Задача N (Div 1). Арифметика

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

Королю Байтландии очень нравится арифметика. Для него это самая простая наука. Ведь нет ничего проще, чем складывать числа. На досуге, он любит выбрать какое-нибудь целое число A и складывать его с самим собой, так, что получается следующий ряд:

$$A,\ 2A,\ 3A,\ 4A,\ \ldots$$

Вчера вечером король успел посчитать только N первых чисел этого ряда, как его прервал придворный математик. Король объяснил ему, что занимается очень важным делом, складывая числа. В ответ, математик, плохо скрывая улыбку, спросил сколько из посчитанных королем чисел этого ряда делятся нацело на N.

Король не смог ответить и попросил вас написать программу, которая ответит на вопрос математика.

Формат входного файла

Входной файл содержит два целых числа A и N, разделенных пробелом. Гарантируется, что $(-10^5 \le A \le 10^5,\ 1 \le N \le 10^9)$

Формат выходного файла

В выходной файл выведите единственное число — количество чисел, посчитанных королем, которые нацело делятся на N.

Пример

arith.in	arith.out
2 4	2
7 3	1

Задача O (Div 1). Морской бой

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

Национальным видом спорта в Байтландии является игра “Морской бой”. В ней два игрока расставляют корабли на поле размером 10 на 10 клеток, а потом по очереди называют координаты клеток и пытаются потопить корабли противника.

По версии Федерации Морского Боя Байтландии каждый игрок должен расставить на своем поле следующие корабли: один трехпалубный, два двухпалубных и три однопалубных. Каждый корабль, это прямоугольник из соответствующего количества клеток, расположенный горизонтально или вертикально. Никакие два корабля не должны касаться друг друга.

С целью облегчения подготовки к игре программист Федерации написал программу, которая позволяет визуально создавать и редактировать расположение кораблей, а также сохранять его в файл и загружать из файла. Клетку поля, в которой есть какой-то корабль, он называет непустой, а в которой нет корабля — пустой.

Однажды он, пользуясь этой программой, по случайности, сохранил очередную позицию в файл, который уже содержал сохраненное ранее положение кораблей. Он открыл файл и увидел, что там не последняя сохраненная позиция, а непонятно что. У него было две версии, что произошло при сохранении, либо вторая позиция наслоилась на первую, либо из-за какой-то ошибки был сохранен случайный “мусор”.

Под наложением двух позиций A и B, он имеет ввиду позицию C, где клетка пустая тогда и только тогда, когда пусты обе соответствующие клетки в A и B.

В связи с этим, он просит вас помочь ему написать программу, которая бы определяла, является ли данная позиция результатом наложения двух корректных позиций кораблей или нет.

Формат входного файла

Входной файл содержит 10 строк по 10 символов в каждой. Это позиция, которую вам передал ваш друг. Пустая клетка обозначена символом “.”(точка), а непустая “#”(решетка). Других символов в файле нет.

Формат выходного файла

Если данная позиция не может быть получена путем наложения двух корректных позиций, то в первой строке выходного файла выведите “NO” (большими буквами, без кавычек). Иначе, выведите “YES” а в последующих 10 строках запишите обе позиции, как показано в примере (используйте ровно три пробела для разделения позиций). Если решений несколько, выведите любое из них.

Пример

battleship.in	battleship.out

`..........`
`...####...`
`..........`
`....##....`
`..........`
`...####...`
`...#..#...`
`...####...`
`...####...`
`..........`
	`YES`
	`..........  ..........`
	`...#.#....  ....#.#...`
	`..........  ..........`
	`....#.....  .....#....`
	`..........  ..........`
	`.....##...  ...##.....`
	`...#......  ......#...`
	`...#.#....  ....#.#...`
	`...#.#....  ....#.#...`
	`..........  ..........`

Задача P (Div 1). Крышки

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

Министерство экологии Байтландии установило новые правила выбрасывания мусора.

Допустим, у вас есть N пакетов и M небольших крышек. В каждый пакет можно складывать как крышки, так и другие пакеты, которые, в свою очередь, тоже могут содержать крышки и пакеты.

Вам необходимо так распределить крышки, чтобы в пакете с номером i было ровно C_i крышек. При этом нужно чтобы все крышки были распределены.

Количество крышек в пакете — это крышки, непосредственно лежащие в данном пакете, а также все крышки во вложенных пакетах.

