Задача A. Снова про простые числа

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

Найдите простое число из отрезка $[a; b]$ с максимальной суммой цифр. Если таких чисел несколько, выберите из них максимальное.

Напомним, что *простым* называется натуральное число, большее единицы и делящееся только на единицу и на само себя.

Формат входного файла

Входной файл содержит два целых числа: a и b $(1 \le a \le b \le 10^8), b - a \le 1000$.

Формат выходного файла

В выходной файл выведите ответ на задачу. Если указанный отрезок не содержит простых чисел выведите в выходной файл `-1`.

Примеры

again.in	again.out
1 13	7
900 1000	997
8 10	-1

Задача B. Игра в фишки

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

Вот уже 10 лет дядя Семен работает сторожем на складе, где хранятся старые процессоры. Его работа чрезвычайно скучна, поэтому все рабочее время он играет в увлекательную игру.

Семен берет A фишек красного цвета, B фишек синего цвета и C зеленого цвета. За один ход он может заменить две фишки разных цветов на одну фишку третьего цвета. Считается, что Семен «*сыграл*» игру, если после некоторого количества ходов осталась одна фишка.

За 10 лет Семен так научился играть в эту игру, что для произвольных неотрицательных A, B, C он сразу может сказать, возможно ли «*сыграть*» игру или нет. Вам предстоит научиться делать это за 3 часа.

Формат входного файла

В первой строке входного файла натуральное число N — число тестов ($1 \le N \le 1000$).

В каждой из последующих N строк содержится тест: три целых числа: A, B и C ($0 \le A, B, C \le 2^{63} - 1$).

Формат выходного файла

Для каждого теста выведите «`Yes`», если «*сыграть*» игру можно, иначе выведите «`No`». Ответ для каждого теста должен располагаться в отдельной строке.

Примеры

counters.in	counters.out
2 1 0 0 1 1 1	Yes No

Задача C. Преобразование ДНК

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

Биологи лаборатории *Advanced Cellular Mechanics Lab. (ACM Lab.)* занимаются исследованиями в области геномов и ДНК. Недавно в этой лаборатории была разработана технология, позволяющая достаточно дешево производить с цепочкой ДНК некоторые преобразования.

Представим себе цепочку ДНК как строку длины n из символов из множества $\{A, G, C, T\}$. Элементарное преобразование, которое умеют проводить биологи лаборатории, представляет собой разворот подстроки с l-ого по r-ый символ (целые числа l и r выбираются так, что $1 \le l \le r \le n$). Таким образом, из строки $a_1 a_2 \ldots a_l a_{l+1} \ldots a_{r-1} a_r \ldots a_n$ получается строка $a_1 a_2 \ldots a_r a_{r-1} \ldots a_{l+1} a_l \ldots a_n$.

Теперь биологи разрабатывают программно-аппаратный комплекс для выполнения преобразований ДНК. Одной из его функций будет преобразование исходной цепочки ДНК в требуемую.

Ваша задача — написать программу, которая по исходной и требуемой цепочкам ДНК будет находить необходимую для этого цепочку элементарных преобразований.

Формат входного файла

Первая строка входного файла содержит описание исходной цепочки ДНК, вторая строка — описание требуемой цепочки ДНК. Длины обеих цепочек совпадают и не превышают 5000. Каждая из цепочек содержит только символы из множества $\{A, G, C, T\}$.

Гарантируется, что искомая последовательность преобразований существует.

Формат выходного файла

В первой строке выходного файла выведите количество k преобразований в построенном решении. Число k должно быть неотрицательным и не должно превышать 4999.

Далее выведите k строк, описывающих построенную последовательность элементарных преобразований. Каждая из строк должны содержать два числа: l_i и r_i — соответственно левый и правый конец разворачиваемого на i-ом шаге отрезка.

Примеры

dna.in	dna.out
AGCT GCAT	2 1 2 1 3
AGCTA ATCGA	1 1 5

Задача D. Поместье

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

И молвил тогда Король: "Ты храбро сражался, Рыцарь, и твой подвиг не будет забыт в веках. За доблесть твою я дарую тебе сей замок и земли вокруг него. Однако нарушен был тобой строжайший из запретов — все войны видели, как ты сражался без Шляпы на голове подобно дикарям, и их злые духи могли вселиться в тебя. Ты знаешь, что закон предков велит отправлять на небеса души подобных тебе, пока зло, которое могло укорениться в них, не вырвалось наружу. Но в моей воле пощадить тебя, ибо я вижу, что ты достаточно силен чтобы не позволить этому злу проникнуть в мысли и душу твои. Ты должен дать обет три месяца и три дня не покидать своей земли и каждый день три часа после захода солнца молить добрых духов о защите. Дела торопят меня, и не могу я препроводить тебя до замка. Поэтому я дарую тебе и дорогу от этого места до замка. А сейчас иди, и возвращайся по истечении срока."

