

## Задача 1. Разность квадратов

Вы участвуете в разработке программного модуля для системы символьных вычислений. Модуль будет использоваться для решения специального вида диофантовых уравнений.

По заданному целому неотрицательному целому числу  $n$  разрабатываемый модуль должен найти два натуральных числа  $x$  и  $y$ , для которых выполнено равенство  $x^2 - y^2 = n$ . Найденные числа не должны превышать  $2^{62} - 1$ .

Требуется написать программу, которая по заданному целому неотрицательному числу  $n$  находит натуральные числа  $x$  и  $y$ , не превышающие  $2^{62} - 1$ , разность квадратов которых равна  $n$ .

### Формат входных данных

В единственной строке дано одно целое число  $n$  ( $0 \leq n \leq 2^{60}$ ).

Обратите внимание, что заданное во вводе число не помещается в 32-битный тип данных, необходимо использовать 64-битный тип данных (например, «long long» в C++, «int64» в паскале, «long» в Java).

### Формат выходных данных

Если искомые  $x$  и  $y$  существуют, то необходимо вывести две строки: в первой строке выведите слово «Yes», а во второй — искомые  $x$  и  $y$ .

Если подходящих пар  $x$  и  $y$  несколько, можно вывести любую из них, но должно выполняться условие  $1 \leq x, y \leq 2^{62} - 1$ .

Если решения нет, в единственной строке необходимо вывести слово «No».

### Система оценивания

Баллы за каждую подзадачу начисляются только в случае, если все тесты для этой подзадачи и необходимых подзадач успешно пройдены.

Подзадача	Баллы	Ограничения	Необходимые подзадачи	Информация о проверке
1	10	$0 \leq n \leq 2^{10}$		полная
2	20	$0 \leq n \leq 2^{20}$	1	полная
3	30	$0 \leq n \leq 2^{30}$	1, 2	полная
4	40	$0 \leq n \leq 2^{60}$	1, 2, 3	полная

### Примеры

стандартный ввод	стандартный вывод
3	Yes 2 1
2	No

## Задача 2. Превышение скорости

Превышение скорости является опасным нарушением, значительно увеличивающим вероятность трагических последствий транспортных происшествий. К сожалению контроль скорости с использованием радаров и камер не решает проблему полностью. Притормаживая перед камерами, водители едут со значительным превышением на участках дорог, где контроль не ведётся. С целью предотвращения такого поведения используется назначение штрафа за гарантированное превышение скорости, основанное на времени проезда дороги.

Рассмотрим дорогу, состоящую из  $n$  участков, пронумерованных от 1 до  $n$ . Длина  $i$ -го участка составляет  $l_i$  метров. На  $i$ -м из участков установлено ограничение по скорости в  $v_i$  м/с.

За превышение скорости предусмотрены штрафы. В зависимости от превышения, установлены различные штрафы, величина штрафа вычисляется следующим образом.

Пусть  $e$  — максимальное превышение разрешённой скорости в процессе пребывания автомобиля на всей дороге, то есть максимальная разница между скоростью автомобиля и максимальной разрешённой скоростью на участке, где он в этот момент находится. Если превышения скорости не было, то штраф не взимается. В противном случае штраф вычисляется так:

- если  $0 < e \leq a_1$ , то штраф составляет  $f_1$  денежных единиц;
- если  $a_1 < e \leq a_2$ , то штраф составляет  $f_2$  денежных единиц;
- ...
- если  $a_{m-2} < e \leq a_{m-1}$ , то штраф составляет  $f_{m-1}$  денежных единиц;
- если  $a_{m-1} < e$ , то штраф составляет  $f_m$  денежных единиц.

Таким образом, есть  $m$  диапазонов превышения скорости и соответствующие им штрафы.

Автоматическая система назначения штрафов получила данные о  $q$  автомобилях. Для удобства пронумеруем их от 1 до  $q$ . Известно, что  $i$ -й автомобиль заехал на дорогу в момент времени  $s_i$ , проехал все  $n$  участков, после чего выехал с нее в момент времени  $t_i$ . Отсчёт времени будем вести в секундах с открытия дороги.

Для каждого из автомобилей система должна определить, какой максимальный штраф можно гарантированно выписать этому автомобилю, основываясь только на времени заезда на дорогу и выезда с нее.

