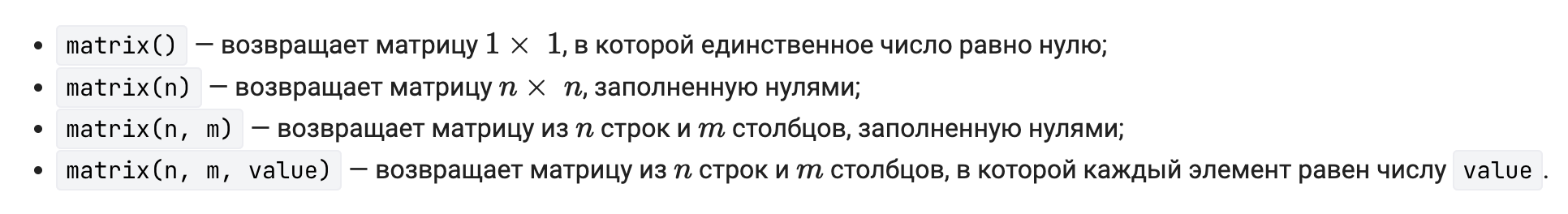
**1РАЗ необязательные и именованные аргументы**

Напишите функцию matrix(), которая создает, заполняет и возвращает матрицу заданного размера. При этом (в зависимости от переданных аргументов) она должна вести себя так:

При создании функции пользуйтесь аргументами по умолчанию.

**Примечание 1.** Приведенный ниже код:

print(matrix()) # матрица 1 × 1 из 0

print(matrix(2, 5)) # матрица 2 × 5 из 0

print(matrix(3, 4, 9)) # матрица 3 × 4 из 9

должен выводить:

[[0]]

[[0, 0, 0, 0, 0], [0, 0, 0, 0, 0]]

[[9, 9, 9, 9], [9, 9, 9, 9], [9, 9, 9, 9]]

**2РАЗфункции с переменным количеством аргументов**

Напишите функцию count\_args(), которая принимает произвольное количество аргументов и возвращает количество переданных в нее аргументов.

**Примечание 1.** Обратите внимание, что функция должна принимать не список, а именно произвольное количество аргументов.

**Примечание 2.** Следующий программный код:

print(count\_args())

print(count\_args('stepik', 'beegeek'))

print(count\_args([], (''), 'a', 12, False))

должен выводить:

0

2

5

**3РАЗфункции с переменным количеством аргументов**

Напишите функцию sq\_sum(), которая принимает произвольное количество числовых аргументов и возвращает сумму их квадратов.

**Примечание 1.** Обратите внимание, что функция должна принимать не список, а именно произвольное количество аргументов.

**Примечание 2.** Следующий программный код:

print(sq\_sum()) - 0

print(sq\_sum(1, 2, 3)) - 14

print(sq\_sum(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)) - 385

**4РАЗ функции с переменным количеством аргументов**

Напишите функцию greet(), которая принимает произвольное количество аргументов строк имен (как минимум одно) и возвращает приветствие в соответствии с образцом.

**Примечание 1.** Обратите внимание, что функция должна принимать не список, а именно произвольное количество аргументов.

**Примечание 2.** Следующий программный код:

print(greet('Timur'))

print(greet('Timur', 'Roman'))

print(greet('Timur', 'Roman', 'Ruslan'))

должен выводить:

Hello, Timur!

Hello, Timur and Roman!

Hello, Timur and Roman and Ruslan!

**Примечание 3.** Функция greet() должна принимать как минимум один обязательный аргумент!

**5РАЗ функции с переменным количеством аргументов**

Напишите функцию info\_kwargs(), которая принимает произвольное количество именованных аргументов и печатает именованные аргументы в соответствии с образцом: <имя аргумента>: <значение аргумента>, при этом имена аргументов следуют в алфавитном порядке (по возрастанию).

**Примечание 1.** Обратите внимание, что функция должна принимать не список, а именно произвольное количество именованных аргументов.

**Примечание 2.** Следующий программный код:

info\_kwargs(first\_name='Timur', last\_name='Guev', age=28, job='teacher')

должен выводить:

age: 28

first\_name: Timur

job: teacher

last\_name: Guev

**6РАЗ функции как объекты**

Список athletes содержит сведения о спортсменах в виде кортежей: (имя, возраст, рост, вес).

Напишите программу сортировки списка спортсменов по указанному полю:

* 1: по имени;
* 2: по возрасту;
* 3: по росту;
* 4: по весу.

**Формат входных данных**  
На вход программе подается натуральное число от 1 до 4 – номер поля по которому требуется отсортировать список.

**Формат выходных данных**  
Программа должна вывести отсортированный по заданному полю список в соответствии с примерами.

**Примечание.** Решите задачу **без** использования условного оператора.

athletes = [['Дима', 10, 130, 35], ['Тимур', 11, 135, 39], ['Руслан', 9, 140, 33], ['Рустам', 10, 128, 30], ['Амир', 16, 170, 70], ['Рома', 16, 188, 100], ['Матвей', 17, 168, 68], ['Петя', 15, 190, 90]]

**Sample Input 1:**

3

**Sample Output 1:**

Рустам 10 128 30

Дима 10 130 35

Тимур 11 135 39

Руслан 9 140 33

Матвей 17 168 68

Амир 16 170 70

Рома 16 188 100

Петя 15 190 90

**Sample Input 2:**

2

**Sample Output 2:**

Руслан 9 140 33

Дима 10 130 35

Рустам 10 128 30

Тимур 11 135 39

Петя 15 190 90

Амир 16 170 70

Рома 16 188 100

Матвей 17 168 68

**7РАЗ функции высшего порядка**

Напишите программу, которая с помощью функций filter() и map() отбирает из заданного списка numbers трёхзначные числа, дающие при делении на 55 остаток 22, и выводит их кубы, каждый в отдельной строке.

