Задача A. Задача о назначениях

Имя входного файла:	`assign.in`
Имя выходного файла:	`assign.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Одной из классических задач комбинаторной оптимизации является так называемая *«задача о назначениях»*. Формулируется она следующим образом.

Есть n работников, пронумерованных числами от 1 до n, и n работ, также пронумерованных числами от 1 до n. Если i-ый работник выполняет j-ую работу, то ему выплачивается зарплата в размере c_{ij} денежных единиц. Необходимо найти такое назначение работников на работы (каждый работник выполняет ровно одну работу, каждая работа выполняется ровно одним работником), что суммарная зарплата работников минимальна (соответствующая сумма называется *стоимостью назначения*).

Напишите программу, решающую задачу о назначениях.

Формат входного файла

Первая строка входного файла содержит целое число n ($1 \le n \le 10$).

Последующие n строк содержат по n чисел каждая. При этом j-ое число $(i+1)$-ой строки равно c_{ij} ($1 \le c_{ij} \le 1000$).

Формат выходного файла

В выходной файл выведите ответ на задачу — минимальную возможную стоимость назначения.

Примеры

assign.in	assign.out
2 1 2 2 1	2
2 1 2 3 4	5

Задача B. Автоспорт

Имя входного файла:	`cars.in`
Имя выходного файла:	`cars.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

В кружке автомоделирования проводятся соревнования. Они проходят на прямом участке трассы длиной L и шириной W. N участников выставляют свои модели на стартовые позиции. По свистку все модели начинают движение.

Цель состязания — быстрее всех доехать до линии финиша. Однако в пути каждую модель подстерегают опасности. А именно, модель может врезаться в борта трассы и тем самым прекратить движение, или две и более модели могут столкнуться и также остановиться.

Введем систему координат, в которой ось x будет направлена вдоль протяжения трассы, а ось y — поперек ему. Тогда трасса является прямоугольником, ограниченным прямыми $y = 0$, $y = W$, $x = 0$, $x = L$.

Модель считается врезавшейся в борт трассы, если в некоторый момент времени её y-координата будет равняться либо 0, либо W. Две модели считаются столкнувшимися, если в некоторый момент времени их координаты совпадают. Модели можно считать материальными точками.

Модель побеждает в соревновании, если она успешно преодолела линию финиша (прямую $x = L$) и сделала это не позже любой другой модели. Возможно, что побеждает сразу несколько моделей, в этом случае, как говорится, «побеждает дружба». Возможно также, что ни одна модель по тем или иным причинам не сумеет преодолеть линию финиша.

Заметим отдельно, что если модель проходит через какую-либо из точек $(L, 0)$ или (L, W), то считается, что она врезается в борт.

Поведение большинства моделей довольно предсказуемо, поэтому часто можно предугадать ход состязания. В данной задаче мы будем приближённо считать, что все модели начинают двигаться из своих начальных точек с заданными, постоянными на протяжении соревнования, векторами скорости.

Вам дана информация, известная на начало соревнования. Предскажите исход соревнования.

Формат входного файла

В первой строке входного файла находятся три числа N, L, W. ($1 \le N \le 1000$, $1 \le L \le 10^4$, $2 \le W \le 10^4$).

В последующих N строках находятся описания соревнующихся моделей — по 4 целых числа x_i, y_i, vx_i, vy_i, $1 \le i \le N$. (x_i, y_i) — это координаты стартовой точки модели с номером i, (vx_i, vy_i) — вектор скорости этой модели.

Гарантируется, что стартовые точки всех моделей различны, находятся на трассе, и не располагаются ни на каком-либо борту трассы, ни на линии финиша. Координаты векторов скорости не превышают 10^4 по абсолютному значению.

Формат выходного файла

В первой строке выходного файла выведите количество победителей. Во второй строке выведите их номера в порядке возрастания.