Требуется написать программу, которая по описанию границ диапазонов превышения скорости, соответствующих штрафов и временам въезда/выезда автомобилей определяет для каждого автомобиля максимальный штраф, который можно выписать этому автомобилю.

### Формат входных данных

Первая строка входных данных содержит единственное целое число  $n$  — количество участков на дороге ( $1 \leq n \leq 10$ ).

Вторая строка содержит  $n$  целых чисел  $v_i$  — ограничения скорости на участках ( $1 \leq v_i \leq 10^9$ ).

Третья строка содержит  $n$  целых чисел  $l_i$  — длины участков ( $1 \leq l_i \leq 10^9$ ).

Четвертая строка содержит единственное целое число  $m$  — количество границ диапазонов превышения скорости ( $1 \leq m \leq 10^5$ ).

Пятая строка содержит  $m - 1$  целых чисел  $a_i$  — границы диапазонов превышения скорости ( $1 \leq a_i \leq 10^9$ ). Гарантируется, что значения  $a_i$  строго возрастают. Обратите внимание, что если  $m = 1$ , то пятая строка ввода пустая.

Шестая строка содержит  $m$  целых чисел  $f_i$  — штрафы за диапазоны превышения скоростей ( $1 \leq f_i \leq 10^9$ ). Гарантируется, что значения  $f_i$  возрастают.

Седьмая строка содержит единственное целое число  $q$  — количество автомобилей, которые надо обработать ( $1 \leq q \leq 10^5$ ).

Каждая из следующих  $q$  строк содержит два целых числа  $s_i$  и  $t_i$  — время заезда на трассу и выезда с неё  $i$ -го из рассматриваемых автомобилей ( $1 \leq s_i < t_i \leq 10^9$ ).

## Формат выходных данных

Для каждого из  $q$  автомобилей выведите в отдельной строке максимальный штраф, который гарантированно можно выписать этому автомобилю, основываясь только на временах его заезда на дорогу и выезда с нее. Если возможна ситуация, что автомобиль ни разу не превысил разрешённую скорость, следует вывести 0.

Гарантируется, что если время въезда или выезда автомобиля изменить не более чем на  $10^{-5}$ , штраф, который можно ему выписать, не изменится.

## Система оценивания

Баллы за каждую подзадачу начисляются только в случае, если все тесты для этой подзадачи и необходимых подзадач успешно пройдены.

Подзадача	Баллы	Дополнительные ограничения	Необходимые подзадачи	Информация о проверке
1	5	$n = 1, m = 1$		первая ошибка
2	10	$m = 1$	1	первая ошибка
3	9	$n = 1, m \leq 10$	1	первая ошибка
4	12	$n = 1$	1, 3	первая ошибка
5	13	$m \leq 10, a_i \leq 10$		первая ошибка
6	14	$m \leq 10$	1, 2, 3, 5	первая ошибка
7	37		1 – 6	первая ошибка

## Пример

стандартный ввод	стандартный вывод
3	0
10 20 30	800
400 500 600	600
6	
1 5 10 12 16	
100 300 600 800 1000 1500	
3	
10 100	
20 70	
45 100	

## Задача 3. Борьба с рутинной

Важным элементом повышения эффективности работы сотрудников является борьба с рутинной. Построим математическую модель разнообразия типов заданий, выполняемых сотрудником в компании.

Рассмотрим работу сотрудника в течение  $n$  последовательных рабочих дней. Будем считать, что каждый день сотрудник выполняет ровно один тип заданий, обозначим тип заданий, выполняемый сотрудником в  $i$ -й день, целым числом  $a_i$ .

Для оценки рутинности работы сотрудника будем использовать следующую характеристику. Зафиксируем целое число  $d$  и рассмотрим все отрезки из  $d$  подряд идущих рабочих дней. Для каждого такого отрезка найдём количество различных типов заданий, которые работник выполнял на протяжении этих дней, и просуммируем эти значения. Полученную величину обозначим как  $S_d$  и будем называть её  $d$ -разнообразием. Чем  $d$ -разнообразие выше, тем больше различных типов заданий выполнял сотрудник. *Профилем вариативности* сотрудника будем называть массив значений  $[S_1, S_2, \dots, S_n]$ .

Требуется написать программу, которая по заданной последовательности  $a_1, a_2, \dots, a_n$  типов выполняемых сотрудником заданий вычисляет его профиль вариативности.

