Контест 1

Конспект:
https://colab.research.google.com/drive/14CH88ceOlchi59c1AVuzQp55BlXKZoKq?usp=sharing

1. Быстрая черепашка

Какая из ниже приведенных команд отвечает за самую быструю прорисовку следа от черепашки:

- [ ] `pendown()`
- [ ] `speed(10)`
- [x] `speed(13)`
- [ ] `forward(100)`

2. Вопрос о фигуре

Какая фигура будет нарисована после выполнения следующего кода?

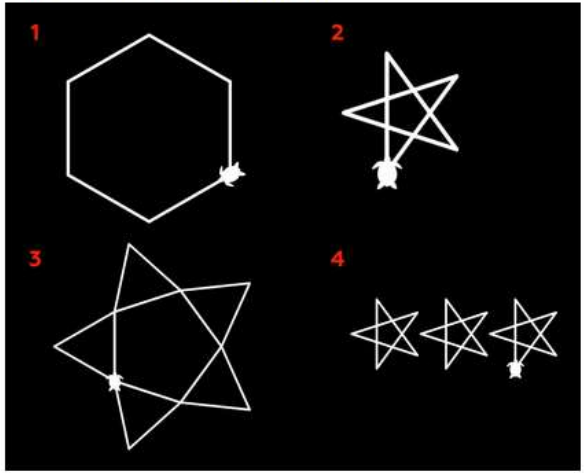
```
from ColabTurtle.Turtle import *
initializeTurtle()

speed(10)

for step in 1, 2, 3, 4, 5, ..., 15:
    forward(120)
    right(720 / 5)
```

Формат вывода

В ответе укажите цифру с правильным ответом:



Примечания

Вы можете проверять свои решения в Google Collab. Для запуска Черепашки нужно подключать библиотеку Collab Turtle. Как это сделать корректно - см. лекцию (раздел: "Особенности подключения модуля").

- [ ] 1
- [x] 2
- [ ] 3
- [ ] 4

3. Послание

Запустите нижеприведенную программу. Какое сообщение нарисует черепашка?

```
from ColabTurtle.Turtle import *
initializeTurtle()

speed(10)

penup()
backward(100)
pendown()
forward(100)
right(150)
forward(60)
left(120)
forward(60)
left(30)
backward(100)
right(90)
penup()
forward(60)
pendown()
left(360/4)
forward(100)
right(360/4)
for i in 1, 2, 3, 4:
    forward(25)
    right(360/8)
forward(25)
for i in 1, 2, 3, 4, 5:
    forward(25)
    right(360/8)
right(360/8)
penup()
forward(100)
left(360/4)
forward(60)
pendown()
left(360/4)
forward(40)
right(30)
forward(69)
right(180)
forward(59)
left(60)
forward(56)
right(180)
forward(56)
left(120)
forward(10)
right(150)
forward(60)
```

Формат вывода

В ответе напишите сообщение в **нижнем** регистре.

мфк

4. Построй дом!

Начинающий программист решил нарисовать с помощью библиотеки turtle изображение домика. У него получился такой код:

```
from ColabTurtle.Turtle import *
initializeTurtle() #1
speed(10) #2
pensize(2) #3

right(90) #4
forward(100) #5
right(90) #6
forward(100) #7
right(90) #8
forward(100) #9
right(90) #10
forward(100) #11

right(60) #12
forward(57) #13
right(30) #14
forward(20) #15

left(120) #16
forward(20) #17
right(90) #18
forward(20) #19
right(90) #20
forward(20) #21
forward(11) #22
left(60) #23
forward(14) #24

penup() #25
forward(50) #26
```

И не совсем желаемый результат. Исправьте код таким образом, чтобы после запуска программы черепашка рисовала домик.

Формат вывода

В ответ через запятую без пробела запишите:

- номер строки первоначальной программы, в которой была ошибка,
- исправленную строку.

Пример: 1,penup()

Формат вывода

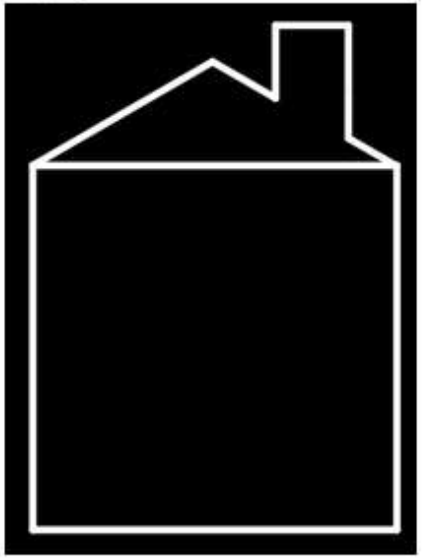
В ответ через запятую без пробела запишите:

- номер строки первоначальной программы, в которой была ошибка,
- исправленную строку.

Пример: 1,penup()

Примечания

1. Нужно исправить **только одну** строчку.
2. Если, по вашему мнению, можно изменить несколько строк, в ответе следует указать исправление, которое нужно сделать **в строке с наименьшим номером**.
3. В результате исправления кода должен получиться дом, изображённый на рисунке:



14,right(60)

Задачи Посылки

5. Звездочный квадрат

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	стандартный ввод
Вывод	стандартный вывод

Напишите программу, которая считывает число n с клавиатуры и распечатывает квадрат размера $n \times n$ из звездочек.

Пример 1

Ввод	Вывод
1	*

Пример 2

Ввод	Вывод
2	** **

Примечания

1. n вводится **с клавиатуры** с помощью input() *(см. рекомендации к вводу-выводу)*.
2. n - это **целое число**, т.е. после ввода его нужно преобразовать с помощью функции int() *(см. рекомендации к вводу-выводу)*.
3. Не забывайте, что n не обязательно положительное число. В случае нулевого либо отрицательного n ничего на экран выводить **не нужно**!
4. В выводе **не нужно** разделять звёздочки пробелами.
5. В ходе работы программы нужно вывести **ровно N строк**, в каждой из которых будет **ровно N звёздочек**.
6. Обратите внимание, что в этой задаче **не нужно** использовать черепашку, о которой рассказывалось на лекции (библиотеки ColabTurtle и turtle **не требуются**!)

Дополнительные условия:

7. Для решения задачи используйте циклы for и функцию range().

Набрать здесь | Отправить файл

```
n = int(input())
for e in range(n):
    print('*'*n)
```

Контест 2

Конспект:
https://colab.research.google.com/drive/1bWvnRzsjiFneg5KU5DfA6ts_F9l37ZEj?usp=sharing

1. Количество символов в строке

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	стандартный ввод или input.txt
Вывод	стандартный вывод или output.txt

Задана строка s, состоящая из символов. Необходимо посчитать их количество $|s|$.

Формат ввода

Строка s.

Формат вывода

Для каждой строки выведите одно число $|s|$, равное длине s.

Пример

Ввод	Вывод
Hello World!	12

Hello World!	12

Примечания

1. Обратите внимание, что входные данные считываются или из стандартного ввода (с клавиатуры), или из файла input.txt *(см. рекомендации к вводу-выводу)*.
2. Обратите внимание на формат вывода. Результат работы программы нужно выводить с помощью **print()** *(см. рекомендации к вводу-выводу)*.
3. Не забывайте, что пробел и \n - тоже символы. Их тоже нужно учитывать при подсчёте количества символов.

Набрать здесь | Отправить файл

```
1 s=str(input())
2 o=len(s)
3 print(o)
```

```
s=str(input())
o=len(s)
print(o)
```

2. Строка в обратном порядке

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	стандартный ввод или input.txt
Вывод	стандартный вывод или output.txt

Измените порядок символов в строке *s* на обратный.

Формат ввода

Строка *s*.

Формат вывода

Строка *s* в обратном порядке.

Пример 1

Ввод	Вывод
Hello World!	!dlroW olleH

Пример 2

Ввод	Вывод
Довод	довоД

Пример 1

Ввод	Вывод
Hello World!	!dlroW olleH

Пример 2

Ввод	Вывод
Довод	довоД

Примечания

Для решения задачи предлагается использовать срезы.

Набрать здесь | Отправить файл

```
s=str(input())
print(s[::-1])
```

Отправить осталось 100 попыток

Предыдущая Следующая

Время посылки	ID	Задача	Компилятор	Вердикт	Тип посылки	Время

```
s=str(input())
print(s[::-1])
```

3. Какой логин?

Ограничение времени	2 секунды
Ограничение памяти	64Mb
Ввод	стандартный ввод или input.txt
Вывод	стандартный вывод или output.txt

Из почтового адреса нужно достать логин.

Пример

Ввод	Вывод
ivan@lenta.ru	ivan

Примечания

1. Длина логина **не обязательно** равняется 4 символам, как это указано в примере (на вход вашей программе подается неизвестный почтовый адрес).
2. Для поиска подстроки можно использовать *find()* и срезы строк.
3. Описание метода *find()* можно найти с помощью *help()*.

Набрать здесь | Отправить файл

```
s=str(input())
a=s.find("@")
print(s[:a])
```

```
s=str(input())
a=s.find("@")
print(s[:a])
```

4. Подстрока в двойных кавычках

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	стандартный ввод или input.txt
Вывод	стандартный вывод или output.txt

Найдите подстроку, заключенную в двойных кавычках в строке *s*.

Пример 1

Ввод	Вывод
Hel"lo Wor"ld!	lo Wor

Пример 2

Ввод	Вывод
"Kto tam?" – sprosil dyadya Fyodor	Kto tam?

Примечания

1. Гарантируется, что такая подстрока **существует** и притом **единственна**.
2. Двойные кавычки в результат включать **не нужно**.
3. На вход вашей программе могут подавать строчки, **отличные** от указанных в примерах, поэтому делать решение, которое работает только на примере – это неправильный путь. Вам нужно **найти кавычки в любой строке**.
4. Для поиска подстроки можно использовать find() и срезы строк.
5. Описание метода find() можно найти с помощью help(), как это было показано в лекции.

```
s=str(input())
a=s.find('"')
b=s.rfind('"')
print(s[a+1:b])
```


```
s=str(input())
a=s.find("")
b=s.rfind("")
print(s[a+1:b])
```

5. Поменялись!

Ограничение времени	2 секунды
Ограничение памяти	64Mb
Ввод	стандартный ввод или input.txt
Вывод	стандартный вывод или output.txt

В строке записаны два числа, разделенные пробелом. Поменяйте их местами и напечатайте результат.

Пример

Ввод	Вывод
3254543 793423410	793423410 3254543

s=str(input())
a=s.find(' ')
print(s[a+1:],s[:a+1],sep=" ")

Набрать здесь | Отправить файл

```
1 s=str(input())
2 a=s.find(' ')
3 print(s[a+1:],s[:a+1],sep=" ")
```

6. Как? Как угодно!

Ограничение времени	2 секунды
Ограничение памяти	64Mb
Ввод	стандартный ввод или input.txt
Вывод	стандартный вывод или output.txt

Человек вводит на сайте номер телефона, ему позволено для удобства использовать кроме плюса и цифр знаки '-', ')', '(' и пробелы. Уберите их из ввода.

Формат ввода

На вход подаётся строка s.

Формат вывода

Строка s без лишних символов, указанных в задании.

Пример

Ввод	Вывод
+7 (812) 134-12-324	+781213412324

Примечания

1. Условие **не ограничивает** количество и порядок использования указанных символов. Их может быть сколько угодно. Ваша цель - просто удалить из входной строки **все** такие символы.
2. Для склеивания двух строк можно использовать операцию конкатенации (с помощью +), рассмотренную на лекции.

```
s=str(input())
a=s.replace("-","")
b=a.replace("(", "")
c=b.replace(")","")
d=c.replace(" ","")
print(d)
```


```
s=str(input())
a=s.replace("-","")
b=a.replace("(",
"")
c=b.replace(")","")
d=c.replace(" ","")
print(d)
```

7. Щйхс Чёибса

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	стандартный ввод или input.txt
Вывод	стандартный вывод или output.txt

В книге "Жизнь двенадцати цезарей" историк Светоний писал про хитрый метод, который использовал Гай Юлий Цезарь для передачи секретных сообщений.

Позже этот метод, делающий текст понятным только отправителю и получателю, был назван ***Шифр Цезаря***.

Кодирование таким шифром происходит следующим образом:

1. Каждая буква входного сообщения заменяется на букву, следующую за ней в алфавите: "А"->"Б", "е"->"ё" и т.д.
2. Любые другие символы остаются без изменений.

Вам предлагается написать программу, которая помогла бы Цезарю зашифровать его сообщение по указанному алгоритму. Сообщение записано на **английском языке**.

Формат ввода

Строка *s* на **английском языке**.

Формат вывода

Строка *s*, преобразованная шифром Цезаря.

Пример

Ввод	Вывод
Veni, vidi, vici	Wfoj, wjej, wjdj

Примечания

1. Строка вводится на **английском языке**.
2. **Размер букв сохраняется**: строчная буква превращается в строчную, прописная – в прописную.
3. Для простоты преобразования считаем, что во входной строке нет букв *z* или *Z*.
4. Для решения можно использовать:
   - итерацию по строке;
   - для проверки, является ли символ буквой – операцию str.isalpha() (подробнее – см. help());
   - операцию ord() для получения кода символа,
   - операцию chr() для превращения кода символа в символ (операция обратна к ord(): chr(ord(c))==c, где c – некоторый символ).
5. Считаем, что и строчные, и прописные буквы связаны следующим **соотношением**: *выражение ord(c)+1* ***всегда*** *даёт код буквы, следующей за данной буквой c в алфавите*. Например: ord('a')+1 – код символа 'b', ord('C')+1 – код символа 'D'.

Набрать здесь | Отправить файл

```
1  letter = str(input())
2  shift = 1
3  s = ""
4  for l in letter:
5      if l.isalpha():
6          a = ord(l) + shift
7          s += chr(a)
8      else:
9          s += l
10
11 print(s)
```

```
letter = str(input())
shift = 1
s = ""
for l in letter:
    if l.isalpha():
        a = ord(l) + shift
        s += chr(a)
    else:
        s += l
print(s)
```

Контест 3

Конспект:
https://colab.research.google.com/drive/1nDTBYMi4rHCa0R0kFax7SAOwyn1i6x-O?usp=sharing

1. Задом наперед

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	input.txt
Вывод	стандартный вывод или output.txt

Писатель Яша купил себе новую пишущую машинку и сразу бросился писать свой новый роман. Закончив пятнадцатую главу и решив оценить свое творение, Яша заметил, что в пишущей машинке был очень необычный заводской брак: все слова она писала задом-наперёд, хотя и в нужном порядке. У Яши больше нет сил, чтобы переписывать роман с самого начала.

Помогите опечаленному Яше справиться с его бедой: напишите программу, которая *развернет все слова в строке*, не меняя их порядка.

Формат ввода

Дана строка *s*.

Формат вывода

Выведите строку, в которой каждое слово строки *s* будет написано задом-наперёд, но порядок слов в строке не поменяется. Обратите внимание на примеры.

Пример 1

Ввод	Вывод
Что такое осень - это небо	отЧ еокат ьнесо - отэ обен

Пример 2

Ввод	Вывод
А роза упала на лапу Азора	А азор алапу ан упал арозА

Пример 3

Ввод	Вывод
Корабли лавировали-лавировали, да н	илбароК ,илаворивал-илаворивал ад е

Примечания

1. Разрешается ввод **только** через input.txt *(см. рекомендации к вводу-выводу)*.
2. Вывод осуществляется **на экран** *(см. рекомендации к вводу-выводу)*.
3. В файле input.txt содержится **ровно одна строка**, которую и должна считать Ваша программа. Её можно считать с помощью readline(), например *(см. рекомендации к вводу-выводу)*.
4. Обратите внимание, что в исходной строке нужно переворачивать **каждое слово**, а **не строку целиком**.
5. Не забывайте, что метод split() применяется **к отдельной строке**, а не к списку строк.