Примеры

cars.in	cars.out
1 1 2 0 1 1 0	1 1
2 10 3 0 1 2 0 5 2 1 0	2 1 2

Задача C. Скидка

Имя входного файла:	`discount.in`
Имя выходного файла:	`discount.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Межгалактическая сеть гипермаркетов OJ устраивает сезонную распродажу. В связи с этим на каждую покупку предоставляется скидка. Однако правила предоставления скидки весьма необычны.

Во-первых, для предоставления скидки количество товаров в покупке должны быть не меньше m. Во-вторых, скидка предоставляется в размере 50% на выбранные покупателем три товара из покупки, но при этом суммарная стоимость этих трех товаров не должна превышать c межгалактических денежных единиц.

Задан список товаров, входящих в покупку. Необходимо найти, за какую минимальную сумму можно их приобрести, воспользовавшись скидкой (если ей, конечно, можно воспользоваться) или не воспользовавшись. При этом все товары должны остаться в одной покупке.

Формат входного файла

Первая строка входного файла содержит три целых числа: n, m, c ($1 \le n, m \le 100$, $1 \le c \le 10^8$).

Вторая строка входного файла содержит n целых чисел: стоимости товаров, входящих в покупку. Стоимость товара — неотрицательное число, не превосходящее 10^8.

Формат выходного файла

В выходной файл выведите ответ на задачу.

Примеры

discount.in	discount.out
6 6 6 2 2 2 2 2 2	9
6 7 8 2 2 2 2 2 2	12

Задача D. Сумма делителей

Имя входного файла:	`divsum.in`
Имя выходного файла:	`divsum.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Пусть x — натуральное число. Назовем y его *делителем*, если $1 \leq y \leq x$ и остаток от деления x на y равен нулю.

Задано число x. Найдите сумму его делителей.

Формат входного файла

Первая строка входного файла содержит целое число x $(1 \leq x \leq 10^{18})$. Все простые делители числа x не превосходят тысячу.

Формат выходного файла

В выходной файл выведите ответ на задачу.

Примеры

divsum.in	divsum.out
12	28
239	240

Задача E. Поток в сети

Имя входного файла:	`flow.in`
Имя выходного файла:	`flow.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

При решении большого числа задач на графах часто бывает полезным введение понятий *сети* и *потока* в сети.

Сеть — это ориентированный граф $G = (V, E)$, каждому ребру $(u, v) \in V$ которого поставлено в соответствие число $c(u, v) \geq 0$, называемое *пропускной способностью* ребра. В случае $(u, v) \notin V$ удобно положить $c(u, v) = 0$. В сети выделены две вершины: *исток* s и *сток* t.

Поток в сети G — это функция $f : V \times V \to \mathcal{R}$, обладающая следующими тремя свойствами:

- *Подчиненность пропускной способности* — $\forall u, v \in V \;\; f(u, v) \leq c(u, v)$;
- *Антисимметричность* — $\forall u, v \in V \;\; f(u, v) = -f(v, u)$;
- *Сохранение потока* — $\forall u \in V \setminus \{s, t\} \;\; \sum_{v \in V} f(u, v) = 0$.

Вам дана сеть и некоторая функция на парах вершин этой сети. Проверьте, является ли эта функция потоком в этой сети.

Формат входного файла

В первой строке входного файла дано число вершин в сети N ($2 \leq N \leq 100$). Вершины в сети будут носить номера от 1 до N.

В следующих N строках заданы пропускные способности ребер сети — по N чисел в каждой строке. j-е число в $i + 1$-ой строке задает $c(i, j)$. Пропускные способности неотрицательны и не превосходят 10^4. Гарантируется, что $c(i, i) = 0$.

Далее идет пустая строка.

В следующих N строках заданы значения функции f в аналогичном формате. Эти значения не превосходят 10^4 по абсолютному значению.

Истоком сети служит вершина с номером 1, стоком — вершина с номером N.

Формат выходного файла

В выходной файл выведите `YES`, если функция является потоком, и `NO`, если не является.

