Python (BSU FAMCS Fall'19)

Семинар 2

Преподаватель: Дмитрий Косицин

Задание 1. (0.5 балла). Напишите функцию, которая принимает один аргумент n – натуральное число – и считает (возвращает) сумму $1^2 * 2 + 2^2 * 3 + ... + (n-1)^2 * n$. Используйте одно comprehension выражение для решения задачи. Асимптотическая сложность алгоритма должна быть $O(n)$.
Замечание. Функцию назовите *calculate_special_sum*. Программу сохраните в файле *special_sum.py*.

Пример

```
assert calculate_special_sum(3) == 14
```

Задание 2. (0.5 балла). Напишите функцию, которая принимает на вход последовательность (кортеж или список) и возвращает список пар из уникальных элементов и количества раз, сколько они встретились в переданной последовательности. Парой называется кортеж из двух элементов. Порядок пар в списке не важен.
Замечание. Функцию назовите *compress*. Программу сохраните в файле *unique.py*.

Пример

```
expected_sorted = [(1, 2), (2, 1)]
actual_sorted = sorted(compress([1, 2, 1]))
assert expected_sorted == actual_sorted
```

Задание 3. (1.5 балла). Напишите функцию, которая принимает один аргумент n – натуральное число – и возвращает все простые числа, не превосходящие n. Используйте comprehensions для решения задачи.

Полный балл за задачу будет выставлен только в случае написания функции из одного comprehension выражения.

Функцию назовите *get_primes*. Программу сохраните в файле *primes.py*.

Пример

```
assert [2, 3, 5, 7, 11] == sorted(get_primes(11))
```

Задание 4. (1.5 балла). Напишите функцию, которая для выборки, заданной списком, и заданного натурального числа k посчитает и выведет гистограмму распределения шириной k. Другими словами, найдет максимум и минимум, поделит интервал на k частей ($1 \leqslant k \leqslant d$, d – количество различных элементов в списке) и посчитает, сколько элементов выборки попало в каждый интервал.

Верните список длины k, содержащий количество элементов в каждой ячейке. Левую границу интервалов считайте включая, правую – исключая (кроме последнего интервала). Желательно реализовать алгоритм со сложностью $O(n)$, где n – длина входного списка.
Замечание. Функцию назовите *distribute*. Программу сохраните в файле *hist.py*.

Пример

```
assert distribute([1.25, 1, 2, 1.75], 2) == [2, 2]
```