Задача A. Странная программа

Имя входного файла:	`awful.in`
Имя выходного файла:	`awful.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

При работе в НИИ Данных Строк Вася столкнулся с большими сложностями, ибо исходники хеширования паролей, состоящих из целых чисел, были написаны на неизвестном для него языке программирования.

```
[define[reversed_range n][if [< n 1] '[] [cons n [reversed_range [- n 1]]]]]
[define [range n] [reverse [reversed_range n]]]
; [range n] - список целых чисел от 1 до n
[define [f l]
  [if [null? [cdr l]]
      [car l]
      [f [append [cdr[cdr l]] [list [car l]]]]
  ]
]
[define [hash n] [f [range n]]]
```

Васе жизненно необходимо уметь вычислять `[hash n]`. Помогите ему разобраться с языком программирования *Awful*. Предполагается, что у интерпретатора достаточно стековой памяти, чтобы вычислить `[hash n]` для всех n.

Выражение	*Результат*	*Выражение*	*Результат*
`[- 10 1]`	9	`[list 1]`	`[1]`
`[car '[1 2 3]]`	1	`[cdr '[1]]`	`[]`
`[cdr '[1 2 3]]`	`[2 3]`	`[cons 1 '[2 3]]`	`[1 2 3]`
`[append '[1 2] '[3 4]]`	`[1 2 3 4]`	`[reverse[range 5]]`	`[5 4 3 2 1]`
`[cdr[cdr[range 5]]]`	`[3 4 5]`	`[if [< 0 1] 1 0]`	1
`[if [null? '[1]] 1 0]`	0	`[if [null? '[]] 1 0]`	1
`[range 5]`	`[1 2 3 4 5]`	`[reversed_range 5]`	`[5 4 3 2 1]`

Формат входного файла

Ввод состоит не более, чем из тысячи тестов. Каждый тест состоит из одной строки, в которой записано одно положительное целое число n ($1 \leqslant n \leqslant 10^{100}$). В конце ввода будет помещён тест с $n = 0$, который не требуется обрабатывать.

Формат выходного файла

Для каждого теста выведите в отдельной строке одно целое число — значение `[hash n]`.

Примеры

awful.in	awful.out
1 2 3 0	1 1 3

Задача B. Количество решений

Имя входного файла:	`count.in`
Имя выходного файла:	`count.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Васе поручили вычислить количество целых неотрицательных решений уравнения:

$$x_1 + 2x_2 + 3x_3 + \ldots + nx_n = n$$

Вася составил таблицу для небольших n:

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Количество решений	1	2	3	5	7	11	15	22	30	42	56	77	101	135	176

Помогите ему вычислить количество решений для больших n. Ответ следует вычислить по модулю m.

Формат входного файла

Ввод состоит не более, чем из 10 000 тестов. Каждый тест состоит из одной строки, в которой записано два целых числа n и m ($1 \leqslant n \leqslant 50\,000$, $2 \leqslant m \leqslant 10^9$). Гарантируется, что m свободно от квадратов, то есть не существует такого целого $a > 1$, что m делится нацело на a^2. Также гарантируется, что максимальный простой делитель числа m не превосходит 100. В конце ввода будет помещён тест с $n = m = 0$, который не требуется обрабатывать.

Формат выходного файла

Для каждого теста выведите в отдельной строке одно целое число — остаток от деления числа решений уравнения на m.

Примеры

count.in	count.out
1 3 4 3 15 210 0 0	1 2 176

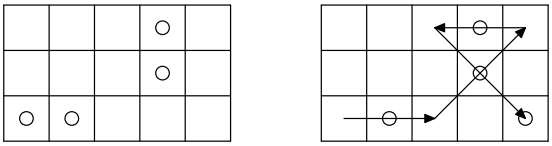
Задача C. Блохи и собака

Имя входного файла:	`jumps.in`
Имя выходного файла:	`jumps.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Васе, работнику НИИ Данных Строк, поручено написать симуляцию игры «Блохи и собака».

На бесконечной шахматной доске находятся n блох и одна спящая собака. На одной клетке может находиться более одной блохи. За один ход одна блоха может переместиться на соседнюю свободную клетку доски или выполнить один или несколько прыжков.