### Формат входных данных

В первой строке находится единственное целое число  $n$  — количество последовательных рабочих дней, которые необходимо проанализировать ( $1 \leq n \leq 2 \cdot 10^5$ ).

Во второй строке находится  $n$  целых чисел  $a_1, a_2, \dots, a_n$  — типы заданий, которое выполнял сотрудник ( $1 \leq a_i \leq 10^9$ ).

### Формат выходных данных

Выведите  $n$  целых чисел:  $S_1, S_2, \dots, S_n$ .

### Система оценивания

Баллы за каждую подзадачу начисляются только в случае, если все тесты для этой подзадачи и необходимых подзадач успешно пройдены.

Подзадача	Баллы	Ограничения	Необходимые подзадачи	Информация о проверке
1	12	$1 \leq n \leq 50, 1 \leq a_i \leq 50$		первая ошибка
2	10	$1 \leq n \leq 50, 1 \leq a_i \leq 10^9$	1	первая ошибка
3	10	$1 \leq n \leq 500, 1 \leq a_i \leq 10^9$	1, 2	первая ошибка
4	12	$1 \leq n \leq 5000, 1 \leq a_i \leq 5000$	1	первая ошибка
5	10	$1 \leq n \leq 5000, 1 \leq a_i \leq 10^9$	1 – 4	первая ошибка
6	16	$1 \leq n \leq 2 \cdot 10^5, 1 \leq a_i \leq 50$	1	первая ошибка
7	30	$1 \leq n \leq 2 \cdot 10^5, 1 \leq a_i \leq 10^9$	1 – 6	первая ошибка

### Примеры

стандартный ввод	стандартный вывод
5 1 3 2 1 2	5 8 8 6 3
3 10 10 10	3 2 1

## Замечание

Рассмотрим, как вычисляются значения  $S_d$  в первом примере.

**1-разнообразие:** необходимо просуммировать количество различных типов заданий, выполняемых сотрудником по всем отрезкам, состоящим из одного дня.

Отрезок дней	Типы заданий	Количество различных
1 – 1	1	1
2 – 2	3	1
3 – 3	2	1
4 – 4	1	1
5 – 5	2	1

Значение 1-разнообразия равно  $S_1 = 1 + 1 + 1 + 1 + 1 = 5$ .

**2-разнообразие:** необходимо просуммировать количество различных типов заданий, выполняемых сотрудником по всем отрезкам, состоящим из двух дней.

Отрезок дней	Типы заданий	Количество различных
1 – 2	1, 3	2
2 – 3	3, 2	2
3 – 4	2, 1	2
4 – 5	1, 2	2

Значение 2-разнообразия равно  $S_2 = 2 + 2 + 2 + 2 = 8$ .

**3-разнообразие:** необходимо просуммировать количество различных типов заданий, выполняемых сотрудником по всем отрезкам, состоящим из трех дней.

Отрезок дней	Типы заданий	Количество различных
1 – 3	1, 3, 2	3
2 – 4	3, 2, 1	3
3 – 5	2, 1, 2	2

Значение 3-разнообразия равно  $S_3 = 3 + 3 + 2 = 8$ .

**4-разнообразие:** необходимо просуммировать количество различных типов заданий, выполняемых сотрудником по всем отрезкам, состоящим из четырех дней.

Отрезок дней	Типы заданий	Количество различных
1 – 4	1, 3, 2, 1	3
2 – 5	3, 2, 1, 2	3

Значение 4-разнообразия равно  $S_4 = 3 + 3 = 6$ .

**5-разнообразие:** необходимо просуммировать количество различных типов заданий, выполняемых сотрудником по всем отрезкам, состоящим из пяти дней.

Отрезок дней	Типы заданий	Количество различных
1 – 5	1, 3, 2, 1, 2	3

Значение 5-разнообразия равно  $S_5 = 3$ .

## Задача 4. Олимпиада для роботов

Жюри чемпионата по скоростному вычислению булевых функций среди роботов готовит задания для участников.

Задание для роботов представляет собой таблицу из  $m$  строк и  $n$  столбцов, каждая ячейка которой содержит целое число. Обозначим число в  $i$ -й строке,  $j$ -м столбце таблицы как  $x_{i,j}$ . В каждом столбце значения в ячейках таблицы образуют перестановку чисел от 0 до  $m - 1$ . Иначе говоря, числа в каждом столбце различны: если  $i \neq k$ , то  $x_{i,j} \neq x_{k,j}$  для всех  $j$ , и выполнено условие  $0 \leq x_{i,j} < m$ .

