Задача 1. Спокойной ночи 100 баллов

Ограничение по времени – 1 секунда.
Ограничение по памяти – 1 мегабайт.
Входные данные – стандартный ввод.
Выходные данные – стандартный вывод.

Кот Матроскин очень любит подремать у телевизора. Он прекрасно понимает, что, когда он уже уснул, смысла в работе телевизора нет, поэтому Матроскин освоил таймер отключения, позволяющий указать необходимое количество минут, через которое телевизор будет автоматически выключен. Чисто теоретически нашему коту интересно, во сколько это произойдёт, если он прекрасно помнит время, в которое он установил таймер. Поможем коту реализовать его чисто теоретический интерес.

Требуется написать программу, которая по известному времени, когда был установлен таймер, и значению таймера N минут, позволяет определить, в какое время телевизор выключится.

Формат входных данных

Первая строка содержит время, когда был установлен таймер, в формате «чч:мм» ($00:00 \le$ чч:мм $\le 23:59$). Вторая строка содержит целое число N ($0 \le N \le 10^5$) – значение таймера в минутах.

Формат выходных данных

Одна строка – время, в которое будет выключен прибор в формате «чч:мм».

Пример входных и выходных данных

Стандартный ввод	Стандартный вывод
22:10 53	23:03
23:10 70	00:20

Описание системы оценивания

Баллы начисляются за каждый пройденный тест.

Решение

Для решения задачи воспользуемся целочисленными арифметическими операциями. При решении задачи необходимо быть внимательным относительно форматов представления данных.

Пример программы для решения задачи на языке программирования Python.

```
h, m = map(int, input().split(':'))
d = int(input())
h1, m1 = (h*60+m+d)//60%24, (h*60+m+d)%60
print(f'{h1:02}:{m1:02}')
```

Задача 2. «Сочувствующая строка» 100 баллов

Ограничение по времени – 1 секунда.
Ограничение по памяти – 1 мегабайт.
Входные данные – стандартный ввод.

Выходные данные – стандартный вывод.

Хватайка после усердных занятий с Матроскиным начал говорить «Кто там?». Будучи чрезвычайно разумным галчонком, он стал внимательно изучать различные слова и с удивлением обнаружил, что есть такие слова, как палиндромы. Хватайке стало интересно, можно ли получить палиндром из любого слова или даже фразы, убрав из него всего пару букв или символов. Хватайка даже придумал термин для такого текста – «сочувствующая палиндрому строка». Это строка, из которой получается палиндром путём удаления двух символов.

Требуется написать программу, которая определяет, является ли исходная строка «сочувствующей палиндрому», и найти все палиндромы, которые могут из неё получиться путём удаления двух символов. Нумерация символов строки начинается с 1, слева направо.

Палиндром – это число, буквосочетание, слово или текст, одинаково читающийся в обоих направлениях.

Формат входных данных

Одна строка, состоящая из символов латиницы, цифр и знаков препинания для проверки на «сочувствие палиндрому». Количество символов в строке – N ($3 \le N \le 10^5$).

Формат выходных данных

В первой строке одно натуральное число M – количество найденных палиндромов, полученных путём удаления двух символов. Далее M строк, каждая из которых состоит из двух натуральных чисел, разделённых пробелом – номера символов, которые необходимо удалить, в порядке возрастания. Если таких вариантов несколько, то строки записываются в порядке возрастания номеров. При отсутствии вариантов – одна строка с сообщением NO.

Пример входных и выходных данных

Стандартный ввод	Стандартный вывод
abbac	2 2 5 3 5
abc	NO

Описание системы оценивания

Баллы начисляются за каждый пройденный тест.

Решение

Следует обратить внимание, что решение задачи требует аккуратности в программировании, так как используются различные способы чтения данных: имена авторов – в одну строку через пробел, словарь названий – построчно.

Пример программы для решения задачи на языке программирования Python.

```
# функция для определения палиндрома
def plnd(a):
    return a == a[::-1]

n = input()
# Проверка наличия ответа
flag = True
k = 0
slv = []
# Перебор с удалением двух возможных символов
```

```
for i in range(len(n)):
    # удаляем первый символ
    b = n[:i]+n[i+1:]
    for j in range(i,len(b)):
        # удаляем второй символ
        b1 = b[:j]+b[j+1:]
        if plnd(b1):
            # Увеличиваем счётчик и запоминаем
            # номера символов  в список
            k += 1
            slv.append(str(i+1)+' '+str(j+2))
            flag = False
# Вывод данных
if flag:
    print('NO')
else:
    print(k)
    for i in slv:
        print(i)
```

Задача 3. Громкое имя — 100 баллов

Ограничение по времени – 1 секунда.
Ограничение по памяти – 1 мегабайт.
Входные данные – стандартный ввод.
Выходные данные – стандартный вывод.

Жители деревни Простоквашино очень любят изобретать различные приспособления, которые могут быть полезны в хозяйстве. В создание очередного приспособления простоквашинцы вкладывают много сил, энергии и души, поэтому каждому устройству присваивается своё имя.