Клетки (x_1, y_1) и (x_2, y_2) считаются соседними, если $\max(|x_1 - x_2|, |y_1 - y_2|) = 1$. Прыжком называется перемещение блохи из клетки (x_1, y_1) в клетку $(2x_2 - x_1, 2y_2 - y_1)$, когда клетка (x_2, y_2) занята и является соседней к клетке (x_1, y_1), а клетка $(2x_2 - x_1, 2y_2 - y_1)$ свободна. Пример прыжков показан на рисунке.



Цель игры заключается в том, чтобы какая-то блоха достигла клетки, в которой находится собака.

Ваша задача заключается в том, чтобы найти минимальное количество ходов, необходимое для достижения цели.

Формат входного файла

Ввод состоит из одного или нескольких тестов. Первая строка содержит три целых числа n, x_d, y_d — количество блох и положение собаки ($1 \leqslant n \leqslant 100\,000$). В последующих n строках заданы координаты блох. В конце ввода будет помещён тест с $n = x_d = y_d = 0$, который не требуется обрабатывать.

Сумма всех n во входном файле не превышает 100 000. Все координаты — целые числа, по абсолютной величине не превышающие 10^9.

Формат выходного файла

Для каждого теста выведите в отдельной строке одно целое число — минимальное количество ходов, необходимое для того, чтобы блоха достигла клетки, в которой спит собака.

Примеры

jumps.in	jumps.out
4 0 -2 0 0 1 0 3 1 3 2 1 0 0 0 0 0 0 0	2 0

Задача D. Головоломка с трубками

Имя входного файла:	`oriental.in`
Имя выходного файла:	`oriental.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мебибайт

Вася работает в НИИЗТ (Научно-Исследовательский Институт Задач с Трубками). Он изучает новую головоломку, состоящую из нескольких трубок, частично заполненных некоторой жидкостью.

Есть несколько коробочек, прозрачные пластиковые трубки соединяют некоторые из них. Каждая трубка внутри покрыта специальным веществом, которое, когда по трубке протекает жидкость, начинает светиться. Есть две особенные коробочки — первая, в которой изначально находится вся жидкость, и последняя, где жидкость должна остаться при правильном решении головоломки.

Требуется так расположить коробочки и трубки, чтобы жидкость, когда она будет выпущена из первой коробочки, прошлась по всем трубкам и собралась в последней коробочке. Жидкость из коробочки течёт по всем выходящим из неё трубкам, которые идут в коробочки, расположенные ниже. То есть, нужно расположить коробочки в столбик одну над другой так, чтобы из первой можно было добраться в любую другую, двигаясь только вниз, и из любой можно было добраться в последнюю.

Помогите Васе решить головоломку: для каждой трубки определите, какая из соединяемых ей коробочек должна быть выше.

Формат входного файла

Входной файл состоит из одного или более тестов. Каждый тест начинается со строки, содержащей два числа: n и k ($2 \leqslant n \leqslant 100\,000$, $0 \leqslant k \leqslant 100\,000$) — это количество коробочек и трубок соответственно. Следующие k строк содержат описания трубок: пары чисел (a_i, b_i), означающие, что соответствующая трубка соединяет коробочки с номерами a_i и b_i. Первая коробочка имеет номер 1, а последняя — номер n. Гарантируется, что ни одна трубка не соединяет коробочку с ней самой; тем не менее, между парой различных коробочек может быть больше одной трубки.

Входной файл заканчивается строкой из двух нулей, которую обрабатывать не следует. Сумма n во всех тестах, как и сумма k, не превосходит 100 000.

Формат выходного файла

Для каждого теста выведите либо «`No`», если соответствующую головоломку решить невозможно, либо «`Yes`», после чего выведите ориентацию трубок. Нужно, как и во входном файле, для каждой трубки вывести две коробочки, которые она соединяет, но эти две коробочки должны идти в правильном порядке — сначала та, которая выше. Трубки нужно выводить в том же порядке, в каком они даны во входном файле.