Для каждого столбца таблицы задаётся значение порога — целое неотрицательное число  $z_j$  от 0 до  $m$ . В качестве аргументов булевых функций, которые будут вычислять участники олимпиады, используются значения логических выражений  $x_{i,j} < z_j$ . Значение такого логического выражения равно 1, если неравенство выполнено, иначе оно равно 0.

В процессе соревнования участники вычисляют значения  $m$  булевых функций — по одному для каждой строки. Каждая булева функция задаётся в виде *бесповторной монотонной линейной программы* (БМЛП).

Рассмотрим БМЛП для  $i$ -й строки таблицы. Она представляет собой последовательность из  $n - 1$  инструкции, пронумерованных от 1 до  $n - 1$ ,  $p$ -я инструкция задаётся тремя числами:  $a_p$ ,  $b_p$  и  $op_p$ . Число  $op_p$  принимает два возможных значения: 1 для операции **and** — логическое «и», 2 для операции **or** — логическое «или». Числа  $a_p$  и  $b_p$  являются номерами аргументов  $p$ -й инструкции, выполнены неравенства  $1 \leq a_p, b_p < n + p$ .

Рассмотрим массив  $val[1..2n - 1]$ , каждое из значений которого равно 0 или 1. Проинициализируем значения  $val[1..val[n]]$  с использованием порогов,  $val[j] = 1$ , если  $x_{i,j} < z_j$ , иначе  $val[j] = 0$ . Значение  $val[n + p]$  вычисляется с использованием  $p$ -й инструкции.

- Если  $op_p = 1$ , то  $val[n + p] = (val[a_p] \text{ and } val[b_p])$ , то есть значение  $val[n + p]$  равно 1 если и только если каждое из значений  $val[a_p]$  и  $val[b_p]$  равно 1.
- Если  $op_p = 2$ , то  $val[n + p] = (val[a_p] \text{ or } val[b_p])$ , то есть значение  $val[n + p]$  равно 1 если и только если хотя бы одно из значений  $val[a_p]$  и  $val[b_p]$  равно 1.

При этом программа является бесповторной, а именно все  $2n - 2$  значений  $a_p$  и  $b_p$  для  $p$  от 1 до  $n - 1$  различны. Иначе говоря,  $a_p \neq b_p$ , а если  $p \neq q$ , то  $a_p \neq a_q$ ,  $a_p \neq b_q$ ,  $b_p \neq a_q$  и  $b_p \neq b_q$ .

Результатом исполнения программы является значение  $val[2n - 1]$ .

Жюри олимпиады подготовило таблицу  $x_{i,j}$ , выбрало булевы функции для каждой строки и записало их в виде БМЛП. Теперь осталось выбрать значение порога для каждого столбца, чтобы получить задание для олимпиады. Жюри считает задание сбалансированным, если ровно  $s$  из  $m$  программ для строк таблицы возвращают единицу, а остальные  $m - s$  возвращают ноль. Ваша задача — помочь жюри найти подходящие значения порогов.

Требуется написать программу, которая по заданным значениям в ячейках таблицы и БМЛП для строк таблицы определяет такие значения порогов  $z_j$ , чтобы значение ровно  $s$  из  $m$  заданных функций было равно 1. Можно доказать, что при описанных в условии задачи ограничениях требуемые значения порогов всегда можно подобрать.

### Формат входных данных

В первой строке входных данных заданы целые числа  $n$ ,  $m$  и  $s$  ( $1 \leq n \leq 3 \cdot 10^5$ ,  $1 \leq m \leq 3 \cdot 10^5$ ,  $n \cdot m \leq 3 \cdot 10^5$ ,  $0 \leq s \leq m$ ).

Далее следует  $m$  блоков по  $n - 1$  строке в каждом, каждый блок задает бесповторную монотонную линейную программу для одной строки таблицы. В каждом блоке  $p$ -я строка содержит 3 целых числа:  $a_p$ ,  $b_p$  и  $op_p$  ( $1 \leq a_p < n + p$ ,  $1 \leq b_p < n + p$ , гарантируется, что в одном блоке все значения  $a_p$  и  $b_p$  попарно различны,  $op_p = 1$  или  $op_p = 2$ ).

