

Обработка массива целых чисел из файла

В лесополосе осуществляется посадка деревьев. Причем саженцы высаживают рядами на одинаковом расстоянии.

Через какое-то время осуществляется аэросъемка, в результате которой определяется, какие саженцы прижились. Необходимо определить ряд с максимальным номером, в котором есть подряд ровно 11 неприжившихся саженцев, при условии, что справа и слева от них саженц прижились.

В ответе запишите сначала наибольший номер ряда, затем наименьший номер из неприжившихся мест.

Входные данные:

В первой строке входного файла 26.txt находится число N - количество занятых мест (натуральное число, не превышающее 10 000). Каждая из следующих N строк содержит два натуральных числа, не превышающих 100 000: номер ряда и номер занятого места.

Выходные данные:

Два целых неотрицательных числа: максимальный номер ряда, где нашлись обозначенные в задаче места, и минимальный номер подходящего места.

Пример входного файла:

```
7
40 30
40 34
50 125
50 129
50 64
50 68
```

Ответ для примера (при поиске 3 подряд идущих неприжившихся саженцев):

```
50 65
```

ряд	40
	50

МЕСТА

30... 34

125... 129

Комм	надо	место	2	3	4	5	6	7	8	9	10	11	12	место
		1	}											13
		занято	}											занято
			11 место											

=> разница 12 т.к. 13-1

	A	B	C
1	100	2681	
2	100	7639	
3	102	5840	
4	103	5912	
5	103	967	
6	103	6977	
7	104	5465	
8	104	8978	
9	106	5791	
10	107	9417	
11	107	890	
12	110	8306	
13	111	2241	
14	112	7560	
15	112	579	
16	113	3299	
17	114	917	
18	114	4615	

1) Используя настраиваемую сортировку. Сортируем оба столбца по возрастанию

2) В ячейку C2 вводим формулу **=ЕСЛИ(И(A2=A1;B2-B1=12);1;0)**

2221	2260	4749	
2222	2261	9660	
2223	2261	634	
2224	2261	5592	
2225	2261	2666	
2226	2261	5086	
2227	2261	5098	1
2228	2266	6681	
2229	2269	4369	
2230	2269	3952	

Ответ 1) 2261 это ряд

2) Первое место которое будет не засажено

5087

Решение на ПИТОНЕ

- в языке Python для чтения данных удобно использовать менеджер контекста (`with ... as`), который открывает файл и закрывает его; например, код

```
with open("26.txt") as Fin: # программа и файл в одной папке
    ... # какие-то операции с файлом
    # при завершении работы менеджера контекста
    # файл автоматически закрывается
```

равносилен такому

```
Fin = open("26.txt") # открытие файла
... # какие-то операции с файлом
Fin.close() # закрытие файла
```

- если в текущей строке файла находится целое число, то прочитать его в переменную `x` можно так:

```
x = int( Fin.readline() )
```

- если в строке записаны два числа, после чтения (`Fin.readline()`) строку нужно разбить на отдельные части по пробелам между числами (каждая часть – символьная запись числа) и затем каждую часть преобразовать в целое число; например, чтение двух чисел:

```
s = Fin.readline()
symData = s.split()
x, y = map( int, symData )
```

или в компактной форме

```
x, y = map( int, Fin.readline().split() )
```

Пример

1

Автомат фиксирует пассажиров некоторого автобуса по ходу рейса. У каждого пассажира фиксируется время входа и выхода с момента начала рейса. Необходимо узнать максимальное количество пассажиров, одновременно находящихся в автобусе, и общее время, когда в автобусе был хотя бы один пассажир. Временем входа и выхода в автобус пренебречь.