Примеры

flow.in	flow.out
4 0 1 3 0 0 0 0 2 0 0 0 4 0 0 0 0 0 1 2 0 -1 0 0 1 -2 0 0 2 0 -1 -2 0	YES
4 0 1 3 0 0 0 0 2 0 0 0 4 0 0 0 0 0 2 1 0 -2 0 0 2 -1 0 0 1 0 -2 -1 0	NO

Задача F. Сравнение дробей

Имя входного файла:	`fraction.in`
Имя выходного файла:	`fraction.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Заданы две дроби. Необходимо их сравнить. Результатом сравнения должна быть строка `More`, если первая дробь больше второй, строка `Less`, если первая дробь меньше второй, и строка `Equal`, если дроби равны.

Формат входного файла

Первая строка входного файла содержит два целых числа: a_1, b_1 — соответственно, числитель и знаменатель первой дроби.

Вторая строка входного файла содержит два целых числа: a_2, b_2 — соответственно, числитель и знаменатель второй дроби.

Все числа во входном файле положительны и не превосходят 100.

Формат выходного файла

В выходной файл выведите ответ на задачу.

Примеры

fraction.in	fraction.out
1 2 1 3	More
1 3 1 2	Less
1 2 2 4	Equal

Задача G. Три поросенка

Имя входного файла:	`game.in`
Имя выходного файла:	`game.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Три поросенка играют в следующую игру.

Первый поросенок придумывает натуральное число N_1 и сообщает его второму поросенку.

После этого второй поросенок придумывает такое натуральное число N_2, что существуют такие $A_1 \geq 2$, $B_1 \geq 2$, что $N_2 = A_1 + B_1$ и $A_1 \times B_1 = N_1$, и сообщает его третьему поросенку. Тот аналогично придумывает число N_3 и сообщает его первому поросенку, и т.д.

Если какой-нибудь поросенок называет число N_i, для которого следующий поросенок не может придумать числа A_i и B_i, то он выигрывает. Если же игра никогда не заканчивается, то считается, что наступила ничья.

Дано число N. Определите, кто выигрывает, если первый поросенок начинает и придумывает число N.

Поросята играют оптимально. То есть, если для поросенка существует ход, приводящий к его выигрышу, то он его делает; если такого хода не существует, но существует ход, ведущий к ничье, то поросенок делает этот ход; иначе, он делает ход, максимизирующий его число N_i.

Формат входного файла

В первой строке дано натуральное число N. $2 \leq N \leq 10^6$.

Формат выходного файла

В выходной файл выведите порядковый номер поросенка, выигрывающего игру, или 0, если наступает ничья.

Примеры

game.in	game.out
5	1
16	3
4	0

Задача H. Ладья в лабиринте

Имя входного файла:	`rook.in`
Имя выходного файла:	`rook.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Ладья — это шахматная фигура, которая за один ход может переместиться на любое количество клеток по горизонтали или вертикали. При этом она не может «перепрыгивать» через стоящие на ее пути фигуры.

Вася недавно соорудил на шахматной доске своеобразный лабиринт — в некоторые клетки доски он поставил пешки (самые «слабые» шахматные фигуры). Теперь он хочет знать, за какое минимальное количество ходов ладья может добраться из одной клетки в другую.

Он размышляет над этим вопросом уже несколько дней, однако найти ответ не может. Поэтому он решил обратиться за помощью к Вам. Напишите программу, находящую ответ на Васину задачу.

Формат входного файла

Первая строка входного файла содержит два натуральных числа: n и m ($1 \le n, m \le 500$) — размеры лабиринта.

Каждая из последующих n строк содержит m символов. j-ый символ i-ой из этих строк соответствует клетке с координатами (i, j). Он равен «`.`» (точка), если клетка пуста, `P`, если занята пешкой, `S`, если это начальная клетка для ладьи, `F`, если это конечная клетка.

Формат выходного файла

В выходной файл выведите минимальное количество ходов, требуемое ладье для того, чтобы из начальной клетки попасть в конечную. Если конечная клетка недостижима из начальной — выведите -1.

Примеры

rook.in	rook.out

`4 4`
`F.PS`
`.PP.`
`.PP.`
`....`
	`3`

`4 4`
`F.PS`
`.PP.`
`.PP.`
`.P..`
	`-1`