Задача A. Расстановка ферзей

Имя входного файла:	`стандартный ввод` или `input.txt`
Имя выходного файла:	`стандартный вывод` или `output.txt`
Ограничение по времени:	2 секунды
Ограничение по памяти:	1024 мегабайта

Захар очень любит играть в шахматы и программировать. Поэтому больше всего на свете Захар любит задачи по программированию на шахматы.

У Захара есть шахматная доска, представляющая собой квадратную таблицу из m строк и m столбцов, а так же n ферзей. Строки таблицы пронумерованы целыми числами от 1 до m сверху вниз, а столбцы — слева направо.

Захар пытается решить задачу о расстановке ферзей на доску: необходимо расставить ферзей так, чтобы никакой ферзь не бил другого ферзя. Напомним, что шахматный ферзь бьет все клетки, расположенные в той же строке, в том же столбце и на тех же диагоналях, что и сам ферзь. У Захара есть n ферзей, i-го из них он собирается поставить на клетку с координатами (x_i, y_i), где x_i — номер столбца, а y_i — номер строки. Гарантируется, что все координаты ферзей попарно различны.

К сожалению, Захар не научился проверять расстановку на корректность. Его интересует q отрезков ферзей с номерами с l_i по r_i включительно. Помогите Захару и определите для каждого из этих отрезков, существует ли на нем пара ферзей, которые бьют друг друга. Формально расстановка ферзей из отрезка $[l_i, r_i]$ считается корректной, если не существует двух ферзей с номерами j и k таких, что $l_i \leqslant j < k \leqslant r_i$ и j-й ферзь бьет k-го ферзя.

Формат входных данных

Первая строка содержит два целых числа n и m ($2 \leqslant n \leqslant 200\,000$, $2 \leqslant m \leqslant 10^9$) — количество ферзей и размер поля, соответственно.

В следующих n строках описываются позиции ферзей. В i-й из них содержатся два целых числа x_i и y_i ($1 \leqslant x_i, y_i \leqslant m$) — координаты клетки, в которую будет поставлен i-й ферзь.

Следующая строка содержит одно целое число q ($1 \leqslant q \leqslant 10^6$) — количество запросов.

В i-й из следующих q строк содержатся два целых числа l_i и r_i ($1 \leqslant l_i \leqslant r_i \leqslant n$) — границы отрезка i-го запроса.

Гарантируется, что все координаты ферзей попарно различны.

Формат выходных данных

В q строках выведите ответы на запросы.

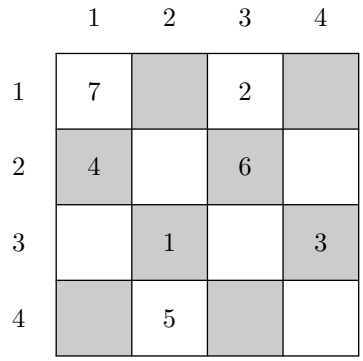
Если расстановка ферзей из отрезка $[l_i, r_i]$ корректна (то есть не существует пары ферзей с номерами из отрезка, которые бьют друг друга), в i-й строке выведите «`Yes`» (без кавычек), иначе в i-й строке выведите «`No`» (без кавычек).

Пример

стандартный ввод	стандартный вывод
7 4 3 2 1 3 3 4 2 1 4 2 2 3 1 1 8 1 2 1 3 2 5 3 5 3 6 4 7 5 7 7 7	Yes No Yes Yes No No Yes Yes

Замечание

На картинке изображена шахматная доска для примера. В клетках указаны номера соответствующих ферзей.



В первом запросе ферзи 1 и 2 не бьют друг друга.
Во втором запросе ферзи 1 и 3 бьют друг друга.
В третьем запросе ферзи 2 – 5 не бьют друг друга.
В четвертом запросе ферзи 3 – 5 не бьют друг друга.
В пятом запросе ферзи 3 и 6, а также 4 и 6 бьют друг друга.
В шестом запросе ферзи 4 и 6, а также 4 и 7 бьют друг друга.
В седьмом запросе ферзи 5 – 7 не бьют друг друга.
В восьмом запросе присутствует только один ферзь.

Система оценки

Тесты к этой задаче состоят из семи групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов некоторых из предыдущих групп. Обратите внимание, что прохождение тестов из условия не требуется для некоторых групп.

Группа	Баллы	Доп. ограничения			Необх. группы	Комментарий
		n	m	q		
0	0	–	–	–	–	Тесты из условия
1	7	–	–	–	–	Для всех ферзей $x_i = 1$
2	12	$n \leqslant 100$	$m \leqslant 100\,000$	$q \leqslant 100$	0	
3	16	$n \leqslant 5\,000$	$m \leqslant 100\,000$	$q \leqslant 5\,000$	0, 2	
4	23	$n \leqslant 100\,000$	$m \leqslant 100\,000$	$q \leqslant 100\,000$	0, 2, 3	
5	17	–	$m \leqslant 100\,000$	–	–	Во всех запросах $l_i = 1$
6	10	–	–	–	5	Во всех запросах $l_i = 1$
7	15	–	–	–	0 – 6	

Задача B. История

Имя входного файла:	`стандартный ввод` или `input.txt`
Имя выходного файла:	`стандартный вывод` или `output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	1024 мегабайта

Традиционно на пары по истории приходило мало студентов, поэтому преподаватель решил изменить систему и ввести обязательные групповые проекты для студентов, которые выполнять можно только очно на парах. Теперь на пары ходят все n студентов, пронумерованных от 1 до n. Преподавателю известно, что уровень знаний i-го студента равен a_i.

Для выполнения группового проекта студентам необходимо разбиться на пары. Чтобы это не было слишком просто, преподаватель выдвинул следующее требование: два человека с номерами $i \neq j$ могут быть в паре, если либо $a_i + a_j = S$, либо $a_i \oplus a_j = X$, где $\oplus$ обозначает операцию побитового исключающего ИЛИ (XOR).

