которого параллельны осям координат. Известно, что любая прямая, пересекающая основание памятника и параллельная одной из осей координат, в пересечении с основанием образует один отрезок.

Для установки памятника необходимо выбрать место на площади таким образом, чтобы количество удалённых плиток было минимальным. При выборе места основание разрешается только передвигать параллельно осям координат.

Требуется написать программу, вычисляющую минимальное количество плиток, которые придётся удалить.

Формат входных данных

Первая строка входных данных содержит два числа n и k — количество вершин в основании памятника и размер плитки.

Каждая из последующих n строк содержит два целых числа x_i, y_i — координаты i-й вершины основания. Координаты перечислены в порядке обхода против часовой стрелки.

Формат выходных данных

Единственная строка выходных данных должна содержать минимально возможное количество плиток, которые необходимо удалить для размещения памятника на площади.

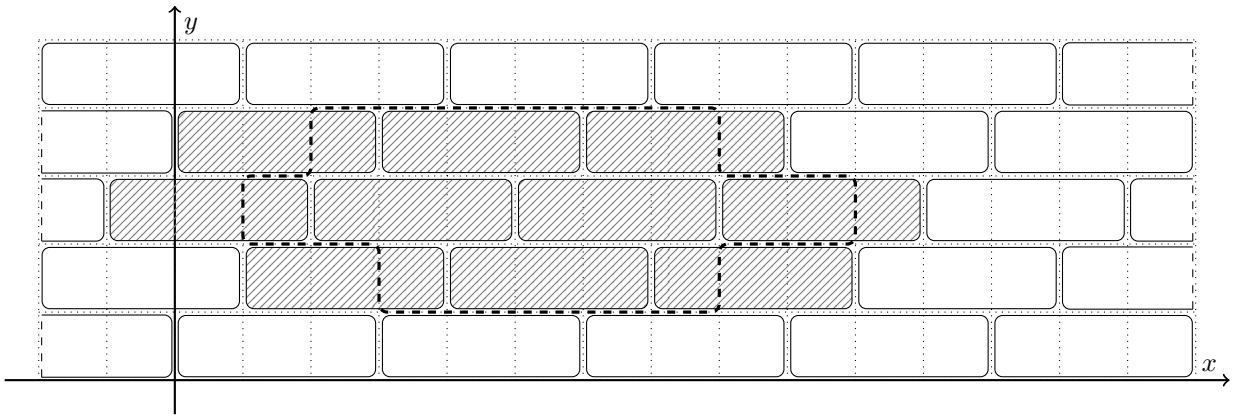
Таблица системы оценивания

Номер подзадачи	Баллы	Ограничения		
		n	k	x_i, y_i
1	32	$1 \leqslant n \leqslant 50$	$1 \leqslant k \leqslant 50$	$0 \leqslant x_i, y_i \leqslant 50$
2	37	$1 \leqslant n \leqslant 1000$	$1 \leqslant k \leqslant 1000$	$0 \leqslant x_i, y_i \leqslant 1000$
3	31	$1 \leqslant n \leqslant 100\,000$	$1 \leqslant k \leqslant 100\,000$	$0 \leqslant x_i, y_i \leqslant 1000\,000$

