Python. Семинар 2

Преподаватели: Дмитрий Косицин, Светлана Боярович и Анастасия Мицкевич

Задание 1. Напишите функцию, которая принимает один аргумент n – натуральное число – и считает (возвращает) сумму $1*2+2*3+...+(n-1)*n$. Используйте comprehensions для решения задачи.

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

Функию назовите calculate_special_sum. Программу сохраните в файле special_sum.py.

Задание 2. Напишите функцию, которая принимает один аргумент n – натуральное число – и возвращает все Пифагоровские тройки, т.е. тройки натуральных чисел (x, y, z), что $x^2 + y^2 = z^2$ и $1 \leqslant x \leqslant n, 1 \leqslant y \leqslant n, 1 \leqslant z \leqslant n$. Тройкой называется кортеж из трех элементов. Используйте comprehensions для решения задачи.

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

Функцию назовите get_pythagoras_triples.py. Функции сохраните в файле pythagoras.py.

Задание 3. 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.

Задание 4. 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.

Пример. Вызов compress($[1, 2, 1]$) вернет список $[(1, 2), (2, 1)]$.

Функцию назовите compress. Программу сохраните в файле unique.py.

Задание 5. Реализуйте сортировку слиянием (merge sort, асимптотическая сложность алгоритма – $O(n \log(n))$). Программа должна корректно работать для списков (возвращать новый отсортированный список) и кортежей (возвращать кортеж). Использовать встроенные функции сортировки и/или слияния запрещается.

Полный балл за задачу будет выставлен только в случае отсутствия итерирования по сортируемой коллекции с помощью индексов. Иначе говоря, рекомендуется обращаться только к нулевому и последнему элементам коллекций, а произвольной индексации избегать.

Функцию назовите sort. Программу сохраните в файле merge_sort.py.

Общие замечания к заданиям.

Задания следует выполнять, используя Python 2. При особом желании написать на Python 3, следует об этом написать отдельно.

Оценивается количество сделанных задач, их корректность, стиль кода, наличие нерекомендованных конструкций (например, использование связки reduce + lambda). Делать все не обязательно, успейте, главное, до дедлайна. Оценка работ после дедлайна остается на усмотрение проверяющего, но, по умолчанию, такие работы не проверяются.

Для тестирования интерфейса ваших заданий выложен специальный скрипт. Если тесты не проходят, задание проверяться не будет. Просьба также не оставлять debug print'ы в ваших программах.

Если условие на ваш взгляд допускает неоднозначность – спросите при сдаче у проверяющего! Мы знаем ответы!

Списывание работ не рекомендуется, ведь проверяющие могут на вас обидеться. В любом случае, то, как вы усвоите материал – на вашей совести. Все задания нужны именно для отработки пройденного.

О работе с Anytask.

Не следует загружать архивы, как вы уже могли понять.

Желательно поставить фото, поскольку так легче распознать вас на паре.

Письма можно писать в системе – либо в контексте задачи (там же, где загружать файлы), либо воспользоваться общей системой сообщений (см. конвертик в верхнем правом углу).