Помогите студентам узнать, можно ли организоваться по парам так, чтобы требования преподавателя были удовлетворены?

Формат входных данных

Первая строка содержит три целых числа n, S и X ($2 \leqslant n \leqslant 500\,000$, $0 \leqslant S, X < 2^{30}$, n **четно**) — количество студентов и необходимые значения суммы или XOR в паре соответственно.

Следующая строка содержит n целых чисел $a_1,\ a_2,\ \ldots,\ a_n$ ($0 \leqslant a_i < 2^{30}$) — уровни знаний студентов.

Формат выходных данных

Если пары организовать можно, то в первой строке выведите «`Yes`» (без кавычек).

В следующих $\frac{n}{2}$ строках выведите сами пары, в i-й строке должно содержаться два целых числа c_i, d_i ($1 \leqslant c_i, d_i \leqslant n, c_i \neq d_i$), означающих, что студентов с номерами c_i и d_i нужно объединить в пару. Каждый номер студента, описывающийся числом от 1 до n, должен встречаться среди пар ровно один раз.

Если нельзя организовать пары так, чтобы все условия были удовлетворены, в одной единственной строке выведите «`No`» (без кавычек).

Примеры

стандартный ввод	стандартный вывод
6 7 0 1 2 9 9 5 6	Yes 1 6 2 5 4 3
4 6 2 1 5 2 3	No

Замечание

Покажем, что разбиение на пары в первом примере корректно.

- $a_1 + a_6 = 1 + 6 = 7$
- $a_2 + a_5 = 2 + 5 = 7$
- $a_4 \oplus a_3 = 9 \oplus 9 = 0$

Система оценки

Тесты к этой задаче состоят из восьми групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов некоторых из предыдущих групп. Обратите внимание, что прохождение тестов из условия не требуется для некоторых групп. **Offline-проверка** означает, что результаты тестирования вашего решения на данной группе станут доступны только после окончания соревнования.

Группа	Баллы	Доп. ограничения	Необх. группы	Комментарий
		n		
0	0	–	–	Тесты из условия
1	9	$n \leqslant 20$	0	
2	15	$n \leqslant 100$	0 – 1	
3	7	–	–	$S \leqslant 1,\ a_i \geqslant 1$
4	17	–	–	$X = 0$
5	10	$n \leqslant 2\,000$	0 – 2	
6	21	–	–	Все числа в a различны
7	11	$n \leqslant 100\,000$	0 – 2, 5	
8	10	–	0 – 7	**Offline-проверка**

Задача C. Мармеладки

Имя входного файла:	`стандартный ввод` или `input.txt`
Имя выходного файла:	`стандартный вывод` или `output.txt`
Ограничение по времени:	3 секунды
Ограничение по памяти:	1024 мегабайта

Петя очень любит мармеладки. Есть n магазинов с мармеладками, в i-м магазине продается бесконечное количество мармеладок аппетитности a_i. Петя питается только мармеладками, поэтому его друг Саша ответственно следит за его питанием.

Каждый день происходят события двух типов:

1. В магазинах с номерами от l до r аппетитность мармеладок увеличивается на x:

$$a_i := a_i + x \quad \text{для всех } i \in [l, r].$$

2. Покупка мармеладок. Петя рассматривает магазины на отрезке с номерами от l до r по очереди в таком порядке. В каждом магазине Петя может либо купить ровно одну мармеладку, либо пропустить и не покупать ничего.

   Пусть Петя купил мармеладки в магазинах с номерами $l \leqslant i_1 < i_2 < \ldots < i_m \leqslant r$. Рассмотрим аппетитности мармеладок в порядке покупки, обозначим $b_j := a_{i_j}$. Среди всех способов закупить мармеладки Петя выбирает такой, при котором последовательность b будет лексикографически максимальной.

   После покупки Саша хочет узнать значение b_k (то есть аппетитность k-й мармеладки, которую купит Петя), либо узнать, что длина выбранной последовательности меньше k.

Помогите Саше разобраться с питанием Пети!

Напомним, что последовательность $(b_1, b_2, \ldots, b_m)$ лексикографически больше последовательности $(c_1, c_2, \ldots, c_k)$, если существует позиция i такая, что $b_i > c_i$ и для всех $j < i$ выполнено $b_j = c_j$, либо если $(b_1, \ldots, b_k)$ является префиксом $(c_1, \ldots, c_k)$ и $m > k$.

Формат входных данных

Первая строка содержит два целых числа n и q ($1 \leqslant n, q \leqslant 500\,000$) — количество магазинов и количество дней.

Вторая строка содержит n целых чисел $a_1, a_2, \ldots, a_n$ ($1 \leqslant a_i \leqslant 10^9$) — изначальные аппетитности мармеладок.

Следующие q строк содержат описания запросов. Сначала вводится число t_i ($1 \leqslant t \leqslant 2$) — тип i-го запроса.

Если $t = 1$, то далее следуют три числа l_i, r_i и x_i ($1 \leqslant l_i \leqslant r_i \leqslant n, 1 \leqslant x_i \leqslant 10^9$) — увеличение аппетитности мармеладок в магазинах с номерами $l_i, l_i + 1, \ldots, r_i$ на число x_i.

Если $t = 2$, то далее следуют три числа l_i, r_i и k_i ($1 \leqslant l_i \leqslant r_i \leqslant n, 1 \leqslant k_i \leqslant n$) — Петя рассматривает мармеладки в магазинах $l_i, l_i + 1, \ldots, r_i$, Саша хочет узнать, какой аппетитности мармеладку Петя купит k_i-й по счету.

Формат выходных данных

На каждый запрос второго типа выведите, какой аппетитности мармеладку купил Петя k-й по счету.