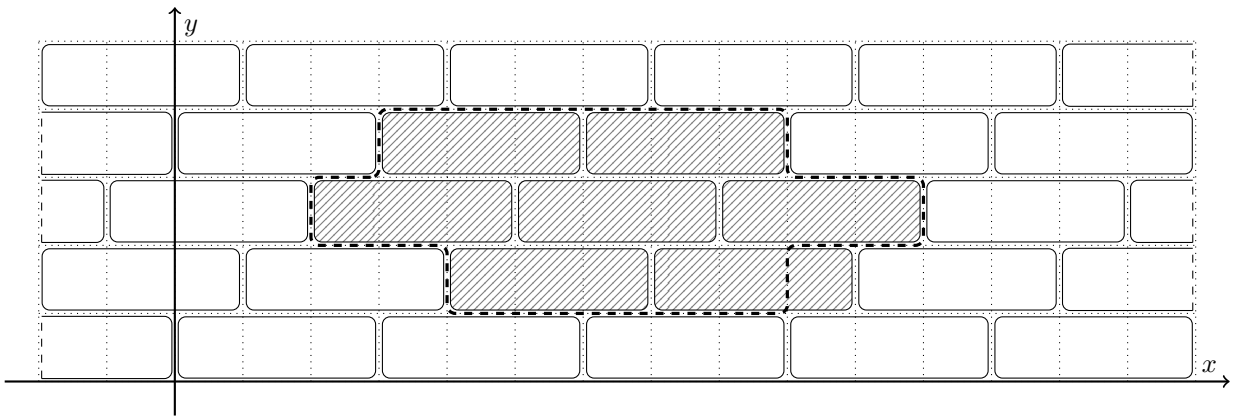
Примеры

monument.in	monument.out
12 3 2 3 1 3 1 2 3 2 3 1 8 1 8 2 10 2 10 3 8 3 8 4 2 4	7
8 4 1 0 5 0 5 2 4 2 4 3 2 3 2 1 1 1	5

Замечание



Исходное расположение основания памятника в первом примере.



Оптимальное расположение основания памятника в первом примере.

Задача 3. Вышивка жемчугом

Имя входного файла:	`repair.in`
Имя выходного файла:	`repair.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 МБ

Со стародавних времён в поморских деревнях рукодельницы вышивали жемчугом на прямоугольных полотенцах, состоящих из одинаковых клеток. Вышивка начиналась с пришивания жемчужины к полотенцу в центре одной из клеток. Чтобы пришить новую жемчужину, рукодельница делала стежок из клетки, уже содержащей жемчужину, в соседнюю с ней по горизонтали или вертикали свободную клетку. Новая жемчужина пришивалась в центре клетки на конце стежка. Этот процесс повторялся, пока не заканчивались жемчужины.

Одно из таких праздничных полотенец находится в музее. К сожалению, некоторые части узора были утеряны, но описание полотенца сохранилось. Дирекция музея планирует восстановить один из прямоугольных фрагментов полотенца, но не ещё не решила какой именно. Затраты на восстановление фрагмента зависят от количества связных частей узора, попавших на этот фрагмент. Часть узора считается связной, если от любой её жемчужины можно по стежкам перейти к любой другой её жемчужине, не выходя за границы фрагмента. Дирекция всегда относит любые две жемчужины, между которыми можно перейти по стежкам, к одной и той же связной части узора.

Требуется написать программу, вычисляющую количество связных частей узора для каждого из заданных фрагментов.

Формат входных данных

Первая строка входных данных содержит два целых числа a и b — размеры полотенца в клетках по горизонтали и вертикали.

Вторая строка содержит два числа n и q — количество жемчужин в узоре и количество фрагментов соответственно.

Следующие $(n-1)$ строк содержат описания стежков. Каждый стежок имеет один из следующих видов:

- `h` x y означает, что клетки с координатами (x, y) и $(x+1, y)$ содержат жемчужины, соединённые горизонтальным стежком $(1 \leqslant x \leqslant a-1;\ 1 \leqslant y \leqslant b)$;
- `v` x y означает, что клетки с координатами (x, y) и $(x, y+1)$ содержат жемчужины, соединённые вертикальным стежком $(1 \leqslant x \leqslant a;\ 1 \leqslant y \leqslant b-1)$.

Так как неизвестно в каком порядке рукодельница наносила стежки, их описания следуют в произвольном порядке. При этом гарантируется, что узор был получен в результате процесса, описанного в условии задачи.

Следующие q строк описывают фрагменты. Каждое описание содержит четыре целых числа x_1, y_1, x_2 и y_2 — координаты левой нижней и правой верхней клетки фрагмента $(1 \leqslant x_1 \leqslant x_2 \leqslant a;\ 1 \leqslant y_1 \leqslant y_2 \leqslant b)$.

Формат выходных данных

Выходные данные должны содержать q строк, где i-я строка содержит количество связных частей узора в i-м фрагменте.