Набрать здесь | Отправить файл

```
s = open('input.txt')
c = s.readlines()
b = ''.join(c)
a = b.split( )
nw = [i [:: - 1] for i in a]
d = ' '.join(nw)
print(d)
s.close()
```

```
s = open('input.txt')
c = s.readlines()
b = ''.join(c)
a = b.split( )
nw = [i [:: - 1] for i in a]
d = ' '.join(nw)
print(d)
s.close()
```

2. Буква "ё"

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	input.txt
Вывод	стандартный вывод или output.txt

Буква **ё** в рукописных и печатных текстах по различным причинам часто заменяется на **е**. Эту особенность традиции письменности в русском языке можно отслеживать и использовать для установления оригинальности текстов, для поиска грамматических ошибок, для других целей, связанных с анализом текстов. Напишите программу, которая будет искать строки в текстовом файле, в котором есть буква **ё** и выводить их на экран.

Формат ввода

Набор строк, часть из которых содержит букву **ё**.

Формат вывода

Вывести строки, содержащие букву **ё**.

Гарантируется, что **существует хотя бы одна** такая строка.

Пример

Ввод	Вывод
В лесу родилась ёлочка, В лесу она росла, Зимой и летом стройная, зеленая бы	В лесу родилась ёлочка,

Примечания

1. Разрешается ввод **только** через input.txt *(см. рекомендации к вводу-выводу)*.
2. Подсказка: используйте .filter() и *lambda-функции*.
3. Не забывайте, что метод readlines() получает строки **вместе с символом перехода на новую строку**. Поэтому при выводе строк проверяйте, чтобы в результате не появлялось лишних пустых строк.
4. Если буквы "ё" нет в строке - на экран ничего выводить **не нужно**.
5. Строку необходимо выводить **ровно в том виде, в котором она находится во входном файле** (заглавная буква остаётся заглавной, строчная - строчной).

Набрать здесь | Отправить файл

```
1 s = open('input.txt')
2 c = s.readlines()
3 b = ''.join(c)
4 a = b.split("\n")
5 stripped_lines= [line.strip() for line in a]
6 filtered_lines = [line for line in stripped_lines if 'ё' in line]
7 d = '\n'.join(filtered_lines)
8 print(d)
9 s.close()
```

```
s = open('input.txt')
c = s.readlines()
b = ''.join(c)
a = b.split("\n")
stripped_lines= [line.strip() for line in a]
filtered_lines = [line for line in stripped_lines if 'ё' in line]
d = '\n'.join(filtered_lines)
print(d)
```

3. Убрать половину

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	input.txt
Вывод	стандартный вывод или output.txt

Нотариус Зоя помогает юридически неграмотным гражданам оформлять их жалобы и запросы в государственные службы и коммерческие компании. У Зои трудная работа: многие граждане совсем не умеют не только составлять юридически верные тексты, но ещё и оформлять свои мысли в сколько-нибудь связной форме.

Однажды клиентом Зои стал подросток Вадим. Вадим очень любил копаться в компьютерах и прочей технике и очень не любил уроки литературы, которые активно прогуливал. Вадим принес Зое папку жалоб на разные компании, в которых он заказывал компоненты компьютеров — они часто доставлялись в неудовлетворительном состоянии.

Прочитав первую жалобу, Зоя обомлела: слова-паразиты встречались в тексте Вадима буквально через слово.

Зоя поняла, что исправить все записки Вадима вручную — задача непосильная. Помогите Зое, *исключив каждое второе слово* из текстов Вадима.

Формат ввода

Дан набор строк (записки Вадима).

Формат вывода

Вывести строки, *исключив каждое второе слово* из каждой строчки текста Вадима.

Знак препинания, следующий за словом, которое будет удалено, удаляется вместе со словом.

Пример

Ввод	Вывод
Мне типа привезли этот товар ну в Плата там разбита прямо. Я вот хочу типа вернуть это, деньг	Мне привезли товар в плохом состоя Плата разбита Я хочу вернуть деньги.

Примечания

1. Разрешается ввод **только** через input.txt *(см. рекомендации к вводу-выводу)*.
2. Обратите внимание, что во входном файле может быть несколько строк.
3. Перед отправкой решения не забывайте проверять ситуации, когда в строке может быть как чётное число слов, так и нечётное.

Набрать здесь | Отправить файл

```
1 with open('input.txt') as file:
2     for s in file:
3         g = s.split()
4         print(" ".join(g[::2]))
```

with open('input.txt') as file:
for s in file:
g = s.split()
print("

4. Дни рождения

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	input.txt
Вывод	output.txt

Дан текстовый файл input.txt, содержащий следующую информацию:

в первой строке записано число N,

в следующих N строках содержатся имена студентов,

а в следующих N строках (см. примеры) — их даты рождения.

Выведите построчно информацию из этого файла и запишите в файл output.txt, при этом разделяйте имена и даты рождения при помощи специального символа табуляции '\t'.

Формат ввода

Данные вида:

$$N$$

$$Name_1$$

$$\vdots$$

$$Name_N$$

$$Birthday_1$$

$$\vdots$$

$$Birthday_N$$

Формат вывода

Данные в следующем виде:

$Name_1$ \t $Birthday_1$

$Name_2$ \t $Birthday_2$

$\vdots$

$Name_N$ \t $Birthday_N$

Пример

Ввод	Вывод
3 Dania Petya Egor 13 oct 28 mar 26 feb	Dania 13 oct Petya 28 mar Egor 26 feb

Примечания

1. Обратите внимание, что разрешается ввод и вывод только через input.txt и output.txt соответственно! *(см. рекомендации к вводу-выводу)*.
2. Подсказка: используя функцию zip(), соедините построчно информацию из файла input.txt (имена и даты рождения).
3. Обратите внимание, что что имена во входном файле печатаются на отдельной строке, а в результирующем файле нужно разделять имя и дату рождения символом **табуляции** \t.
4. Чтобы в выводе использовать символ табуляции в качестве разделителя, можно использовать параметр sep функции print(): print('string1', 'string2', sep='\t').
5. Не забывайте, что если в print() **не задавать** параметр sep, то параметры в выводе будут разделяться **пробелами**. Т.е. в результате выполнения кода: print(x, '\t', y)
   - после x будет выведен пробел, потом \t, потом снова пробел, и потом y. Это **неверно**!

6. При работе с датами не требуется никакая дополнительная обработка.

7. Тестовая система сравнивает результат работы программы с эталонным **посимвольно**. Поэтому, при отправке решения проверяйте, что ваш код **не печатает никаких лишних символов** (в частности, пробелов).

8. Не забывайте, что функция write(), в отличие от print(), принимает **только один аргумент - строку**, которая должна быть выведена! Если хотите выводить параметры в файл с указанием их через запятую - используйте print(): print(x, y, sep='\t', file=f) (f - это открытый файл).

Набрать здесь | Отправить файл

```
s = open('input.txt')
c = s.readlines()
d="/".join(c)
d=d.replace('\n',"")
d=d.split("/")
n=int(c[0])
a=d[1:n+1]
b=d[n+1:]
g=""
q=open("output.txt", "a")
for x, y in zip(a, b):
    q.write(str(x))
    q.write("\t")
    q.write(str(y))
    q.write("\n")
s.close()
q.close()
```


```
s = open('input.txt')
c = s.readlines()
d="/".join(c)
d=d.replace('\n',"")
d=d.split("/")
n=int(c[0])
a=d[1:n+1]
b=d[n+1:]
g=""
q=open("output.txt",
"a")
for x, y in zip(a, b):
    q.write(str(x))
    q.write("\t")
    q.write(str(y))
    q.write("\n")
s.close()
q.close()
```

5. Комбинации дежурных

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	studygroup.txt
Вывод	стандартный вывод

Классу 8 "Б" было поручено организовать школьные дежурства в новом учебном году. Ученики должны дежурить по трое, и перед классным руководителем была поставлена задача организовать тройки дежурных, составив план дежурств.

8 "Б" — очень дружный класс. Когда пришло время назначать план школьных дежурств, каждый из учеников 8 "Б" напрочь отказался принимать в этом участие, пока ему не пообещают, что в течение года он подежурит с каждым из своих друзей.

Классный руководитель 8 "Б" и учитель истории Мария Ивановна — замечательный человек и прекрасный педагог. Она решила пойти навстречу своим ученикам. Мария Ивановна сразу поняла, что ей предстоит решить сложную комбинаторную задачу и, не мешкая, приступила к опросу своих учеников.

К окончанию опроса выяснилось, что все ребята дружат со всеми. Мария Ивановна сразу сообразила, что ей предстоит организовать C^3_N троек, где N — количество ребят в классе. Помогите Марии Ивановне написать программу, составляющую *всевозможные тройки дежурных*.

Дан текстовый файл studygroup.txt, содержащий имена школьников 8 "Б". При помощи itertools.combinations вывести все способы назначения трёх дежурных.

Гарантируется, что в 8 "Б" классе не менее 3 человек.

Формат ввода

Имена школьников перечислены в одной строке через пробелы в файле studygroup.txt.

Формат вывода

Выведите несколько строк вида

1: student_A 2: student_B 3: student_C

Имена школьников и сами строки должны быть выведены в том порядке, в котором их предполагает вывод функции itertools.combinations().

Используйте стандартный вывод на экран, то есть через print.

Между двоеточием и именем ученика - один пробел, между именем ученика и цифрой - один пробел.

Пример

Ввод	Вывод
Dima Petya Sasha Misha Egor	1: Dima 2: Petya 3: Sasha 1: Dima 2: Petya 3: Misha 1: Dima 2: Petya 3: Egor 1: Dima 2: Sasha 3: Misha 1: Dima 2: Sasha 3: Egor 1: Dima 2: Misha 3: Egor 1: Petya 2: Sasha 3: Misha 1: Petya 2: Sasha 3: Egor 1: Petya 2: Misha 3: Egor 1: Sasha 2: Misha 3: Egor

Примечания

1. Обратите внимание, что разрешается ввод только через studygroup.txt, а вывод только на экран (обычной вывод через print)! *(см. рекомендации к вводу-выводу)*.
2. В этой задаче **не нужно** применять сортировку. Имена школьников в каждой строке и сами строки должны выводиться ровно в том порядке, в котором предлагает вывод функции из библиотеки itertools.
3. Не забывайте в начале кода делать импорт библиотеки itertools: import itertools as it
4. В данной задаче **не нужен** модуль fractions!
5. Не забывайте, что readlines() **всегда** возвращает **список**. Если в исходном файле всего одна строка, readlines() вернёт список **из одного** элемента.

Примечания

1. Обратите внимание, что разрешается ввод только через studygroup.txt, а вывод только на экран (обычной вывод через print)! *(см. рекомендации к вводу-выводу)*.
2. В этой задаче **не нужно** применять сортировку. Имена школьников в каждой строке и сами строки должны выводиться ровно в том порядке, в котором предлагает вывод функции из библиотеки itertools.
3. Не забывайте в начале кода делать импорт библиотеки itertools: import itertools as it
4. В данной задаче **не нужен** модуль fractions!
5. Не забывайте, что readlines() **всегда** возвращает **список**. Если в исходном файле всего одна строка, readlines() вернёт список **из одного** элемента.
6. Не забывайте, что split() применяется **к строкам** (**не** к спискам)

Набрать здесь | Отправить файл

```
import itertools as it
s = open('studygroup.txt')
c = s.read()
d=c.split(" ")

for x, y, z in it.combinations(d, 3):
    print('1:', x,'2:', y,'3:', z, sep=' ', end='\n')
s.close()
```


```
import itertools as it
s = open('studygroup.txt')
c = s.read()
d=c.split(" ")
for x, y, z in it.combinations(d, 3):
    print('1:', x,'2:', y,'3:', z, sep=' ', end='\n')
s.close()
```

Контест 4

Конспект:
https://colab.research.google.com/drive/1ltKt_-m29veUzT0ol4agjsc3nYd1_jO8?usp=sharing

1. Перевод из двоичной системы счисления в десятичную

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	стандартный ввод или input.txt
Вывод	стандартный вывод или output.txt

Перевести число из двоичной системы счисления в десятичную.

Формат ввода

s — число в двоичном представлении, записанное при помощи цифр 0 и 1, длиной не более 8 символов.

Формат вывода

Число a ($0 \le a \le 255$) в десятичном представлении.

Пример 1

Ввод	Вывод
00000001	1

Пример 2

Ввод	Вывод
11111111	255

```
n=input()
dec=int(n, 2)
print(dec)
```

2. Вычисление дроби

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	стандартный ввод или input.txt
Вывод	стандартный вывод или output.txt

Вычислить при помощи класса Fraction библиотеки fractions значение следующего выражения:

$$\frac{b}{a} + \frac{b}{a+c} - \frac{c}{c-a}$$

Формат ввода

6 чисел, по 2 числа на строку (через пробел):

a_n, a_d – числитель и знаменатель a, то есть $a = \frac{a_n}{a_d}$;

b_n, b_d – числитель и знаменатель b, то есть $b = \frac{b_n}{b_d}$;

c_n, c_d – числитель и знаменатель c, то есть $c = \frac{c_n}{c_d}$;

Формат вывода

Численное выражение $\frac{b}{a} + \frac{b}{a+c} - \frac{c}{c-a}$ с точностью до 4-х знаков после запятой.

Пример

Ввод	Вывод
-48 29 -538 363 -643 518	4.4065

Примечания

1. В данной задаче подразумевается ввод **со стандартного потока ввода (с клавиатуры) или из файла input.txt!** *(см. рекомендации к вводу-выводу)*.
2. Гарантируется, что ответ существует, то есть **деление на 0 не происходит**.
3. Не забывайте, что один вызов input() считывает **только одну строку**. По условию, входные данные вводятся **не в одну строку**, а по 2 числа в каждой строке. Т.е. нужно считать 3 строки.
4. Можно построить объект Fraction, передав в него либо пару чисел (Fraction(1, 3)), либо строковое представление дроби, где числитель и знаменатель разделены '/' (Fraction('3/7')).
5. В данной задаче **обязательно** использовать модуль fractions! От этого зависит точность вычислений, соответственно, результат, выводимый в качестве ответа.
6. Для форматированного вывода можно использовать так называемые f-строки. Подробнее - см. *лекции* и *рекомендации к вводу-выводу*.

Набрать здесь | Отправить файл

```python
from fractions import Fraction
a_inp = input()
b_inp = input()
c_inp = input()
a = Fraction(a_inp.replace(" ", "/"))
b = Fraction(b_inp.replace(" ", "/"))
c = Fraction(c_inp.replace(" ", "/"))
f = str(float(b/a +b/(a+c) - c/(c-a)))
n = f.find(".")
num = f[0:n+5]
last = int(f[n+5])
if last < 5:
    sol_1 = num
    print(sol_1)
else:
    x = num[n+4]
    last_s = str(int(x) + 1)
    num_1 = num[::-1]
    num_2 = num_1.replace(x, last_s, 1)
    sol_2 = num_2[::-1]
    print(sol_2)
```

```
from fractions import Fraction
a_inp = input()
b_inp = input()
c_inp = input()
a = Fraction(a_inp.replace(" ", "/"))
b = Fraction(b_inp.replace(" ", "/"))
c = Fraction(c_inp.replace(" ", "/"))
f = str(float(b/a +b/(a+c) - c/(c-a)))
n = f.find(".")
num = f[0:n+5]
last = int(f[n+5])
if last < 5:
        sol_1 = num
        print(sol_1)
else:
        x = num[n+4]
        last_s = str(int(x) + 1)
        num_1 = num[::-1]
        num_2 = num_1.replace(x,
```

3. Сумма последовательности чисел

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	стандартный ввод или input.txt
Вывод	стандартный вывод или output.txt

Вычислить сумму последовательности целых чисел, вводимых пользователем с клавиатуры, которая завершается **нулём**.

