Задача A. Резервное копирование

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

Гениальные британские ученые в очередной раз потрясли мир новым великим изобретением. На этот раз они представили инновационную версию системы резервного копирования. Правда, как и у всех их изобретений, у нее есть один существенный недостаток. Этот недостаток заключается в том, что пока что она может быть реализована лишь для строк, состоящих из строчных символов латинского алфавита. Но когда и кого такие проблемы останавливали?

Перед вами поставлена задача реализовать эту систему. Суть системы достаточно проста. На вход подается по одному символу строки. Как только система встречает символ, который уже есть в текущей версии строки, она ищет его последнее вхождение и стирает (ну, естественно, не просто так стирает, а резервно копирует) все, что было записано в текущей версии строки после этого последнего вхождения. После чего поступивший символ дописывается к текущей версии. Естественно, что если последнее вхождение было непосредственно перед тем, как поступил текущий символ, то ничего не стирается и резервно не копируется.

Утверждается, что после всех этих действий, имея все резервно скопированные строки и то, что в итоге осталось в текущей версии, можно восстановить строку, которая подавалась на вход. Но мы не спрашиваем вас, как это сделать, а просим, наоборот, по строке, которая подается системе на вход, вывести все, что было из нее удалено и скопировано в том порядке, в котором это было сделано, а также вывести то, что в итоге осталось в текущей версии.

Формат входного файла

Во входном файле содержится одна строка, состоящая из строчных латинских символов — то, что системе подается на вход. Длина строки не превышает 200000 символов.

Формат выходного файла

Каждый раз, когда происходит резервное копирование, выведите в новой строке то, что было резервно скопировано. После того, как строка закончилась, выведите текущую версию.

Примеры

backup.in	backup.out
abcabcdbcd	bc cd aabbcd

Задача B. Шифровка

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

Петя и Вася любят играть в шпионов. А какие игры в шпионов обходятся без секретных шифровок! Вот и Вася недавно придумал новый способ шифрования своих сообщений. Информацию о том, как он шифрует, он оставил в секрете и сообщил Пете только способ расшифровки, чтобы тот всегда мог им воспользоваться.

Для расшифровки Пете необходимо всего лишь найти лексикографически максимальную подстроку в зашифрованном сообщении, которое ему передает Вася. Именно эта подстрока и будет исходным текстом. Поскольку Петя справляется с этим поиском не так быстро, как ему хотелось бы, он просит Вас написать программу, которая поможет ему в этом!

Формат входного файла

Первая и единственная строка входного файла содержит непустое зашифрованное сообщение, состоящее только из строчных букв латинского алфавита. Сообщение имеет длину не более 1000 символов.

Формат выходного файла

В выходной файл выведите расшифрованное сообщение.

Примеры

cipher.in	cipher.out
abacaba	caba
babb	bb

Задача C. Совпадения случайны

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

Теперь в Берляндии новый национальный спорт — спортивное программирование! Для его популяризации и организации была создана Организация, целью которой стало проведение онлайн-соревнований. Однако получилось так, что сначала Организация провела регистрацию n участников, присвоила каждому некоторый рейтинг и провела m соревнований, а только потом приняла регламент проведения соревнований и правила участия в них.

Самым главным новшеством стало введение запрета использования нескольких различных аккаунтов одним человеком, и сейчас Организация пытается определить всех потенциальных нарушителей. Как показал многолетний опыт, участник A может быть нарушителем только в том случае, если есть участник B такой, что выполнены все следующие условия:

1. A и B никогда не участвовали в одном соревновании
2. рейтинг B после последнего соревнования меньше, чем рейтинг A
3. A и B писали соревнования с одного IP-адреса
4. B выступал на уровне A во всех соревнованиях, в которых принимал участие. Считается, что участник B выступил на уровне участника A в некотором соревновании, если для любого $i < k$ верно что $highest(i) \le points(b)$ и $highest(k) > points(b)$, где $points(b)$ — результат участника B в этом соревновании, k — конечный рейтинг участника A, $highest(j)$ — маскимальное число баллов, набранное в текущем соревновании участником, чей конечный рейтинг равен j. При этом считается, что участник, набравший в соревновании масимальное число баллов, выступил в нем на уровне участника с рейтингом десять. Если участник не участвовал ни в одном соревновании, его уровень считается равным нулю.