Таблица системы оценивания

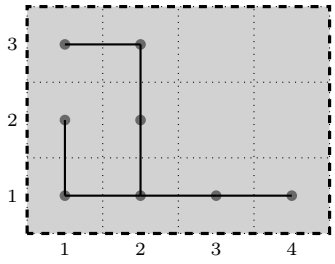
Номер подзадачи	Баллы	Ограничения		
		a, b	n	q
1	28	$1 \leqslant a, b \leqslant 100$	$2 \leqslant n \leqslant 100$	$1 \leqslant q \leqslant 100$
2	27	$1 \leqslant a, b \leqslant 3000$	$2 \leqslant n \leqslant 3000$	$1 \leqslant q \leqslant 3000$
3	23	$1 \leqslant a, b \leqslant 3000$	$2 \leqslant n \leqslant 100\,000$	$1 \leqslant q \leqslant 100\,000$
4	22	$1 \leqslant a, b \leqslant 150\,000$	$2 \leqslant n \leqslant 150\,000$	$1 \leqslant q \leqslant 150\,000$

Примеры

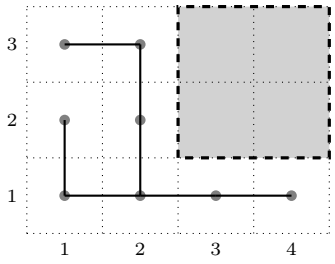
repair.in	repair.out
4 3 8 4 v 1 1 h 1 1 h 2 1 v 2 1 v 2 2 h 1 3 h 3 1 1 1 4 3 3 2 4 3 3 1 3 1 1 2 3 3	1 0 1 2

Замечание

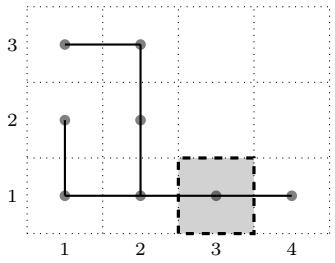
Пояснение к тесту из условия.



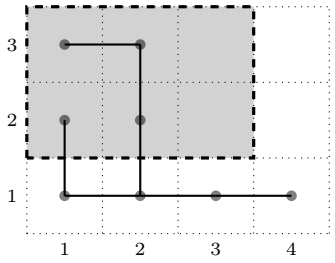
$x_1 = 1$, $y_1 = 1$, $x_2 = 4$, $y_2 = 3$



$x_1 = 3$, $y_1 = 2$, $x_2 = 4$, $y_2 = 3$



$x_1 = 3$, $y_1 = 1$, $x_2 = 3$, $y_2 = 1$



$x_1 = 1$, $y_1 = 2$, $x_2 = 3$, $y_2 = 3$

Задача 4. Пингвиноведение

Имя входного файла:	`penguins.in`
Имя выходного файла:	`penguins.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 МБ

На кафедре пингвиноведения Южного Антарктического университета проводятся исследования популяций пингвинов. Фотографии скоплений плотно стоящих пингвинов обрабатываются студентами. Распознавание пингвинов на снимках производится следующим образом: на фотографии выбирается *характерная полоса* высотой в один пиксель, каждый пиксель которой входит в изображение одного из пингвинов.

У всех пингвинов исследуемой популяции живот белый, а спина и крылья — чёрные. Таким образом, если у пингвина на фотографии видна только спина, то на характерной полосе ему соответствует отрезок из чёрных пикселей, а если только живот, то из белых. В остальных случаях, например, когда чёрные крылья видны поверх белого живота, пингвину соответствует отрезок из чёрных и белых пикселей. Для продолжения исследований необходимо, чтобы каждому пингвину соответствовал отрезок, состоящий либо только из чёрных, либо только из белых пикселей.

Для i-й фотографии известно максимальное количество пингвинов k_i, изображение которых могло попасть на характерную полосу. Поэтому эту полосу пикселей необходимо заменить на упрощённую полосу той же длины, которая будет состоять не более чем из k_i отрезков, каждый из которых либо полностью чёрный, либо полностью белый. Из всех возможных упрощённых полос нужно выбрать оптимальную — то есть ту, которая получается из характерной путём изменения цвета минимального числа пикселей.

Требуется написать программу, решающую поставленную задачу.

Формат входных данных

В первой строке входных данных содержится число t — количество фотографий. Далее следуют t пар строк, i-я пара строк описывает i-ю фотографию.

Первая строка описания фотографии содержит два числа: n_i — длину характерной полосы i-й фотографии, и k_i — максимальное количество пингвинов, которые могут быть на ней изображены ($k_i \leqslant n_i$).

Вторая строка описания состоит из n_i символов `0` и `1`, где `0` обозначает чёрный, а `1` — белый пиксель.

Формат выходных данных

Выходные данные должны содержать t строк, где i-я строка состоит из n_i символов `0` и `1` и описывает упрощённую полосу, полученную из характерной полосы i-й фотографии. Если оптимальных упрощённых полос несколько, выведите любую из них.