Формат ввода

Построчно вводятся целые числа. Последовательность заканчивается нулем.

Формат вывода

Вывести 2 числа через пробел: количество чисел (не считая 0) и сумму чисел.

Пример 1

Ввод	Вывод
1 2 3 0	3 6

Пример 2

Ввод	Вывод
0	0 0

Примечания

1. Обратите внимание на **Пример 2**!
2. Используйте цикл while.
3. Не забывайте, что входная последовательность может быть пустой.
4. Во входных данных могут быть и **отрицательные числа**.

Набрать здесь | Отправить файл

```
a=open('input.txt')
c = a.readlines()
d="/".join(c)
d=d.replace('\n',"")
d=d.split("/")
h=0
i=0
t=0
q=len(d)
for h in d:
    n=int(h)
    if n != 0:
        i+=1
        t+=n
print(i, t)
```

Попробуй решить через while, мы не смогли придумать и сделали через for

```
a=open('input.txt')
c = a.readlines()
d="/".join(c)
d=d.replace('\n',"")
d=d.split("/")
h=0
i=0
t=0
q=len(d)
for h in d:
    n=int(h)
    if n != 0:
        i+=1
```

4. Элемент геометрической прогрессии

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	стандартный ввод или input.txt
Вывод	стандартный вывод или output.txt

Дана геометрическая прогрессия, n-ый член которой задается формулой $b_n = b_1 \cdot q^{n-1}$,

b_1 - первый член геометрической прогрессии, q - знаменатель прогрессии.

Найти номер наименьшего члена геометрической прогрессии, превышающего заданное пользователем число a.

Формат ввода

Даны 3 числа, введенные через пробел:

$b_1 \in (0, 10)$ — первый член прогрессии,

$q \in (1, 256]$ — знаменатель прогрессии,

$a \in [0, 10^6]$ — заданное число.

Формат вывода

Наименьший номер n, такой что: $b_n > a$.

Пример

Ввод	Вывод
1 2 3	3

Примечания

1. Прогрессия возрастающая, то есть: $q > 1$.
2. $b_1 > 0$.
3. В условии не сказано, что входные данные - целые числа. Значит, все числа **могут быть дробными**. Для преобразования таких строк в число следует использовать float() (а **не** int()).
4. Входные данные вводятся в одной строке.
5. В данной задаче можно обойтись без округления.
6. Обратите внимание, что в задаче требуется найти **номер** n, а **не** само значение b_n.

Набрать здесь | Отправить файл

```python
s=open('input.txt')
c = s.read()
d=c.split(" ")
b1=float(d[0])
q=float(d[1])
a=float(d[2])
n=1
t=b1
while a>t:
    n+=1
    t=b1*q**(n-1)
print(n)
s.close()
```


```python
s=open('input.txt')
c = s.read()
d=c.split(" ")
b1=float(d[0])
q=float(d[1])
a=float(d[2])
n=1
t=b1
while a>t:
    n+=1
    t=b1*q**(n-1)
```

5. Банковский вклад*

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	стандартный ввод или input.txt
Вывод	output.csv

На лекции вы изучили, как расчитывать таблицы для погашения кредита в банке.

По аналогии создайте программу для расчёта банковского вклада на заданный срок. Вклад без ежемесячного пополнения, но с ежемесячным начислением процентов и капитализацией (начислением процентов на проценты). Результат сохраните в файле **output.csv**.

Формат ввода

3 числа, введенные через пробел:

$amount$ ($10^6 \le amount \le 10^7$) — целое число, равное сумме вклада;

$period$ ($1 \le period \le 10$) — целое число, равное сроку (в годах) вклада;

$rate$ ($rate \in [1, 12]$) — число (не обязательно целое), равное процентной ставке вклада (за год).

Формат вывода

Таблица CSV, состоящая из 3-х столбцов с названиями: "Месяц", "Сумма на вкладе", "Начисление"; Необходимо заполнить таблицу соответствующими величинами (см. примечание) с точностью до 2-х знаков после запятой.

Формат вывода

Таблица CSV, состоящая из 3-х столбцов с названиями: "Месяц", "Сумма на вкладе", "Начисление"; Необходимо заполнить таблицу соответствующими величинами (см. примечание) с точностью до 2-х знаков после запятой.

Пример

Ввод	Вывод
1000000 2 10	Месяц,Сумма на вкладе,Начисление 1,1008333.33,8333.33 2,1016736.11,8402.78 3,1025208.91,8472.80 4,1033752.32,8543.41 5,1042366.92,8614.60 6,1051053.31,8686.39 7,1059812.09,8758.78 8,1068643.86,8831.77 9,1077549.22,8905.37 10,1086528.80,8979.58 11,1095583.21,9054.41 12,1104713.07,9129.86 13,1113919.01,9205.94 14,1123201.67,9282.66 15,1132561.68,9360.01 16,1141999.70,9438.01 17,1151516.36,9516.66 18,1161112.33,9595.97 19,1170788.27,9675.94 20,1180544.83,9756.57 21,1190382.71,9837.87 22,1200302.56,9919.86 23,1210305.09,10002.52 24,1220390.96,10085.88

Примечания

Обратите внимание, что при записи в output.csv разделителем служит **только запятая** без пробелов и прочих символов. В качестве символа завершения строки используется символ '\n'. В качестве разделителя для целой и дробной части вещественного числа используется '.'.

Дополнительные комментарии:

1. Процентная ставка может быть вещественной.
2. Обратите внимание, что период вводится в годах, в таблицу же нужно выводить начисления за каждый месяц!
3. Ставка так же вводится за год и в процентах. Поэтому для удобства можно посчитать дробное значение ставки за месяц: rate_month = rate / 12.0 / 100.0.
4. В задаче можно пользоваться только теми библиотеками и средствами, которые были изучены на лекции.
5. Проверяющая система сравнивает файл, который получается у Вас, с эталонным файлом, который получается после работы верного решения. И сравнение это происходит **посимвольно**. Т.е. если хоть в каком-то символе будет различие, то решение не засчитается, к сожалению. Поэтому проверяйте корректность названий колонок!
6. Для округления в этой задаче **не** рекомендуется использовать round(). Потому что он **не гарантирует вывод двух знаков после запятой** (т.е. число 1.5 выведется как «1.5» - а нужно чтобы вывелось «1.50»). Тестовая система сравнивает Ваш результат с эталонным **посимвольно**. Поэтому, если хоть в каком-то символе будет различие - решение не засчитывается. Лучше использовать форматный вывод (f-строки). Подробнее - см. *рекомендации к вводу-выводу*.
7. Обратите внимание, что ставку нужно рассчитывать ровно на тот период, который указывается во входных данных (а **не** всегда на 24 месяца, как в примере).

```python
s=open('input.txt')
c = s.read()
d=c.split(" ")
amount =float(d[0])
period=float(d[1])
rate =float(d[2])
period=period*12
amount1=amount
i=1
f=open('output.csv', 'a')
print("Месяц", "Сумма на вкладе", "Начисление", sep=",", file=f)
while i<=period:
    amount2 = amount1
    amount1=amount*(1+(rate/12/100))**i
    gain=amount1-amount2
    print(f"{i},{amount1:0.2f},{gain:0.2f}", sep=",", file=f)
    i+=1
s.close()
f.close()
```

```
s=open('input.txt')
c = s.read()
d=c.split(" ")
amount =float(d[0])
period=float(d[1])
rate =float(d[2])
period=period*12
amount1=amount
i=1
f=open('output.csv', 'a')
print("Месяц", "Сумма на вкладе", "Начисление", sep=",", file=f)
while i<=period:
    amount2 = amount1
    amount1=amount*(1+(rate/12/100))**i
    gain=amount1-amount2
    print(f"{i},{amount1:0.2f},{gain:0.2f}", sep=",", file=f)
    i+=1
s.close()
f.close()
```

Контест 5

Конспект:
https://colab.research.google.com/drive/1NstnAsm0qS3f6FQHgJNV-PpMpvHsrKIL?usp=sharing

1. Две команды

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	стандартный ввод или weights.txt
Вывод	team.txt

Для игры в американский футбол собрались 22 мальчика с разным весом. Их вес (у всех он разный) и их имена (тоже у всех разные) даны в текстовом файле. Требуется создать две команды следующим образом: отсортировать всех ребят по *убыванию веса*, а затем 1-го, 3-го, 5-го и т.д. игроков отобрать в 1-ю команду, а оставшихся — во 2-ю команду. Вывести список команды №1 в первые 11 строк файла **team.txt**, а список команды №2 в последние 11 строк файла **team.txt**. В списках команд указывать и имена, и значения веса.

Формат ввода

Текстовый файл **weights.txt** содержит в каждой строке пары ИМЯ ВЕС (через пробел). Например, John 57. Вес может быть вещественным числом.

Формат вывода

Файл **team.txt**, содержащий строки исходного файла, отсортированные в том порядке, который требуется в данном задании.

Формат вывода

Файл **team.txt**, содержащий строки исходного файла, отсортированные в том порядке, который требуется в данном задании.

Пример

Ввод	Вывод
A 1.5	V 23.5
B 2.5	T 21.5
C 3.5	R 19.5
D 4.5	P 17.5
E 6.5	N 15.5
F 7.5	L 13.5
G 8.5	J 11.5
H 9.5	H 9.5
I 10.5	F 7.5
J 11.5	D 4.5
K 12.5	B 2.5
L 13.5	U 22.5
M 14.5	S 20.5
N 15.5	Q 18.5
O 16.5	O 16.5
P 17.5	M 14.5
Q 18.5	K 12.5
R 19.5	I 10.5
S 20.5	G 8.5
T 21.5	E 6.5
U 22.5	C 3.5
V 23.5	A 1.5

Примечания

1. Для сортировки можно использовать sorted() с указанием параметров (см. лекцию).
2. Для сортировки по убыванию веса можно указать в качестве параметра sorted() значение reverse=True.
3. В задаче **не** разрешается использовать библиотеку NumPy.

Набрать здесь | Отправить файл

```
s = open('weights.txt')
c = s.readlines()
d = "/".join(c)
d = d.replace('\n', "")
d = d.split("/")
d = sorted(d, key=lambda w: float(w.split()[1]), reverse=True)
team_1 = "\n".join(d[::2])
team_2 = "\n".join(d[1::2])
fi = open('team.txt', 'a')
fi.writelines(team_1 + "\n" + team_2)
fi.close()
s.close()
```


```
s = open('weights.txt')
c = s.readlines()
d = "/".join(c)
d = d.replace('\n', "")
d = d.split("/")
d = sorted(d, key=lambda w: float(w.split()[1]), reverse=True)
team_1 = "\n".join(d[::2])
team_2 = "\n".join(d[1::2])
fi = open('team.txt', 'a')
fi.writelines(team_1 + "\n" + team_2)
```

2. Неопубликованный шедевр

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	стандартный ввод или poe_unpublished.txt
Вывод	poe_decode_attempt.txt

Журналист обнаружил в архиве ранее не публиковавшуюся шифровку Эдгара Алана По **poe_unpublished.txt**. Чтобы не просто представить её публике, а совершить сенсацию, он хочет расшифровать её и для этого нанял вас. Вы разобрались, что в тексте намеренно перепутаны строки, а в каждой строке перепутаны слова. Для применения алгоритма расшифрования нужно упорядочить строки по возрастанию количества слов в каждой строке (слова разделяются пробелами), а внутри каждой строки слова нужно упорядочить по количеству букв в них. Результат нужно сохранить в файл **poe_decode_attempt.txt**

Формат ввода

Список зашифрованных строк, записанных в файле **poe_unpublished.txt**

Формат вывода

Список расшифрованных строк, записанных в файл **poe_decode_attempt.txt**

Пример

Ввод	Вывод
I was angry with my friend	I my was with angry friend
I told my wrath my wrath did end	I my was foe with angry
I was angry with my foe	I my my did end told wrath wrath
I told it not my wrath did grow	I it my not did told grow wrath

Примечания

1. В конце выводимых строк не нужны лишние пробелы!
2. Обратите внимание, что строки по умолчанию сортируются в лексикографическом порядке, по условию же их нужно сортировать **по количеству слов**.
3. При считывании данных через readlines() вместе с каждой строкой сохраняется завершающий ее символ \n. Это нужно не забывать и учитывать при вычислении длины слов (символ \n учитываться не должен).

Набрать здесь | Отправить файл

```python
s = open('poe_unpublished.txt')
c = s.readlines()
d = "/".join(c)
d = d.replace('\n', "")
d = d.split("/")
list_1 = []
for words in d:
    words = words.split()
    words.sort(key=len)
    list_1.append(words)
list_1 = sorted(list_1, key=len)
fi = open('poe_decode_attempt.txt', 'a')
for stings in list_1:
    t=" ".join(stings)
    fi.writelines(t+ "\n")
fi.close()
s.close()
```

```
s = open('poe_unpublished.txt')
c = s.readlines()
d = "/".join(c)
d = d.replace('\n', "")
d = d.split("/")
list_1 = []
for words in d:
    words = words.split()
    words.sort(key=len)
    list_1.append(words)
list_1 = sorted(list_1, key=len)
fi = open('poe_decode_attempt.txt', 'a')
for stings in list_1:
    t=" ".join(stings)
    fi.writelines(t+ "\n")
fi.close()
s.close()
```

3. Транспонирование

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	стандартный ввод или med_research.txt
Вывод	output.txt

Биолог провёл исследование устойчивости нескольких видов бактерий стрептококка к нескольким антибиотикам и получил численные оценки, характеризующие эффективность действия каждого препарата. Его данные хранятся в формате текстовой таблицы дробных чисел (с плавающей точкой) **med_research.txt**, при этом по горизонтали изменяются виды бактерий, а по вертикали — виды антибиотиков. Однако для отчёта требуется, чтобы виды антибиотиков шли по горизонтали, а виды бактерий — по вертикали. Биолог узнал, что эта операция называется в математике транспонированием, но он не знает, как её сделать автоматически. Если вы не поможете, то ему предстоит провести ночь, вручную переставляя числа в огромном листе. *В файле хранятся только сами числа*. Список видов бактерий и список антибиотиков вас не интересуют, биолог вставит их сам.

Формат ввода

Ввод из файла **med_research.txt**, все значения в строках разделены пробелом. Количество строк в исходной матрице >= 2, количество столбцов в исходной матрице >= 2.

Формат вывода

Вывод в файл **output.txt**.

Формат вывода

Вывод в файл **output.txt**.

Пример

Ввод	Вывод
1 2 3 4 5 6 7 8 9 10	1 6 2 7 3 8 4 9 5 10

Примечания

1. Матрица не обязательно должна состоять из 5-и столбцов или 2-х строк, может быть 3x3, 4x10, 1000x261 и т.д., матрица может быть **любого** размера, с учетом ограничения:

количество строк в исходной матрице >= 2, количество столбцов в исходной матрице >= 2.