Вам как ведущему аналитику Организации было поручено по результатам соревнований определить всех участников, которые могут быть нарушителями.

Формат входного файла

В первой строке входного файла записано два целых числа: n и m ($1 \le n, m \le 200$) — количество зарегистрированных участников и количество соревнований. В следующих n строках записаны данные учасников — в i-й строке записан логин s_i и текущий рейтинг r_i ($0 \le r_i \le 10$) участника с номером i, записанные через пробел. Логин участника может содержать только латинские буквы и может быть не длиннее 20 символов.

Далее находится m блоков, описывающих результаты соревнований. В первой строке кажого блока записано одно целое число k ($1 \le k \le n$) — количество участников, принимавших участие в этом соревновании. В следующих k строках записаны результаты каждого участника в следующем формате: логин участника s_j, целое число, показывающее количество набранных им в этом соревновании баллов, b_j ($0 \le b_j \le 2000$), целое число, описавающее изменение его рейтинга, d_j ($-10 \le d_j \le 10$) и IP-адрес, с которого этот участник писал контест, разделенные пробелами. При этом рейтинг участника не может опускаться ниже нуля и подниматься выше десяти. IP-адреса участников в течение серии соревнований не менялись. Каждый IP-адрес содержит только точки и цифры.

Формат выходного файла

В первой строке выходного файла выведите число l подозрительных участников. В следующих l строках выведите их логины в лексикографическом порядке.

Примеры

coincidence.in	coincidence.out
3 2 Petya 1 Vasya 2 Gena 3 2 Vasya 2 -2 10.0.0.1 Petya 10 2 10.0.0.2 2 Gena 100 0 10.0.0.1 Petya 10 0 10.0.0.2	1 Gena
2 2 Petya 10 Gena 3 1 Gena 1000 0 10.0.0.1 1 Petya 1000 0 10.0.0.1	1 Petya

Задача D. Марсианский друг

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

Десятиклассник Вася дружит с марсианином Хрулем. И сегодня Хруль попросил Васю помочь решить следующая задачу: необходимо найти на какое количество нулей заканчивается число $N!$. Вася сначала обрадовался, ведь он решал такую задачу в пятом классе на олимпиадном кружке по математике. Но вовремя вспомнил, что на Марсе k-ичная система счисления, и задумался.

Помогите Васе посчитать количество нулей в записи число $N!$ в k-ичной системе счисления.

Формат входного файла

Входной файл содержит два целых числа $N(1 \le N \le 10^{18})$ и $k(2 \le k \le 10^9)$.

Формат выходного файла

В выходной файл выведите одно целое число — ответ на задачу.

Примеры

mars.in	mars.out
20 7	2

Задача E. Деловые встречи

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

Алексей — успешный предприниматель и в течении одного дня у него бывает много встреч с разными деловыми партнерами. К сожалению, встречи бывают разные и не все приносят ему радость и после них настроение улучшается. Также, на многие встречи не стоит приходить в слишком плохом или хорошем настроении — результат таких встреч может быть не таким, какой хочется Алексею.

К счастью, недавно Алексей научился оценивать свое настроение с помощью целых чисел. После этого для каждой встречи он оценил при каком максимальном и минимальном настроении стоит на нее приходить, а также как изменится его настроение после этой встречи. Теперь он хочет распланировать порядок встреч так, чтобы в течении дня совершить максимальное число встреч.

Ваша задача — написать программу, которая по информации о всех встречах и настроении Алексея в начале дня находит порядок проведения встреч такой, что их количество при этом максимально.

Формат входного файла

Первая строка входного файла содержит два целых числа n и k ($1 \le n \le 20$, $|k| \le 100$) — количество встреч и настроение Алексея в начале дня.

Следующие n строк содержат по три целых числа a_i, b_i и c_i ($|a_i|, |b_i|, |c_i| \le 100$) — минимальное, максимальное настроение при котором встреча возможна и изменение настроения по окончании встречи, соответственно.

