# Неделя-2

## 2-1: Key-value хранилище

На этой неделе мы с вами реализуем собственное **key-values** 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
```

Вашей задачей будет написать реализацию утилиты***storage.py***.
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.

Для работы с аргументами командной строки используйте модуль argparse. Хранить данные в файле мы рекомендуем в формате JSON с помощью использования модуля стандартной библиотеки  json. Прежде чем отправлять ваше решение на проверку, протестируйте работу вашей утилиты на добавление нескольких ключей и разных значений.

Файл следует создавать с помощью модуля tempfile.

```
import os
import tempfile
storage_path = os.path.join(tempfile.gettempdir(), 'storage.data')
with open(storage_path, 'w') as f:
  ...
```

**Пример работы:**

```
$ python storage.py --key key_name --val value
$ python storage.py --key key_name
value
$ python storage.py --key multi_key --val value1
$ python storage.py --key multi_key --val value2
$ python storage.py --key multi_key
value1, value2
```

Часто задаваемые вопросы и полезные ссылки по данному заданию - **FAQ**.