Пример

oriental.in	oriental.out
6 8 1 3 3 6 1 2 2 4 4 1 4 5 5 1 2 3 0 0	Yes 1 3 3 6 1 2 4 2 1 4 5 4 1 5 2 3

Задача E. Многоугольник

Имя входного файла:	`polygon.in`
Имя выходного файла:	`polygon.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

У Васи есть шарнирный многоугольник. Это значит, что у многоугольника жёсткие стороны, а вершины соединены шарнирно, т. е. углы многоугольника могут меняться. Вася хочет добиться того, чтобы многоугольник был описанным. Многоугольник называется описанным, если существует окружность, которая касается всех сторон многоугольника. Касание в вершине запрещено.

Формат входного файла

Входной файл состоит из нескольких наборов входных данных. Каждый из них описывается одной строкой из целых чисел. Первым числом в строке является $3 \leqslant N \leqslant 1000$ — количество сторон многоугольника. Далее в строке записано N натуральных чисел — длины сторон в порядке обхода многоугольника по часовой стрелке. Длины сторон не превосходят 100. Последняя строка файла содержит единственное число 0.

Сумма всех N из входного файла не будет превосходить 100 000.

Формат выходного файла

Для каждого теста выведите в первой строке «`YES`», если Васе удастся сделать задуманное, и «`NO`», если нет. В случае, если Васе это удастся, во второй строке выведите радиус окружности, а в третьей строке выведите $2N$ вещественных чисел — координаты вершин многоугольника. Вершины должны быть выведены в порядке обхода по часовой стрелке. Первая сторона из входного файла должна соответствовать стороне между вершинами 1 и 2. Центр вписанной окружности должен быть в точке $(0, 0)$. В случае отрицательного ответа оставьте вторую и третью строку пустыми.

Ответ считается правильным, если длины сторон многоугольника отличаются от заданных не более чем на 10^{-6}.

Примеры

polygon.in	polygon.out
4 1 1 1 2 4 1 1 1 1 0	NO YES 0.5 -0.5 -0.5 -0.5 0.5 0.5 0.5 0.5 -0.5

Задача F. Полиномиальный хеш

Имя входного файла:	`polyhash.in`
Имя выходного файла:	`polyhash.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Вася использует полиномиальное хеширование. Если дана строка с символами $s_0 s_1 s_2 \dots s_n$, то её хешом будет число $h = (\mathrm{ord}(s_0) + \mathrm{ord}(s_1)p + \mathrm{ord}(s_2)p^2 + \dots + \mathrm{ord}(s_n)p^n) \bmod q$, где p и q — это произвольные различные простые числа, а $\mathrm{ord}(c)$ — ASCII-код символа c.

По данному хешу найдите какую-нибудь строку с таким хешом. Строка состоит из строчных букв латинского алфавита (коды символов 97–122).

Формат входного файла

Входной файл состоит из одного набора входных данных. Он состоит из трёх чисел q, p и h, где $3 \leqslant q \leqslant 10^7$, $0 \leqslant h < q$, $2 \leqslant p < q$, $2 \leqslant p \leqslant 10^6$. Гарантируется, что p и q будут простыми.

Формат выходного файла

Выведите строку длины не более 10^6, имеющую хеш h.

Примеры

polyhash.in	polyhash.out
533009 239 244670	spbsuchampionship
53309 239 6636	abacaba

Задача G. Установка памятника

Имя входного файла:	`roll.in`
Имя выходного файла:	`roll.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Директор НИИ Данных Строк решил установить памятник в вестибюле. Васе поручено рассчитать смету установочных работ.

Памятник представляет собой очень массивный шар. Вестибюль — это прямоугольник размера h на w. Изначально шар находится в нижнем левом углу вестибюля, т. е. в клетке с координатами $(1, 1)$, а шар требуется установить в клетке с координатами $(h-1, w-1)$. Грузчики могут раскачать и толкнуть шар либо вверх, либо направо. Если шар начал движение, то он остановится, только когда встретит препятствие на своем пути. Поэтому, чтобы переместить шар в место назначения, грузчикам необходимо временно поставить в вестибюле несколько плит. К сожалению, стоимости установки плит в разных клетках вестибюля не всегда совпадают.

Васе известна стоимость установки плиты для каждой клетки вестибюля. Помогите ему определить минимальную стоимость транспортировки шара в нужную клетку.