Формат входного файла

Первая строка входного файла содержит два целых числа N и M — количество пакетов и крышек соответственно ($1 \le N, M \le 1000$). Во второй строке, через пробел, идет N целых чисел C_i — требуемое количество крышек в соответствующем пакете ($0 \le C_i \le 1000$). Пакеты нумеруются от 1 до N.

Формат выходного файла

Если распределить крышки требуемым образом невозможно, то в первой строке выходного файла выведите "NO" (большими буквами и без кавычек). Иначе, выведите "YES", а в следующих N строках выведите содержимое пакетов. Содержимое пакета выводится следующим образом, сначала количество крышек непосредственно в данном пакете, потом количество пакетов непосредственно в данном пакете, а затем номера этих пакетов в порядке возрастания. Все числа в строке должны быть разделены одним пробелом. Если правильных ответов несколько, то выведите любой из них.

Вложенность пакетов не должна противоречить смыслу, т.е. каждый пакет может быть непосредственно вложен не более чем в один другой пакет, а также не должно быть циклической вложенности.

Пример

covers.in	covers.out
2 10 5 10	YES 5 0 5 1 1
4 10 6 1 1 4	YES 4 2 2 3 1 0 1 0 4 0
2 5 3 3	NO

Примечание: В первом примере первый пакет содержит пять крышек, а второй пять крышек и первый пакет. Таким образом, все десять крышек распределены нужным образом.

Во втором примере первый пакет содержит 4 крышки и два других пакета, в каждом из которых по одной крышке. Четвертый пакет содержит только 4 крышки.

В третьем примере крышки требуемым образом распределить нельзя.

Задача Q (Div 1). Бар "Троица"

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

Бар "Троица" — самое популярное заведение в Байтландии. Его название, вопреки мнению многих, связано не с тем, что там частенько соображают "на троих", а с тем, что в этом баре каждый коктейль состоит равно из трех различных ингредиентов.

Хозяин бара знает, что всего у него имеется N различных ингредиентов. Из них он хочет составить максимальное количество различных коктейлей, чтобы привлечь как можно больше клиентов. Два коктейля он считает различными, если в них не больше одного совпадающего ингредиента.

Конечно, сам он этим заниматься не намерен, и поручил вам составить список коктейлей, такой чтобы выполнялись два условия:

1. Все коктейли должны быть попарно различными
2. Для любых двух имеющихся ингредиентов должен быть коктейль, в котором они присутствуют

Напишите программу, которая в зависимости от количества ингредиентов находит нужный список коктейлей, или говорит, что его не существует.

Формат входного файла

В первой строке входного файла задано единственное целое число N — количество различных ингредиентов ($3 \le N \le 300$).

Формат выходного файла

Если требуемый список коктейлей составить невозможно, то в первой строке выходного файла выведите -1. Иначе, выведите количество коктейлей в списке, а в следующих строках сами коктейли в любом порядке. Коктейль нужно выводить как три ингредиента, разделенные пробелом. Ингредиенты нумеруются от 1 до N. Если возможных ответов несколько, то выведите любой из них.

Пример

drink.in	drink.out
3	1 1 2 3
4	-1
7	7 1 2 7 1 3 6 1 4 5 2 3 5 2 4 6 3 4 7 5 6 7

Задача R (Div 1). Затмение

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

Астрономы Байтландии давно хотят понять, как именно предугадывать затмения. У них есть история наблюдений за много лет, но они не могут точно определить когда произойдет следующее затмение.

После одного неприятного инцидента, связанного с неправильно предсказанным затмением, кое-кто из жителей королевства стал подозревать, что король на самом деле не является наместником бога-солнца на земле.

Именно тогда король и приказал ученым полностью разобраться в этом вопросе. После многих лет изысканий ученые свели задачу предсказания следующего затмения к задаче отыскания количества треугольников определенного вида.

А именно, вам дано целое число N. Надо найти количество различных целочисленных треугольников (тех у которых длины сторон являются целыми положительными числами) у которых наибольшая сторона равна N. Все треугольники должны иметь площадь больше нуля.

От вас требуется написать программу, решающую данную задачу.

Формат входного файла

Входной файл содержит единственное целое число N ($1 \le N \le 10^6$).

Формат выходного файла

В выходной файл выведите единственное число — ответ на вопрос задачи.

Пример

eclipse.in	eclipse.out
14	56
12	42
2008	1009020

Задача S (Div 1). Фирма

Имя входного файла:	`firm.in`
Имя выходного файла:	`firm.out`
Ограничение по времени:	3 секунды
Ограничение по памяти:	64 мегабайта

Крупнейшая в Байтландии фирма “Веники Ltd.” имеет иерархическую структуру. То есть у каждого работника, кроме директора, есть ровно один непосредственный начальник. У директора фирмы, само собой, начальника нет.