В случае, если Петя купил меньше k мармеладок, выведите -1.

Примеры

стандартный ввод	стандартный вывод
5 5 1 3 2 3 2 2 1 5 3 2 3 5 1 1 3 3 2 2 2 5 2 2 1 5 4	2 3 3 -1
5 6 5 2 5 5 2 1 3 5 10 2 2 2 1 2 3 5 1 1 2 3 11 2 3 4 1 2 2 4 1	2 15 26 26

Замечание

Рассмотрим первый пример.

Изначальные аппетитности мармеладок равняются $[1, 3, 2, 3, 2]$.

В первом запросе Петя закупает мармеладки на всем массиве. Лексикографически максимальная последовательность b, которую он может получить, равняется $[3, 3, 2]$. От нас спрашивают третий элемент этой последовательности, он равняется двум.

Во втором запросе Петя закупает мармеладки на отрезке $[3, 5]$. Лексикографически максимальная последовательность b, которую он может получить, равняется $[3, 2]$. От нас спрашивают первый элемент этой последовательности, он равняется трем.

Далее аппетитность в третьем магазине увеличивается на 2, и аппетитности становятся $[1, 3, 4, 3, 2]$.

Далее Петя закупает мармеладки на отрезке $[2, 5]$. Лексикографически максимальная последовательность b, которую он может получить, равняется $[4, 3, 2]$. От нас спрашивают второй элемент этой последовательности, он равняется трем.

В последнем запросе Петя закупает мармеладки на всем массиве. Лексикографически максимальная последовательность b, которую он может получить, равняется $[4, 3, 2]$. От нас спрашивают четвертый элемент последовательности, и мы выводим -1, потому что его не существует.

Система оценки

Тесты к этой задаче состоят из семи групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов некоторых из предыдущих групп. Обратите внимание, что прохождение тестов из условия не требуется для некоторых групп. **Offline-проверка** означает, что результаты тестирования вашего решения на данной группе станут доступны только после окончания соревнования.

Обозначим за m количество мармеладок, которых купит Петя в очередном запросе.

Группа	Баллы	Доп. ограничения		Необх. группы	Комментарий
		n	q		
0	0	–	–	–	Тесты из условия
1	8	$n \leqslant 100$	$q \leqslant 100$	0	
2	16	$n \leqslant 300\,000$	$q \leqslant 300\,000$	0	Гарантируется, что в запросах второго типа $k \leqslant 50$
3	15	$n \leqslant 300\,000$	$q \leqslant 300\,000$		Нет изменений, в запросах второго типа $k \in \{m, m+1\}$
4	21	–	–	3	Нет изменений
5	14	$n \leqslant 400\,000$	$q \leqslant 400\,000$	3	В запросах второго типа $k \in \{m, m+1\}$
6	11	$n \leqslant 300\,000$	$q \leqslant 300\,000$	0, 1, 2, 3	
7	15	–	–	0 – 6	**Offline-проверка**

Задача D. Прямоугольная квартира

Имя входного файла:	`стандартный ввод` или `input.txt`
Имя выходного файла:	`стандартный вывод` или `output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	1024 мегабайта

Недавно черепашка сняла квартиру. По счастливому стечению обстоятельств она оказалась прямоугольной формы. Квартира разбита на $n \times m$ квадратиков одинакового размера: n строк по m квадратиков в каждой. Строки пронумерованы сверху вниз, а столбцы — слева направо. Обозначим за (i, j) квадрат в i-й строке и j-м столбце.

В некоторых местах квартиры стоит мебель. Описание квартиры задано матрицей a размера $n \times m$:

- Если $a_{i,j} =$ «`#`», то квадратик (i, j) занят мебелью.
- Если $a_{i,j} =$ «`.`», то квадратик (i, j) свободен.

Черепашка долго тренировалась и научилась двигаться не только вправо или вниз, но и вверх тоже. Чтобы закрепить свои успехи в изучении нового движения, она решила делать зарядку каждое утро. Зарядка описывается строкой s и выглядит следующим образом:

1. Черепашка встаёт в квадратик (i, j).
2. Далее для каждого i от 1 до $|s|$ черепашка перемещается в другой квадратик. Предположим, что сейчас она находится в (x, y), тогда:
   - Если $s_i =$ «`D`», черепашка переходит в $(x + 1, y)$.
   - Если $s_i =$ «`R`», черепашка переходит в $(x, y + 1)$.
   - Если $s_i =$ «`U`», черепашка переходит в $(x - 1, y)$.

Само собой, во время зарядки черепашка не может выходить за границы квартиры или стоять в занятом мебелью квадратике. Таким образом, если в какой-то момент черепашка пытается перейти в квадратик, которого не существует, или он занят мебелью, то зарядка проваливается. Квадратик, с которого черепашка начинает делать зарядку, соответственно, тоже обязан быть пустым.

Помогите черепашке найти количество квадратиков (i, j), начиная с которых она сможет выполнить свою зарядку полностью.

Формат входных данных

Первая строка содержит два целых числа n и m ($2 \leqslant n, m \leqslant 500$) — размеры квартиры.

Вторая строка содержит строку s ($1 \leqslant |s| \leqslant 2nm$, $s_i \in \{\texttt{D}, \texttt{R}, \texttt{U}\}$) — описание зарядки.

i-я из следующих n строк содержит $a_{i,1}$, $a_{i,2}$, …, $a_{i,m}$ ($a_{i,j} \in \{\texttt{\#}, \texttt{.}\}$) — описание квартиры черепашки.

Формат выходных данных

Выведите единственное число — количество квадратиков (i, j), начиная с которых она сможет выполнить свою зарядку полностью.

Примеры

стандартный ввод	стандартный вывод
5 6 RDUUR .#.... .....# .#.#.. ....#....	3
4 2 RR	0

Замечание

В первом примере черепашка может выполнить зарядку, начиная с квадратиков $(2, 2)$, $(2, 4)$ и $(4, 4)$.