2. Элементы исходной матрицы **не обязаны быть целыми**.
3. Библиотеками numpy и pandas пользоваться **запрещено**!
4. Тип float по умолчанию выводится с дробной частью. Вам же в задании совершенно не важно, какого типа число - целое или нет. Просто оставляйте его в виде строки. Т.е. Число 2 должно записываться в выходной файл как "2", а не как "2.0".

```python
s = open('med_research.txt')
c = s.readlines()
d = "/".join(c)
d = d.replace('\n', "")
d = d.split("/")
list_of_lists  = []
transposed=list()
for num in d:
    num = num.split()
    list_of_lists.append(num)
for i in range(len(list_of_lists[0])):
    row = list()
    for sublist in list_of_lists:
        row.append(sublist[i])
    transposed.append(row)
fi = open('output.txt', 'a')
for item in transposed:
    t = " ".join(item)
    fi.writelines(t + "\n")
fi.close()
s.close()
```


```python
s = open('med_research.txt')
c = s.readlines()
d = "/".join(c)
d = d.replace('\n', "")
d = d.split("/")
list_of_lists  = []
transposed=list()
for num in d:
    num = num.split()
    list_of_lists.append(num)
for i in
range(len(list_of_lists[0])):
    row = list()
    for sublist in list_of_lists:
        row.append(sublist[i])
    transposed.append(row)
fi = open('output.txt', 'a')
for item in transposed:
    t = " ".join(item)
    fi.writelines(t + "\n")
```

4. Логирование звонков

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	стандартный ввод или the_calls.txt
Вывод	calls.txt

Руководитель хочет узнать, какой из двух его заместителей A или B больше времени тратит на звонки по телефону. Для этого он установил шпионское ПО на их смартфоны и получил файл **the_calls.txt**, в котором для каждого звонка по телефону указано: дата в формате YYYY.MM.DD, длительность звонка в секундах (целое число), буква A или B, а также номер телефона, на который был совершён звонок (без пробелов, но со значком + и скобками). Поля в строке разделяются знаком табуляции.

Руководителю более подозрительны звонки, *длившиеся дольше всего*, поэтому строки следует отсортировать в обратном порядке по длительности звонка (сначала самые длительные).

Распределите строки из файла на два упорядоченных указанным образом списка, и запишите их в файл **calls.txt**: Сначала группу звонков A, затем - группу звонков B.

Формат ввода

Строки, записанные в файл **the_calls.txt** в описанном в условии формате.

Формат вывода

Вывод в файл **calls.txt**

Пример

Ввод

```
2021.12.12	3	B	+7
2019.10.03	18	A	+7
2020.05.04	13	B	+7
2020.05.04	4	A	+7
```

Вывод

```
2019.10.03	18	A	+7
2020.05.04	4	A	+7
2020.05.04	13	B	+7
2021.12.12	3	B	+7
```

Примечания

1. Не забывайте, что в первоначальном виде длительность звонка - это строка. Поэтому к ключу сортировки нужно добавлять вызов метода int().
2. Во входном файле данные разделяются не пробелом, а **знаком табуляции**. Нужно делать str.split('\t').
3. Проверьте, что в выходном файле **не нужна** лишняя пустая строка в конце.
4. Разрешается ввод и вывод **только через файл** *(см. рекомендации к вводу-выводу)*.
5. Обратите внимание, что в вашем коде названия входного и выходного файла должны совпадать с названиями, указанными в условии.
6. Проверьте корректность перехода на новую строку после вывода всех номеров от A и **перед** выводом номеров от B. В частности, проверьте ситуацию, когда во входном файле на последней строке номер от B (и он **не самый короткий**) (без \n в конце).

Набрать здесь | Отправить файл

```python
s = open('the_calls.txt')
c = s.readlines()
d = "/".join(c)
d = d.replace('\n', "")
d = d.split("/")
d = sorted(d, key=lambda w: (str(w.split("\t")[2]), -int( w.split("\t") [1])))
p='\n'.join(d)
fi=open('calls.txt',"a")
fi.writelines(p)
fi.close()
s.close()
```

```
s = open('the_calls.txt')
c = s.readlines()
d = "/".join(c)
d = d.replace('\n', "")
d = d.split("/")
d = sorted(d, key=lambda w: (str(w.split("\t")[2]), -int( w.split("\t") [1])))
p='\n'.join(d)
fi=open('calls.txt',"a")
fi.writelines(p)
fi.close()
s.close()
```

Контест 6

Конспект:
https://colab.research.google.com/drive/1ru8FIjKqeeZ5ccpMty9_ctt3Q9uFRQyx?usp=sharing

1. Структуры данных в Python

Ниже приведены данные 3-х различных типов. Сопоставьте типы данных и значения, которые они могут принимать.

1. ["Python", "C++", "Java"]
2. {1, 2, 3, 5, 7, 11, 13, 17}
3. {"Россия": "Москва", "Франция": "Париж", "Япония": "Токио"}

Примечания

В первой колонке нужно отметить, к какой структуре относятся данные под цифрой 1 (["Python", "C++", "Java"]): список, словарь или множество.

	1	2	3
Список (list)	●	○	○
Словарь (dict)	○	○	●
Множество (set)	○	●	○

2. Пересечение множеств

Даны два множества:

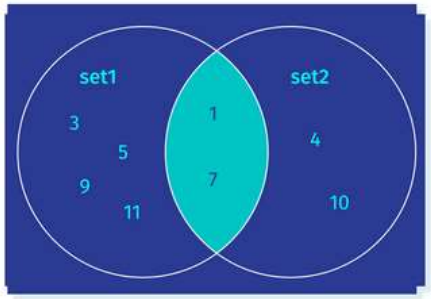
set1 = {1, 3, 5, 7, 9, 11}

set2 = {1, 4, 7, 10}

С помощью каких инструментов Python можно найти пересечение этих множест

Примечания

Пересечения множеств set1 и set2 – это те элементы, которые принадлежат множествам set1 и set2 одновременно, то есть числа 1 и 7.



Подробнее про методы работы с множествами: см. документ.

- [x] set1.intersection(set2)
- [ ] set1 | set2
- [x] set1 & set2
- [ ] set1.difference(set2)

3. Словари

Каким образом можно добавить новый элемент в словарь the_dict = {"Россия": "Москва", "Япония": "Токио", "Италия": "Рим"}?

Примечания

Об этом можно прочитать в лекции.

- (●) `the_dict["Франция"] = "Париж"`
- ( ) `the_dict.append("Париж")`
- ( ) `the_dict.add("Париж")`

4. Наиболее часто встречающаяся цифра

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	стандартный ввод или input.txt
Вывод	стандартный вывод или output.txt

Провести частотный анализ числа и найти наиболее часто встречающуюся цифру.

Формат ввода

Подается длинное число, состоящее из n ($1 \le n \le 1000$) цифр.

Формат вывода

Самая часто встречающая цифра $b \in \{0, 1, \ldots, 9\}$.

Пример

Ввод	Вывод
2021	2

Примечания

Гарантируется, что такая цифра в числе одна, то есть ответ однозначный.

Подсказка: см. документ.

```
1 from collections import Counter
2 def most_frequent(List):
3     occurence_count = Counter(List)
4     return occurence_count.most_common(1)[0][0]
5 List=list(input())
6 print(most_frequent(List))
```


```
from collections import Counter
def most_frequent(List):
    occurence_count = Counter(List)
    return occurence_count.most_common(1)[0][0]
List=list(input())
print(most_frequent(List))
```

5. Все виды животных

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	input.txt
Вывод	стандартный вывод

В определенном зоопарке находятся животные, каждому из которых присвоены следующие параметры:
ID, вид, пол, кличка, дата рождения и дата поступления в зоопарк. Исходя из этих данных, вывести все виды животных, которые есть в данном зоопарке.

Формат ввода

Имеется текстовый файл input.txt, который содержит информацию про конкретного животного в каждой строчке:

$$ID_1 \quad animal_1 \quad gender_1 \quad nickname_1 \quad dateofbirth_1 \quad dateofarrive_1$$

$$ID_2 \quad animal_2 \quad gender_2 \quad nickname_2 \quad dateofbirth_2 \quad dateofarrive_2$$

$$\vdots$$

$$ID_n \quad animal_n \quad gender_n \quad nickname_n \quad dateofbirth_n \quad dateofarrive_n$$

$1 \le n \le 1000$.

Формат вывода

Вывести через строчку названия всех видов животных в порядке увеличения длины названия, т. е.:

$animal_{k_1}$

$animal_{k_2}$

$\vdots$

$animal_{k_m}$

Где $|animal_{k_1}| < |animal_{k_2}| < \ldots < |animal_{k_m}|$, $m \le n$.

Ввод	Вывод
0042 cat male Pushok 13.09.2015 31 0003 elephant male Archi 03.03.201:	cat elephant

Пример 2

Ввод	Вывод
7910 leopard male Leo 04.06.2001 0: 9315 orangutan male Hiha 01.04.200 2226 leopard female Lia 07.28.2007	leopard orangutan

Примечания

1. Входные данные считываются из input.txt, а вывод стандартный, то есть через print() *(см. рекомендации к вводу-выводу)*.
2. Гарантируется, что вывод **однозначный** (**не существует** разных названий животных с одинаковой длиной названия);
3. Порядок вывода **имеет значение**.
4. Подсказки по решению: *см. документ*.
5. Не забывайте, что по умолчанию строки сортируются *в лексикографическом порядке*. В условии требуется сортировка **по длине строки**. Для этого используйте параметр key в функции sort.

Набрать здесь | Отправить файл

```python
s = open('input.txt')
c = s.readlines()
d = "/".join(c)
d = d.replace('\n', "")
d = d.split("/")
animals = set()
list_1 = []
for words in d:
    words = words.split()
    list_1.append(words)
for an in list_1:
    f = an[1]
    animals.add(f)
animals = sorted(animals, key=len)
print(*animals, sep='\n')
s.close()
```

```
s = open('input.txt')
c = s.readlines()
d = "/".join(c)
d = d.replace('\n', "")
d = d.split("/")
animals = set()
list_1 = []
for words in d:
    words = words.split()
    list_1.append(words)
for an in list_1:
    f = an[1]
    animals.add(f)
animals = sorted(animals,
key=len)
print(*animals, sep='\n')
s.close()
```

6. Создание потомства

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	input.txt
Вывод	стандартный вывод

В определенном зоопарке находятся животные, каждому из которых присвоены следующие параметры:
ID, вид, пол, кличка, дата рождения и дата поступления в зоопарк. Исходя из этих данных, вывести все виды животных, которые могут размножаться в данном зоопарке (то есть виды животных, которые представлены в зоопарке как мужским, так и женским полом).

Формат ввода

Имеется текстовый файл input.txt, который содержит информацию про конкретного животного в каждой строчке:

$$ID_1 \quad animal_1 \quad gender_1 \quad nickname_1 \quad dateofbirth_1 \quad dateofarrive_1$$

$$ID_2 \quad animal_2 \quad gender_2 \quad nickname_2 \quad dateofbirth_2 \quad dateofarrive_2$$

$$\vdots$$

$$ID_n \quad animal_n \quad gender_n \quad nickname_n \quad dateofbirth_n \quad dateofarrive_n$$

$1 \le n \le 1000$.

Формат вывода

Вывести через строчку названия всех животных, которые могут давать потомство, **в порядке увеличения длины названия:**

$animal_{k_1}$

$animal_{k_2}$

$\vdots$

$animal_{k_m}$.

Где $|animal_{k_1}| < |animal_{k_2}| < \ldots < |animal_{k_m}|$, $m \le n$.

В противном случае вывести 0.

Пример 1

Ввод

```
0042 cat male Pushok 13.09.2015 31
0043 dog female Dora 29.01.2015 31
0003 elephant male Archi 03.03.201:
```

Вывод

```
0
```

Пример 2

Ввод

```
0042 cat male Pushok 13.09.2015 31
0043 dog female Dora 29.01.2015 31
0003 cat female Eva 03.03.2017 31.:
```

Вывод

```
cat
```

Примечания

1. Обратите внимание, что входные данные считываются из файла input.txt (*см. рекомендации к вводу-выводу*).
2. Гарантируется, что вывод **однозначный** (не существует животных с одинаковой длиной названия);
3. Порядок вывода **имеет значение**.
4. Подсказка к решению: *см. документ*.
5. Каждое животное нужно выводить **в отдельной строке**.
6. Если животных, которые могут дать потомство, нет, **нужно выводить 0**.
7. Рассмотрите ситуацию, когда во входных данных вообще **нет мальчиков или девочек**.

Набрать здесь | Отправить файл

```python
s = open('input.txt')
c = s.readlines()
d = "/".join(c)
d = d.replace('\n', "")
d = d.split("/")
animals = set()
animals2 = set()
animals1 = set()
list_1 = []
for words in d:
    words = words.split()
    list_1.append(words)
for an in list_1:
    if an[2]=='male':
        f = an[1]
        animals1.add(f)
    if an[2]=='female':
        f = an[1]
        animals2.add(f)
animals=animals1.intersection(animals2)
if animals!=set():
    animals = sorted(animals, key=len)
    print(*animals, sep='\n')
else:
    print('0')
s.close()
```


```python
s = open('input.txt')
c = s.readlines()
d = "/".join(c)
d = d.replace('\n', "")
d = d.split("/")
animals = set()
animals2 = set()
animals1 = set()
list_1 = []
for words in d:
    words = words.split()
    list_1.append(words)
for an in list_1:
    if an[2]=='male':
        f = an[1]
        animals1.add(f)
    if an[2]=='female':
        f = an[1]
        animals2.add(f)
animals=animals1.intersection(animals2)
if animals!=set():
    animals = sorted(animals, key=len)
    print(*animals, sep='\n')
else:
    print('0')
s.close()
```

7. Наиболее часто встречающееся животное

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	input.txt
Вывод	стандартный вывод

В определенном зоопарке находятся животные, каждому из которых присвоены следующие параметры:
ID, вид, пол, кличка, дата рождения и дата поступления в зоопарк. Исходя из этих данных, вывести количество животных для каждого вида в порядке убывания.

Формат ввода

Имеется текстовый файл input.txt, который содержит информацию про конкретного животного в каждой строчке:

$ID_1 \ animal_1 \ gender_1 \ nickname_1 \ dateofbirth_1 \ dateofarrive_1$

$ID_2 \ animal_2 \ gender_2 \ nickname_2 \ dateofbirth_2 \ dateofarrive_2$

$\vdots$

$ID_n \ animal_n \ gender_n \ nickname_n \ dateofbirth_n \ dateofarrive_n$

$1 \le n \le 1000$.

Формат вывода

Вывести через строчку количество животных для каждого вида в порядке убывания:

$animal_{k_1} - a_1$

$animal_{k_2} - a_2$

$\vdots$

$animal_{k_m} - a_m$.

$m \le n$, $a_1 > a_2 > \ldots > a_m$.