Формат входного файла

Ввод состоит из одного или нескольких тестов. Каждый тест начинается со строки, содержащей два целых числа h и w ($3 \leqslant h, w \leqslant 100$). В следующих h строках задаётся матрица стоимости установки плит. Стоимость задаётся целым числом от 0 до 10^9. Если указана отрицательная стоимость, это означает, что в данной ячейке невозможно возвести плиту. Гарантируется, что нельзя возвести плиту в клетке $(1, 1)$.

В конце ввода будет помещён тест с $h = w = 0$, который не требуется обрабатывать. Сумма всех h и w во входном файле не превышает 1000.

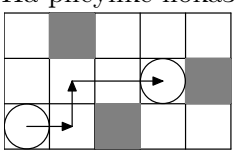
Формат выходного файла

Для каждого теста выведите в отдельной строке одно целое число — минимальную стоимость транспортировки, или же фразу «`Mission impossible`», если транспортировка невозможна.

Примеры

roll.in	roll.out
3 5 5 1 5 5 5 5 5 5 5 1 -1 5 1 5 5 3 3 -1 -1 -1 -1 -1 -1 -1 -1 -1 0 0	3 Mission impossible

На рисунке показано, как Васе надо расставить стены в первом тесте примера.



Задача H. Ёлка на новый год

Имя входного файла:	`spruce.in`
Имя выходного файла:	`spruce.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мебибайт

В городе Ма́нерске готовятся к проведению нового года. На главной площади города нужно поставить большую красивую ёлку. Ответственное задание по выбору ёлки поручили Васе, известному сотруднику какого-то там исследовательского института.

Лесничий по имени Андрей знает, как Вася выбирает деревья. Конечно, он хочет найти самую высокую ёлку. Но лес слишком большой, и Вася не хочет заблудиться в нём. Поэтому перед тем, как идти исследовать деревья, он спросит у Андрея, какую бы ёлку ему спилить и унести на площадь. Потом Вася залезет на эту ёлку и посмотрит, что растёт в округе. С ёлки высоты h метров виден весь лес в квадрате $2h$ метров $\times$ $2h$ метров, с центром в этой ёлке.

Если Вася увидит ёлку бо́льшей высоты, чем та, на которой он сейчас, он перейдёт к такой ёлке (самой высокой из тех, которые увидит) и заберётся на неё. Это будет продолжаться, пока Вася не окажется на ёлке, с которой он не увидит более высоких ёлок. Тогда он слезет с этой ёлки, спилит её и отнесёт на площадь.

Лесничий Андрей знаёт всё про ёлки и лес: координаты и высоту каждой из n ёлок. Помогите ему сохранить лес — сделать так, чтобы Вася спилил ёлку как можно меньшей высоты.

Формат входного файла

Входной файл состоит из нескольких тестов. В первой строке каждого теста содержится число n ($1 \leqslant n \leqslant 50\,000$). Далее следуют n строк по три числа в каждой. В $(i+1)$-й строке содержатся числа x_i, y_i и h_i. Это координаты и высота i-й ёлки. Все числа во входном файле целые и не превосходят 10^9 по модулю. В одном тесте нету двух ёлок одинаковой высоты.

Входной файл завершается строкой с одним нулём, которую обрабатывать не нужно. Сумма n по всем тестам не превосходит $50\,000$.

Формат выходного файла

Для каждого теста выведите два числа на отдельной строке: номер ёлки, которую Андрей должен показать Васе, и высоту ёлки, которая будет в результате спилена. Ёлки нумеруются в том порядке, в котором они даны во входном файле, начиная с единицы, отдельно для каждого теста.

Пример

spruce.in	spruce.out
3 -1 1 2 -1 -1 3 0 3 4 0	2 3

Задача I. Новые фильмы скачать бесплатно

Имя входного файла:	`torrent.in`
Имя выходного файла:	`torrent.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Антон решил с друзьями посмотреть ещё раз один старый, но очень хороший фильм. О нет! Старый DVD-диск с фильмом уже настолько расцарапан, что не читается. Придётся качать фильм из интернета.

