# Функции с переменным числом аргументов

# Матрица

Kлассная работа

макс. 1 балл.

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 64Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

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

* matrix() — создает матрицу 1×11×1, в которой единственное число равно нулю.
* matrix(n) — создает матрицу n×nn×n, заполненную нулями.
* matrix(n, m) — создает матрицу из nn строк и mm столбцов, заполненную нулями.
* matrix(n, m, a) — создает матрицу из nn строк и mm столбцов, в которой каждый элемент равен aa.

При создании функции пользуйтесь аргументами по умолчанию.

### Пример 1

| Ввод | Вывод |
| --- | --- |
| rows = matrix()  for row in rows:  print(\*row) | 0 |

### Пример 2

| Ввод | Вывод |
| --- | --- |
| rows = matrix(2)  for row in rows:  print(\*row) | 0 0  0 0 |

# Бариста

Kлассная работа

макс. 1 балл.

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 64Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

На заводе «Кофейный» открывается новое кафе. Изначально есть некоторое количество кофейных зерен, молока и взбитых сливок.

Надо написать функцию **choose\_coffee(preference1, preference2,..., preferenceN)**, которая возвращает напиток, который можно приготовить из имеющихся продуктов (ingredients). На вход функция принимает заранее неизвестное количество предпочтений посетителя. Все напитки перечислены в порядке убывания предпочтений и гарантированно не повторяются. Бариста готовит наиболее предпочитаемый напиток из доступных.

Для Эспрессо требуется: 1 порция кофейных зерен.

Для Капучино требуется: 1 порция кофейных зерен и 3 порции молока.

Для Маккиато требуется: 2 порции кофейных зерен и 1 порция молока.

Для Кофе по-венски требуется: 1 порция кофейных зерен и 2 порции взбитых сливок.

Для Латте Маккиато требуется: 1 порция кофейных зерен, 2 порции молока и 1 порция взбитых сливок.

Для Кон Панна требуется: 1 порция кофейных зерен и 1 порция взбитых сливок.

При приготовлении напитка ингредиенты расходуются.

Если недостаточно ингредиентов, то вернуть сообщение: «К сожалению, не можем предложить Вам напиток».

### Пример 1

| Ввод | Вывод |
| --- | --- |
| ingredients = [1, 2, 3] | Эспрессо  К сожалению, не можем предложить Вам напиток |
| print(choose\_coffee("Эспрессо", "Капучино", "Маккиато", "Кофе по-венски", "Латте Маккиато", "Кон Панна")) |
| print(choose\_coffee("Эспрессо", "Капучино", "Маккиато", "Кофе по-венски", "Латте Маккиато", "Кон Панна")) |

### Пример 2

| Ввод | Вывод |
| --- | --- |
| ingredients = [4, 4, 0] | Капучино  Маккиато  Эспрессо |
| print(choose\_coffee("Капучино", "Маккиато", "Эспрессо")) |
| print(choose\_coffee("Капучино", "Маккиато", "Эспрессо")) |
| print(choose\_coffee("Капучино", "Маккиато", "Эспрессо")) |

## Примечания

ingredients – список длины три, в котором хранится количество порций кофейных зерен, молока и сливок соответственно.

# Дартс

Kлассная работа

макс. 1 балл.

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 64Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Стандартная мишень для игры в дартс разделена на 20 ячеек с номерами от 1 до 20. В центре расположено «яблочко», попадание в которое приносит игроку 50 очков. Вокруг него — зелёное кольцо, при попадании в которое засчитывается 25 очков. Попадание во внешнее (узкое) кольцо мишени удваивает число сектора, а во внутреннее — утраивает.

Эти правила показались игрокам слишком простыми, поэтому они решили присвоить секторам во внешнем и внутреннем кольцах случайные значения. В глобальной переменной **scoring** хранится словарь для подсчета очков (обратите внимание, что в случае внутреннего и внешнего колец значениями являются словари, ключами в которых являются номера сектора, а значениями –– количество очков):

Яблочко: 50

Зеленое кольцо: 25

Внешнее кольцо: 1: 8, 2: 6, 3: 42,…, 20: 50

Внутреннее кольцо: 1: 2, 2: 9, 3: 56,…, 20: 3

Напишите функцию **score()**, которая принимает на вход 1 (если это «Яблочко» или «Зеленое кольцо») или 2 аргумента (если это внутреннее или внешнее кольцо, то название кольца и номер сектора) и возвращает количество очков.