Если черепашка начинает зарядку в квадратике $(2, 2)$, то её путь выглядит следующим образом: $(2, 2) \to (2, 3) \to (3, 3) \to (2, 3) \to (1, 3) \to (1, 4)$.

Система оценки

Тесты к этой задаче состоят из пяти групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов некоторых из предыдущих групп. Обратите внимание, что прохождение тестов из условия не требуется для некоторых групп.

Группа	Баллы	Доп. ограничения	Необх. группы	Комментарий
0	0	–	–	Тесты из условия
1	17	$n, m \leqslant 50$	0	
2	14	$s_i = \texttt{R}$	–	
3	19	$s_i \in \{\texttt{D}, \texttt{U}\}$	–	
4	23	$s_i \in \{\texttt{R}, \texttt{D}\}$	2	
5	27	–	0 – 4	

Задача E. Простая задача

Имя входного файла:	`стандартный ввод` или `input.txt`
Имя выходного файла:	`стандартный вывод` или `output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	1024 мегабайта

Дано неориентированное дерево из n вершин. Также у каждой вершины v записано целое неотрицательное число a_v ($0 \leqslant a_v < 2^k$).

Множество вершин называется хорошим, если побитовое ИЛИ значений a вершин в этом множестве равно $2^k - 1$.

Стоимостью множества называется максимальное из попарных расстояний между вершинами множества, где расстоянием между вершинами считается количество ребер на единственном простом пути между ними.

Вам нужно найти минимальную стоимость хорошего множества или сказать, что такого нет.

Формат входных данных

Первая строка содержит два целых числа n и k ($2 \leqslant n \leqslant 100\,000$, $1 \leqslant k \leqslant 20$) — количество вершин дерева и число k соответственно.

Вторая строка содержит n целых чисел a_i ($0 \leqslant a_i < 2^k$) — значения вершин.

Следующие $n - 1$ строк описывают ребра дерева.

i-я из них содержит два целых числа v_i и u_i ($1 \leqslant v_i, u_i \leqslant n$) — номера вершин, соединенных i-м ребром.

Формат выходных данных

В случае, если хорошее множество существует, в единственной строке выведите минимальную стоимость хорошего множества.

В случае, если не существует ни одного хорошего множества, выведите -1.

Примеры

стандартный ввод	стандартный вывод
5 3 1 2 6 0 4 1 2 2 3 1 4 3 5	2
3 3 0 1 2 1 2 2 3	-1

Замечание

В первом примере можно выбрать множество вершин $\{1, 2, 3\}$.

Во втором примере максимальное ИЛИ, которое можно получить, равняется 3.

Система оценки

Тесты к этой задаче состоят из девяти групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов некоторых из предыдущих групп. Обратите внимание, что прохождение тестов из условия не требуется для некоторых групп. **Offline-проверка** означает, что результаты тестирования вашего решения на данной группе станут доступны только после окончания соревнования.

Группа	Баллы	Доп. ограничения		Необх. группы	Комментарий
		n	k		
0	0	–	–	–	Тесты из условия
1	12	$n \leqslant 15$	–	0	
2	9	$n \leqslant 1\,000$	–	–	$v_i = i$, $u_i = i + 1$
3	14	–	–	2	$v_i = i$, $u_i = i + 1$
4	6	–	$k = 1$	–	–
5	10	$n \leqslant 1\,000$	$k = 2$	–	–
6	12	–	$k = 2$	5	–
7	9	$n \leqslant 100$	$k \leqslant 5$	0	–
8	16	$n \leqslant 1\,000$	–	0 – 2, 5, 7	–
9	12	–	–	0 – 8	**Offline-проверка**

Задача F. Минимумы на дугах

Имя входного файла:	`стандартный ввод` или `input.txt`
Имя выходного файла:	`стандартный вывод` или `output.txt`
Ограничение по времени:	3 секунды
Ограничение по памяти:	1024 мегабайта

Данную задачу можно решать только на языке программирования C++.

По кругу расположено n различных чисел от 1 до n, в i-й вершине при обходе против часовой стрелки записано число p_i. Для каждой пары значений x и y есть два способа добраться из x в y — пойти по часовой стрелке или против. В этой задаче n всегда нечетное, поэтому эти два пути будут иметь различную длину.

Скажем, что $f(x, y)$ — это минимальное значение, которое мы встретим по пути, если добираться из числа x в число y по кратчайшему из путей.

Например, если $p = [1, 5, 3, 2, 7, 4, 6]$, то значения 5 и 7 соединены двумя путями, и значения вершин на этих путях, соответственно, равны $[5, 3, 2, 7]$ и $[5, 1, 6, 4, 7]$. Кратчайшим является первый указанный путь, а значит $f(5, 7) = \min\{5, 3, 2, 7\} = 2$.

Скажем, что две перестановки p и q — *эквивалентные*, если значение функции f для этих перестановок совпадает при всех x и y. Вам загадана перестановка p, требуется отгадать любую эквивалентную ей перестановку q, задавая запросы к функции $f(x, y)$.

Формат решения

Это необычная задача. Она имеет формат тестирования с грейдером, где вам необходимо реализовать только функцию `guess` с решением. Эта функция будет вызвана тестирующей программой жюри (грейдером), и возвращаемое значение функции будет принято как решение задачи.

В частности, это означает, что в отправленном вами коде **не должно быть ввода или вывода**. Ваш код **не должен** содержать функцию `main`. При необходимости вы можете реализовать произвольное количество вспомогательных функций, структур, классов и глобальных переменных, но весь код вашего решения должен находиться в одном файле.

Вы должны реализовать следующую функцию:

```
std::vector<int> guess(int n);
```

Функция `guess` принимает на вход число n — длину перестановки.

