1. Напишите функцию lensort, которая принимает список строк и сортирует его в порядке возрастания их длины. Возвращает отсортированный таким образом список. Исходный список должен остаться неизменным.



Что нужно знать:

* Kак происходит передача параметра в функцию: По ссылке? По значению? По копии ссылки?
* Разница между функцией sort и методом sorted класса list
* Лямбда-выражения и как их использовать вместе с sort

1. Напишите функцию unique, которая удаляет дубликаты из списка и возвращает результат



Что нужно знать:

* Какую структуру данных реализует set и как (и почему) с его помощью можно избавиться от дубликатов?

1. Напишите функцию my\_enumerate, которая принимает произвольный список, и возвращает список кортежей, в каждом из которых два элемента: элемент списка и порядковый номер данного элемента



Подсказки:

* range
* zip

1. Напишите функцию, принимающую имя файла с текстом и подсчитывающую частоту встречающихся в нём слов. Каждая линия вывода имеет формат:  
   «<Слово>: <Сколько раз встречается это слово>»

Что нужно знать:

* Что представляет собой тип dict и как им пользоваться
* Как читать файлы
* Как разделить строку с текстом на слова

1. Напишите декоратор, который измеряет время выполнения функции и выводит его в консоль. Проверьте его действие на трёх функциях, принимающих список целых чисел и возвращающих список их квадратов, но выполняющих эту задачу тремя способами: через цикл for, через list comprehension и с использованием встроенной функции map

Подсказки:

* Чтобы узнать, сколько времени, можно использовать модуль time из стандартной библиотеки
* list comprehension – это конструкция типа i = [i for i in …]

1. Написать **key-value** хранилище. Данные будут сохраняться в файле **storage.data**. Добавление новых данных в хранилище и получение текущих значений осуществляется с помощью утилиты командной строки **storage.py**. Пример работы утилиты:

**Сохранение значения value по ключу key\_name:**

$ storage.py --key key\_name --val value

**Получение значения по ключу key\_name:**

$ storage.py --key key\_name

Утилита может вызваться со следующими параметрами:

--key <имя ключа>*,* где <имя ключа> - ключ по которому сохраняются/получаются значения

--val <значение>*,* где <значение> - сохраняемое значение.

Если при запуске утилиты переданы оба ключа, происходит добавление переданного значения по ключу и сохранение данных в файле. Если передано только имя ключа, происходит чтение файла хранилища и вывод на печать значений, которые были сохранены по данному ключу.  Обратите внимание, что значения по одному ключу не перезаписываются, а добавляются к уже сохраненным. Другими словами - по одному ключу могут храниться несколько значений. При выводе на печать, значения выводятся в порядке их добавления в хранилище. Формат вывода на печать для нескольких значений:

value\_1, value\_2

Обратите внимание на пробел после запятой. Если значений по ключу не было найдено, выведите пустую строку или None.

Подсказки:

* Для работы с аргументами командной строки используйте модуль [argparse](https://docs.python.org/3/howto/argparse.html" \o "" \t "_blank).
* Хранить данные в файле можно в формате JSON с помощью модуля стандартной библиотеки  [json](https://docs.python.org/3/library/json.html" \o "" \t "_blank).
* Файл следует можно с помощью модуля [tempfile](https://docs.python.org/3/library/tempfile.html).