Входные данные представлены в файле **26-75.txt** следующим образом. В первой строке входного файла находится число N – общее количество пассажиров (натуральное число, не превышающее 10 000). В следующих N строках находится по два числа. Первое число – время входа пассажира от начала рейса (натуральное число, не превышающее 1 000 000). Второе число – время выхода пассажира от начала рейса (натуральное число, не превышающее 1 000 000). Запишите в ответе два числа: количество пассажиров, одновременно находящихся в автобусе и общее время, когда в автобусе был хотя бы один пассажир.

Пример входного файла:

```
7
10 40
50 130
70 130
75 90
120 170
140 170
150 180
```

В приведённом примере пассажиры были в временных отрезках 10-40 и 50-180. Максимальное количество пассажиров одновременно 3. Ответ: 3 160.

```
f = open('26-75.txt')
n = int(f.readline())
a = [0]*1000001
for i in range(n):
    st, end = map(int, f.readline().split())
    a[st] += 1
    a[end] -= 1

k = 0
m, count = 0, 0
for i in range(1000001):
    if k > 0: count += 1
    k += a[i]
    m = max(m, k)
print(m, count)
```

Пример

2

(А. Кабанов) Маркетплейс с оптового склада каждый день отправляет заказанные товары в точки выдачи. Маркетплейс имеет множество видов различных товаров, каждый из которых имеет какой-то вес. Для отправки склад выделяет транспорт таким образом, чтобы отправить все возможные типы товаров и при этом отправить как можно больше каждого из них, но не более, чем определённый вес S . Нужно определить, сколько всего товаров останется на складе и тип товара с самым большим остатком. Если таких товаров несколько, вывести товар с наименьшим кодом.

Входные данные представлены в файле **26-71.txt** следующим образом. В первой строке входного файла записаны два числа, разделённые пробелом: число N – количество доступных товаров (натуральное число, не превышающее 10000) и число S – вес, не более которого можно отправить каждый тип товара (натуральное число, не превышающее 10^8). В каждой из следующих N строк записаны по два числа, разделённые пробелом: код товара (натуральное число, не превышающее 10^9) и его вес (натуральное число, не превышающее 10^5).

Известно, что количество различных кодов товаров в файле не превышает тысячи.

Запишите в ответе два числа: сначала количество товаров, оставшихся на складе, а затем код товара с самым большим остатком.

Пример входного файла:

```
8 13
150 8
237 3
237 6
150 4
237 5
237 6
150 3
150 3
```

При таких исходных данных имеется всего два вида товаров (с кодами 150 и 237). Товаров с кодом 150 можно погрузить три штуки (3, 3 и 4), останется 1 штука (8). Товаров с кодом 237 можно погрузить две штуки (за 3 и 5), останется 2 штуки (5 и 6). Ответ: 3 237.

1)

Сортировка

Добавить уровень | Удалить уровень | Копировать уровень | Параметры... | Мои данные содержат заголовки

Столбец: Сортировать по: Столбец А | Значения ячеек | По возрастанию | Порядок: По возрастанию

Затем по: Столбец В | Значения ячеек | По возрастанию

	А	В	С
1	10166618	34	
2	10166618	41	
3	10166618	48	
4	10166618	52	
5	10166618	54	
6	10166618	58	
7	10166618	63	
8	10166618	67	
9	10166618	75	
10	10166618	77	
11	10166618	78	
12	10166618	89	
13	10166618	99	
14	10166618	108	
15	11606873	34	
16	11606873	36	
17	11606873	36	

2) Считаем суммарный вес группы товаров **а)** В1 копируем в С1 вводим **б)** в ячейку С2 =ЕСЛИ(А2=А1;С2+В2;В2)

	А	В	С	Д	Е	Ф	Г
1	10166618	34	34				
2	10166618	41	75				
3	10166618	48	123				
4	10166618	52	175				
5	10166618	54	229				
6	10166618	58	287				
7	10166618	63	350				
8	10166618	67	417				
9	10166618	75	492				
10	10166618	77	569				
11	10166618	78	647				
12	10166618	89	736				
13	10166618	99	835				
14	10166618	108	943				
15	11606873	34	34				
16	11606873	36	70				
17	11606873	36	106				