Директор фирмы, услышав послание президента, в котором говорилось о необходимости внедрения инновационных технологий, решил применить, придуманную им самим, новую систему оценки качества работы сотрудников.

Она заключается в следующем. В течении месяца, каждый работник имеет право опускать в специальную урну запросы следующих двух видов.

1. Изменить зарплату всех моих подчиненных на X рублей.
2. Изменить зарплату всех моих начальников на X рублей.

Здесь под всеми подчиненными работника понимаются его непосредственные подчиненные, подчиненные их подчиненных и т.д. А под всеми начальниками — непосредственный начальник, его начальник и т.д. вплоть до директора. Под изменением зарплаты понимается прибавление к ней X рублей. Если работник хочет уменьшить зарплату, то X будет отрицательным.

Каждый работник может опускать в урну произвольное количество запросов.

В конце месяца нужно будет проанализировать все поступившие запросы и найти новое значение зарплаты для каждого сотрудника. Ваша задача — написать программу, которая бы автоматизировала процесс обработки запросов.

Формат входного файла

В первой строке входного файла задано единственное число N — количество людей в фирме ($1 \le N \le 10^5$). Во второй строке, через пробел, идут N целых чисел A_i. Каждое число означает начальную зарплату соответствующего работника ($1 \le A_i \le 10^5$). Работники нумеруются от 1 до N. В третьей строке, через пробел, заданы $N-1$ целых чисел P_i. Первое из них — номер непосредственного начальника второго работника, второе — номер начальника третьего работника, и так далее. Директор фирмы не имеет начальника и его номер равен 1. В следующей строке идет одно целое число M — количество запросов в урне ($1 \le M \le 10^5$). В следующих M строках заданы сами запросы. Каждый запрос задается тремя целыми числами A, B, X. A — это вид запроса, если 1, то запрос для всех подчиненных, а если 2, то для всех начальников. B — номер работника, отправившего запрос. X — количество на которое надо увеличить зарплату. $1 \le A \le 2$, $1 \le B \le N$, $|X| \le 10^5$.

Формат выходного файла

В выходной файл выведите N чисел, по одному в строке — итоговую зарплату каждого работника, после выполнения всех запросов. Она может оказаться отрицательной.

Пример

firm.in	firm.out
5 100 50 50 50 50 1 1 3 3 3 1 3 -2 2 4 5 1 2 1000	105 50 55 48 48

Задача T (Div 1). Игра

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

В Байтландии очень популярна следующая простая игра. На полоске в N клеток расставляются фишки белого и черного цветов. Некоторые клетки могут быть пусты. В одной клетке может быть только одна фишка.

Игрок, начинающий игру, ходит белыми фишками, а его оппонент черными. Игроки ходят поочередно. Ход заключается в продвижении одной из своих фишек на свободную слева клетку, если таковая имеется. Если игрок не может сделать ход, то он проигрывает.

Вам требуется написать программу, которая по заданной начальной позиции определяет кто выиграет, если оба соперника будут играть оптимально.

Формат входного файла

В первой строке входного файла заданы три целых числа N — количество клеток на игровой полоске, W и B — количества белых и черных фишек соответственно ($2 \le N \le 10^6$, $1 \le W, B \le 1000$). Во второй строке, через пробел, заданы координаты белых фишек в произвольном порядке. Клетки полоски нумеруются от 1 до N, слева направо. Аналогично, в третьей строке заданы координаты черных фишек. Гарантируется, что координаты будут в пределах полоски и не будут совпадать.

Формат выходного файла

Если при оптимальной игре выигрывает начинающий, то выведите слово “WHITE” (большими буквами и без кавычек). Иначе, выведите “BLACK”.

Пример

game.in	game.out
6 2 1 2 6 4	WHITE
6 2 2 2 6 5 4	BLACK

Задача U (Div 1). Счастливые билетики

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

В Байтландии автобусный билетик считается счастливым, если в его номере сумма цифр на нечетных местах делится нацело на сумму цифр на четных местах. Например, билет с номером 10221 счастливый, т.к. $1 + 2 + 1$ делится на $0 + 2$. А билеты с номерами 8, 124, 1040 не являются счастливыми.

Ваша задача — написать программу, которая находила бы количество счастливых билетов с номерами в промежутке от L до R, включительно.

