**Практическая работа 5**

**Питон. Строки.**

***Указание:*** для решения задачи написать хотя бы одну вспомогательную функцию.

**Вариант 1**

**Задача 1**

Вася получает доступ в Интернет с помощью мобильного телефона. Однако такая связь очень нестабильна, поэтому каждый раз после подключения ему приходиться проверять подключение. Для этого он просто в командной строке набирает примерно следующую фразу: «ping name», где name – это имя удаленного сервера, который точно находится в сети. Затем идет обмен пакетами с сервером и выдается статистика. Однако у Васи недавно сломался модуль, отвечающий за подсчет и вывод статистики. Вам надо будет помочь Васе – написать аналогичный модуль.

После вызова команды «ping» на удаленный сервер по очереди посылаются 4 пакета по 32 байта. Как только удаленный сервер получил пакет, он отвечает на него. Если пакет не уложился в определенное время (он должен дойти до удаленного сервера и вернуться обратно) в силу тех или иных причин (низкая скорость, отсутствие подключения и т.д.), он считается утерянным.

Дана информация обо всех 4 пакетах. Требуется определить количество потерянных пакетов, максимальное, минимальное и среднее время обмена одного пакета.

**Входные данные**

Входные данные содержат ровно 5 строк. В первой строке находится фраза «ping name», где name – это имя сервера. Имя сервера представляет собой IP адрес. IP-адрес – это 4 однобайтных числа (т.е. числа от 0 до 255), отделенные друг от друга точкой. В каждой из следующих 4 строк содержится либо фраза «Time out», если пакет считается утерянным, либо «Reply from name Time=number», где name – это имя удаленного сервера, а number – время за которое вернулся пакет (number – целое число, 0 ≤ number ≤ 104).

**Выходные данные**

Выведите статистику по обмену пакетами с удаленным сервером. Следуйте формату, приведенному в примере. Среднее время округлите до целого числа по математическим правилам. Если все 4 пакета утеряны, то выведите только первые две строки.

Пример

|  |  |  |
| --- | --- | --- |
| **№** | **INPUT.TXT** | **OUTPUT.TXT** |
| 1 | ping 209.85.135.147 Time out Reply from 209.85.135.147 Time=100 Reply from 209.85.135.147 Time=300 Reply from 209.85.135.147 Time=200 | Ping statistics for 209.85.135.147: Packets: Sent = 4 Received = 3 Lost = 1 (25% loss) Approximate round trip times: Minimum = 100 Maximum = 300 Average = 200 |

**Задача 2**

Билет называется счастливым, если его можно разрезать прямой линией между цифр на две части таким образом, что оказавшиеся на них числа имеют одинаковые цифровые корни. Чтобы вычислить цифровой корень числа, его цифры складывают, если в результате получится число большее или равное 10, то цифры складывают снова и так далее, пока не получится число от 0 до 9 – это и есть цифровой корень. Например, билет с номером 0015420 является счастливым, так как разрезав его на части с числами 0015 и 420 имеем у этих чисел одинаковые цифровые корни.

Требуется написать программу, которая определит, является ли счастливым билет с заданным номером.

**Входные данные**

Содержат номер счастливого билета. Номер может начитаться с нулей и содержит от 1 до 100 цифр.

**Выходные данные**

Выведите «YES», если билет счастливый и «NO» иначе.

**Примеры**

|  |  |  |
| --- | --- | --- |
| **№** | **INPUT.TXT** | **OUTPUT.TXT** |
| 1 | 0015420 | YES |
| 2 | 00100 | NO |

**Вариант 2**

**Задача 1**

На одной из лекций по информатике студент Петя узнал про новый шифр - простой замены. Он и на самом деле прост: в тексте каждая буква алфавита заменяется некоторой другой буквой того же алфавита (может быть, той же самой).

Петя написал письмо своему другу Васе. Письмо - это текст из нескольких строк, написанный на английском языке, с использованием только строчных английских букв и пробелов. В произвольное место, отдельной строкой Петя вставил ключевую фразу: "the quick brown fox jumps over the lazy dog", о которой они с Васей договорились заранее. После чего зашифровал письмо. Известно, что пробелы в письме не шифруются. Получив такое письмо, Вася сумеет его расшифровать и прочесть. Иногда Петя ошибается, и забывает вставить ключевую фразу. Увы, в этом случае прочесть письмо невозможно.