Как известно, передача файлов через торрент-протокол (он называется BitTorrent, так же как и первый клиент) работает следующим образом: файл разбивается на сегменты, после чего каждый из сегментов скачивается отдельно. При этом возможно скачивание разных сегментов из разных мест для увеличения скорости.

Антон знает, на сегменты какого размера разбит нужный ему файл. Также он знает, с какой скоростью фильм будет качаться, а с какой воспроизводиться. Так, один сегмент скачивается за t_d секунд, и его хватает на t_v секунд просмотра фильма. Поскольку Антон очень хорошо разбирается в работе многих сетевых протоколов (в частности, BitTorrent), то он заранее подсчитал, в каком порядке будут скачиваться сегменты. Особенность торрент-клиента, который использует Антон, состоит в том, что он скачивает сегмент в оперативную память, и только после того, как скачает сегмент целиком, выгружает его на диск.

Теперь он хочет узнать, в какой момент можно будет начинать смотреть фильм. Вдруг, если начать смотреть фильм до полного скачивания, пока Антон с друзьями будут смотреть первые сегменты, остальные как раз успеют докачаться?

Формат входного файла

В первой строке входного файла содержатся три целых числа: n, t_d и t_v ($1 \leqslant n \leqslant 100\,000$, $1 \leqslant t_d, t_v \leqslant 10^9$). Следующая строка содержит перестановку из n чисел — порядок, в котором сегменты будут скачиваться.

Формат выходного файла

В выходной файл выведите одно число — время (в секундах), которое пройдёт с начала скачивания до момента, когда можно будет смотреть фильм.

Примеры

torrent.in	torrent.out
3 10 20 1 3 2	10
3 10 20 2 3 1	30
3 10 15 1 3 2	15

Задача J. Ксерокс

Имя входного файла:	`xerox.in`
Имя выходного файла:	`xerox.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

В НИИ Данных Строк идёт активная подготовка к празднованию нового года. Васе поручена почётная обязанность — вывесить транспарант-поздравление. Транспарант традиционно оформляется очень нетрадиционным шрифтом, и поэтому руководство решило сэкономить, и получить новый транспарант с помощью ксерокопирования частей старого. А именно, разрешается взять некоторую подстроку старого транспаранта (к сожалению, от него осталась одна фотография), длина которой лежит в пределах k_1 от k_2 (иначе буквы получатся слишком разного размера), и отксерокопировать соответствующий кусок фотографии на лист формата A4. После этого получившиеся листы можно будет склеить так, чтобы получился нужный лозунг.

Проблема в том, что ксерокс одного листа формата A4 стоит 2 рубля. Помогите Васе получить текст заданного транспаранта, отксерив минимальное число листов. Если потребуется, ксерокопируемые части могут перекрываться и даже совпадать.

Формат входного файла

Входной файл состоит из одного или нескольких тестов. Каждый тест состоит из трёх строк — в первой указаны длина n прошлогоднего транспаранта и длина m транспаранта, который нужно получить ($1 \leqslant n, m \leqslant 10\,000$), а также числа k_1 и k_2 ($1 \leqslant k_1 \leqslant k_2 \leqslant n$). Во второй и третьей строках приводятся соответственно тексты прошлогоднего транспаранта и транспаранта, который нужно вывесить в этом году. Все тексты состоят из маленьких латинских букв, общая длина всех текстов во входном файле не превосходит 10 000 символов. Входной файл завершается строкой из четырёх нулей, которую не нужно обрабатывать.

Формат выходного файла

Для каждого из тестов выведите разбиение на кусочки, которые необходимо отксерить, как показано в примере. Если задача для данного теста неразрешима ни за какие деньги, выведите слово «`IMPOSSIBLE`».

Примеры

xerox.in
51 12 1 2 snovymgodomsnovymschastiemsprazdnikomdorogiekollegi pozdravlyaem 51 12 1 2 snovymgodomsnovymschastiemsprazdnikomdorogiekollegi happynewyear 0 0 0 0
xerox.out
Case #1: p|o|zd|ra|v|l|y|a|em Case #2: IMPOSSIBLE

Задача K. Coin Stacks (Division 2 Only!)