Формат входного файла

Входной файл содержит два целых числа L и R, разделенных пробелом. Гарантируется, что $(1 \le L \le R \le 10^{18})$.

Формат выходного файла

В выходной файл выведите единственное число — ответ на вопрос задачи.

Пример

happy.in	happy.out
1 21	2
50 100	15

Примечание: В первом примере два счастливые билета имеют номера 11 и 21. Во втором примере счастливыми будут следующие номера: 51, 55, 61, 62, 63, 66, 71, 77, 81, 82, 84, 88, 91, 93, 99.

Задача V (Div 1). Маршрут

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

Королевство Байтландия разделена на N княжеств, в каждом из которых есть своя столица. Периодически король должен посещать столицы княжеств, чтобы быть в курсе всех дел, которые там происходят.

Дороги в королевстве плохие, а расстояния протяженные, поэтому король предпочитает передвигаться воздушным транспортом. Так получается быстрее и надежнее. Вот только одна беда — уж больно дорого стоит топливо для самолетов.

Король хочет облететь все столицы княжеств таким маршрутом, который бы имел наименьшую длину.

Напишите программу, которая по расположению городов находит минимальное расстояние, которое придется преодолеть королю, чтобы вылететь из его дворца, облететь все города и вернуться обратно. Дворец короля имеет координаты $(0, 0)$.

Формат входного файла

Первая строка входного файла содержит единственное число N — количество княжеств в королевстве Байтландия ($1 \le N \le 9$). В следующих N строках заданы координаты столиц княжеств. Гарантируется, что координаты будут целыми числами, по модулю не превосходящими 1000.

Формат выходного файла

В выходной файл выведите единственное число — минимаьное расстояние, которое должен преодолеть король. Ответ должен отличаться от правильного не более чем на 10^{-6}.

Пример

itinerary.in	itinerary.out
3 1 0 0 1 1 1	4
4 1 2 3 4 5 6 7 8	21.35149516

Задача W (Div 1). Объединение домов

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

В столице Байтландии строится новый микрорайон. В нем все дома и коммуникации будут сделаны по последнему слову техники. В частности, каждый дом будет находиться на пересечении двух перпендикулярных улиц, а все улицы будут двух типов, одни будут идти с севера на юг (вертикальные), а другие с запада на восток (горизонтальные).

Все здания надо объединить в единую энергосеть. Для этого вдоль каждой улицы идет линия электропередач, к которой можно подключать дома, находящиеся на данной улице.

Каждый дом можно подключить к одной из двух линий, идущих вдоль улиц, пересекающихся возле данного дома. Но, чтобы сбалансировать нагрузку, было принято следующее решение. Если на улице стоит X домов, то половина из них должна быть подключена к линии на этой улице, а остальные дома должны быть подключены к линиям, идущим по перпендикулярным улицам. Если X — нечетное и его нельзя поровну разделить на два, то подключить можно как $\lfloor X/2 \rfloor$, так и $\lfloor X/2 \rfloor + 1$ домов. Здесь $\lfloor X \rfloor$ означает округление числа X вниз. Т.е. наибольшее целое число, непревосходящее X.

Например, если на улице 6 домов, то к линии на этой улице надо обязательно подключить ровно 3 дома. А если 7 домов, то можно подключить либо 3, либо 4 дома. Дома для подключения можно выбирать произвольно.

Вам даны дома уже построенные в новом микрорайоне. Ваша задача — написать программу, которая для каждого из них определяла бы к какой из двух линий его надо подключить, чтобы в итоге для всех улиц выполнялось вышеприведенное требование.

Формат входного файла

В первой строке входного файла задано целое число N — количество домов ($1 \le N \le 200$). В следующих N строках заданы адреса домов. Адрес дома — это пара чисел (X, Y), которые задают номера горизонтальной и вертикальной улицы, на пересечении которых и стоит дом ($1 \le X, Y \le 200$). Гарантируется, что все адреса различны.

Формат выходного файла

Если подключить дома требуемым образом невозможно, то в первой строке выведите "NO" (большими буквами, без кавычек). Иначе, в первой строке выведите "YES", а во второй строке N символов, означающих к какой линии подключен соответствующий дом. Для обозначения горизонтальных улиц используйте символ "H", а для вертикальных "V". Если правильных ответов несколько, то выведите любой из них.

Пример

junction.in	junction.out
4 1 1 2 2 3 1 2 1	YES VVVH
3 1 1 2 2 3 3	YES HHV