Так как процесс расшифровки трудоемок, Вася просит написать программу, с помощью которой он сможет быстро расшифровывать письмо от Пети.

**Входные данные**

Первая строка входного файла INPUT.TXT содержит целое число N – количество строк в письме (1 ≤ N ≤ 200). Далее идет N строк письма (пустые строки отсутствуют, в каждой строке не более 80 символов).

**Выходные данные**

В выходной файл OUTPUT.TXT в случае присутствия в тексте ключевой фразы выведите N строк расшифрованного сообщения. Если ключевой фразы нет, следует вывести «No solution» (без кавычек). Гарантируется, что есть не более одного способа расшифровки текста из входных данных.

**Примеры**

|  |  |  |
| --- | --- | --- |
| **№** | **INPUT.TXT** | **OUTPUT.TXT** |
| 1 | 3 vtz ud xnm xugm itr pyy jttk gmv xt otgm xt xnm puk ti xnm fprxq xnm ceuob lrtzv ita hegfd tsmr xnm ypwq ktj frtjrpgguvj otvxmdxd prm iev prmvx xnmq | now is the time for all good men to come to the aid of the party the quick brown fox jumps over the lazy dog programming contests are fun arent they |
| 2 | 3 vtz ud xnm xugm itr pyy jttk gmv xt otgm xt xnm puk ti xnm fprxq xnm fffff lrtzv iia wwwfd tsmr xnm ypwq ktj frtjrpgguvj otvxmdxd prm iev prmvx xnmq | No solution |

**Задача 2**

Натуральные числа записаны в строку без каких-либо разделителей. Начало этой строки имеет вид 123456789101112131415161718192021… .

Требуется написать программу, которая определит первое вхождение десятичной записи заданного числа N в этой строке.

**Входные данные**

Содержат заданное число N (1 ≤ N ≤ 104).

**Выходные данные**

Выведите одно число – позицию, начиная с которой в строке записано первое вхождение заданного числа. Нумерация позиций начинается с единицы.

**Примеры**

|  |  |  |
| --- | --- | --- |
| **№** | **INPUT.TXT** | **OUTPUT.TXT** |
| 1 | 45 | 4 |
| 2 | 101 | 10 |
| 3 | 142 | 73 |

**Вариант 3**

**Задача 1**

Циклическим сдвигом строки s называется строка sksk+1sk+2…s|s|s1s2…sk-1 для некоторого k, здесь |s| - длина строки s. Подстрокой строки s называется строка sisi+1…sj-1sj для некоторых i и j. Вам даны две строки a и b. Выведите количество подстрок строки a, являющихся циклическими сдвигами строки b.

**Входные данные**

Первая строка содержит строку a (1 ≤ |a| ≤ 1000). Во второй строке записана строка b (1 ≤ |b| ≤ min(100,|a|)). Обе строки состоят только из символов английского алфавита и цифр.

**Выходные данные**

Выведите целое число – ответ на задачу.

**Примеры**

|  |  |  |
| --- | --- | --- |
| **№** | **INPUT.TXT** | **OUTPUT.TXT** |
| 1 | abcabc abc | 4 |
| 2 | abcabc acb | 0 |
| 3 | aaaaaaa aa | 6 |
| 4 | aAaa8aaAa aAa | 4 |

**Задача 2**

Будем рассматривать только строчки, состоящие из заглавных английских букв. Например, рассмотрим строку AAAABCCCCCDDDD. Длина этой строки равна 14. Поскольку строка состоит только из английских букв, повторяющиеся символы могут быть удалены и заменены числами, определяющими количество повторений. Таким образом, данная строка может быть представлена как 4AB5C4D. Длина такой строки 7. Описанный метод мы назовем упаковкой строки.

Напишите программу, которая берет упакованную строчку и восстанавливает по ней исходную строку.

**Входные данные**

Входные данные содержат одну упакованную строку. В строке могут встречаться только конструкции вида nA, где n — количество повторений символа (целое число от 2 до 99), а A — заглавная английская буква, либо конструкции вида A, то есть символ без числа, определяющего количество повторений. Строка содержит от 1 до 80 символов.

**Выходные данные**

Выведите восстановленную строку. При этом строка должна быть разбита на строчки длиной ровно по 40 символов (за исключением последней, которая может содержать меньше 40 символов).