Пример

Ввод	Вывод
0042 cat male Pushok 13.09.2015 31 0043 dog female Dora 29.01.2015 31 0003 dog male Max 03.03.2011 31.12	dog - 2 cat - 1

Примечания

1. Обратите внимание, что входные данные считываются из файла input.txt *(см. рекомендации к вводу-выводу)*.
2. Гарантируется, что вывод **однозначный**. Т.е. животных с одинаковым количеством упоминаний **нет**.
3. В выводе используются **пробелы** (не символы табуляции и не какие-то другие символы).
4. Название животного нужно искать **не во всей строке** строке, а только **в графе с названием животного**. Потому что может быть животное с названием Tomcat. Тут есть подстрока 'cat' - но он не обязан быть котом.

```python
from collections import Counter
def most_frequent(List):
    occurence_count = Counter(List)
    return occurence_count.most_common(1)[0][0]
s = open('input.txt')
c = s.readlines()
d = "/".join(c)
d = d.replace('\n', "")
d = d.split("/")
animals = list()
list_1 = []
for words in d:
    words = words.split()
    list_1.append(words)
for an in list_1:
    f = an[1]
    animals.append(f)
List = animals
count = (Counter(List))
animal = list(count.keys())
value = list(count.values())
value = str(value)
value = value.replace("[", "")
value = value.replace("]", "")
value = value.replace(",", "")
value = value.split(' ')
zipped_animals = list(zip(animal, value))
zip_an = []
for t in zipped_animals:
    t = " - ".join(t)
    zip_an.append(t)
zip_an = sorted(zip_an, key=lambda w: int(w.split()[-1]), reverse=True)
print(*zip_an, sep="\n")
s.close()
```


```python
from collections import Counter
def most_frequent(List):
    occurence_count = Counter(List)
    return occurence_count.most_common(1)[0][0]
s = open('input.txt')
c = s.readlines()
d = "/".join(c)
d = d.replace('\n', "")
d = d.split("/")
animals = list()
list_1 = []
for words in d:
    words = words.split()
    list_1.append(words)
for an in list_1:
    f = an[1]
    animals.append(f)
List = animals
count = (Counter(List))
animal = list(count.keys())
value = list(count.values())
value = str(value)
value = value.replace("[", "")
value = value.replace("]", "")
value = value.replace(",", "")
value = value.split(' ')
zipped_animals = list(zip(animal, value))
zip_an = []
for t in zipped_animals:
    t = " - ".join(t)
    zip_an.append(t)
```

8. ID животных*

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	input.txt
Вывод	стандартный вывод

В определенном зоопарке находятся животные, каждому из которых присвоены следующие параметры:
ID, вид, пол, кличка, дата рождения и дата поступления в зоопарк. Исходя из этих данных, вывести ID всех животных для каждого вида.

Формат ввода

Имеется текстовый файл input.txt, который содержит информацию про конкретного животного в каждой строчке:

$ID_1 \ animal_1 \ gender_1 \ nickname_1 \ dateofbirth_1 \ dateofarrive_1$

$ID_2 \ animal_2 \ gender_2 \ nickname_2 \ dateofbirth_2 \ dateofarrive_2$

$\vdots$

$ID_n \ animal_n \ gender_n \ nickname_n \ dateofbirth_n \ dateofarrive_n$

$1 \le n \le 1000$.

Формат вывода

Вывести через строчку ID всех животных определенного вида следующим образом:

$animal_{k_1}$: ID_1, ID_2, …
$animal_{k_2}$: ID_1, ID_2, …
⋮
$animal_{k_m}$: ID_1, ID_2, …

Где длина названия вида животных увеличивается с каждой строчки, то есть:
$|animal_{k_1}| < |animal_{k_2}| < \ldots < |animal_{k_m}|$.

Также ID выводится в порядке увеличения значения, то есть:
$ID_1 \le ID_2 \le \ldots$

$m \le n$.

Пример

Ввод	Вывод
0042 lion male Pushok 13.09.2015 3 0043 dog female Dora 29.01.2015 31 0003 dog male Max 03.03.2011 31.12	dog: 0003, 0043 lion: 0042

Примечания

1. Обратите внимание, что входные данные считываются из файла input.txt (см. рекомендации к вводу-выводу).
2. Гарантируется, что вывод **однозначный** (не существует животных с одинаковой длиной названия);
3. Порядок вывода **имеет значение**.
4. Идентификаторы животных **могут повторяться** во входных данных. Соответственно, в выходных данных они должны так же повторяться.
5. Нужно сортировать не только имена животных, но и **идентификаторы**.

Набрать здесь | Отправить файл

```
result = {}
with open('input.txt') as file:
    for line in file:
        data = line.split()
        result.setdefault(data[1], []).append(data[0])
for key in sorted(result, key=len):
    print(f'{key}: {", ".join(sorted(result[key]))}')
```

```
result = {}
with open('input.txt') as file:
    for line in file:
        data = line.split()
        result.setdefault(data[1],
[]).append(data[0])
```

Контест 7

Конспект:
https://colab.research.google.com/drive/1q2h_xhO9hPWaoi59thzhwBEgjuz3WG30?usp=sharing

1. Underflow error

Физик Леонид придумал, как решить все проблемы современной физики. Он придумал новую формулу, но для того, чтобы воспользоваться ей, ему нужно сложить число порядка 1e-12 и число порядка 1e-55

(1e-12 - это 10^{-12} , 1e-55 - это 10^{-55}).

Попробовав сделать это, удивленный Леонид обнаружил, что его программа выдает ему какой-то неправильный результат:

1e-12 + 1e-55 = 1e-12

(проверьте на Python).

Леонид спросил совета у своего друга-программиста, и тот рассказал ему, что все дело в том, что вещественные числа в представлении ЭВМ - очень большое, но все же дискретное множество, и что пытаясь сложить два очень маленьких числа, Леонид вызывает т. н. **underflow problem**. Расстроенный Леонид решил, что может пренебречь точностью вычислений, но ему нужно знать, какой минимальный порядок числа он может прибавить к числу 1e-12, чтобы программа смогла исполниться корректно, то есть нужно найти такое число n, что:

1e-12 + 1e-n - вычисляется корректно, то есть 1e-12 + 1e-n $\neq$ 1e-12,

а 1e-12 + 1e-m = 1e-12 , где m = n+1.

Помогите Леониду найти такое число.

Формат вывода

Напишите число (вручную) n, где n — это показатель степени в числе 1e-n, являющемся ответом к задаче.

Примечания

1. Вы можете задать число типа float при помощи вызова функции float(str), где str будет иметь вид '1e-n'.
2. Информация по данной теме: см. 32 минуту лекции.
3. Обратите внимание, что количество попыток по данной задаче **меньше, чем обычно** (чтобы избежать решения методом подбора).
4. В задании требуется написать **одно число** в поле ответа.

Ответ: 27

2. Управление пекарней

Пекарь Иван работает в кондитерской. Иван с самого утра готовит много выпечки, чтобы в течение дня продать ее. Иногда бывает, что Иван готовит слишком много выпечки, и она пропадает, а иногда - слишком мало, и клиенты остаются недовольны. В обоих случаях Иван несет убытки. Он обратился к другу-математику Григорию, чтобы тот помог ему. Проанализировав данные о посещаемости пекарни Ивана за последний год, Григорий выяснил, что функция зависимости количества посетителей пекарни от дня месяца задается следующим выражением:

$$f(day) = e^{sin(day/100)\cdot day/10} \cdot 20.$$

Помогите Ивану посчитать число клиентов, которые будут у него 18 числа месяца, реализовав функцию при помощи инструментов python.

Формат вывода

Напишите число (вручную) n, где n — это число клиентов, которые будут у него 18 числа месяца.

Примечания

1. Ответ округлите **до целого** при помощи int().
2. Про работу с математическими функциями: см. конспект лекции.

27

3. Битва при Азенкуре 2.0

Ограничение времени	10 секунд
Ограничение памяти	476.8 Мб
Ввод	стандартный ввод или input.txt
Вывод	стандартный вывод

Обратите внимание, что перед отправкой решения данной задачи нужно в качестве языка выбрать Make.

Это можно сделать в нижней части данной страницы, как показано на рисунке:

В процессе археологических раскопок было найдено много документов. Откапывая все новые и новые рукописи, археологи датировали их и складывали в одну стопку. У одного из археологов родилась новая гипотеза о причинах *поражения Французов в битве 1415 года при Азенкуре*. Однако, для проверки гипотезы необходимо найти документ 1415 года. Вам дан список дат, которые приписаны к каждому документу. Даты упорядочены *строго* по возрастанию. При помощи бинарного поиска найдите индекс соответствующего документа. Гарантируется, что документ 1415 года есть в списке.

Вам нужно реализовать **функцию** с именем bin_search(lst) со следующими требованиями:

1. Имя функции - **строго фиксировано** (bin_search).
2. Функции подаётся на вход список lst
   - список **строк**, каждая из которых является **строковым представлением соответствующей даты**. Например: ['1000', '1280', '1415']
3. Функция возвращает **число**
   - искомый индекс 1415 года во входном списке.
4. В качестве отступов в функции используется 4 пробела.
5. Функция реализует алгоритм бинарного поиска.

Приведём пример функции, которую следует сдавать в тестовую систему (в данном случае реализован не бинарный поиск, а обычный, поэтому такое решение засчитано **не будет**):

```
def bin_search(lst):
    for i in range(len(lst)):
        if int(lst[i])==1415:
            return i
```

Формат ввода

Входной параметр функции представляет список **строк**- дат.

Формат вывода

Вывести в качестве возвращаемого функцией bin_search(lst) значения (через return) индекс *i* элемента со значением '1415'.

Пример

Ввод	Вывод
1414 1414 1415	2

Примечания

1. Отсчёт индексов начинается с **нуля**.
2. Обратите внимание, что входной параметр функции - список **строк**.
3. Нужно реализовать **бинарный поиск**, который работает достаточно быстро. Остальные, менее эффективные алгоритмы (например, с find()) не пройдут по ограничению времени. В этом случае будет получена ошибка Runtime Error (RE).
4. Не забывайте, что во входных данных числа представлены в виде **строк**. Обычное сравнение строк происходит в лексикографическом порядке. Т.е. '111'<'12'. Нам же нужно сравнивать именно числа. Поэтому внутри бинарного поиска перед сравнением нужно выполнять преобразование строк к int.
5. Преобразование **всех** элементов исходного списка к int занимает много времени. В бинарном поиске нам **не нужны** все значения списка. Мы только смотрим на середины отдельных срезов исходного списка. Поэтому нужно выполнять преобразование строки к int только для тех элементов, которые мы сравниваем с числом 1415.
6. Вообще говоря, бинарный поиск работает только с **отсортированным** списком.
7. В нашем случае не нужно дополнительно сортировать исходный список, т.к. он **уже отсортирован** (см. условие).
8. При проверке собственной реализации бинарного поиска рассмотрите ситуации:
   - список из одного элемента,
   - список из чётного количества элементов,
   - список из нечётного количества элементов (больше одного).

Язык Make

Набрать здесь | Отправить файл

```
1  def bin_search(lst):
2      val=1415
3      first = 0
4      last = len(lst)-1
5      index = -1
6      while (first <= last) and (index == -1):
7          mid = (first+last)//2
8          if int(lst[mid]) == val:
9              index = mid
10         else:
11             if val<int(lst[mid]):
12                 last = mid -1
13             else:
14                 first = mid +1
15     return index
```

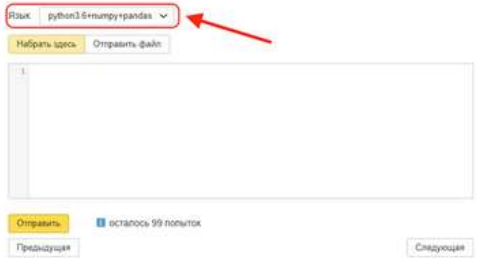
```
def bin_search(lst):
    val=1415
    first = 0
    last = len(lst)-1
    index = -1
    while (first <= last) and (index == -1):
        mid = (first+last)//2
        if int(lst[mid]) == val:
            index = mid
        else:
            if val<int(lst[mid]):
                last = mid -1
            else:
                first = mid +1
    return index
```

4. Менеджмент

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	input.txt
Вывод	output.txt

Обратите внимание, что перед отправкой решения данной задачи нужно в качестве языка выбрать Python.

Это можно сделать в нижней части данной страницы, как показано на рисунке:



Менеджер крупной столичной фирмы Алексей готовит отчет о прибыльности проекта. Для отчета о прибыли фирмы Алексей проанализировал много данных и построил графики дневной прибыли.

По плану, который был составлен до старта проекта, прибыль должна была увеличиваться линейно каждый день. Если за первый день работы проекта прибыль была 5 у.е., за последний — 15 у.е., между ними прошло 10 дней, то прибыль за второй день составит 6 у.е., за третий — 7 у.е. и так далее.

На деле оказалось, что участники проекта очень не любят понедельники, и их продуктивность по понедельникам (то есть каждый 7-ой день) снижается в 3 раза по сравнению с ожидаемой, а в пятницу — наоборот: участники проекта чувствуют душевный подъем и работают в 2 раза активнее. Из-за этого прибыль по этим дням изменяется пропорционально продуктивности.

Пример: ожидаемая прибыль проекта (план)

-пн: 6 у.е.
-вт: 8 у.е.
-ср: 10 у.е.
-чт: 12 у.е.
-пт: 14 у.е.
-сб: 16 у.е.
-вс: 18 у.е.
-пн: 20 у.е.
-вт: ... и т.д.

Фактическая прибыль проекта (с поправкой на продуктивность):

-пн: 2 у.е.
-вт: 8 у.е.
-ср: 10 у.е.
-чт: 12 у.е.
-пт: 28 у.е.
-сб: 16 у.е.
-вс: 18 у.е.
-пн: 6.666 у.е.
-вт: ... и т.д.

Вам дана информация об ожидаемой по плану прибыли в первый день запуска проекта, об ожидаемой по плану прибыли в последний день запуска проекта и количестве дней работы проекта. Считаем, что понедельник — всегда первый день запуска проекта. При помощи np.linspace и присваиваний в срез, найдите фактическую прибыль проекта с поправкой на продуктивность участников за каждый день работы проекта.

Формат ввода

В файле **input.txt** Вам даны три числа: прибыль (по плану) проекта в первый день (в понедельник), прибыль (по плану) проекта в последний день и количество дней.

Формат вывода

Выведите в файл **output.txt** массив, содержащий данные по ежедневной фактической прибыли. Прибыль за каждый день представляет собой **вещественное** число с точностью 2 знака после запятой, каждое число записывается в **отдельной строке**.

Пример

Ввод	Вывод
10 50 5	3.33 20.00 30.00 40.00 100.00

Примечания

1. Обратите внимание, что в выходном файле нужно сохранять числа с точностью **2 знака после точки**.
2. Во входных данных может быть больше одного понедельника и пятницы.
3. Не забывайте делать import numpy!
4. Значения прибыли проекта во входных данных могут быть нецелыми числами (вещественными).
5. Пятница - **не всегда** последний день. И она может быть **не одна**.
6. Проверяйте, что в выходных данных нет лишних пробелов.
7. Обратите внимание, что у функции numpy.linspace(start, end, len) первые 2 аргумента (start и end) - начало и конец списка - **могут быть нецелыми**. А вот третий аргумент - len
   - длина списка, **должен быть целым числом**.
8. Метод round() округляет так, что не оставляет нулей в конце. Например, round(5.101, 2) выдаст 5.1, хотя нужно выводить 5.10. Чтобы побороть эту проблему, можно отсекать лишние разряды в самом методе print() с помощью f-string: f'{x:.2f}'. Подробнее: см. *рекомендации к вводу-выводу*.

```
import numpy as np
data = np.loadtxt('input.txt')
start, end, lenght = data
lenght = int(lenght)
profit = np.linspace(start, end, lenght)
mon, fri = np.ones(lenght), np.ones(lenght)
mon[::7], fri[4::7] = 3, 2
np.savetxt('output.txt', profit / mon * fri, fmt='%.2f')
```

Контест 8

Конспект:
https://colab.research.google.com/drive/1NeMnmR5ZEpf9vZGSyvSmnK_UgEngra5M?usp=sharing

1. Немного статистики

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	input.txt
Вывод	стандартный вывод или output.txt

В тот самый момент, когда стажер-аналитик Павел собирается пойти пообедать, директор по связям с общественностью обращается к нему с вопросом: может ли он предоставить для веб-сайта какие-нибудь примечательные факты об уровне зарплат исследователей данных? Данные имеют конфиденциальный характер, и поэтому Павла снабдили анонимным набором данных, содержащим зарплату каждого сотрудника в долларах в месяц. Но это не проблема, подумал Павел, так как достаточно будет посчитать некоторые статистические показатели по предоставленным ему данным, а именно: медиану, среднее арифметическое, а также среднеквадратическое отклонение зарплат. Помогите Павлу сделать задание и со спокойной душой пойти на перерыв.