В реализации функции `guess` вы можете использовать функцию `min_value`, которую реализует грейдер. Данная функция принимает на вход два значения x и y, и возвращает $f(x, y)$ в качестве результата. Если сделать некорректный запрос или исчерпать количество запросов, то программа автоматически завершит свою работу.

Для того, чтобы получить доступ к функции `min_value` в решении, первой строчкой вашего кода нужно подключить заголовочный файл следующей строчкой:

```
#include "checkpoint.h"
```

Определение функции `min_value` в заголовочном файле следующее:

```
int min_value(int x, int y);
```

Все параметры (значения x, y, а так же возвращаемая вами перестановка q) заданы в **1-индексации**.

Ваша функция `guess` с помощью вызовов функции `min_value` должна определить неизвестную перестановку p, с точностью до указанной эквивалентности. То есть если жюри загадали перестановку p, а функция `guess` вернула любую эквивалентную ей перестановку q, то такой ответ считается корректным.

За один запуск грейдера **жюри может делать несколько вызовов функции** `guess`, в таком случае функция `min_value` будет возвращать значения $f(i, j)$, определенные для текущей загаданной перестановки p в рамках конкретного вызова функции `guess`.

Грейдер не является адаптивным, то есть перестановки p фиксируются заранее и не зависят от реализации функции `guess`.

Тестирование

Вам предоставлен шаблон решения `checkpoint.cpp`, а так же заголовочный файл `checkpoint.h`, содержащий определение функций `min_value` и `guess`.

Для удобства тестирования вам предоставлен грейдер — файл `grader.cpp`. В этом файле реализован ввод входных данных со стандартного потока ввода, запуск функции `guess` и вывод на стандартный поток вывода возвращаемого значения функции `guess`. В тестирующей системе эти файлы грейдеров могут отличаться.

Чтобы скомпилировать ваш код `checkpoint.cpp` на языке C++, используйте команду

```
g++ -std=c++20 grader.cpp checkpoint.cpp -o grader
```

После выполнения данной команды будет создан исполняемый файл грейдера `grader` или `grader.exe`, в зависимости от вашей операционной системы, запустив который можно ввести тест в формате, указанном далее.

Если компиляция через команды вызывает у вас трудности, для локального тестирования вы можете скопировать реализацию функции `guess` в файл `grader.cpp` и запустить файл `grader.cpp`. При этом перед отправкой решения в тестирующую систему вам нужно будет оставить только реализацию функции `guess`, не забыв подключить заготовочный файл в начале кода.

Формат входных данных

Грейдер читает тест в следующем формате:

В первой строке указано число t ($1 \leqslant t \leqslant 30\,000$) — количество наборов входных данных.

Первая строка каждого набора входных данных содержит **нечетное** число n ($1 \leqslant n \leqslant 30\,000$) — длину перестановки.

Вторая строка содержит n различных чисел $p_1, \ldots, p_n$ ($1 \leqslant p_i \leqslant n$) — загаданную перестановку.

Формат выходных данных

Грейдер выводит результаты функции `guess` — угаданную перестановку для каждого набора входных данных.

В файле `grader.cpp` есть переменная `verbose`, которая исходно равна 0. При увеличении ее значения грейдер будет более подробно писать информацию о вашем решении и его запросах.

Пример

стандартный ввод	стандартный вывод
2 3 1 2 3 5 1 4 2 3 5	queries count => 3 queries count => 10

Система оценки

Тесты к этой задаче состоят из четырех групп.

Баллы за каждую из первых трех групп ставятся только при прохождении всех тестов группы и всех тестов некоторых из предыдущих групп. Балл за последнюю группу равняется минимальному из баллов, полученных за каждый из тестов в четвертой группе.

Для каждого теста обозначим за N сумму n по всем наборам входных данных, за Q — ограничение на суммарное количество вызовов функции `min_value` по всем наборам входных данных.

Группа	Баллы	Доп. ограничения		Необх. группы	Комментарий
		N	Q		
0	0	–	$Q = 4950$	–	Тесты из условия
1	10	$N \leqslant 100$	$Q = 4950$		$p_{2i-1} = \frac{n-1}{2} + i$ для $1 \leqslant i \leqslant \frac{n+1}{2}$
2	20	$N \leqslant 100$	$Q = 4950$	0, 1	
3	20	$N \leqslant 1\,000$	$Q = 100\,000$	0, 1, 2	
4	60	$N \leqslant 30\,000$	$Q = 1\,000\,000$	–	

При этом последняя группа оценивается по формуле. Если в тесте вы сделали x запросов, то ваш балл будет равен:

$$\text{score} = \min\left(60,\ \left\lfloor 95 \cdot \left(1 - \frac{x}{10^6}\right)\right\rfloor\right)$$

Ваш балл за последнюю группу равняется минимальному из полученных баллов на тестах четвертой группы.

Задача G. «Титаник»

Имя входного файла:	`стандартный ввод` или `input.txt`
Имя выходного файла:	`стандартный вывод` или `output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	1024 мегабайта

Андрей очень любит игровые автоматы, в этот раз он нашел новинку — игровой автомат «Титаник». В нем есть n шлюпок, расположенные на плоскости, i-я шлюпка находится в точке с координатами (x_i, y_i). Так же есть спасательная лодка, которая после запуска игрового автомата начинает плыть с постоянной скоростью из точки A в точку B.

На спасательной лодке есть крюк, способный выстреливать перпендикулярно движению лодки в обе стороны. Крюк выстреливает с невероятно быстрой скоростью, и если на пути крюка оказывается шлюпка, то он ее цепляет и начинает тащить к лодке. Сматывается крюк не быстро — если расстояние между вашей лодкой и шлюпкой в момент выстрела и зацепа было равно d, то спасательная лодка проплывет расстояние d, пока крюк будет сматываться. Пока крюк сматывается, выстреливать, чтобы спасти другие шлюпки, нельзя.