**Примеры**

|  |  |  |
| --- | --- | --- |
| **№** | **INPUT.TXT** | **OUTPUT.TXT** |
| 1 | 3A4B7D | AAABBBBDDDDDDD |
| 2 | 22D7AC18FGD | DDDDDDDDDDDDDDDDDDDDDDAAAAAAACFFFFFFFFFF FFFFFFFFGD |
| 3 | 95AB | AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA AAAAAAAAAAAAAAAB |
| 4 | 40AB39A | AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA BAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA |

**Вариант 4**

**Задача 1**

Археологами найден набор древних копий старинных манускриптов с мифами – различными историями о древних богах. К несчастью, переписчики этих манускриптов не отличались особой грамотностью и умудрились в каждом имени сделать ровно по одной орфографической ошибке – т.е ровно одну из букв божественного имени заменили какой-то другой буквой. Археологи смогли составить список правильных написаний имен богов, так же им удалось выписать из манускриптов все имена собственные. Однако сопоставлять два списка – свыше их сил. Помогите им в этом!

**Входные данные**

Первая строка содержит число N – количество имен богов в списке. Следующие N строк – имена богов. Далее идет строка, содержащая число M – количество «подозрительных» слов, выписанных из манускриптов. Следующие M строк – «подозрительные» слова. Каждое из имен богов и «подозрительных» слов – последовательность из K заглавных букв английского алфавита (1 ≤ N, M, K ≤ 30).

**Выходные данные**

Выведите N чисел – для каждого божьего имени выводится **число** “подозрительных” слов, которые являются именем данного бога с одной ошибкой.

**Пример**

|  |  |  |
| --- | --- | --- |
| **№** | **INPUT.TXT** | **OUTPUT.TXT** |
| 1 | 3 ZEUS POSEIDON AFINA 4 ZEVS POSEYDON AVYNA ZERS | 2 1 0 |

**Задача 2**

Руны — это древние магические знаки, которые наши предки использовали как буквы. Говорят, что рунные знаки обладают магическими свойствами, а при сложении рун в слова их магическая сила многократно возрастает. Если кузнец изготовит доспехи и начертит там определенные руны в определенном порядке, то доспехи будут наделены необычайными магическими силами.

Для того, чтобы стать обладателем таких доспехов достаточно просто принести кузнецу начертания этих рунных знаков. А вот, чтобы стать обладателем рунного знака приходилось немало потрудиться. Воины добывали начертания рун других языков и наречий в боях или получали их в качестве наград в благодарность за оказанные услуги.

Но так или иначе и в этом деле развелись жулики. По подозрениям ученых кузнец Игнатус Мошеникус изготавливал благородным воинам фальшивые рунные слова. Из древних преданий ученым стало достоверно известно, что каждая руна записывается из двух, трех или четырех английских букв. Причем первая буква рунного слова всегда записывается как заглавная, а все остальные являются маленькими. Ученые перевели несколько, выкованных этим кузнецом, рунных слов на английский язык и теперь нуждаются в Вашей помощи. Проверьте, является ли приведенное слово рунным.

**Входные данные**

В единственной строке содержится слово. Оно представляет собой непустую строку, длиной не более 100000 символов, содержащую только большие и маленькие буквы английского алфавита.

**Выходные данные**

Выведите «Yes», если слово является рунным и «No» в противном случае.

**Пример**

|  |  |  |
| --- | --- | --- |
| **№** | **INPUT.TXT** | **OUTPUT.TXT** |
| 1 | IoIsTheBest | Yes |
| 2 | IoItIsWaste | No |

**Вариант 5**

**Задача 1**

Уравнение для пятиклассников представляет собой строку длиной 5 символов. Второй символ строки является либо знаком '+' (плюс) либо '-' (минус), четвёртый символ — знак '=' (равно). Из первого, третьего и пятого символов ровно два являются цифрами из диапазона от 0 до 9, и один — буквой x, обозначающей неизвестное.

Требуется написать программу, которая позволит решить данное уравнение относительно x.

**Входные данные**

Входные данные состоят из одной строки, в которой записано уравнение.

**Выходные данные**

Выведите целое число — значение x.

**Примеры**

|  |  |  |
| --- | --- | --- |
| **№** | **INPUT.TXT** | **OUTPUT.TXT** |
| 1 | x+5=7 | 2 |
| 2 | 3-x=9 | -6 |

**Задача 2**

В одном очень известном университете один очень известный профессор очень быстро произносил свои лекции, так, что ничего невозможно было разобрать. Но недавно студент Вилли решил провести исследование по изучению словарного запаса профессора. С этой целью он даже посетил одну лекцию и записал всё сказанное на ней на диктофон. Затем, прокручивая дома запись с десятикратным замедлением, Вилли смог записать всё, что сказал профессор.