Формат ввода

В файле input.txt дана последовательность целых чисел в диапазоне от 1000 до 10000 включительно.

Числа разделены одним символом пробел.

Количество чисел меньше 100000.

Формат вывода

В стандартный поток вывода или в файл output.txt выведите 3 вещественных числа (медиану, среднее арифметическое и среднеквадратическое отклонение) с точностью 2 знака после запятой, разделив их одним пробелом.

Примечания

1. Использование numpy является **обязательным**.
2. Для нахождения статистик по данным воспользуйтесь соответствующими функциями из библиотеки numpy:
   - numpy.median(),
   - numpy.mean(),
   - numpy.std().
3. Чтобы вывести вещественное число с двумя знаками после запятой воспользуйтесь форматированием строк:
   - print("{:.2f}".format(3.14159)) -> 3.14. Подробнее - см. *рекомендации к вводу-выводу*.
4. В задаче **не разрешается** использовать pandas.
5. Для подсчёта суммы отдельно по столбцам или по строкам можно использовать numpy.sum() с указанием параметра axis (см. официальную документацию).

Набрать здесь | Отправить файл

```
import numpy
s = open('input.txt')
c = s.read()
c=c.replace('\n', "")
d = c.split(" ")
e=[eval(i) for i in d]
print("{:.2f}".format(numpy.median(e)), "{:.2f}".format(numpy.mean(e)), "{:.2f}".for
s.close()
```

```
import numpy
s = open('input.txt')
c = s.read()
c=c.replace('\n', "")
d = c.split(" ")
e=[eval(i) for i in d]
print("{:.2f}".format(numpy.median(e)), "{:.2f}".format(numpy.mean(e)), "{:.2f}".format(numpy.std(e)), sep=" ")
s.close()
```

2. Мода

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	input.csv
Вывод	стандартный вывод или output.txt

В один приятный вечер три подруги-модницы встретились, чтобы обсудить последние новости моды. Каждая из них пришла в платье своего цвета: Вилена пришла в зеленом платье, Марианна — в салатовом, а Снежанна — в платье цвета лесной листвы. Они принялись спорить, чей цвет моднее в этом сезоне. Для того, чтобы решить спор, подруги "откопали" 150 выпусков журнала "Мода модных модниц", и стали считать, сколько раз известные звезды появлялись в том или ином цвете. Считав массив, содержащий в себе информацию об этих подсчетах, определите, какой же цвет самый модный.

Формат ввода

В файле input.csv содержится таблица, состоящая из 150 строк и 3 столбцов. Одна строка данной таблицы относится к одному модному журналу. Каждый столбец относится к определенному цвету: 1-ый — зеленый, 2-ой — салатовый, 3-ий — цвет лесной листвы. Значение в некоторой строке i и некотором столбце j — это число, равное количеству появлений в модном журнале i звезд в цвете соответствующем столбцу j.

Формат вывода

В стандартный поток вывода или в файл output.txt необходимо вывести единственное число, равное номеру (не путайте номер с ИНДЕКСОМ) наиболее часто встречающегося цвета:

- 1 — для зеленого цвета,
- 2 — для салатового цвета,
- 3 — для цвета лесной листвы.

Примечания

1. Разделитель в файле input.csv — **запятая без пробелов**.
2. Использование numpy является **обязательным**!
3. Для считывания исходных данных и формирования списка можно использовать стандартную работу с файлами (open(), readlines(), split(), int()) либо метод numpy.genfromtxt().

Набрать здесь | Отправить файл

```
import numpy as np
s =np.genfromtxt('input.csv', delimiter=',')
i=s[:,0].sum()
j=s[:,1].sum()
k=s[:,2].sum()
def maximum(a, b, c):
    if (a >= b) and (a >= c):
        return 1
    elif (b >= a) and (b >= c):
        return 2
    else:
        return 3
print(maximum(i, j, k))
```


```
import numpy as np
s =np.genfromtxt('input.csv', delimiter=',')
i=s[:,0].sum()
j=s[:,1].sum()
k=s[:,2].sum()
def maximum(a, b, c):
    if (a >= b) and (a >= c):
        return 1
    elif (b >= a) and (b >= c):
        return 2
    else:
```

3. Две чугунные гири

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	input.csv
Вывод	стандартный вывод или output.txt

В спортивном клубе "Две чугунные гири" есть два типа клиентов:

- одни из них всегда делают примерно одинаковое количество подходов, планомерно развивая свои физические способности,
- другие же часто ленятся и делают существенно меньше, чем нужно, а затем их мучает совесть, и они начинают делать существенно больше.

Внимательные тренеры ведут скрупулезные и подробные записи о каждом посетителе клуба. Отличить одних от других можно при помощи подсчета стандартного отклонения по статистике количества сделанных подходов за последние дни. Нормальным разбросом вокруг среднего значения подходов для последовательных посетителей считается +/-4 подхода.

По записям за последние 35 дней определите кого в клубе больше — клиентов 1 или 2 типа. Гарантируется, что одних из них строго больше.

Формат ввода

В файле input.csv содержатся записи о подходах, сделанных клиентами за последние 35 дней. Массив содержит N строк и 35 столбцов, где N — это количество спортсменов в клубе.

Формат вывода

В стандартный поток вывода или в файл output.txt запишите единственное число 1 или 2, в зависимости от Вашего решения. Если в клубе больше спортсменов, совершающих примерно одно и то же число подходов, запишите 1, иначе — запишите 2.

Примечания

1. Разделитель в файле input.csv — **запятая без пробелов**.
2. Использование numpy является **обязательным**.
3. Для подсчета разброса вокруг среднего значения подходов воспользуйтесь функцией библиотеки numpy:
   - numpy.std().

```
import numpy as np
s =np.genfromtxt('input.csv', delimiter=',')
s=s.astype(int)
i=0
j=0
f=np.std(s, axis = 1)
f=f.tolist()
for d in range(len(f)):
    if f[int(d)]>4:
        i+=1
    else:
        j+=1
if i>j:
    print('2')
else:
    print('1')
```


```
import numpy as np
s =np.genfromtxt('input.csv', delimiter=',')
s=s.astype(int)
i=0
j=0
f=np.std(s, axis = 1)
f=f.tolist()
for d in range(len(f)):
    if f[int(d)]>4:
        i+=1
    else:
        j+=1
if i>j:
    print('2')
else:
```

4. Очистка логов

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	input.csv
Вывод	output.csv

Программист Иван очень общительный человек, поэтому в рабочее время он любит общаться с друзьями в социальных сетях. Недавно программист Иван узнал, что его начальники логируют время, проведенное сотрудниками в социальных сетях, и в тот же момент понял, что его карьера висит на волоске. Он выяснил у своего друга и коллеги, как можно получить доступ к лог-файлам и в каком формате логируются данные. Оказалось, что данные логируются в виде двумерного массива размерностью 30x12, где каждый элемент массива содержит информацию о том, сколько времени сотрудник провел в социальных сетях в i-й рабочий день j-го месяца, в часах. Ради удобства начальство Ивана считает, что каждый месяц состоит ровно из 30 рабочих дней.

Программист Иван знает, что начальство поймет, что что-то не так, если Иван просто обнулит все свои логи. Иван внимательно с ними ознакомился и узнал, что наибольшее время в социальных сетях он проводил по четным числам нечетных месяцев, и по нечетным числам в четные месяцы. Помогите Ивану занизить логи, уменьшив ровно в 2 раза элементы массива, имеющие один четный и один нечетный индекс.

Вам дан csv-файл с логами, его нужно считать и при помощи операций с массивами (мы рекомендуем Вам использовать присваивание в срез) изменить его так, как сказано в условии задачи. А затем сохранить результат в новый csv-файл.

Формат ввода

В файле input.csv содержится массив размерностью 30 строк на 12 столбцов с информацией о проведенных часах в социальных сетях.

Формат вывода

Сохраните модифицированный файл в output.csv.

Пример

Ввод

```
6.8,7,3.7,7.7,3.1,6.8,6.3,4.7,5.4,
6.2,0.1,7.9,4.4,4.2,7,6.8,0.9,2.2,
0.5,6.8,0.8,0.8,4.4,3.9,4.1,3.8,6.
7.1,0.5,1.7,2.2,7.9,5.3,5.2,3.9,4.
3.1,4.1,7.5,1.5,3.9,3.5,3.8,4.5,3.
3.2,0.7,3.8,6.1,4.1,5.5,3.5,5.1,1.
2.4,0.9,6.2,5.6,2.6,5.9,1.7,7.2,5.
5.4,5,7.2,2.6,3.3,3.7,0.8,7.6,5.9,
2,3.7,2.3,6.1,4.5,2.5,0.9,1.6,5.6,
0.6,6.4,4.7,7.5,7.3,4.9,4.7,4.4,0.
2.5,4.4,0.4,5.5,1,0.3,2.9,5.8,7.9,
1.5,1.9,1.2,7.3,0.2,0.7,5.5,2.8,3.
2.1,3.8,2.1,2.9,1.1,5.5,4.1,7.8,7.
6.6,2.6,3.4,5.2,2.7,1.2,1,7.5,7.8,
6.1,4,2.6,6.6,3.5,7.4,6.1,1.6,5.3,
0.7,4.6,1.6,5,4.5,0.5,2.1,4.6,1.5,
2.4,4.3,7.3,3.9,2.5,0.4,7.8,1.6,3.
1.5,2.9,4,5,0.3,7.6,3,2.5,1.5,3.7,
7.3,8,2.6,1,6.7,2.8,1.4,2.4,7.5,0.
4.9,6.1,1.2,6.1,4.6,3.3,3.1,6.5,6.
2.1,2,2.5,4.9,2.5,4.1,7.7,2.1,1.4,
2,1.3,3.7,1.5,6.7,1.6,6.3,6.7,7.9,
6.7,2.7,5.9,5.7,5.4,0,1.4,6.1,7.5,
6,7.8,7.1,3.3,6.2,3.2,0.8,3.1,2.5,
4.2,1.8,1.5,2.8,8,1,6.4,4.3,6.6,7.
6.4,4.5,7.3,7.4,4.8,7.2,1.8,6.8,4.
1.3,4.4,1.6,6.9,3.1,7,1.7,4.6,0.3,
7.7,4,0.8,4.2,4.1,2.4,0.8,2.6,3,6.
2.8,3.8,0.6,6.4,1.1,7,1.4,7.3,0.1,
1.9,3.4,2.5,7.4,6.2,0.3,3.1,4.2,5.
```

Вывод

```
6.8,3.5,3.7,3.85,3.1,3.4,6.3,2.35,
3.1,0.1,3.95,4.4,2.1,7,3.4,0.9,1.1
0.5,3.4,0.8,0.4,4.4,1.95,4.1,1.9,6
3.55,0.5,0.85,2.2,3.95,5.3,2.6,3.9
3.1,2.05,7.5,0.75,3.9,1.75,3.8,2.2
1.6,0.7,1.9,6.1,2.05,5.5,1.75,5.1,
2.4,0.45,6.2,2.8,2.6,2.95,1.7,3.6,
2.7,5,3.6,2.6,1.65,3.7,0.4,7.6,2.9
2,1.85,2.3,3.05,4.5,1.25,0.9,0.8,5
0.3,6.4,2.35,7.5,3.65,4.9,2.35,4.4
2.5,2.2,0.4,2.75,1,0.15,2.9,2.9,7.
0.75,1.9,0.6,7.3,0.1,0.7,2.75,2.8,
2.1,1.9,2.1,1.45,1.1,2.75,4.1,3.9,
3.3,2.6,1.7,5.2,1.35,1.2,0.5,7.5,3
6.1,2,2.6,3.3,3.5,3.7,6.1,0.8,5.3,
0.35,4.6,0.8,5,2.25,0.5,1.05,4.6,0
2.4,2.15,7.3,1.95,2.5,0.2,7.8,0.8,
0.75,2.9,2,5,0.15,7.6,1.5,2.5,0.75
7.3,4,2.6,0.5,6.7,1.4,1.4,1.2,7.5,
2.45,6.1,0.6,6.1,2.3,3.3,1.55,6.5,
2.1,1,2.5,2.45,2.5,2.05,7.7,1.05,1
1,1.3,1.85,1.5,3.35,1.6,3.15,6.7,3
6.7,1.35,5.9,2.85,5.4,0,1.4,3.05,7
3,7.8,3.55,3.3,3.1,3.2,0.4,3.1,1.2
4.2,0.9,1.5,1.4,8,0.5,6.4,2.15,6.6
3.2,4.5,3.65,7.4,2.4,7.2,0.9,6.8,2
1.3,2.2,1.6,3.45,3.1,3.5,1.7,2.3,0
3.85,4,0.4,4.2,2.05,2.4,0.4,2.6,1.
2.8,1.9,0.6,3.2,1.1,3.5,1.4,3.65,0
0.95,3.4,1.25,7.4,3.1,0.3,1.55,4.2
```

Примечания

1. Использование numpy является **обязательным!**
2. Данные в файле input.csv представлены в стандартном виде .csv:
   - строки, разделённые '\n',
   - разделитель в каждой строке – **запятая без пробелов**,
   - в качестве разделителя десятичной и дробной части вещественного числа используется **точка**.
3. Для вывода количества часов, проведенных в социальных сетях, воспользуйтесь форматированием: "%g".
4. Для вывода результата можно использовать numpy.savetxt().

Набрать здесь | Отправить файл

```
import numpy as np
s = np.genfromtxt('input.csv', delimiter=',')
s[::2,1::2]=s[::2,1::2]/2
s[1::2,::2]=s[1::2,::2]/2
np.savetxt('output.csv', s,fmt='%g', delimiter=',')
```


```
import numpy as np
s = np.genfromtxt('input.csv', delimiter=',')
s[::2,1::2]=s[::2,1::2]/2
s[1::2,::2]=s[1::2,::2]/2
np.savetxt('output.csv', s,fmt='%g',
delimiter=',')
```

5. Дуэльный клуб

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	input.csv
Вывод	output.csv

В выручай-комнате Гарри Поттер и его соратники тренировались сражаться на волшебных палочках, устраивая дуэли. Результаты дуэлей тщательно записывались в волшебный список. К концу месяца друзья составляли списки лучших волшебников. Однажды оказалось, что Гарри не стал лучшим волшебником, и руководство дуэльного клуба в лице Гарри и его ближайших друзей, решило, что это плохо для имиджа клуба, поэтому список нужно немного скорректировать следующим образом. В каждом месяце Гарри должен выигрывать в полтора раза больше матчей, чем ребята выигрывали в среднем (при подсчете среднего количества матчей результаты Гарри учитываются). Этот список волшебный и изменить его можно только с помощью змеиного языка Python, которым Гарри, как известно, отлично владеет. Напишите для Гарри заклинание, которое он должен произнести, чтобы имидж клуба не пострадал.

Вам дан csv-файл с количеством побед каждого из участников клуба (их 15) за каждый из 30 дней месяца. Присваиванием в срез замените значение в первой строке (конечно, эта строка принадлежит Гарри) на среднее значение побед по этому дню, умноженное на 1.5.