— так записано в Зеленой Книге Лет.

Помимо этого из Зеленой Книги Лет известно, что земли, вместе с которыми был дарован замок, имели форму круга. Король был очень мудр и, во избежание лишних разбирательств относительно права на землю всегда даровал только области земли, на карте имеющие выпуклую форму. Недавно в распоряжении историков оказалась информация о том, где располагался замок и где происходил этот исторический разговор. Их интересует, участок земли какой площади получил Рыцарь в предположении, что дорога до замка была идеально прямой.

Формат входного файла

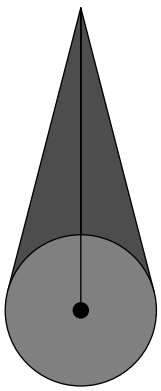
Первая входного файла содержит два вещественных числа x_k и y_k — координаты места, в котором происходил диалог. Во второй строке записаны три вещественных числа x_c, y_c и r_c — координаты замка и и радиус окружности, ограничивающей дарованную вместе с ним землю. Все числа во входном файле по модулю не превосходят 10000.

Формат выходного файла

В выходной файл выведите одно вещественное число — площадь земельного участка, полученного Рыцарем, с точностью не менее трех знаков после десятичной точки.

Примеры

estate.in	estate.out
2 5 2 1 1	5.69646



На рисунке светло-серым цветом показана территория, изначально дарованная рыцарю, а темно-серым, та, которая перешла к нему в результате того, что короля даровал ему дорогу.

Задача E. Задача о рюкзаке

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

Одной из классических NP-полных задач является так называемая *Задача о рюкзаке*. Формулируется она следующим образом.

Дано n предметов, каждый из которых характеризуется весом w_i и полезностью p_i. Необходимо выбрать некоторый набор этих предметов так, чтобы суммарный вес этого набора не превышал W, а суммарная полезность была максимальна.

Ваша задача состоит в том, чтобы написать программу, решающую задачу о рюкзаке.

Формат входного файла

Первая строка входного файла содержит натуральные числа n ($1 \le n \le 20$) и W ($1 \le W \le 10^9$). Каждая из последующих n строк содержит описание одного предмета. Каждое описание состоит из двух чисел: w_i — веса предмета и p_i — его полезности ($1 \le w_i, p_i \le 10^9$).

Формат выходного файла

В первой строке выходного файла выведите количество выбранных предметов и их суммарную полезность. Во второй строке выведите через пробел их номера в возрастающем порядке (предметы нумеруются с единицы в порядке, в котором они перечислены во входном файле).

Если искомых наборов несколько, выберите тот, в котором наименьшее число предметов. Если же после этого ответ по-прежнему неоднозначен, выберите тот набор, в котором первый предмет имеет наименьший возможный номер, из всех таких выберите тот, в котором второй предмет имеет наименьший возможный номер, и т.д.

Примеры

knapsack.in	knapsack.out
2 10 10 100 9 80	1 100 1
5 100 80 1000 50 550 50 550 50 550 50 550	2 1100 2 3
6 100 80 1000 50 550 50 550 50 550 50 550 100 1100	1 1100 6

Задача F. Четно-нечетная задача

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

Значит, чет благоприятен,
Нечет лишний в нашем деле.
(из песен группы «Зимовье зверей»)

Задано n чисел $a_1, a_2, \ldots, a_n$. Выберите из них четные числа, у которых третья справа цифра в их представлении в восьмеричной системе счисления нечетна.

Выбранные числа отсортируйте по неубыванию и выведите в выходной файл.

Формат входного файла

Первая строка входного файла содержит целое число n ($1 \le n \le 100000$). Вторая строка входного файла содержит n целых чисел: $a_1, a_2, \ldots, a_n$. Они отделены друг от друга пробелами и заданы в десятичной системе счисления. Для всех $i \in 1 \ldots n$ верно неравенство $64 \le a_i \le 10^9$.