### Пример 1

| Ввод | Вывод |
| --- | --- |
| print(score("Яблочко")) | 50 |

### Пример 2

| Ввод | Вывод |
| --- | --- |
| print(score("Внешнее\_кольцо", 1)) | 8 |

## Примечания

В этой задаче не надо использовать параметры \*args и \*\*kwargs

# Цезарь

Kлассная работа

макс. 3 балл.

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 64Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Напишите функцию encrypt\_caesar(msg, shift), которая кодирует сообщение шифром Цезаря и возвращает его. Шифр Цезаря заменяет каждую букву в тексте на букву, которая отстоит в алфавите на некоторое фиксированное число позиций.

В функцию передается сообщение и сдвиг алфавита. Если сдвиг не указан, то пусть ваша функция кодирует сдвиг алфавита на 3 позиции:

А→Г,А→Г,

Б→Д,Б→Д,

В→Е,В→Е,

……

Э→А,Э→А,

Ю→Б,Ю→Б,

Я→ВЯ→В

Все символы, кроме русских букв должны остаться неизменными. Маленькие буквы должны превращаться в маленькие, большие — в большие.

Напишите также функцию декодирования decrypt\_caesar(msg, shift), также использующую сдвиг по умолчанию. При написании функции декодирования используйте вашу функцию кодирования.

### Пример 1

| Ввод | Вывод |
| --- | --- |
| msg = "Да здравствует салат Цезарь!"  shift = 3  encrypted = encrypt\_caesar(msg, shift)  decrypted = decrypt\_caesar(encrypted, shift)  print(encrypted)  print(decrypted) | Зг кзугефхецих фгогх Щикгуя!  Да здравствует салат Цезарь! |

### Пример 2

| Ввод | Вывод |
| --- | --- |
| msg = "Да здравствует салат Цезарь!"  shift = 5  encrypted = encrypt\_caesar(msg, shift)  decrypted = decrypt\_caesar(encrypted, shift)  print(encrypted)  print(decrypted) | Йе мйхезцчзшкч цереч Ыкмехб!  Да здравствует салат Цезарь! |

## Примечания

Символы русского алфавита расположены в стандартной для Python таблице кодировки подряд, то есть номера, выдаваемые функцией ord(symbol), идут подряд.

Буква «ё» идёт в таблице кодировки отдельно от основного алфавита. При решении задачи считайте, что буквы «ё» в русском алфавите нет.

# Частичные суммы

Kлассная работа

макс. 3 балл.

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 64Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

kk-ой частичной суммой списка называется сумма первых kk элементов списка. Напишите функцию partial\_sums, которая принимает неограниченное число аргументов, а возвращает список частичных сумм этих элементов: на нулевой позиции — 0, на первой позиции — первое число, на второй — сумму первого и второго чисел, затем — сумму первого, второго и третьего и т.д.

Обратите внимание, что функция должна принимать не список, а именно неограниченное число аргументов.

### Пример 1

| Ввод | Вывод |
| --- | --- |
| print(partial\_sums(13)) | [0, 13] |

### Пример 2

| Ввод | Вывод |
| --- | --- |
| print(partial\_sums(1, 0.5, 0.25, 0.125)) | [0, 1, 1.5, 1.75, 1.875] |

## Примечания

Данная задача дополнительно проверяется преподавателем.

# Уравнения степени не выше второй — часть 2

Kлассная работа

макс. 3 балл.

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 64Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Сделайте функцию solve(\*coefficients), которая умеет решать уравнения степени не выше второй (квадратные и линейные).

* Если у функции три аргумента, их надо трактовать как a, b и c в уравнении ax2+bx+c=0.
* Если два — как коэффициенты b и c в уравнении bx+c=0.
* Если один — как коэффициент c в уравнении c = 0.
* Если список коэффициентов пуст или коэффициентов больше трёх, то функция должна вернуть None.

Значения корней возвращайте в том же виде, как их возвращала функция roots\_of\_quadratic\_equation(a,b,c) из задачи «Уравнения степени не выше второй» Для решения можно как переписать ту функцию, так и использовать её внутри новой функции.

### Пример 1

| Ввод | Вывод |
| --- | --- |
| print(sorted(solve(1, 2, 1))) | [-1.0] |

### Пример 2

| Ввод | Вывод |
| --- | --- |
| print(sorted(solve(1, -3, 2))) | [1.0, 2.0] |