Формат ввода

В файле input.csv записан двумерный массив обозначенного в условии формата:

- количество строк в файле - 15
- количество чисел в каждой строке файла - 30
- файл содержит только целые числа

Формат вывода

Сохранить модифицированный массив в файл output.csv.

Пример

Ввод

```
22,28,33,17,21,33,27,34,27,31,16,1
21,22,32,22,34,31,25,28,29,26,20,2
30,28,20,32,21,23,29,28,24,24,19,2
26,20,27,28,17,26,21,25,23,26,19,2
32,34,30,25,30,28,21,25,22,28,15,1
34,17,15,31,23,21,34,34,33,20,19,2
33,32,29,22,33,17,30,29,30,22,27,2
17,21,22,28,15,28,31,19,22,24,20,1
30,29,28,34,25,18,15,31,31,18,23,2
25,33,33,15,17,25,21,31,32,32,29,2
25,19,17,28,27,29,25,30,30,28,16,1
16,25,17,22,24,19,34,18,25,24,32,3
32,26,31,21,16,16,25,19,18,28,18,3
29,32,26,32,22,18,31,27,18,32,25,1
28,28,15,22,32,25,15,17,26,28,30,2
```

Вывод

```
40,39,37,37,35,35,38,39,39,39,32,3
21,22,32,22,34,31,25,28,29,26,20,2
30,28,20,32,21,23,29,28,24,24,19,2
26,20,27,28,17,26,21,25,23,26,19,2
32,34,30,25,30,28,21,25,22,28,15,1
34,17,15,31,23,21,34,34,33,20,19,2
33,32,29,22,33,17,30,29,30,22,27,2
17,21,22,28,15,28,31,19,22,24,20,1
30,29,28,34,25,18,15,31,31,18,23,2
25,33,33,15,17,25,21,31,32,32,29,2
25,19,17,28,27,29,25,30,30,28,16,1
16,25,17,22,24,19,34,18,25,24,32,3
32,26,31,21,16,16,25,19,18,28,18,3
29,32,26,32,22,18,31,27,18,32,25,1
28,28,15,22,32,25,15,17,26,28,30,2
```

Примечания

1. Использование numpy является **обязательным!**
2. Данные в файле input.csv представлены в стандартном виде .csv:
   - строки, разделённые '\n',
   - разделитель в каждой строке - **запятая без пробелов**.
3. Обратите внимание, что значения в первой строке нужно заменять на среднее значение **по дню**.
4. При вычислении среднего значения по дню **нужно** включать данные Гарри.
5. После умножения среднего значения побед на 1.5 нужно отбрасывать дробную часть.
   - Пример: 35.7 -> 35.
6. В результирующий файл нужно записывать **всю** таблицу (а не только первую строку с изменёнными данными).

```
import numpy as np
data = np.genfromtxt('input.csv', delimiter=',')
avg_wins = np.mean(data, axis=0)
data[0, :] = np.floor(avg_wins * 1.5)
np.savetxt('output.csv', data, fmt="%g",
delimiter=',')
```

Контест 9

Конспект:
https://colab.research.google.com/drive/1GXHEAgyj0sVyFw3owTeYgY7eI-bLaPNU?usp=sharing

1. Железная логика

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	
Вывод	стандартный вывод или output.txt

Какие различные наборы логических переменных $x1$, $x2$, $x3$ приводят к ИСТИННОМУ значению приведенное ниже логическое высказывание?

$$(\neg x1 \wedge x2 \wedge \neg x3) \vee (\neg x1 \wedge x2 \wedge x3) \vee (x1 \wedge \neg x2 \wedge \neg x3)$$

Формат ввода

Обратите внимание, что на вход программе ничего не подается.

Формат вывода

На стандатный поток вывода или в файл output.txt выведите все наборы. Каждый набор выводится на отдельной строке. Каждый набор состоит из последовательности значений True или False, разделенных одним пробелом между собой.

Для пояснения, представим, что во всех наборах False заменили на 0, True на 1. Тогда каждый набор можно рассматривать как бинарный вектор: $(x1, x2, x3)$, где $xi \in \{0, 1\}$. Все наборы можно отсортировать по возрастанию, если рассматривать их как числа в двоичной системе:

- $(0, 0, 1)$
- $(0, 1, 0)$
- $(0, 1, 1)$

...

- $(1, 1, 0)$
- $(1, 1, 1)$

Если теперь сделать обратную замену 0 и 1 на False и True соответственно, то получилм упорядоченные наборы.

Примечания

1. Обратите внимание, что в данной задаче **не нужно** ничего передавать в качестве входных данных.
2. Формат вывода:
   - набор значений вида "True" или "False", разделённых **пробелом**,
   - разделитель строк: символ '\n'.
3. Выводимые значения нужно сортировать **в порядке возрастания соответствующих двоичных чисел** (True==1, False==0 => True>False).
4. Примечание о значениях операторов:
   - ¬ — логическое отрицание (not),
   - ∧ — логическая конъюнкция (and),
   - ∨ — логическая дизъюнкция (or).
5. Перебор значений $x1$, $x2$, $x3$ может быть осуществлён как при помощи вложенных циклов, так и с помощью itertools.
6. При поиске ответа на задачу может быть полезна информация о приоритетах логических операций, определяющих порядок их выполнения:
   - Сначала выполняются операции в скобках.
   - В каждой скобке сначала выполняются все унарные not.
   - После вычисляются все AND.
   - Затем выполняется OR.
   - Если есть несколько операций с одним приоритетом, то они выполняются **слева направо**.
7. Можно использовать библиотеку itertools. При этом **не забывайте** делать соответствующий import.

Набрать здесь | Отправить файл

```
for i in range(8):
    x1, x2, x3 = bool(i & 4), bool(i & 2), bool(i & 1)
    if (not x1 and x2 and not x3) or (not x1 and x2 and x3) or (x1 and not x2 and no
        print(x1, x2, x3)
```

```
for i in range(8):
        x1, x2, x3 = bool(i & 4), bool(i & 2), bool(i & 1)
        if (not x1 and x2 and not x3) or (not x1 and x2 and x3) or (x1 and not x2 and
not x3):
                print(x1, x2, x3)
```

2. Ошибка выжившего

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	input.csv
Вывод	стандартный вывод или output.txt

В математической статистике есть много интересных парадоксов и загадок. Одним из самых известных и интересных примеров является так называемыя "ошибка выжившего". Этот парадокс имеет много интерпретаций: например, есть известная история о том, как во время Второй мировой войны американские инженеры принимали решение о том, каким образом нужно укрепить части самолетов. У них были ресурсы для укрепления лишь одной части: двигателя, лобового стекла, крыльев или хвоста самолета. Большинство самолетов возвращались с повреждениями в районе хвоста и крыльев — логичным решение было укреплять одну из этих частей. Но все не так просто: в воздушном бою снаряды попадают по самолетам равномерно, поэтому факт того, что большинство самолетов возвращается с повреждениями в этих областях, говорит лишь о некритичности попаданий в эти области, поскольку самолетам все же хватает ресурсов для возвращения на базу. Тогда укреплять нужно область с минимальным количеством попаданий.

Вам предлагается отследить "ошибку выжившего" в следующем формате.

Вам дан csv-файл, в котором содержится информация об N вернувшихся на базу самолетах. Мы условно разделили самолет на 10 областей, в двумерном массиве размерностью N x 10 приведено количество попаданий в каждую область из этих областей у вернувшегося самолета. Посчитайте среднее количество попаданий для каждой области по всем самолетам и примите решение о том, какую из 10 областей Вы бы стали укреплять.

Формат ввода

Файл input.csv, содержащий информацию о попаданиях в разные области самолета в виде двумерного массива: N строк — записи об N самолетах, 10 столбцов — количество попаданий в каждую из 10 областей.

Формат вывода

В стандартный поток вывода или в файл output.txt выведите число от 1 до 10, представляющее собой Ваш выбор. Обратите внимание, что области нумеруются не с 0, а с 1.

Примечания

1. Разделитель в файле input.csv — запятая.
2. Файл input.csv не содержит заголовка.
3. Использование numpy является **обязательным**.
4. Не забывайте делать import pandas as pd.
5. Обратите внимание, что по условию нужно брать **не** столбец с максимальным средним, а столбец с **минимальным** средним значением.

Набрать здесь | Отправить файл

```
import numpy as np
import pandas as pd
s =np.genfromtxt('input.csv', delimiter=',')
s=s.astype(int)
i=0
j=0
f=np.average(s, axis = 0)
f=f.tolist()
# function to find minimum and maximum position in list
def minimum(a, n):
    minpos = a.index(min(a))
    print(minpos + 1)
minimum(f, len(f))
```


```
import numpy as np
import pandas as pd
s =np.genfromtxt('input.csv', delimiter=',')
s=s.astype(int)
i=0
j=0
f=np.average(s, axis = 0)
f=f.tolist()
# function to find minimum and maximum position in list
def minimum(a, n):
    minpos = a.index(min(a))
```

3. Электрик Станислав и интернет-казино

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	input.csv
Вывод	стандартный вывод или output.txt

Электрик Станислав любил смотреть фильмы и сериалы, пользуясь различными бесплатными сервисами, вследствие чего невольно стал частью аудитории реклам интернет-казино. Вдохновившись последними, Станислав твердо решил, что в жизни нужно что-то менять, и отправился искать удачу в интернет-казино "3 кабачка". Казино предложило Станиславу приветственный бонус: Станислав мог получить 1000 рублей или же статистику по выигрышам на каждом из игровых автоматов за последние 100 игр. Учитывая, что игровых автоматов в казино "3 кабачка" очень много, а Станислав не привык принимать необдуманные решения, он согласился на второе условие и получил таблицу с данными в файле input.csv. Станислав не учел лишь одного: в программе подготовки электриков никогда не была курса по анализу данных при помощи Python, так что Станислав теперь понятия не имеет, что ему делать со столь ценной информацией.

Помогите Станиславу по данным из файла принять решение, какой из автоматов наиболее прибыльный.

Формат ввода

В input.csv хранятся данные о каждом автомате, а именно:

$$name_1, name_2, \dots, name_n$$

$$win_1^1, win_2^1, \dots, win_n^1$$

$$win_1^2, win_2^2, \dots, win_n^2$$

$$\vdots$$

$$win_1^{100}, win_2^{100}, \dots, win_n^{100}$$

где $name_i$ — название автомата, win_i^k — выигрыш на автомате $name_i$ за k-ю игру.

Обратите внимание, что в примере для удобства приведена статистика только за последние 3 игры.

Формат вывода

Название автомата $name_i$, котороя в среднем дает наибольший выигрыш за последние 100 игр.

Пример

Ввод	Вывод
Вишенка,Бабочка,Кабачок 97.84034494908083,-173.34106120704 30.346779261014778,92.842664370813 -20.927535597697414,-225.060017034	Вишенка

Примечания

1. Воспользуйтесь подходящими функциями из библиотеки pandas.
2. Для считывания входных данных можно использовать либо старый подход (с помощью open(), readlines(), split() и т.д.). **Но лучше считывать с помощью pandas.read_csv()**.
3. При работе с pandas.DataFrame() в текущей версии pandas может не работать конструкция вида: df.columns[df.mean().argmax()] (df=pandas.DataFrame()). Используйте *одну из* следующих конструкций:
   - df.mean().values.argmax(),
   - .idxmax().

Набрать здесь | Отправить файл

```
import numpy as np
import pandas as pd
s1 = open('input.csv')
c = s1.readline()
c=c.replace('\n', "")
c=c.split(',')
s1.close()
s =np.genfromtxt('input.csv', delimiter=',')
s = np.delete(s, (0), axis=0)
s=s.astype(float)
f=np.sum(s, axis = 0)
f=f.tolist()
def maximum(a, n):
    maxpos = a.index(max(a))
    print(c[maxpos])
maximum(f, len(f))
```

```
import numpy as np
import pandas as pd
s1 = open('input.csv')
c = s1.readline()
c=c.replace('\n', "")
c=c.split(',')
s1.close()
s =np.genfromtxt('input.csv', delimiter=',')
s = np.delete(s, (0), axis=0)
s=s.astype(float)
f=np.sum(s, axis = 0)
f=f.tolist()
def maximum(a, n):
    maxpos = a.index(max(a))
    print(c[maxpos])
maximum(f, len(f))
```

4. Инвестор Генадий и новый Мерседес

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	input.csv
Вывод	стандартный вывод или output.txt

Инвестор Генадий копит на новый Мерседес. Для его покупки Генадию необходимо накопить 8 000 000 рублей. Генадий смотрит на эту задачу со здоровым оптимизмом: у Генадия собран портфель из большого числа ценных бумаг. С прошлого сбора дивидендов прошло 20 рассчетных периодов и каждая из этих ценных бумаг принесет Генадию круглую сумму. В файле input.csv указана прибыльность каждой ценной бумаги за каждый рассчетный период.

Вам предлагается оценить, хватит ли Генадию на новый Мерседес, если он соберет все дивиденды за все периоды, указанные в таблице. Найдите среди бумаг Генадия наиболее прибыльную.

Формат ввода

В input.csv хранятся данные о дивидендах за расчетные периоды для каждой бумаги:

$$name_1, name_2, \dots, name_n$$

$$div_1^1, div_2^1, \dots, div_n^1$$

$$div_1^2, div_2^2, \dots, div_n^2$$

$$\vdots$$

$$div_1^{20}, div_2^{20}, \dots, div_n^{20}$$

где $name_i$ – название ценной бумаги, div_i^k – дивиденды за бумагу $name_i$ за k-й расчетный период.

Обратите внимание, что в примере для удобства приведена статистика только за последние 3 расчетных периода.

Формат вывода

Выведите 1, если Генадию хватит на новый Мерседес, если он соберет все дивиденды за все периоды, указанные в таблице. В обратном случае выведите 0.

Также через пробел выведите $name_i$ – название наиболее прибыльной (суммарно) бумаги.

Пример

Ввод	Вывод
Газбанк,Сберпром,Мыло.ру 1095238,1656163,2746738 1103245,1644220,2731470 1116010,1647492,2755662	1 Мыло.ру

Примечания

1. Воспользуйтесь подходящими функциями из библиотеки pandas.
2. Нужно получить 8 миллионов **со всех имеющихся бумаг** *(а не только с самой прибыльной)*.
3. Для считывания данных рекомендуется использовать pandas.read_csv().
4. При использовании numpy.genfromtxt() для считывания входных данных может возникнуть ошибка, связанная с удалением точки в названии колонки 'Мыло.ру'. Это связано с особенностями реализации данной функции *(см. официальную документацию)*. Лучше использовать pandas.read_csv().
5. При работе с pandas.DataFrame() в текущей версии pandas может не работать конструкция вида: df.columns[df.mean().argmax()] (df=pandas.DataFrame()). Используйте *одну из* следующих конструкций:
   - df.mean().values.argmax(),
   - .idxmax().
6. Количество акций **не обязано** равняться трём, оно может быть **любым**.

Набрать здесь | Отправить файл

```
import pandas as pd
df = pd.read_csv('input.csv')
print(int(df.sum().sum() >= 8e6), df.sum().idxmax())
```

```
import pandas as pd
df = pd.read_csv('input.csv')
print(int(df.sum().sum() >= 8e6),
df.sum().idxmax())
```

5. Фильтрация данных

Имеется таблица с данными, которая считывается с помощью инструментов библиотеки Pandas.

```
import pandas as pd
data = pd.read_csv("demodata.csv")
```

Вы хотите отфильтровать данные по дням недели и цене так, чтобы в столбце weekdays были только значения Thu, а в столбце prices значения больше, чем 2 млн.