Таблица системы оценивания

Номер подзадачи	Баллы	Ограничения	
		n_i, k_i	$n = n_1 + n_2 + \cdots + n_t$
1	11	$1 \leqslant k_i \leqslant n_i \leqslant 10$	$1 \leqslant n \leqslant 5000$
2	24	$1 \leqslant k_i \leqslant n_i \leqslant 100$	$1 \leqslant n \leqslant 5000$
3	24	$1 \leqslant k_i \leqslant n_i \leqslant 1000$	$1 \leqslant n \leqslant 50\,000$
4	21	$1 \leqslant k_i \leqslant 5000$	$1 \leqslant n \leqslant 100\,000$
5	20	$1 \leqslant k_i \leqslant 200\,000$	$1 \leqslant n \leqslant 200\,000$

Примеры

penguins.in	penguins.out
3 9 3 000111000 10 3 0111011010 4 4 0001	000111000 0111111000 0001

Общая информация по задачам второго тура

Доступ к результатам проверки решений задач во время тура

В течение тура можно не более 10 раз по каждой задаче запросить информацию о результатах оценивания решения на тестах жюри.

Ограничение на размер исходного кода программы-решения

Во всех задачах размер файла с исходным кодом решения не должен превышать 256 КБ.

Процесс тестирования

Во всех задачах, очередная подзадача будет тестироваться, только если пройдены все тесты всех предыдущих подзадач.

Ограничения

Задача	Ограничение по времени	Ограничение по памяти	Получение результатов во время тура
5. Поможем дикой природе	2 секунды	512 МБ	Для каждой подзадачи сообщаются баллы за эту подзадачу и результат проверки программы на каждом тесте.
6. Подводная лодка	2 секунды	512 МБ	Для каждой подзадачи сообщаются баллы за эту подзадачу и результат проверки программы на каждом тесте.
7. Фонари	2 секунды	512 МБ	Для подзадач 1 и 2 сообщаются баллы за эту подзадачу и результат проверки программы на каждом тесте. Для подзадач 3, 4 и 5 сообщаются только баллы за эту подзадачу.
8. Сигнализация	4 секунды	512 МБ	Для подзадач 1 и 2 сообщаются баллы за эту подзадачу и результат проверки программы на каждом тесте. Для подзадач 3, 4 и 5 сообщаются только баллы за эту подзадачу.

Задача 5. Поможем дикой природе

Имя входного файла:	`grants.in`
Имя выходного файла:	`grants.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 МБ

Фонд изучения дикой природы в течение t лет ежегодно выделяет денежные гранты в поддержку исследований северной фауны. На гранты претендуют три организации, одна из которых занимается изучением тюленей, вторая — оленей, третья — белых медведей.

Для упрощения бухгалтерского учёта фонд принял следующие решения:

- размер любого гранта в денежных единицах должен быть степенью числа 2, то есть равен 2^k для некоторого целого $k \geqslant 0$;
- все гранты, получаемые одной организацией в одном году, должны иметь различные размеры.

В i-м году фонд планирует полностью распределить n_i денежных единиц, выделенных на гранты. Сравнивать результативность использования средств возможно только для грантов одинакового размера, выделенных каждой из трёх организаций. Такие гранты называются *целевыми*. Распределение денежных единиц на гранты между тремя организациям считается оптимальным, если как можно бóльшая часть общей суммы выделена на целевые гранты.

Например, если в текущем году на все гранты выделено 47 денежных единиц, то оптимальным вариантом распределения будет: выделить каждой из организаций целевые гранты размерами по 2 и 8 денежных единиц, что составит в сумме 30 единиц. Остальные 17 единиц можно распределить, например, выделив первой организации 16 денежных единиц, а третьей — 1 денежную единицу. Выделить более 30 денежных единиц на целевые гранты, распределяя 47 денежных единиц, нельзя.

Требуется написать программу, которая по заданной в i-м году общей сумме грантов n_i определяет, сколько денежных единиц следует выделить каждой из трёх организаций при оптимальном распределении грантов.

Формат входных данных

В первой строке входных данных записано целое число t — количество лет ($1 \leqslant t \leqslant 100$). В каждой из последующих t строк записано целое число n_i — общая сумма грантов, которую необходимо полностью распределить в i-м году.