Изначально ваш счет спасенных шлюпок равен 0. В момент, когда крюк смотается полностью, ваш счет увеличится на 1, и вы сможете снова выстреливать им. В момент, когда лодка достигает точки B, игра заканчивается. Если ровно в этот момент крюк успевает смотаться, то спасенная шлюпка засчитывается, если же крюк не успевает смотаться полностью — шлюпка не считается спасенной, и счет не увеличивается.

Андрей задумался, нет ли в автоматах подвоха, и захотел вычислить максимальный счет, который можно в нем набрать. К сожалению, Андрей решил это выяснить опытным путем, помогите ему — вычислите для автомата максимальный счет, который в нем можно набрать, пока Андрей не потратил все ваши жетоны!

Формат входных данных

Каждый тест состоит из нескольких наборов входных данных. Первая строка содержит единственное целое число t ($1 \leqslant t \leqslant 10\,000$) — количество наборов входных данных. Далее следует описание наборов входных данных.

Первая строка каждого набора входных данных содержит целое число n ($1 \leqslant n \leqslant 200\,000$) — количество шлюпок в автомате.

Далее следуют n строк. i-я из них содержит два целых числа x_i, y_i ($-10^9 \leqslant x_i, y_i \leqslant 10^9$) — координаты i-й шлюпки.

В предпоследней строке каждого набора содержатся два целых числа a_x, a_y ($-10^9 \leqslant a_x, a_y \leqslant 10^9$) — координаты точки A.

В последней строке каждого набора содержатся два целых числа b_x, b_y ($-10^9 \leqslant b_x, b_y \leqslant 10^9$) — координаты точки B.

Гарантируется, что точки A и B не совпадают. Гарантируется, что все шлюпки в каждом наборе входных данных находятся в попарно различных точках.

Обозначим за N сумму n по всем наборам входных данных. Гарантируется, что N не превосходит 200 000.

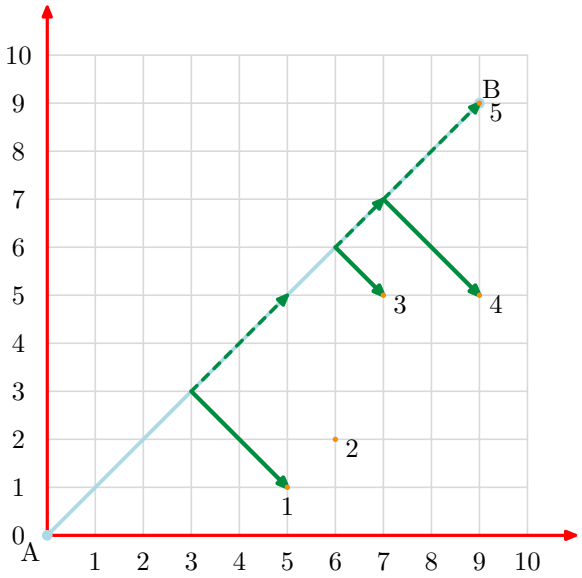
Формат выходных данных

Для каждого набора входных данных выведите максимальный счет, который можно набрать в автомате.

Пример

стандартный ввод	стандартный вывод
2 5 5 1 6 2 7 5 9 5 9 9 0 0 9 9 2 0 1 -1 5 0 0 -5 0	4 1

Замечание



В первом тестовом случае наша лодка плывет из точки $(0, 0)$ в точку $(9, 9)$. Мы подбираем 1-ю шлюпку, находясь в $(3, 3)$, и тянем, пока не окажемся $(5, 5)$. Далее в $(6, 6)$ мы захватываем 3-ю шлюпку и тянем ее, пока не доплывем до $(7, 7)$. Потом мы сразу же захватываем 4-ю шлюпку и тянем ее до $(9, 9)$. В конечной точке мы подбираем 5-ю шлюпку и сразу же ее захватываем. После этого автомат заканчивает работу, и мы имеем счет равный 4.

Во втором тестовом случае наша лодка плывет из точки $(0, 0)$ в точку $(-5, 0)$. В точке $(0, 0)$ зацепляем 1-ю шлюпку и тянем ее к себе до точки $(-1, 0)$, после этого мы можем попытаться зацепить 2-ю шлюпку, но нам не хватит расстояния ее к себе подтянуть — автомат закончит игру раньше, поэтому итоговый счет 1.

Система оценки

Тесты к этой задаче состоят из пяти групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов некоторых из предыдущих групп. Обратите внимание, что прохождение тестов из условия не требуется для некоторых групп.

Группа	Баллы	Доп. ограничения	Необх. группы	Комментарий
		N		
0	0	–	–	Тесты из условия
1	21	$N \leqslant 4000$	–	$a_y = b_y = 0$
2	20	$N \leqslant 4000$	0, 1	
3	32	$N \leqslant 100\,000$	1	$a_y = b_y = 0$
4	14	$N \leqslant 100\,000$	0–3	
5	13	–	0–4	

Задача H. Играем в го

Имя входного файла:	`стандартный ввод` или `input.txt`
Имя выходного файла:	`стандартный вывод` или `output.txt`
Ограничение по времени:	4 секунды
Ограничение по памяти:	1024 мегабайта

Дано n точек на плоскости. Часть из них белые, часть чёрные. Никакие две точки не совпадают. Никакие три точки не лежат на одной прямой.

Вам требуется подвинуть одну из белых точек на расстояние $\leqslant r$, таким образом, чтобы максимизировать площадь выпуклой оболочки множества точек, которое у вас получится после этой операции.

Формат входных данных

Каждый тест состоит из нескольких наборов входных данных. В первой строке находится одно целое число t ($1 \leqslant t \leqslant 1\,000$) — количество наборов входных данных. Далее следует описание наборов входных данных.