Чтобы дать название вновь созданному устройству, авторский коллектив решил применить эзотерический подход. Во-первых, решение принималось на основе данных нумерологии, где особую роль играет так называемое число Пифагора. Для его получения необходимо сложить все цифры, входящие в запись официальной даты создания устройства, цифры полученного числа необходимо снова сложить и так до тех пор, пока не получится однозначное число (цифра) – число Пифагора. При этом авторский коллектив решил, что название их «детища» обязательно должно содержать первые буквы имён авторов. При этом если имена нескольких авторов начинаются с одной и той же буквы, то эта буква должна входить в название то же количество раз.

Название устройства выбиралось из специального перечня (словаря), исходя из следующих соображений:

• количество символов в названии должно быть равно числу Пифагора;

• название должно содержать первые буквы имён всех членов авторского коллектива.

Требуется выбрать из предложенного словаря название, удовлетворяющее условию задачи.

Формат входных данных

В первой строке находится дата создания устройства в формате «дд.мм.гггг». В следующей строке через пробел записаны имена создателей. В третьей строке записано

одно число N ($1 \le N \le 10^3$) – количество названий в перечне. В каждой из следующих строк располагается одно название. Все имена авторов и названия состоят из строчных букв латинского алфавита, содержат не более 100 символов.

Формат выходных данных

Выведите название устройства, удовлетворяющее всем условиям. Если вариантов несколько, выведите первое в алфавитном порядке подходящее название. Если вариантов нет, выведите сообщение NO.

Пример входных и выходных данных

Стандартный ввод	Стандартный вывод
01.02.0310 fedor sharik gavrusha 3 hstgvkl afsgbers eugtfrs	eugtfrs

Описание системы оценивания

Баллы начисляются за каждый пройденный тест.

Решение

Следует обратить внимание, что решение задачи требует аккуратности в программировании, так как используются различные способы чтения данных: имена авторов – в одну строку через пробел, словарь названий – построчно.

Пример программы для решения задачи на языке программирования Python.

```
# Есть ли все буквы списка «a» в слове «w»
def isword(a,w):
    k = 0
    for i in a:
        if i in w:
            w = w.replace(i,'_',1)
            k += 1
    return k == len(a)


f = open('input.in')
# Считываем дату
d = f.readline()
# Найдём сумму всех чисел
s = 0
for i in d:
    if i.isnumeric():
        s += int(i)
if s > 10:
    s = int(str(s)[0])+int(str(s)[1])
# Считаем в список первые буквы
# имён (для проверки нужны только они)
nm = f.readline().rstrip().split(' ')
for i in range(len(nm)):
    nm[i] = nm[i][0]
# Будем считывать возможные имена и проверять их
# на соответствие условиям задачи
```

```
# Запоминаем нужные имена в переменной slv
# с учётом алфавитного порядка
n = int(f.readline())
slv = ''
for i in range(n):
    t = f.readline().rstrip()
    if len(t) == s and isword(nm,t):
        if slv == '':
            slv = t
        elif slv > t:
            slv = t
f.close()
# Вывод результата
if slv != '':
    print(slv)
else:
    print('NO')
```

Задача 4. Точный учёт — 100 баллов

Ограничение по времени – 1 секунда.
Ограничение по памяти – 1 мегабайт.
Входные данные – стандартный ввод.
Выходные данные – стандартный вывод.

Игорь Иванович Печкин, почтальон, работающий в Простоквашинском почтовом отделении, всегда стремится к порядку. И в этом ему способствует его ум, хитрость и занудный характер. Все поступающие в его отделение письма обязательно нумеруются. Причём если письма поступают от одного адресанта, то им присваивается одинаковый номер. Вся поступающая корреспонденция раскладывается в два ящика в произвольном порядке.

Почтальон Печкин для оптимизации процесса решил выяснить есть ли в ящиках письма с одинаковыми номерами, так как, по его умозрению, их надо доставить в первую очередь.

Если мы сможем помочь Игорю Ивановичу в решении этой задачи и быстро определить номера нужных писем в отсортированном в порядке возрастания виде, то будет очень здóрово.

Требуется определить количество значений, встречающихся в обоих неупорядоченных наборах целых чисел (возможно с повторениями) вывести их в порядке возрастания без повторений.

Формат входных данных

В первой строке через пробел находится два целых числа N и M ($1 \le N, M \le 10^5$) – количество элементов первого и второго наборов соответственно. Во второй строке записано N чисел первого набора через пробел. В третьей строке записано M чисел второго набора через пробел. Каждое из этих чисел попадает в промежуток $[0; 10^5]$.

Формат выходных данных

В первой строке одно число, количество чисел, которые есть как в первом, так и втором списках. Во второй строке K чисел, входящих как в первый, так и во второй набор,

разделённые одним пробелом. Если таких чисел нет, то выходной файл должен содержать одну строку NO.