Имя входного файла:	`coin.in`
Имя выходного файла:	`coin.out`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 Mebibytes

У вас есть несколько столбиков, составленных из одинаковых монет. Вы можете применить к любым двум столбикам следующую операцию:

- Если количество монет в столбиках одинаково, собрать из двух столбиков один (поставив их друг на друга).
- Если количество монет в столбиках различно и равно a и b соответственно ($a > b > 0$), то столбики заменяются столбиками размера $2b$ и $a - b$.

Например, если у Вас есть столбики высотой 1, 3, 4, вы сначала можете скомбинировать первый и второй столбики (получив столбики высотой $2, 2, 4$), затем снова скомбинировать первый и второй столбики (получив 4 4) и затем собрать все монеты в столбик высоты 8.

По заданному набору столбиков выясните, каким образом их можно собрать в один столбик с помощью указанной операции.

Формат входного файла

В первой строке входного файла задано целое положительное число $T \leqslant 100$ — количество тестовых примеров. Каждый тестовый пример начинается со строки, содержащей целое число N, ($2 \leqslant N \leqslant 200$) — количество столбиков. Следующая строка содержит N целых положительных чисел — размеры столбиков. Общее количество монет во всех столбиках одного тестового примера равно 2^K, где K — целое число, не превосходящее 30.

Формат выходного файла

Для каждого тестового примера выведите последовательность из не более, чем 10^4 операций, которые приводят к требуемому результату. Каждая операция записывается через пробел двумя числами — размерами столбиков, к которым она применяется. После финальной операции каждого тестового примера выводите два нуля в соответствии с форматом вывода. Если собрать монеты указанным выше способом за не более, чем 10^4, операций невозможно, выведите в качестве финальной (и единственной) операции для данного тестового примера $-1 - 1$.

Пример

coin.in	coin.out
1 3 1 3 4	1 3 2 2 4 4 0 0

Задача L. Martial arts (Division 2 Only!)

Имя входного файла:	`martial.in`
Имя выходного файла:	`martial.out`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 Mebibytes

В восточных единоборствах финалисты определяются в однокруговом турнире на выбывание. Победитель финала получает золотые медали, проигравший — серебряные.

Бронзовые медали разыгрываются в отдельном турнире, в котором участвуют те, кто на протяжении основного турнира проиграл одному из финалистов.

По заданным результатам основного турнира вычислите победителя, серебряного призёра и участников «бронзового» раунда.

Формат входного файла

Входной файл состоит из не более, чем 60 тестовых примеров. Каждый тестовый пример начинается целым числом N. $N = 0$ сигнализирует о завершении входного файла (этот тестовый пример не должен быть обработан). В остальных случаях $2 \leqslant N \leqslant 6$.

Далее заданы $2^N - 1$ строк, состоящих из пар имён (строк длины не менее 1 и не более 10, состоящих из строчных и заглавных латинских букв), обозначающих, что участник, чьё имя указано первым, выиграл свой бой у участника, чьё имя указано вторым.

Гарантируется, что данные корректно описывают однокруговой турнир на выбывание и что имена никаких двух участников не совпадают.

Формат выходного файла

Для каждого тестового примера выведите в соответствии с примером `'Gold: '`, затем имя победителя, затем, на следующей строке, `'Silver: '` и имя победителя, затем, на третьей и последней для данного примера строке, `'Bronze round: '` и отсортированный по алфавиту (при сортировке любая строчная буква идёт после любой прописной) список участников турнира за бронзовые медали.

Пример

martial.in	martial.out
3 A aB CC D F E H G A CC H F A H 0	Gold: A Silver: H Bronze Round: CC F G aB

Задача M. Paperboy (Division 2 Only!)

Имя входного файла:	`paperboy.in`
Имя выходного файла:	`paperboy.out`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 Mebibytes

Посёлок N расположен по одну сторону очень длинного шоссе. Почтальону нужно доставить газеты в некоторые из домов посёлка за минимальное время. На доставку одной газеты он тратит 1 минуту, время на движение между двумя домами с номерами k и l равно $|k-l| \cdot (p+1)$, где p — количество газет, которые почтальон несёт в данный момент.