3) Посмотрим какие товары мы не возьмем в ячейки D1 вводим ЕСЛИ (C1>500;1;0)

- 0 – берем
- 1 – не берем

Ответ: 1) Останется **391** товар (выдели столбец D и посмотри сумму)

2) Ищем максимальный

Formula bar: =ЕСЛИ(D2=1;E1+1;0)

	A	B	C	D	E	F
1	10166618	34	34	0	0	
2	10166618	41	75	0	0	
3	10166618	48	123	0	0	

Formula bar: =ЕСЛИ(D52=1;E51+1;0)

	A	B	C	D	E	F
55	14981976	26	26	0	0	
56	14981976	33	59	0	0	
57	14981976	40	99	0	0	
58	14981976	45	144	0	0	
59	14981976	51	195	0	0	
60	14981976	56	251	0	0	
61	14981976	61	312	0	0	
62	14981976	71	383	0	0	
63	14981976	75	458	0	0	
64	14981976	80	538	1	1	
65	14981976	81	619	1	2	
66	14981976	85	704	1	3	
67	14981976	88	792	1	4	
68	14981976	99	891	1	5	
69	14981976	104	995	1	6	
70	14981976	112	1107	1	7	
71	14981976	122	1229	1	8	
72	15230058	36	36	0	0	
73	15230058	46	82	0	0	
74	15230058	49	131	0	0	
75	15230058	52	183	0	0	
76	15230058	61	244	0	0	
77	15230058	68	312	0	0	
78	15230058	79	391	0	0	
79	15230058	80	471	0	0	
80	15230058	92	563	1	1	
81	15230058	97	660	1	2	

Вставлю строку для фильтрации

Formula bar: 15230058

	A	B	C	D	E	F	G
1							
103	15230058	243	4247	1	23		
1004							
1005							

15230058

Решение на
питоне

```
f = open('26-71.txt')
N, S = map(int, f.readline().split())
a = [list(map(int, f.readline().split())) for i in range(N)]
a.sort()

s = a[0][1]
k, count = 0, 0
mk, mt = 0, 0

for i in range(1, N):
    if a[i][0] == a[i-1][0]:
        s += a[i][1]
        if s > S:
            k += 1
    else:
        s = a[i][1]
        count += k
        if k > mk:
            mk, mt = k, a[i-1][0]
        k = 0
print(count, mt)
```

Пример 3 Дикая задача

На закупку товаров типов А, В, С, D и E выделена определённая сумма денег. Эти товары есть в продаже по различной цене. Необходимо на выделенную сумму закупить как можно больше товаров пяти типов (по общему количеству). Если можно разными способами купить максимальное количество пяти типов товаров, то нужно выбрать способ, при котором будет закуплено как можно больше товаров типа А. Если при этих условиях есть несколько способов закупки, нужно потратить как можно меньше денег.

Определите, сколько будет закуплено товаров типа А и сколько денег останется.

Входные данные представлены в файле **26-64.txt** следующим образом. Первая строка входного файла содержит два целых числа: N – общее количество товаров и M – сумма выделенных на закупку денег (в рублях). Каждая из следующих N строк содержит целое число (цена товара в рублях) и символ (латинская буква), определяющий тип товара. Все данные в строках входного файла отделены одним пробелом.

Запишите в ответе два числа: сначала количество закупленных товаров типа А, затем оставшуюся неиспользованной сумму денег.