Пример входных и выходных данных

Стандартный ввод	Стандартный вывод
5 7 4 25 6 4 8 25 5 7 13 8 8 9	2 8 25
5 7 4 25 6 4 8 9 5 7 13 7 9 7	NO

Описание системы оценивания

Баллы начисляются за каждый пройденный тест.

Решение

Решение задачи удобно провести с использованием множеств как совокупностей неповторяющихся элементов.

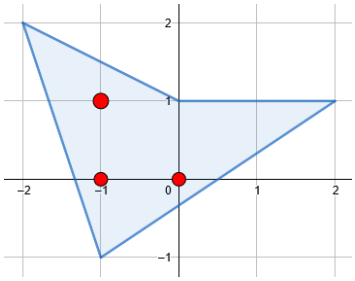
Пример программы для решения задачи на языке программирования Python.

```
f = open('input.in')
# считываем данные
a = f.readline()
# заполняем множества, что позволяет избежать повторений
a = set(map(int, f.readline().rstrip().split()))
b = set(map(int, f.readline().rstrip().split()))
f.close()
# находим пересечение множеств и строим список
c = list(a & b)
# сортируем
c.sort()
# выводим результат
print(len(c))
for i in c:
    print(i,' ',sep='',end='')
```

Задача 5. Хозяйство по уму — 100 баллов

Ограничение по времени – 1 секунда.
Ограничение по памяти – 1 мегабайт.
Входные данные – стандартный ввод.
Выходные данные – стандартный вывод.

Дядя Фёдор хочет посадить ягодные кусты на своём приусадебном участке. Он очень хозяйственный мальчик и поэтому планирует посадки заранее. Было принято решение садить кусты строго горизонтально и вертикально на расстоянии в 1 метр (по горизонтали и вертикали). К сожалению, участок, доставшийся Дяде Фёдору в Простоквашино, имеет сложную форму произвольного многоугольника. Но он не сдаётся и продолжает планировать посадки. С этой целью Дядя Фёдор ввёл прямоугольную систему координат и решил сажать кусты только в те места,



которые имеют целочисленные координаты. При этом если место для посадки куста попадает на границу участка, то там сажать нельзя (там же забор).

Помогите Дяде Фёдору определиться с количеством кустов, которые необходимо закупить для посадок.

Требуется по заданным координатам вершин в порядке обхода произвольного многоугольника подсчитать количество точек с целочисленными координатами, лежащих внутри него.

Формат входных данных

В первой строке содержится N ($3 \le N \le 10^3$) – число вершин многоугольника. В последующих N строках идут целочисленные координаты ($-10^6 \le X_i, Y_i \le 10^6$) вершин многоугольника в порядке их обхода.

Формат выходных данных

В выходной файл вывести одно число — искомое число точек.

Пример входных и выходных данных

Стандартный ввод	Стандартный вывод
4 -1 -1 -2 2 0 1 2 1	3
3 0 0 0 2 2 0	0

Описание системы оценивания

Баллы начисляются за каждый пройденный тест.

Решение

Один из наиболее простых вариантов решения данной задачи – использование формулы Пика. Формула Пика говорит о том, что если вершины многоугольника расположены в точках с целочисленными координатами, то выполнено следующее равенство:

$$S = K + \frac{M}{2} - 1,$$

где S – площадь произвольного многоугольника, K – число целых точек, лежащих внутри многоугольника, M – число целых точек с целочисленными координатами, лежащих на границе многоугольника. Таким образом, требуется найти значения параметров S и M.

Площадь многоугольника S можно найти, например, с помощью метода треугольников (формула Гаусса):

$$S = \frac{1}{2}(x_1y_2 + x_2y_3 + \cdots + x_{n-1}y_n + x_ny_1 - x_2y_1 - x_3y_2 - \cdots - x_ny_{n-1} - x_1y_n).$$

Число точек с целочисленными координатами M можно найти, зная координаты концов отрезка (целые числа). Заметим, что ответом на последний вопрос будет:

$$\text{НОД}(\Delta_x, \Delta_y) + 1,$$

где $\Delta_x = x_2 - x_1$, $\Delta_y = y_2 - y_1$.

Остаётся только найти сумму точек на всех отрезках (сторонах).

Пример программы для решения задачи на языке программирования Python.

```
# Функция НОД(a,b)
```

```
def nod(a,b):
    while a != 0 and b != 0:
         if a >= b:
             a %= b
         else:
             b %= a
    return a or b

p = []
f = open('input.in')
n = int(f.readline())
for i in range(n):
    p.append(tuple(map(float, f.readline().split())))
f.close()
S = 0
M = 0
# цикл расчёта
for i in range(n):
    x1 = p[i][0]
    y1 = p[i][1]
    x2 = p[(i+1)%n][0]
    y2 = p[(i+1)%n][1]
    S += x1 * y2 - x2 * y1
    M += nod(abs(p[i][0]-p[(i+1)%n][0]),
             abs(p[i][1]-p[(i+1)%n][1]))
S = abs(S)/2
K = int(S-M/2+1)
# Вывод ответа
print(K)
```