Первая строка каждого набора входных данных содержит два целых числа n и r ($3 \leqslant n \leqslant 2\,000$, $1 \leqslant r \leqslant 10^9$) — количество точек и максимальное расстояние для сдвига точки.

Следующие n строк описывают точки. i-я из них содержит три целых числа x_i, y_i и c_i ($-10^9 \leqslant x_i, y_i \leqslant 10^9$, $1 \leqslant c_i \leqslant 2$) — координаты и цвет точки i (1 — белый, 2 — черный).

Гарантируется, что сумма n по всем наборам входных данных в каждом тесте не превосходит $2\,000$.

Формат выходных данных

Для каждого набора входных данных в отдельной строке выведите одно число — максимальную площадь выпуклой оболочки этого множества точек, если любую из белых можно сдвинуть не больше чем на r.

Ваш ответ будет считаться правильным, если его абсолютная или относительная ошибка не превосходит 10^{-6}.

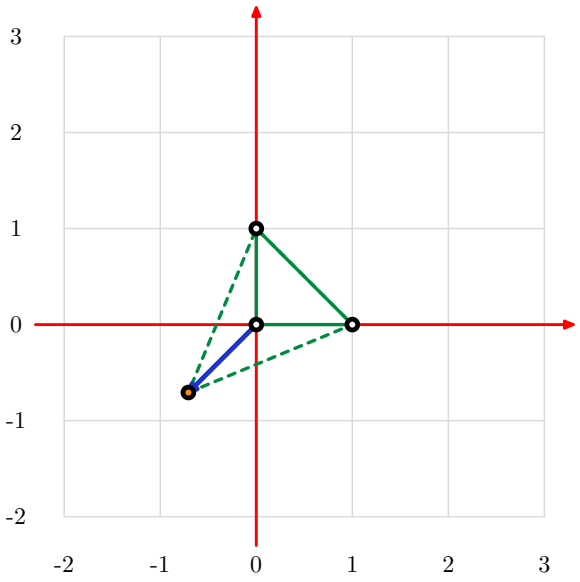
Формально, пусть ваш ответ равен a, а ответ жюри равен b. Ваш ответ будет зачтен, если и только если $\frac{|a-b|}{\max(1,|b|)} \leqslant 10^{-6}$.

Примеры

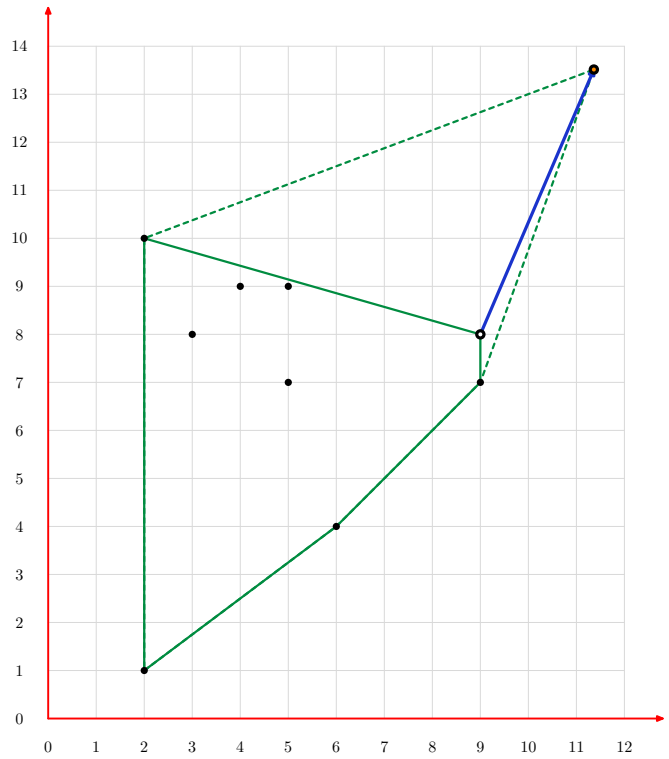
стандартный ввод	стандартный вывод
1 3 1 0 0 1 1 0 1 0 1 1	1.20710678118655
1 9 6 2 10 2 5 7 2 9 8 1 6 4 2 4 9 2 2 1 2 9 7 2 5 9 2 3 8 2	59.34731931759172
1 9 1 8 7 1 1 9 1 3 9 1 4 2 1 7 4 1 10 5 1 3 7 1 4 4 1 7 6 1	37.42442890089805

Замечание

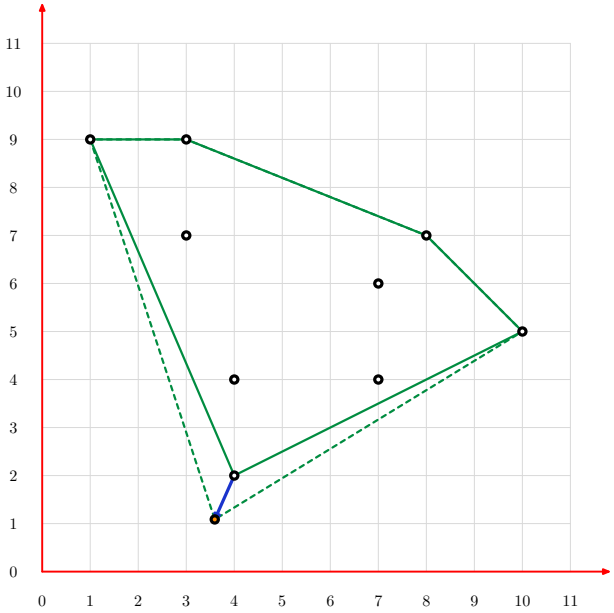
Ниже приведены картинки для примеров из условия. Пунктирной линией обозначены ребра новой выпуклой оболочки, сплошной — ребра старой выпуклой оболочки и общие ребра. Синей стрелкой показано оптимальное движение белой точки.