Пример входного файла:

```
6 110
40 E
50 A
50 D
30 C
20 B
10 A
```

В данном случае можно купить не более четырёх товаров, из них не более двух товаров типа А. Минимальная цена такой покупки 110 рублей (покупаем товары 10 А, 20 В, 30 С, 50 А). Останется 0 рублей. Ответ: 2 0.

```

target = 'A'

with open("26-64.txt") as F:
    N, M = map( int, F.readline().split() )
    data = []
    for i in range(N):
        d = F.readline().split()
        data.append( (int(d[0]), d[1]) )

data.sort()
print( data )

# определяем максимальное общее число деталей
count, S = {}, 0
for i in range(N):
    v, m = data[i]
    if S+v <= M:
        iLast = i
        S += v
        print(v, m)
        count[m] = count.get(m, 0) + 1

# пытаемся увеличить количество target
remainingA = [ (v, m) for v, m in data[iLast+1:]
               if m == target ]

i = iLast
while remainingA and i >= 0:
    v, m = data[i]
    if m != target:
        vA, mA = remainingA.pop(0)
        if S - v + vA <= M:
            count[target] = count.get(target, 0) + 1
            S = S - v + vA
            print( i, '*', -v, m, '<-', vA, mA, ':', M - S )
        else:
            break
    i -= 1

print( count[target], M - S )

```

```

1087 E
1088 A
1088 A
1088 B
94 * -1088 B <- 1089 A : 843
91 * -1087 E <- 1094 A : 836
90 * -1087 B <- 1098 A : 825
89 * -1086 C <- 1107 A : 804
88 * -1084 B <- 1109 A : 779
87 * -1082 B <- 1113 A : 748
86 * -1081 E <- 1114 A : 715
85 * -1080 E <- 1119 A : 676
84 * -1079 E <- 1125 A : 630
83 * -1079 B <- 1129 A : 580
82 * -1079 B <- 1132 A : 527
81 * -1078 D <- 1135 A : 470
80 * -1077 D <- 1137 A : 410
79 * -1076 B <- 1138 A : 348
77 * -1073 E <- 1143 A : 278
76 * -1073 C <- 1145 A : 206
75 * -1072 C <- 1150 A : 128
74 * -1071 E <- 1155 A : 44
35 44

```

Пример

4

Для анализа нагрузки сервера для каждого запроса в журнал записываются время начала и время завершения его обработки (в миллисекундах от момента начала исследований). Если начальное время равно 0, запрос начал обрабатываться до начала исследований, если конечное время равно 0, то обработка запроса закончилась после окончания исследований. Необходимо определить наибольшее количество запросов, которые сервер обрабатывал одновременно в течение суток, начиная с момента K , и суммарное время, в течение которого обрабатывалось это максимальное количество запросов.

Входные данные представлены в файле **26-66.txt** следующим образом. Первая строка входного файла содержит количество записей N и время K . Каждая из следующих N строк содержит два целых числа: время начала и время завершения обработки одного запроса (в миллисекундах).

Запишите в ответе два числа: наибольшее количество запросов, которые сервер обрабатывал одновременно в течение указанных суток, и суммарное время, в течение которого обрабатывалось это максимальное количество запросов.

Пример входного файла (для заданного диапазона от 1000 до 6000):

```
6 1000
1300 2200
0 3700
1300 5700
0 0
5000 0
1800 3400
```

В данном случае наибольшее число запросов (5) выполнялось в интервале времени между 1800 и 2200. Ответ: 5 400.

```

with open('26-66.txt') as f:
    N, START = map(int, f.readline().split())
    END = START + 24*3600*1000
    data = []
    for i in range(N):
        t0, t1 = map(int, f.readline().split())
        if t1 == 0: t1 = 10**10
        data.extend([t0, -t1])

data.extend([-START, START]) # учесть первый интервал
data.append(-END) # учесть последний интервал
data.sort(key=abs)

nProc = maxProc = 0
maxTime = 0
prev = 0
for i, t in enumerate(data):
    nProc += 1 if t >= 0 else -1
    if START <= abs(t) <= END:
        if nProc > maxProc:
            maxProc, maxTime = nProc, 0
        else:
            if prev == maxProc:
                maxTime += abs(t) - abs(data[i-1])
                # print( t, maxProc, maxTime, abs(t) - abs(data[i-1]) )
            prev = nProc

print(maxProc, maxTime)