Последние  $m$  строк задают таблицу,  $i$ -я строка содержит  $n$  целых чисел,  $j$ -е из которых равно  $x_{i,j}$  ( $0 \leq x_{i,j} \leq m - 1$ , в каждом столбце все числа различны, то есть если  $i \neq k$ , то  $x_{i,j} \neq x_{k,j}$  для всех  $j$ ).

## Формат выходных данных

Выведите  $n$  целых чисел — искомые значения порогов  $z_1, z_2, \dots, z_n$  ( $0 \leq z_j \leq m$ ). Если подходящих вариантов несколько, выведите любой из них.

## Система оценивания

Баллы за каждую подзадачу начисляются только в случае, если все тесты для этой подзадачи и необходимых подзадач успешно пройдены.

Подз.	Баллы	Дополнительные ограничения	Необходимые подзадачи	Информация о проверке
1	10	$n \leq 2, m \leq 10^3$		первая ошибка
2	10	$n \leq 2, m \leq 10^5$	1	первая ошибка
3	10	$n \leq 10, m \leq 2$		первая ошибка
4	5	$x_{i,j} = i - 1$		первая ошибка
5	5	$op_p = 1$ , только операции «и»		первая ошибка
6	20	$n \leq 100$	1, 2, 3	первая ошибка
7	10	БМЛП для всех строк одинаковые		первая ошибка
8	30	нет	1 – 7	первая ошибка

## Пример

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

## Замечание

В примере в таблице три строки, каждой соответствует формула. Необходимо найти четыре порога так, чтобы ровно две формулы возвращали 1, а оставшаяся — 0.

Рассмотрим, как будет вычисляться массив  $val$  для первой строки.

Первые четыре значения вычисляются на основе чисел в этой строке и порогов:

- $val[1] = (x_{1,1} < z_1) = (0 < 0) = 0$ ;
- $val[2] = (x_{1,2} < z_2) = (1 < 1) = 0$ ;
- $val[3] = (x_{1,3} < z_3) = (2 < 2) = 0$ ;
- $val[4] = (x_{1,4} < z_4) = (2 < 3) = 1$ .

Далее выполняем линейную программу для первой строки:

- $val[5] = (val[1] \text{ and } val[2]) = (0 \text{ and } 0) = 0$ ;

- $val[6] = (val[3] \text{ and } val[4]) = (0 \text{ and } 1) = 0$ ;
- $val[7] = (val[5] \text{ or } val[6]) = (0 \text{ or } 0) = 0$ .

Таким образом значение булевой функции для первой строки равно 0. Кстати, если эту функцию записать формулой, то получится:

$$((x_{1,1} < z_1) \text{ and } (x_{1,2} < z_2)) \text{ or } ((x_{1,3} < z_3) \text{ and } (x_{1,4} < z_4)).$$

Аналогично, булева функция для второй строки равна:

$$(((x_{2,1} < z_1) \text{ or } (x_{2,2} < z_2)) \text{ and } (x_{2,3} < z_3)) \text{ or } (x_{2,4} < z_4),$$

а для третьей строки:

$$((x_{3,1} < z_1) \text{ and } (x_{3,4} < z_4)) \text{ or } ((x_{3,2} < z_2) \text{ and } (x_{3,3} < z_3)).$$

При подстановке порогов  $z_1 = 0$ ,  $z_2 = 1$ ,  $z_3 = 2$ ,  $z_4 = 3$  получим следующие выражения.

Вторая строка:

$$(((2 < 0) \text{ or } (2 < 1)) \text{ and } (1 < 2)) \text{ or } (0 < 3) = (((0 \text{ or } 0) \text{ and } 1) \text{ or } 1) = (0 \text{ or } 1) = 1,$$

Третья строка:

$$(((1 < 0) \text{ and } (1 < 3)) \text{ or } ((0 < 1) \text{ and } (0 < 2))) = ((0 \text{ and } 1) \text{ or } (1 \text{ and } 1)) = (0 \text{ or } 1) = 1.$$

Заметим, что это не единственный подходящий набор порогов, также подойдут, например, значения  $z_1 = 0$ ,  $z_2 = 0$ ,  $z_3 = 3$ ,  $z_4 = 3$ .