Каким образом вы можете это сделать?

Примечания

1. Обратите внимание на расстановку круглых скобок и приоритеты логических операций.

- [x] `data[(data.weekdays == 'Thu') & (data.prices > 2000000)]`
- [ ] `data[data.weekdays == 'Thu' & data.prices > 2000000]`
- [ ] `data[(weekdays == 'Thu') & (prices > 2000000)]`
- [x] `data.query('weekdays == "Thu" and prices > 2000000')`

6. Неравенство треугольника

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	input.csv
Вывод	стандартный вывод или output.txt

В таблице содержатся длины треугольников – по три **натуральных** (целых положительных) числа в каждой строке. Выясните, какое количество троек чисел может являться сторонами треугольника, то есть удовлетворяет неравенству треугольника: **длина любой стороны треугольника всегда меньше суммы длин двух его других сторон**.

Формат ввода

Данные содержатся в файле input.csv. Столбцы называются "a", "b", "c".

Пример данных:

```
a,b,c
10,20,50
10,41,50
100,40,60
100,40,61
20,50,20
20,50,31
60,60,60
```

Здесь в первой, третьей и пятой строках неравенство треугольника нарушено, а в остальных четырёх – соблюдено.

Формат вывода

Одно целое число – количество треугольников во входных данных.

Для примера выше это число:

4

Пример

Ввод	Вывод
a,b,c 10,20,50 10,41,50 100,40,60 100,40,61 20,50,20 20,50,31 60,60,60	4

Примечания

1. Данные в файле input.csv представлены в стандартном виде .csv:
   - строки, разделённые '\n',
   - разделитель в каждой строке – **запятая без пробелов**,
   - названия столбцов записываются **в первой строке**,
   - названия столбцов: "a", "b", "c".
2. Обратите внимание, что неравенство треугольника должно выполняться **для всех сторон** исходного треугольника.

Набрать здесь | Отправить файл

```
with open('input.csv', mode='r') as f:
    txt=f.readlines()[1:]
s=0
for line in txt:
    a,b,c=map(int, line.split(','))
    a,b,c = sorted([a,b,c])
    s+=a+b>c
print(s)
```

```
with open('input.csv', mode='r') as f:
        txt=f.readlines()[1:]
s=0
for line in txt:
        a,b,c=map(int,
line.split(','))
        a,b,c = sorted([a,b,c])
        s+=a+b>c
print(s)
```

Контекст 10

Конспект:
https://colab.research.google.com/drive/1acvLycZQrtB4Ks-_7gAIEx6lksXZi8cT?usp=sharing

1. Предобработка данных

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	input.csv
Вывод	output.csv

В исходной таблице два столбца с названиями name и score.

Напишите программный код, который преобразует исходную таблицу следующим образом:

1. Необходимо удалить те строки, у которых отсутствуют данные в первом столбце (name).
2. Отсутствующие значения во втором столбце (score) необходимо заменить средним по данному столбцу.

Результат необходимо сохранить в файл output.csv

Формат ввода

На вход подается файл input.csv, в первой строке названия столбцов: name и score, данные в строках разделены с помощью запятых. Данные в колонке score – числовые.

Формат вывода

Результат необходимо сохранить в файл output.csv, разделитель – запятая. Обратите внимания на формат вывода в примере.

Пример

Ввод

```
name,score
name1,87
name2,100
name3,98
name4,
,
,78
name5,67
```

Вывод

```
,name,score
0,name1,87.0
1,name2,100.0
2,name3,98.0
3,name4,88.0
6,name5,67.0
```

Примечания

1. Используйте методы read_csv(), dropna(), fillna(), to_csv().
2. Для поиска среднего значения используйте метод mean, после применения метода округлять результат **не требуется**.
3. Нужно считать среднее по датафрейму **после** удаления лишних строк.
4. При использовании функции dropna() в качестве параметра subset subset передаётся некоторый список. Список названий колонок, которые нужно анализировать. Если в качестве этого параметра передать строку 'name' (dropna(subset='name')), сама по себе эта строка будет восприниматься как список (список отдельных символов). И pandas пытается удалить пропуски в колонках 'n', 'a', 'm' и 'e'. Но таких колонок нет, поэтому появляется ошибка. Поэтому в случае указания одной колонки, следует передавать её название в виде списка из одного элемента (dropna(subset=['name'])).

Набрать здесь | Отправить файл

```
import numpy as np
import pandas as pd
df = pd.read_csv('input.csv')
df.dropna(subset=['name'], inplace=True)
df.fillna(value={'score': df['score'].mean()}, inplace=True)
df.to_csv('output.csv')
```


import numpy as np
import pandas as pd
df = pd.read_csv('input.csv')
df.dropna(subset=['name'], inplace=True)
df.fillna(value={'score': df['score'].mean()}, inplace=True)
df.to_csv('output.csv')

2. Температура

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	input.csv
Вывод	output.csv

В файле input.csv хранится информация о температуре на определеннную дату. Первый столбец называется date, в данном столбце хранится дата. Второй столбец называется temperature_c, здесь хранится информация о температуре по шкале Цельсия. Необходимо добавить новый столбец с названием temperature_f, в котором будет храниться информация о температуре по Фаренгейту в соответствующий день. Результат округлите до целого с помощью round().

Формат ввода

На вход подается файл input.csv, в первой строке названия столбцов: date и temperature_c, данные в строках разделены с помощью запятых. Данные в колонке temperature_c - числовые.

Формат вывода

Результат необходимо сохранить в файл output.csv, разделитель - запятая. Обратите внимания на формат вывода в примере.

Пример

Ввод	Вывод
date,temperature_c	,date,temperature_c,temperature_f
07-12-2021,-9	0,07-12-2021,-9,16
08-12-2021,-11	1,08-12-2021,-11,12
09-12-2021,-13	2,09-12-2021,-13,9

Примечания

1. Формула перевода температуры по шкале Цельсия в температуру по шкале Фаренгейта:

$$t_f = \frac{9}{5} * t_c + 32$$

2. Используйте метод apply.
3. Не забывайте, что в качестве параметра метода apply необходимо передавать либо lambda-функцию, либо имя функции, **возвращающей некоторое значение**.
4. Для генерации новой колонки можно использовать конструкцию: df['new_col_name'] = lst, где
   - df - объект типа pandas.DataFrame,
   - 'new_col_name' - имя колонки, которую нужно добавить,
   - lst - значения, которые должны быть добавлены в новую колонку (список).

Набрать здесь | Отправить файл

```
import numpy as np
import pandas as pd
df = pd.read_csv('input.csv')
temperature_f=df['temperature_c'].tolist()
for d in  range(len(temperature_f)):
    temperature_f[d]=round(float(temperature_f[d])*9/5+32)
df['temperature_f'] = temperature_f
df.to_csv('output.csv')
```


```
import numpy as np
import pandas as pd
df = pd.read_csv('input.csv')
temperature_f=df['temperature_c'].tolist()
for d in  range(len(temperature_f)):
    temperature_f[d]=round(float(temperature_f[d])*9/5+32)
df['temperature_f'] = temperature_f
```

3. Метод loc

Определите результат применения метода loc[2:3] к следующему датафрейму:

	login	score4	score5	score6
1	login3	2	5	5
2	login1	0	3	3
3	login4	5	5	5
4	login5	4	4	4

- ○ login4, login5
- ○ login1, login4, login5
- ● login1, login4

4. Водители склада

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	input.csv
Вывод	output.csv

В файле input.csv дана таблица с информацией о работе водителей склада №7:

ID	Фамилия водителя	Объем груза	Тип операции
1	Уточкин	2726	Вывоз
2	Петров	3581	Привоз
3	Сидоров	3724	Привоз
4	Петров	2683	Привоз
5	Иванов	1637	Вывоз
6	Уточкин	3836	Вывоз
7	Ёжкин	3855	Привоз
8	Иванов	3220	Привоз
9	Сидоров	2584	Вывоз

Необходимо вычислить по каждому водителю сколько килограмм суммарно груза он привёз за вычетом суммарного веса вывезенного груза. Итоговую таблицу отсортировать по убыванию этой суммы, а при равенстве её для двух водителей — по возрастанию фамилий водителей (в лексикографическом порядке). Сохранить в формате CSV в файл output.csv.

Пример

Ввод

```
ID,Фамилия водителя,Объем груза,Тип
1,Уточкин,2726,Вывоз
2,Петров,3581,Привоз
3,Сидоров,3724,Привоз
4,Петров,2683,Привоз
5,Иванов,1637,Вывоз
6,Уточкин,3836,Вывоз
7,Ёжкин,3855,Привоз
8,Иванов,3220,Привоз
9,Сидоров,2584,Вывоз
```

Вывод

```
Фамилия водителя,Объем груза
Петров,6264
Ёжкин,3855
Иванов,1583
Сидоров,1140
Уточкин,-6562
```

Примечания

1. Входные данные хранятся в файле input.csv в формате CSV (разделитель - запятая **без пробелов**, 1 строка содержит названия столбцов) (подробнее - см. Рекомендации по работе с вводом-выводом).
2. Входные данные можно считать с помощью pandas.read_csv().
3. Выходные данные нужно сохранять в файл output.csv также в формате CSV (это можно сделать с помощью DataFrame.to_csv()).
4. Обратите внимание, что в файле input.csv может вообще не быть никакой информации (только названия колонок). В этом случае в output.csv также нужно сохранить только названия колонок (DataFrame.to_csv() также может корректно это выполнить без указания дополнительных параметров).
5. В результирующей таблице названия столбцов следует записать на русском языке, поэтому в DataFrame.to_csv() следует дополнительно указать параметр: encoding="utf8".

Набрать здесь Отправить файл

```python
import pandas as pd

df= pd.read_csv('input.csv', encoding='utf-8')
df= df.drop(columns='ID')
df['Тип операции']= df['Тип операции'].apply(lambda x: 1 if x == 'Привоз' else -1)
df['Объем груза']= df['Объем груза'] * df['Тип операции']
df= df.drop(columns='Тип операции')
df= df.groupby ('Фамилия водителя').sum()
df.sort_values(by=['Объем груза', 'Фамилия водителя'], ascending= [False, True], inp
df.to_csv('output.csv')
```

ЭТО САМАЯ СЛОЖНАЯ ЗАДАЧА КАПЕЦ

```
import pandas as pd

df= pd.read_csv('input.csv', encoding='utf-8')
df= df.drop(columns='ID')
df['Тип операции']= df['Тип операции'].apply(lambda x: 1 if x == 'Привоз' else -1)
df['Объем груза']= df['Объем груза'] * df['Тип операции']
df= df.drop(columns='Тип операции')
df= df.groupby ('Фамилия водителя').sum()
df.sort_values(by=['Объем груза', 'Фамилия водителя'], ascending= [False, True], inplace=True)
df.to_csv('output.csv')
```

5. Лёгкий подъём

Ограничение времени	1 секунда
Ограничение памяти	64.0 Мб
Ввод	input.csv
Вывод	output.csv

В файле input.csv дана таблица с информацией о рейсах авиакомпании "Лёгкий подъём":

ID	Город отправления	Город прибытия	Номер борта
7	Геленджик	Омск	128
8	Москва	Красноярск	191
9	Иркутск	Рязань	170
10	Саратов	Грозный	104
11	Екатеринбург	Хабаровск	197
12	Рязань	Хабаровск	180
13	Новосибирск	Омск	128
14	Санкт-Петербург	Краснодар	182
15	Санкт-Петербург	Грозный	117
16	Москва	Новосибирск	153
17	Москва	Краснодар	138
18	Краснодар	Саратов	139
19	Санкт-Петербург	Пермь	189
20	Пермь	Уфа	128

При помощи группировки и метода nunique() необходимо для каждого номера борта найти количество различных маршрутов, которые он осуществляет.

Итоговую таблицу отсортировать по убыванию количества различных уникальных маршрутов, а при равенстве числа для двух бортов — по возрастанию номеров бортов. Сохранить в формате CSV в файл output.csv.

Пример

Ввод

```
ID,Город отправления,Город прибыти
7,Геленджик,Омск,128
8,Москва,Красноярск,191
9,Иркутск,Рязань,170
10,Саратов,Грозный,104
11,Екатеринбург,Хабаровск,197
12,Рязань,Хабаровск,189
13,Новосибирск,Омск,128
14,Санкт-Петербург,Краснодар,182
15,Санкт-Петербург,Грозный,117
16,Москва,Новосибирск,153
17,Москва,Краснодар,138
18,Краснодар,Саратов,139
19,Санкт-Петербург,Пермь,189
20,Пермь,Уфа,128
```

Вывод

```
Номер борта,Уникальных маршрутов
128,3
189,2
104,1
117,1
138,1
139,1
153,1
170,1
182,1
191,1
197,1
```

Примечания

1. Входные данные хранятся в файле input.csv в формате CSV (разделитель - запятая **без пробелов**, 1 строка содержит названия столбцов) (подробнее - см. Рекомендации по работе с вводом-выводом).
2. Входные данные можно считать с помощью pandas.read_csv().
3. Выходные данные нужно сохранять в файл output.csv также в формате CSV (это можно сделать с помощью DataFrame.to_csv()).
4. Обратите внимание, что в файле input.csv может вообще не быть никакой информации (только названия колонок). В этом случае в output.csv также нужно сохранить только названия колонок (DataFrame.to_csv() также может корректно это выполнить без указания дополнительных параметров).
5. В результирующей таблице названия столбцов следует записать на русском языке, поэтому в DataFrame.to_csv() следует дополнительно указать параметр: encoding="utf8".
6. В разных версиях библиотеки Pandas может возникать неожиданное поведение метода nunique(), из-за чего верные решения не будут приняты системой Яндекс. Контест. Поэтому, если Ваше решение получает статус WA, рекомендуем использовать следующий код для сохранения итоговой таблицы (summary — это объект, получаемый после применения nunique(), сортировку применять к summary **не рекомендуется**):

```
res = pd.DataFrame()
res['Номер борта'] = list(summary.index)
# Если контест не примет решение со строчкой ниже - можно попробовать заменить её на:
# res["Уникальных маршрутов"] = summary.values[:, 0]
res["Уникальных маршрутов"] = summary.values[:, 1]
# Здесь нужно вставить строчку с сортировкой res, указанной в условии
res.to_csv("output.csv", encoding="utf8", index=False)
```

```
import numpy as np
import pandas as pd
df = pd.read_csv('input.csv')
df['Уникальных маршрутов'] = df['Город отправления'] + df['Город прибытия']
a=df.groupby('Номер борта')['Уникальных маршрутов'].nunique ()
a=a.to_frame()
b=a.sort_values(by=['Уникальных маршрутов','Номер борта'], ascending=[False,
True])
b.to_csv('output.csv', encoding="utf8")
```

6. Метод merge

В первой таблице хранится информация о сдаче экзамена №1 в следующем виде:

id (идентификатор студента в базе данных), name (имя), score (оценка).

Во второй таблице хранится информация о сдаче экзамена №2 в следующем виде:

id (идентификатор студента в базе данных), name (имя), score (оценка).

Вам необходимо получить общую таблицу с результатами двух экзаменов. В итоговой таблице должны храниться данные только по тем студентам, которые сдавали оба экзамена (то есть те, о ком есть информация и в первой, и во второй таблице).

Для объединения таблиц вы решили использовать метод merge. Какое значения аргумента how необходимо использовать для корректного отображения результатов?

- ○ left
- ◉ inner
- ○ outer
- ○ right