Формат выходных данных

Выходные данные должны содержать t строк по три целых числа в каждой — суммы грантов, которые следует выделить каждой из трёх организаций в соответствующий год. Если оптимальных вариантов распределения несколько, необходимо вывести любой из них.

Таблица системы оценивания

Номер подзадачи	Баллы	Ограничения
		n_i
1	16	$1 \leqslant n_i < 64$
2	33	$1 \leqslant n_i < 512$
3	17	$1 \leqslant n_i < 2^{17}$
4	34	$1 \leqslant n_i < 2^{60}$

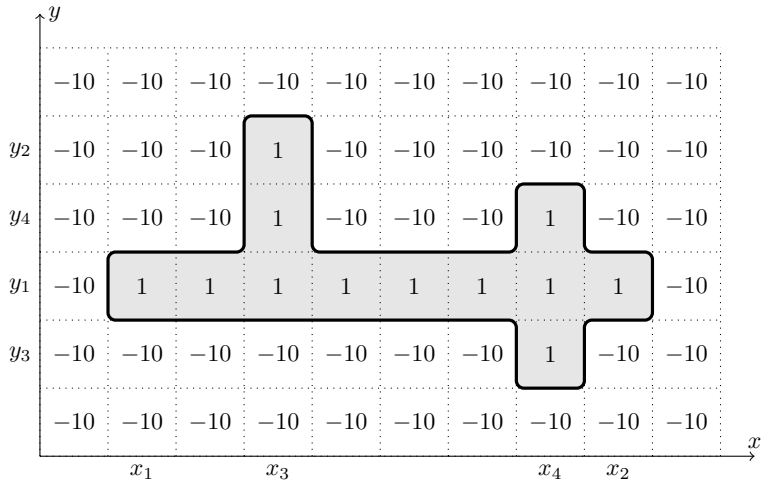
Примеры

grants.in	grants.out
3 4 21 47	0 0 4 7 7 7 26 10 11

Задача 6. Подводная лодка

Имя входного файла:	`submarine.in`
Имя выходного файла:	`submarine.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 МБ

Подводная лодка легла на грунт на мелководье. Для её обнаружения используются данные спутника, который с высокой точностью измеряет отклонение высоты поверхности воды от среднего уровня моря. Снимок, получаемый со спутника, представляет собой массив из h строк по w элементов в каждой строке.



Введём на снимке систему координат с осью абсцисс, направленной вдоль строк снимка слева направо, и осью ординат, направленной вдоль столбцов снимка снизу вверх. Потенциальное изображение подводной лодки представляет собой любое множество элементов массива, состоящее из следующих частей:

- «корпус» — полоса из элементов с координатами от (x_1, y_1) до (x_2, y_1), где $x_1 < x_2$;
- «рубка» — полоса из элементов с координатами от (x_3, y_1) до (x_3, y_2), где $x_1 \leqslant x_3 < x_2$; $y_1 \leqslant y_2$;
- «хвост» — полоса из элементов с координатами от (x_4, y_3) до (x_4, y_4), где $x_3 < x_4 \leqslant x_2$; $y_3 \leqslant y_1 \leqslant y_4$.

Поскольку подводная лодка находится вблизи поверхности в районе с сильным течением, уровень воды над ней немного повышается. Поэтому изображением подводной лодки на снимке будем считать потенциальное изображение с максимально возможной суммой входящих в него элементов массива.

Требуется написать программу, которая находит на снимке изображение подводной лодки и выводит сумму его элементов.

Формат входных данных

Для сжатия передаваемых со спутника данных каждый элемент снимка кодируется строчной буквой английского алфавита. Первая строка входных данных содержит число k — количество использованных для кодирования букв ($k \leqslant 26$). Вторая строка входных данных содержит k целых чисел c_i — значения отклонений соответствующих каждому кодовому символу по порядку букв в английском алфавите от 1 до k-й.

Третья строка входных данных содержит числа h и w — размеры снимка. Последующие h строк содержат по w символов — кодовые значения элементов снимка.

Формат выходных данных

Выходные данные должны содержать единственное целое число — сумму элементов массива, соответствующих изображению подводной лодки.