Формат выходного файла

В первой строке выходного файл выведите количество k искомых чисел. Во второй строке выведите эти числа в указанном порядке в десятичной системе счисления.

Примеры

oddeven.in	oddeven.out
4 100 64 64 130	3 64 64 100
3 128 129 130	0

Задача G. Точки и линии

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

Вася и Петя играют в следующую игру. Вася рисует на плоскости N точек и говорит число M. Петя должен ответить, можно ли соединить эти N точек с помощью M линий, так, чтобы:

1) Каждая линия соединяет ровно две точки.

2) Никакая линия не соединяла точку саму с собой.

3) Для любой пары точек существует не более одной соединяющией их линии.

4) Существуют такие две точки A и B, что из A нельзя добраться вдоль нарисованных линий до B (по каждой линии можно идти в любую сторону).

Если Петя отвечает правильно, то он выигрывает, иначе выигрывает Вася.

Ваша задача — написать программу, которая поможет Пете всегда выигравать.

Формат входного файла

В первой строке входного файла записано число K — количество тестов во входном файле ($1 \le K \le 1000$). В следующих K строках записаны числа N ($1 \le N \le 10^9$) и M ($0 \le M \le 10^9$).

Формат выходного файла

Для каждого из K тестов в отдельной строке запишите правильный ответ на вопрос Васи, строку «`Yes`» (если ответ положительный) или «`No`» (если ответ отрицательный).

Примеры

points.in	points.out
2 1 0 2 2	Yes No

Задача H. Шаблон программы

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

Многие команды, участвующие в командных соревнованиях по программированию, используют так называемый «шаблон программы». Он набирается в самом начале соревнования и содержит общее для всех решений — например, открытие и закрытие входных и выходных файлов.

Трехкратный чемпион мира по версии *AMC* команда *Dream Team* — не исключение. Во многом их успехи связаны с тем, что они очень тщательно готовятся к соревнованиям, продумывая даже очень мелкие детали. Например, перед последним финалом они во время пробного тура рассчитали, сколько джоулей энергии потратится на набор шаблона.

Организаторы финала использовали весьма странные клавиатуры — жесткость различных клавиш была различной. Таким образом, на нажатие разных клавиш требовалось различное количество энергии.

Эксперименты, проведенные командой *Dream Team* во время пробного тура, показали следующее. На набор строчной буквы латинского алфавита требуется количество энергии, равное сумме цифр ее порядкового номера в алфавите (буквы нумерются с единицы). На нажатие клавиши «Shift» требуется 10 джоулей энергии (таким образом набор заглавной буквы латинского алфавита требует на 10 джоулей больше, чем набор соответствующей ей строчной буквы), нажатие клавиши «Пробел» требует 4 джоуля энергии. Набор цифры x требует $13 - x$ джоулей энергии, набор точки — 5 джоулей, точки с запятой — 7 джоулей, запятой — 2 джоуля. Знак равенства, плюс, минус, одинарная и двойная кавычка требуют по 3 джоуля энергии. Закрывающая и открывающая круглые скобки требуют по 1 джоулю, а фигурные, квадратные и угловые (т.е. символы $<$ и $>$) — по 8. При этом для всех упомянутых знаков препинания на клавиауре, используемой на финале, существуют отдельные клавиши, и другой возможности набрать соответствующий символ нет. Нажатие клавиши «Enter» (перевод строки) оказалось настолько легким, что энергозатраты на него можно считать нулевыми.

Ваша задача — написать программу, которая по тексту шаблона вычислит энергозатраты на его набор.

Формат входного файла

Входной файл содежит шаблон программы, энергетические затраты на набор которого необходимо вычислить. Он содержит только цифры, пробелы, строчные и заглавные буквы латинского алфавита, точки, запятые, знаки равенства, плюсы, точки с запятыми, двойные кавычки ("), одинарные кавычки (’), закрывающие и открывающие круглые, фигурные и квадратные скобки.

Его размер не превышает 20000 байт.

Формат выходного файла

В выходной файл выведите ответ на задачу.

Примеры

template.in	template.out
`abCD '+=1;2,3."()[]{}`	128

`program solution;`
`uses`
`  sysutils, math;`
`begin`
`  assign(input, '.in');`
`  assign(output, '.out');`
`end.`
	499