Формат выходного файла

В первой строке выходного файла выведите число m — максимально возможно число встреч. В следующей строке выведите m целых чисел — номера встреч в порядке их проведения. Встречи пронумерованы в порядке описания во входном файле.

Если ответов с максимальным числом встреч несколько, выведите любой.

Примеры

meetings.in	meetings.out
3 0 1 3 3 0 1 2 1 3 1	3 2 3 1
3 1 -10 -5 3 -5 5 -2 -3 2 1	2 3 2

Задача F. Сообщение

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

Петя и Вася продолжают играть в шпионов. На этот раз Петя прислал Васе закодированное сообщение, ключами для декодирования которого являются два простых числа p и q. Петя сообщил Васе, что эти числа связаны соотношением $q = 2p + 1$, и p - это k-е по величине простое число для которого существует простое q, удовлетворяющее выше написанному соотношению.

Помогите Васе прочитать сообщение — найдите такое p.

Формат входного файла

Единственная строка входного файла содержит целое число $k(1 \le k \le 30000)$ — порядковый номер числа p.

Формат выходного файла

В выходной файл выведите единственное число p.

Примеры

message.in	message.out
1	2
4	11

Задача G. Суперзвезда

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

В альтернативной геометрии, изучаемой последователями доктора Иллитида, одной из ключевых фигур является n-угольная суперзвезда. Она используется для того, чтобы произносить заклинания, призывать Ктулху и делать много других крайне полезных вещей. Описание процесса построения фигуры приведено практически в любом справочнике для юных последователей, и состоит всего из трех пунктов:

- возьмите окружность радиусом 10 метров и поставьте на ней n точек так, что длины дуг между всеми парами соседних точек одинаковы
- пронумеруйте точки от 0 до $n - 1$, обходя окружность по часовой стрелке
- соедините отрезками те и только те точки, для которых верно равенство $(a + 2) \mod n = b$, где a и b — номера соединяемых точек

После этого вы можете встать в центр окружности и призвать Ктулху, у которого будет ровно n щупалец. Все, что вам для этого понадобится — правильно назвать площадь нарисованной вами звезды. Напишите программу, решающую эту задачу за вас.

Формат входного файла

Входной файл содержит одно целое число $n (5 \le n \le 100)$ — количество щупалец у Ктулху, которого вы хотите призвать.

Формат выходного файла

В выходной файл выведите одно целое число — ответ на задачу, выведенный с точностью не меньшей, чем 5 знаков после запятой.

Примеры

superstar.in	superstar.out
5	112.25699414489632

Задача H. Путешествие

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

Заскучала Лида в северном городе Санкт-Петербурге. Решила она отправиться путешествовать вместе с друзьями, поехала на железнодоржный вокзал и столкнулась там со следующей системой покупки билетов: один запрос к системе бронирует все свободные места на отрезке от k до r ($1 \le k \le r \le 54$) включительно, игнорируя все занятые места на этом отрезке. Так как Лида очень торопится, то она хочет сделать минимальное количество запросов на покупку билетов, а так, как она ещё и знает на каком месте хочет поехать каждый из ее друзей, то она будет брать билеты только на эти места.

Зная список занятых мест и необходимых Лиде билетов, помогите ей купить их за минимальное число запросов.

Формат входного файла

В первой строке входного файла содержится два целых числа n ($0 \le n \le 54$) — количество уже занятых мест, и l ($0 \le l \le 54$) — сколько людей собралось ехать. Во второй строке содержится n различных чисел a_i ($1 \le a_i \le 54$) — список занятых мест. В третьей строке содержится l различных чисел b_i ($1 \le b_i \le 54$) — места, на которых хотят ехать Лида и её друзья.

Формат выходного файла

В первой строке выходного файла выведите одно целое число p — количество запросов. Далее выведите p строк, каждая из которых содержит пару чисел: левую и правую границы номеров билетов в очередном запросе. Если купить билеты так, как хотят Лида и ее друзья нельзя, то выведете -1.

Примеры

travel.in	travel.out
2 3 2 5 1 3 7	2 1 3 7 7