Определите оптимальное время на доставку всей корреспонденции, если изначально все газеты складированы около дома x, а почтальон может брать с собой любое количество газет и в процессе доставки складировать любое количество газет в любом месте.

Формат входного файла

Входной файл состоит из нескольких тестовых примеров. В первой строке входного файла задано целое положительное число $T \leqslant 60$ — количество тестовых примеров. Каждый тестовый пример занимает одну строку и состоит из целых положительных чисел. Строка начинается с числа $n < 100$ — количества домов, куда почтальону надо доставить газеты. Первые n чисел — номера домов, куда почтальон должен доставить газеты, последнее число — номер дома, около которого складированы все n газет и от которого почтальон начинает свой маршрут. Номера домов не превосходят 10^5.

Формат выходного файла

Для каждого тестового примера выведите минимальное количество времени, которое требуется почтальону, чтобы доставить все газеты.

Пример

paperboy.in	paperboy.out
2 3 10 20 30 10 4 10 20 40 80 30	53 184

Задача N. Qualifiers (Division 2 Only!)

Имя входного файла:	`qual.in`
Имя выходного файла:	`qual.out`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 Mebibytes

В лёгкой атлетике состав финалистов определяется по результатам двух полуфиналов следующим образом: в финал выходят по 3 спортсмена с лучшим результатом из каждого полуфинала, а также два спортсмена с лучшим временем среди не вошедших в тройку в каком-либо полуфинале.

По результатам двух полуфиналов определите, кто из участников прошёл в финал.

Формат входного файла

В первой строке входного файла задано целое положительное число $T \leqslant 1000$ — количество тестовых примеров.

Каждый тестовый пример состоит из двух строк по 8 чисел в каждой — результатов полуфиналов. Каждое время, показанное в полуфинале — число между 10.0 и 55.0, заданное не более, чем с 3 знаками после запятой. Гарантируется, что все времена в рамках одного тестового примера различны.

Формат выходного файла

Для каждого тестового примера в первой строке выведите номер примера, в двух последующих строках — результаты полуфиналов в том порядке, в котором они были заданы во входном файле.

Перед временем каждого участника, вышедшего в финал, должен быть поставлен `-`.

Пример

qual.in	qual.out
2 10 12 14 16 18 20 22 24 11 13 15 17 19 21 23 25 10.0 10.1 10.6 10.9 11.5 11.8 12.5 22.7 10.2 10.3 10.4 10.5 10.7 10.8 11.0 12.2	1 -10.000 -12.000 -14.000 -16.000 18.000 20.000 22.000 24.000 -11.000 -13.000 -15.000 -17.000 19.000 21.000 23.000 25.000 2 -10.000 -10.100 -10.600 10.900 11.500 11.800 12.500 22.700 -10.200 -10.300 -10.400 -10.500 -10.700 10.800 11.000 12.200

Задача O. Splitting (Division 2 Only!)

Имя входного файла:	`split.in`
Имя выходного файла:	`split.out`
Ограничение по времени:	2 seconds
Ограничение по памяти:	256 Mebibytes

В попарно различных точках на большой площади перед стадионом расположились три группы фанатов московского «Спартака» и три группы фанатов ЦСКА. Для предотвращения беспорядков стюарды хотят построить стену в виде прямой, отделяющую фанатов «Спартака» от фанатов ЦСКА. По заданным координатам групп фанатов (каждая группа представляется в виде точки на плоскости) требуется выяснить, возможно ли такое разделение.

Формат входного файла

В первой строке входного файла задано количество тестовых примеров $T \leqslant 1000$. Каждый тестовый пример состоит из двух строк. В первой строке заданы числа $x_1, y_1, x_2, y_2, x_3, y_3$ — координаты групп болельщиков «Спартака», а во второй — координаты групп болельщиков ЦСКА $x_4, y_4, x_5, y_5, x_6, y_6$. Все координаты — целые числа между 1000 и -1000, при этом никакие три точки не лежат на одной прямой.

Формат выходного файла

Выведите `Yes`, если стена может быть построена, и `No` в противном случае.

Пример

split.in	split.out
2 1 1 2 2 2 3 1 0 5 1 4 5 1 0 -2 1 -2 -1 -1 0 2 1 2 -1	Yes No