```

Пример

5

Предприятие производит оптовую закупку изделий A и Z, на которую выделена определённая сумма денег. У поставщика есть в наличии партии этих изделий различных модификаций по различной цене. На выделенные деньги необходимо приобрести как можно больше изделий Z (независимо от модификации). Закупать можно любую часть каждой партии. Если у поставщика закончатся изделия Z, то на оставшиеся деньги необходимо приобрести как можно больше изделий A. Известна выделенная для закупки сумма, а также количество и цена различных модификаций данных изделий у поставщика. Необходимо определить, сколько будет закуплено изделий A и какая сумма останется неиспользованной. Если возможно несколько вариантов решения (с одинаковым количеством закупленных изделий A), нужно выбрать вариант, при котором оставшаяся сумма максимальна.

Входные данные представлены в файле **26-42.txt** следующим образом. Первая строка входного файла содержит два целых числа: N – общее количество партий изделий у поставщика и S – сумма выделенных на закупку денег (в рублях). Каждая из следующих N строк описывает одну партию изделия: сначала записана буква A или Z (тип изделия), а затем – два целых числа: цена одного изделия в рублях и количество изделий в партии. Все данные в строках входного файла разделены одним пробелом.

В ответе запишите два целых числа: сначала количество закупленных изделий типа A, затем оставшуюся неиспользованной сумму денег.

Пример входного файла

```
4 1000
A 14 12
Z 30 7
A 40 24
Z 50 15
```

В данном случае сначала нужно купить изделия Z: 7 изделий по 30 рублей и 15 изделий по 50 рублей. На это будет потрачено 960 рублей. На оставшиеся 40 рублей можно купить 2 изделия A по 14 рублей. Таким образом, всего будет куплено 2 изделия A и останется 12 рублей. В ответе надо записать числа 2 и 12.


```
with open("26-42.txt") as F:
    N, S = map(int, F.readline().split())

    data = []
    for i in range(N):
        model, price, size = F.readline().split()
        data.append((model, int(price), int(size)))

data.sort(key = lambda d: (d[0], -d[1]), reverse = True)

xSum = 0
countA = 0
for d in data:
    if xSum+d[1] > S: break
    thisCount = min(d[2], int((S-xSum)/d[1]))
    xSum += d[1]*thisCount
    print(d, '+' + str(thisCount))
    if d[0] == 'A':
        countA += thisCount

print(countA, S-xSum)
```

```
('A', 338, 68) +68
('A', 338, 30) +30
('A', 339, 53) +53
('A', 341, 69) +69
('A', 345, 63) +63
('A', 345, 26) +26
('A', 346, 50) +50
('A', 351, 33) +33
('A', 352, 70) +70
('A', 356, 14) +14
('A', 359, 39) +39
('A', 361, 34) +34
('A', 362, 21) +21
('A', 363, 23) +23
('A', 365, 51) +51
('A', 369, 59) +59
('A', 369, 18) +14
7354 111
```

Пример

6

(Е. Джобс) Для перевозки партии грузов различной массы выкупают место у компании, которая организует перевозку на грузовых судах. Судно не может принять на борт больше S тонн груза. Известно, что отдельный груз нельзя разделить для перевозки, то есть один груз должен доставляться одним рейсом на одном грузовом судне. В первую очередь перевозятся грузы как можно большей массы. За какое минимальное количество рейсов можно перевезти все грузы? В ответе запишите два числа – минимальное количество рейсов и суммарную массу грузов, которые будут перевезены последним рейсом.

Входные данные представлены в файле **26-55.txt** следующим образом. В первой строке входного файла записаны два целых числа: N – общее количество грузов и S – грузоподъемность судна в тоннах. Каждая из следующих N строк содержит одно целое число $\leq S$ – массу груза в тоннах.