Пример 1



Пример 2



Пример 3

Система оценки

Тесты к этой задаче состоят из девяти групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов некоторых из предыдущих групп. Обратите внимание, что прохождение тестов из условия может не требоваться для некоторых групп. **Offline-проверка** означает, что результаты тестирования вашего решения на данной группе станут доступны только после окончания соревнования.

Обозначим за $\sum n$ — сумму n по всем наборам входных данных данного теста.

Группа	Баллы	Доп. ограничения	Необх. группы	Комментарий
		$\sum n$		
0	0	–	–	Тесты из условия
1	11	$\sum n \leqslant 3$	0	
2	18	$\sum n \leqslant 30$	0, 1	
3	6	$\sum n \leqslant 50$	0, 1, 2	
4	13	$\sum n \leqslant 200$	–	Все точки на выпуклой оболочке черные
5	14	$\sum n \leqslant 200$	0 – 4	
6	8	$\sum n \leqslant 500$	4	Все точки на выпуклой оболочке черные
7	7	$\sum n \leqslant 500$	0 – 6	
8	12	$\sum n \leqslant 2000$	4, 6	Все точки на выпуклой оболочке черные. **Offline-проверка**
9	11	$\sum n \leqslant 2000$	0 – 8	**Offline-проверка**

Задача I. Переезд

Имя входного файла:	`стандартный ввод` или `input.txt`
Имя выходного файла:	`стандартный вывод` или `output.txt`
Ограничение по времени:	3 секунды
Ограничение по памяти:	1024 мегабайта

Рассмотрим город, представляющий собой таблицу из n строк и m столбцов. На пересечении каждой строки и каждого столбца построен дом.

В доме (i, j) на пересечении строки i и столбца j изначально жило $a_{(i,j)}$ людей. На следующий год каждый человек переехал из своего дома в какой-то другой (или, возможно, остался жить там же). Известно, что на следующий год в доме (i, j) стало проживать $b_{(i,j)}$ людей.

Требуется вывести минимальное число x, такое что люди могли переехать так, что расстояние между изначальным и итоговым домом каждого человека не превышало x. Расстояние между клетками (x_1, y_1) и (x_2, y_2) равно $|x_1 - x_2| + |y_1 - y_2|$.

Формат входных данных

Каждый тест состоит из нескольких наборов входных данных. В первой строке находится одно целое число t ($1 \leqslant t \leqslant 100$) — количество наборов входных данных. Далее следует описание наборов входных данных.

Первая строка каждого набора входных данных содержит два целых числа n и m ($1 \leqslant n \leqslant 3$, $1 \leqslant m \leqslant 100\,000$) — количество строк и столбцов таблицы соответственно.

В следующих n строках описывается исходное количество жильцов в домах. i-я из них содержит m целых чисел $a_{(i,1)}, a_{(i,2)}, \ldots a_{(i,m)}$ ($0 \leqslant a_{(i,j)} \leqslant 10^9$) — изначальное количество людей в каждом доме.

В следующих n строках описывается количество жильцов в домах после переезда. i-я из них содержит m целых чисел $b_{(i,1)}, b_{(i,2)}, \ldots b_{(i,m)}$ ($0 \leqslant b_{(i,j)} \leqslant 10^9$) — количество людей в каждом доме после переезда.

Гарантируется, что сумма значений $a_{(i,j)}$ равняется сумме значений $b_{(i,j)}$.

Обозначим за M сумму m по всем наборам входных данных. Гарантируется, что M не превосходит 100 000.

Формат выходных данных

Для каждого набора входных данных выведите минимальное число x, такое что люди могли переехать так, чтобы расстояние между изначальным и итоговым домом при переезде каждого человека не превышало x.

Пример

стандартный ввод	стандартный вывод
1 2 5 0 4 0 4 0 0 0 0 0 0 1 1 1 1 1 0 1 1 1 0	2

Замечание

В примере из дома $(1, 2)$ люди переезжают в $(1, 1), (1, 2), (1, 3), (2, 2)$, а люди из дома $(1, 4)$ переезжают в $(1, 4), (1, 5), (2, 3), (2, 4)$. Максимальное расстояние равняется двум.

Система оценки

Тесты к этой задаче состоят из двенадцати групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов некоторых из предыдущих групп. Обратите внимание, что прохождение тестов из условия может не требоваться для некоторых групп. **Offline-проверка** означает, что результаты тестирования вашего решения на данной группе станут доступны только после окончания соревнования.

Обозначим за S суммарное количество людей в городе (сумма элементов любой из таблиц), за A количество ненулевых значений a_{ij}, и за B количество ненулевых значений b_{ij}.

Группа	Баллы	Доп. ограничения				Необх. группы	Комментарий
		M	S	A, B	n		
0	0	–	–	–	–	–	Тесты из условия
1	8	–	$S \leqslant 7$	–	–	–	
2	9	–	$S \leqslant 50$	–	–	1	
3	8	–	–	$A, B \leqslant 13$	–	1	
4	7	–	–	$A \leqslant 13$	–	1, 3	
5	6	$m \leqslant 50$, $M \leqslant 5\,000$	–	–	–	–	
6	10	$M \leqslant 5\,000$	–	–	–	5	
7	8	$M \leqslant 50\,000$	–	–	$n \leqslant 1$	—	
8	11	$M \leqslant 50\,000$	–	–	$n \leqslant 2$	7	
9	5	$M \leqslant 50\,000$	–	–	–	—	Ответ не превышает 2
10	6	$M \leqslant 50\,000$	–	–	–	9	Ответ не превышает 3
11	12	$M \leqslant 50\,000$	–	–	–	5–10	
12	10	—	–	–	–	1–11	**Offline-проверка**