Таблица системы оценивания

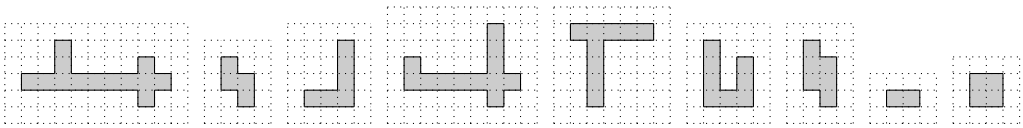
Номер подзадачи	Баллы	Ограничения		Комментарии
		h, w	$\lvert c_i \rvert$	
1	32	$5 \leqslant h, w \leqslant 10$	$\lvert c_i \rvert \leqslant 10$	
2	22	$5 \leqslant h, w \leqslant 100$	$\lvert c_i \rvert \leqslant 100$	
3	23	$5 \leqslant h, w \leqslant 500$	$\lvert c_i \rvert \leqslant 500$	
4	до 23	$5 \leqslant h, w \leqslant 2000$	$\lvert c_i \rvert \leqslant 2000$	Тесты этой подзадачи оцениваются независимо

Примеры

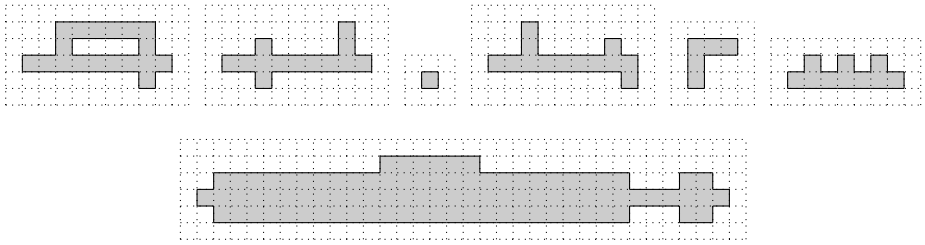
submarine.in	submarine.out	Изображение
2 -10 1 6 11 aaaaaaaaaaa aaabaaaaaaa aaabaaaabaa abbbbbbbbba aaaaaaaabaa aaaaaaaaaaa	13	....b....... ...b....b.. .bbbbbbbbb. ....b.. ....
3 -4 -3 4 5 5 bbabc ccaac accba baccb baaaa	16	..... .c... .cc.. ..c.. .....
3 -2 4 0 5 5 abccb cccac cbcba cccbb accba	24	.b... .c... .b.b. cccbb ...b.
4 -1 -5 -3 0 5 5 bbabc ccaac acdba baccb baaaa	-2	....aa. ....

Пояснение

Для примера ниже приведены несколько потенциальных изображений подводной лодки.



Ниже приведены несколько множеств элементов снимка, которые не являются потенциальными изображениями подводной лодки:



Задача 7. Фонари

Имя входного файла:	`lamps.in`
Имя выходного файла:	`lamps.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 МБ

Улицу Подводный канал освещают n фонарей, пронумерованных вдоль улицы от 1 до n. Один или несколько подряд стоящих фонарей назовём сегментом. Таким образом, общее количество сегментов равно $\frac{n(n+1)}{2}$. Сегмент считается исправным, если лампочки во всех фонарях этого сегмента исправны.

С фонарями регулярно происходят события одного из двух типов:

- в каком-то сегменте из-за скачков напряжения все лампочки одновременно перегорают;
- Архиэнерго выбирает некоторый сегмент и посылает ремонтников, чтобы заменить на нем все перегоревшие лампочки на исправные.

После каждого события мэрия города требует от Архиэнерго предоставить отчёт о количестве исправных сегментов. Для улучшения показателей работы ремонтники включают в отчёт все сегменты, которые исправны сейчас или были исправны когда-либо ранее.

Требуется написать программу, определяющую количество сегментов после каждого события, которые исправны в этот момент или были исправны когда-либо до этого события.

Формат входных данных

В первой строке входных данных содержатся два числа n и q — количество фонарей и количество произошедших событий. Следующая строка входных данных состоит из n символов `0` и `1`, описывающих начальное состояние фонарей, где `1` обозначает фонарь с исправной лампочкой, а `0` — с перегоревшей.

В каждой из последующих q строк содержатся описания событий в виде трёх чисел l_i, r_i и c_i, которые означают, что после этого события все лампочки в фонарях с номерами $l_i, l_i + 1, \ldots, r_i$:

- перегорают при $c_i = 0$,
- становятся исправными при $c_i = 1$.

В описаниях всех событий $1 \leqslant l_i \leqslant r_i \leqslant n$, а c_i принимает значение 0 или 1.