Но вот незадача – профессор говорил так быстро, что, даже прослушивая замедленную запись, нельзя было точно сказать, где он делал паузы между словами. Таким образом, у Вилли есть некоторый текст, состоящий из n маленьких английских букв – лекция, которая была прочитана профессором. Теперь Вилли хочет знать, какое количество различных слов длины m мог использовать в своей лекции профессор.

**Входные данные**

Первая строка содержит два числа n и m (1 ≤ m ≤ n ≤ 100) – длина лекции и длина слова. Вторая строка содержит n английских символов - текст лекции профессора.

**Выходные данные**

Выведите одно число – количество слов длины m, которые профессор мог использовать в своей лекции.

**Примеры**

|  |  |  |
| --- | --- | --- |
| **№** | **INPUT.TXT** | **OUTPUT.TXT** |
| 1 | 3 1 abc | 3 |
| 2 | 10 3 bbaabbbabb | 6 |

**Вариант \***

**Задача 1.**

Для строки T и целого числа n определим n-ю степень строки Tn как конкатенацию n копий строки T. Например, aab4 = aabaabaabaab.

Любая строка S может быть представлена в виде разложения S = S1d1S2d2 ... Skdk. Вообще говоря, такое разложение может быть не единственным. Весом разложения строки S в указанном виде назовем сумму |S1| + |S2| + . . . + |Sk|, где |Z| означает длину строки Z.

По заданной строке S найдите ее разложение с минимальным весом.

**Входные данные**

Входные данные содержат строку S. S состоит из заглавных английских букв и имеет длину не более 5000.

**Выходные данные**

Первая строка должна содержать w – минимальный возможный вес разложения строки S. Пусть k – число элементов в таком разложении. Тогда следующие k строк должны содержать элементы разложения: строку Si и степень di, разделенные ровно одним пробелом.

Если существует несколько оптимальных решений, выведите любое из них.

**Пример**

|  |  |  |
| --- | --- | --- |
| **№** | **INPUT.TXT** | **OUTPUT.TXT** |
| 1 | ABABAAABABA | 5 AB 2 A 3 BA 2 |

**Задача 2**

Строка s называется супрефиксом для строки t, если t начинается с s и заканчивается на s. Например, «abra» является супрефиксом для строки «abracadabra». В частности, сама строка t является своим супрефиксом. Супрефиксы играют важную роль в различных алгоритмах на строках.

В этой задаче требуется решить обратную задачу о поиске супрефикса, которая заключается в следующем. Задан словарь, содержащий n слов t1, t2, …, tn и набор из m строк-образцов s1, s2, …, sm. Необходимо для каждой строки-образца из заданного набора найти количество слов в словаре, для которых эта строка-образец является супрефиксом.

Требуется написать программу, которая по заданному числу n, n словам словаря t1, t2, …, tn, заданному числу m и m строкам-образцам s1, s2, …, sm вычислит для каждой строки-образца количество слов из словаря, для которых эта строка-образец является супрефиксом.

**Входные данные**

Первая строка содержит целое число n (1 ≤ n ≤ 200 000). Последующие n строк содержат слова t1, t2, …, tn, по одному слову в каждой строке. Каждое слово состоит из строчных букв английского алфавита. Длина каждого слова не превышает 50. Суммарная длина всех слов не превышает 106. Словарь не содержит пустых слов.

Затем следует строка, содержащая целое число m (1 ≤ m ≤ 200 000). Последующие m строк содержат строки-образцы s1, s2, …, sm, по одной на каждой строке. Каждая строка-образец состоит из строчных букв английского алфавита. Длина каждой строки-образца не превышает 50. Суммарная длина всех строк-образцов не превышает 106. Никакая строка-образец не является пустой строкой.

**Выходные данные**

Выведите m чисел, по одному на строке. Для каждой строки-образца в порядке, в котором они заданы во входном файле, следует вывести количество слов словаря, для которых она является супрефиксом.

**Пример**

|  |  |  |
| --- | --- | --- |
| **№** | **INPUT.TXT** | **OUTPUT.TXT** |
| 1 | 4 abacaba abracadabra aa abra 3 a abra abac | 4 2 0 |