**Примечание.** Остаток 2 при делении на 5 должно давать само число, а не его куб.

def map(function, items):

result = []

for item in items:

result.append(function(item))

return result

def filter(function, items):

result = []

for item in items:

if function(item):

result.append(item)

return result

numbers = [1014, 1321, 675, 1215, 56, 1386, 1385, 431, 1058, 486, 1434, 696, 1016, 1084, 424, 1189, 475, 95, 1434, 1462, 815, 776, 657, 1225, 912, 537, 1478, 1176, 544, 488, 668, 944, 207, 266, 1309, 1027, 257, 1374, 1289, 1155, 230, 866, 708, 144, 1434, 1163, 345, 394, 560, 338, 232, 182, 1438, 1127, 928, 1309, 98, 530, 1013, 898, 669, 105, 130, 1363, 947, 72, 1278, 166, 904, 349, 831, 1207, 1496, 370, 725, 926, 175, 959, 1282, 336, 1268, 351, 1439, 186, 273, 1008, 231, 138, 142, 433, 456, 1268, 1018, 1274, 387, 120, 340, 963, 832, 1127]

**8РАЗ анонимные функции**

Требовалось написать программу, которая:

* преобразует список floats в список чисел, возведенных в квадрат и округленных с точностью до одного десятичного знака;
* фильтрует список words  и оставляет только палиндромы длиной более 44 символов;
* находит произведение чисел из списка numbers.

Программист торопился и написал программу неправильно. Доработайте его программу.

from functools

floats = [4.35, 6.09, 3.25, 9.77, 2.16, 8.88, 4.59, 34.23, 12.12, 4.67, 2.45, 9.32]

words = ['racecar', 'akinremi', 'deed', 'temidayo', 'omoseun', 'civic', 'TATTARRATTAT', 'malayalam', 'nun']

numbers = [4, 6, 9, 23, 5]

# Исправьте этот код

map\_result = list(map(lambda num: num, floats))

filter\_result = list(filter(lambda name: name, words))

reduce\_result = reduce(lambda num1, num2: num1 + num2, numbers, 0)

print(map\_result)

print(filter\_result)

print(reduce\_result)

**9 РАЗ** Анонимные функции

Напишите программу, которая с помощью встроенных функций filter(), map(), sorted() и reduce() выводит в алфавитном порядке список primary городов с населением более 1000000010000000 человек, в формате:

Cities: Beijing, Buenos Aires, ...

**Примечание 1.** Тестирующая система никак не "покарает" вас за неиспользование встроенных функций filter(), map(), sorted() и reduce(), однако лучше сделать это задание честно 😃.

**Примечание 2.** Ставить запятую в конце вывода не нужно.

data = [['Tokyo', 35676000, 'primary'],

['New York', 19354922, 'nan'],

['Mexico City', 19028000, 'primary'],

['Mumbai', 18978000, 'admin'],

['Sao Paulo', 18845000, 'admin'],

['Delhi', 15926000, 'admin'],

['Shanghai', 14987000, 'admin'],

['Kolkata', 14787000, 'admin'],

['Los Angeles', 12815475, 'nan'],

['Dhaka', 12797394, 'primary'],

['Buenos Aires', 12795000, 'primary'],

['Karachi', 12130000, 'admin'],

['Cairo', 11893000, 'primary'],

['Rio de Janeiro', 11748000, 'admin'],

['Osaka', 11294000, 'admin'],

['Beijing', 11106000, 'primary'],

['Manila', 11100000, 'primary'],

['Moscow', 10452000, 'primary'],

['Istanbul', 10061000, 'admin'],

['Paris', 9904000, 'primary']]

**10РАЗ**

Напишите функцию is\_non\_negative\_num, используя синтаксис анонимных функций, которая принимает строковый аргумент и возвращает значение True, если переданный аргумент является **неотрицательным числом** (целым или вещественным) и False в противном случае.

**Примечание 1.** Следующий программный код:

print(is\_non\_negative\_num('10.34ab'))

print(is\_non\_negative\_num('10.45'))

print(is\_non\_negative\_num('-18'))

print(is\_non\_negative\_num('-34.67'))

print(is\_non\_negative\_num('987'))

print(is\_non\_negative\_num('abcd'))

print(is\_non\_negative\_num('123.122.12'))

print(is\_non\_negative\_num('123.122'))

должен выводить:

False

True

False

False

True

False

False

True

**Примечание 2.**

Неотрицательные числа — это положительные числа и число нуль.

**11 РАЗ** Встроенные функции

Используя параллельную итерацию сразу по трем спискам countries, capitals и population выведите информацию о стране в формате:

<capital> is the capital of <country>, population equal <population> people.

Moscow is the capital of Russia, population equal 145934462 people.

Washington is the capital of USA, population equal 331002651 people.

Для каждой страны информацию выводить на отдельной строке.

countries = ['Russia', 'USA', 'UK', 'Germany', 'France', 'India']

capitals = ['Moscow', 'Washington', 'London', 'Berlin', 'Paris', 'Delhi']

population = [145\_934\_462, 331\_002\_651, 80\_345\_321, 67\_886\_011, 65\_273\_511, 1\_380\_004\_385]