Формат выходных данных

В первой строке выходных данных выведите единственное число — количество исправных сегментов в начальном состоянии. Затем по одному в строке выведите q чисел: для каждого из произошедших событий выведите количество сегментов, указываемых в отчёте после этого события.

Таблица системы оценивания

Номер подзадачи	Баллы	Ограничения		Комментарии
		n	q	
1	17	$1 \leqslant n \leqslant 50$	$1 \leqslant q \leqslant 150$	
2	19	$1 \leqslant n \leqslant 500$	$1 \leqslant q \leqslant 250$	
3	до 12	$1 \leqslant n \leqslant 5000$	$1 \leqslant q \leqslant 1000$	Тесты этой подзадачи оцениваются независимо
4	до 20	$1 \leqslant n \leqslant 50\,000$	$1 \leqslant q \leqslant 1000$	Тесты этой подзадачи оцениваются независимо
5	до 32	$1 \leqslant n \leqslant 300\,000$	$1 \leqslant q \leqslant 300\,000$	Тесты этой подзадачи оцениваются независимо

Примеры

lamps.in	lamps.out
7 4 1100101 4 6 1 3 6 0 3 4 1 5 7 1	5 13 13 19 28

Задача 8. Сигнализация

Имя входного файла:	`alarm.in`
Имя выходного файла:	`alarm.out`
Ограничение по времени:	4 секунды
Ограничение по памяти:	512 МБ

Подземный бункер состоит из n комнат, соединённых $n-1$ коридорами. Каждый коридор соединяет две различные комнаты и имеет определённую длину. Бункер устроен таким образом, что из любой комнаты i можно дойти в любую другую комнату j. Заметим, что существует единственный такой путь, не проходящий по одному и тому же коридору дважды. Сумма длин коридоров, составляющих этот путь, называется расстоянием между комнатами i и j и обозначается $\rho(i, j)$.

Каждая комната бункера оборудована звуковой сигнализацией, состоящей из сирены и датчика звука, который её включает. Сирена, включённая в комнате i, активирует датчик звука в каждой комнате, расстояние до которой не превосходит расстояние d_i, определяемое мощностью этой сирены. Другими словами, включение сирены в комнате i автоматически включает сирену во всех комнатах j, таких что $\rho(i, j) \leqslant d_i$. Эта сирена, в свою очередь, может вызвать автоматическое включение других сирен и так далее.

В случае возникновения чрезвычайной ситуации некоторые сирены необходимо включить вручную, после чего звук от них автоматически включит сирены в других комнатах. Правила безопасности предписывают выбор такого набора сирен для ручного включения, который в конце концов приведёт к автоматическому включению сирен во всех комнатах.

Требуется написать программу, которая определяет минимальное количество сирен в наборе, удовлетворяющем правилам безопасности.

Формат входных данных

Первая строка входных данных содержит единственное число n — количество комнат.

Вторая строка содержит последовательность из n целых чисел d_i, i-е из них равно максимальному расстоянию, на котором расположенная в комнате i сирена активирует датчики ($0 \leqslant d_i \leqslant 10^9$).

Последующие $n-1$ строк описывают коридоры бункера. В i-й из них находятся три целых числа: u_i, v_i, l_i, где u_i, v_i — номера различных комнат, соединённых коридором i, а l_i — длина этого коридора ($1 \leqslant u_i, v_i \leqslant n$; $1 \leqslant l_i \leqslant 10^9$).

Формат выходных данных

Выходные данные должны состоять из единственного числа — минимального количества сирен, которые необходимо включить вручную.

Таблица системы оценивания

Номер подзадачи	Баллы	Ограничения
		n
1	16	$1 \leqslant n \leqslant 15$
2	23	$1 \leqslant n \leqslant 100$
3	17	$1 \leqslant n \leqslant 3000$
4	24	$1 \leqslant n \leqslant 100\,000$
5	20	$1 \leqslant n \leqslant 300\,000$

Примеры

alarm.in	alarm.out
10 1 2 2 2 6 3 4 5 4 3 1 2 5 2 3 1 2 4 5 4 5 2 4 6 4 4 7 3 1 8 1 8 9 5 8 10 4	3

Замечания

В тесте из примера сирена в комнате 4 включает сирену в комнате 5, которая, в свою очередь, включает сирены в комнатах 6 и 7. Сирена в комнате 2 включает сирену в комнате 3. Сирена в комнате 8 включает сирены в комнатах 1, 9 и 10.