Пример входного файла:

6 500

140

150

160

200

220

240

Первым рейсом будет отправлено 2 груза – 240 и 220, вторым – 200, 160 и 140, третьим – 150. При таких входных данных ответ будет 3 и 150.

```
with open('26-55.txt') as F:
    N, S = map(int, F.readline().split())
    data = []
    for i in range(N):
        data.append(int(F.readline()))

data.sort(reverse=True)

count = 1
while True:
    total = 0
    i = 0
    while i < len(data):
        if total + data[i] <= S:
            total += data[i]
            #print('+', data[i])
            del data[i]
        else:
            i += 1
    #print(total, i)
    if not data: break
    count += 1
    total = 0

print(count, total)
```

Пример

7

На закупку товаров типов Q и Z выделена определённая сумма денег. Эти товары есть в продаже по различной цене. Необходимо на выделенную сумму закупить как можно больше товаров двух типов (по общему количеству). Если можно разными способами купить максимальное количество двух товаров, то нужно выбрать способ, при котором будет закуплено как можно больше товаров типа Q. Если при этих условиях есть несколько способов закупки, нужно потратить как можно меньше денег.

Определите, сколько будет закуплено товаров типа Q и сколько денег останется.

Входные данные представлены в файле **26-62.txt** следующим образом. Первая строка входного файла содержит два целых числа: N – общее количество товаров и M – сумма выделенных на закупку денег (в рублях). Каждая из следующих N строк содержит целое число (цена товара в рублях) и символ (латинская буква Q или Z), определяющий тип товара. Все данные в строках входного файла отделены одним пробелом.

Запишите в ответе два числа: сначала количество закупленных товаров типа Q, затем оставшуюся неиспользованной сумму денег.

Пример входного файла:

```
6 110
40 Z
50 Q
50 Z
30 Z
20 Q
10 Z
```

В данном случае можно купить не более четырёх товаров, из них не более двух товаров типа Q.

Минимальная цена такой покупки 110 рублей (покупаем товары 10 Z, 20 Q, 30 Z, 50 Q). Останется 0 рублей. Ответ: 2 0.

```
target = 'Q'
```

```
with open("26-62.txt") as F:
```

```
    N, M = map( int, F.readline().split() )
```

```
    data = []
```

```
    for i in range(N):
```

```
        d = F.readline().split()
```

```
        data.append( (int(d[0]), d[1]) )
```

```
data.sort()
```

```
print( data )
```

```
# определяем максимальное общее число
```

```
деталей
```

```
count, S = {}, 0
```

```
for i in range(N):
```

```
    v, m = data[i]
```

```
    if S+v <= M:
```

```
        iLast = i
```

```
        S += v
```

```
        print(v, m)
```

```
        count[m] = count.get(m, 0) + 1
```

```
# пытаемся увеличить количество target
```

```
remainingA = [ (v, m) for v, m in data[iLast+1:]
```

```
                if m == target ]
```

```
i = iLast
```

```
while remainingA and i >= 0:
```

```
    v, m = data[i]
```

```
    if m != target:
```

```
        vA, mA = remainingA.pop(0)
```

```
        if S - v + vA <= M:
```

```
            count[target] = count.get(target, 0) + 1
```

```
            S = S - v + vA
```

```
            print( i, '*', -v, m, '<-', vA, mA, ':', M - S )
```

```
        else:
```

```
            break
```

```
    i -= 1
```

```
print( count[target], M - S )
```

```
60 * -1034 Z <- 1108 Q : 330
57 * -1051 Z <- 1107 Q : 474
53 * -1048 Z <- 1110 Q : 412
49 * -1046 Z <- 1112 Q : 346
47 * -1044 Z <- 1114 Q : 276
41 * -1039 Z <- 1116 Q : 199
40 * -1038 Z <- 1119 Q : 118
39 * -1036 Z <- 1121 Q : 33
68 33
```

```
Process finished with exit code 0
```