**Лабораторная работа 2. Автоматизированный сбор данных. Работа со строками.**

**Общие требования к лабораторным работам**

1. Лабораторная работа выполняется в виде отдельного python-скрипта.
2. Каждая лабораторная работа должна быть загружена в отдельный git-репозиторий.
3. Взаимодействие с репозиторием должно производиться посредством работы с IDE, а не через сайт.
4. Репозитории с одним коммитом к проверке не принимаются.
5. Сообщения коммитов должны быть осмысленными и отражать процесс выполнения задания.
6. Код должен соответствовать требованиям соглашения PEP8.

**Web scraping**

Веб-скрейпинг (или скрепинг, или скрапинг← англ. web scraping) — это технология получения веб-данных путем извлечения их со страниц веб-ресурсов. Веб-скрейпинг может быть сделан вручную пользователем компьютера, однако термин обычно относится к автоматизированным процессам, реализованным с помощью кода, который выполняет GET-запросы на целевой сайт.

[Некоторая вспомогательная информация (не является руководством к действию)](https://tproger.ru/translations/skraping-sajta-s-pomoshhju-python-gajd-dlja-novichkov/)

Для парсинга html разрешено использовать **BeautifulSoup**.

**Получение html кода веб-страницы**

import os

import requests

URL = "https://yandex.ru/"

html\_page = requests.get(URL, headers={"User-Agent":"Mozilla/5.0"})

*# html\_page.text - хранит html код веб-страницы*

Для работы с загруженными изображениями (не обязательно с расширением, характерным для изображения!) вам могут потребоваться следующие инструкции:

import cv2 *# импорт библиотеки, предназначенной для работы с изображениями*

image = cv2.imread(path\_to\_file) *# прочтение изображения из файла, path\_to\_file - путь до файла-изображения*

cv2.imwrite(path\_to\_save\_image, image) *# сохранение изображения по заданному пути, например, path\_to\_folder/image\_name.jpg*

print(image.shape) *# распечатать размер прочитанного изображения*

*# инструкции для просмотра изображения*

cv2.imshow(window\_name, image)

cv2.waitKey(0)

**Числовые данные**

**Вариант 1.**

С использованием веб-сайта <https://www.cbr-xml-daily.ru> получить курс доллара по дням на максимально возможный период. Результат сохранить в выходной файл dataset.csv, где каждая строка будет содержать дату и курс, разделенные запятой.

**Примечания**

Пример ссылки для получения данных:<https://www.cbr-xml-daily.ru/archive/2022/09/08/daily_json.js>

**Вариант 2.**

С использованием сервиса **gismeteo** получите данные о погоде в Кишиневе за максимально возможный период. Данные необходимо сохранить в выходном файле dataset.csv, где каждой строке будет соответсвовать отдельный день, а в строке через запятую будут указаны дата, температура, давление, данные о ветре.

**Примечания**

Пример ссылки для получения данных: <https://www.gismeteo.ru/diary/4976/2023/9/>

**Изображения**

**Вариант 3.**

С использованием страницы <https://yandex.ru/images/> сформировать запросы для поиска изображений, контент на которых соответствует классам \***cat**\* и \***dog***. Для каждого класса должно быть загружено не менее 1000 изображений. Изображения для каждого класса должны находиться в подпапке папки***dataset**\* с соответсвующим названием.

**Не допускается**:

1. Создание папок вручную. В коде должен быть отражен процесс создания папок и перемещения/загрузки в них файлов.
2. Дублирование изображений для класса.

**Примечания**

Каждое изображение должно иметь расширение *.jpg*

Именовать файлы необходимо порядковым номером (от 0 до 999).

Для дальнейшего удобства необходимо дополнять имя файла ведующими нулями (например, 0000, 0001, ..., 0999). Для этого необходимо использовать один из методов класса **str**.

После загрузки всех изображений, необходимо их просмотреть на соответствие классу. В случае замеченных несоответствий необходимо будет дополнить набор данных до минимального размера. Для избежания подобных ситуаций рекомендуется загружать изображения с запасом.

Вариант подразумевает два уровня сложности:

1. Для первого уровня сложности достаточно загрузить лишь миниатюры изображений.
2. Для второго уровня сложности необходимо загрузить полноразмерные изображения.

**Вариант 4.**

С использованием страницы <https://yandex.ru/images/> сформировать запросы для поиска изображений, контент на которых соответствует классам \***rose**\* и \***tulip***. Для каждого класса должно быть загружено не менее 1000 изображений. Изображения для каждого класса должны находиться в подпапке папки***dataset**\* с соответсвующим названием.

**Не допускается**:

1. Создание папок вручную. В коде должен быть отражен процесс создания папок и перемещения/загрузки в них файлов.
2. Дублирование изображений для класса.

**Примечания**

Каждое изображение должно иметь расширение *.jpg*

Именовать файлы необходимо порядковым номером (от 0 до 999).

Для дальнейшего удобства необходимо дополнять имя файла ведующими нулями (например, 0000, 0001, ..., 0999). Для этого необходимо использовать один из методов класса **str**.

После загрузки всех изображений, необходимо их просмотреть на соответствие классу. В случае замеченных несоответствий необходимо будет дополнить набор данных до минимального размера. Для избежания подобных ситуаций рекомендуется загружать изображения с запасом.

Вариант подразумевает два уровня сложности:

1. Для первого уровня сложности достаточно загрузить лишь миниатюры изображений.
2. Для второго уровня сложности необходимо загрузить полноразмерные изображения.

**Вариант 5.**

С использованием страницы <https://yandex.ru/images/> сформировать запросы для поиска изображений, контент на которых соответствует классам \***polar bear**\* и \***brown bear***. Для каждого класса должно быть загружено не менее 1000 изображений. Изображения для каждого класса должны находиться в подпапке папки***dataset**\* с соответсвующим названием.

**Не допускается**:

1. Создание папок вручную. В коде должен быть отражен процесс создания папок и перемещения/загрузки в них файлов.
2. Дублирование изображений для класса.

**Примечания**

Каждое изображение должно иметь расширение *.jpg*

Именовать файлы необходимо порядковым номером (от 0 до 999).

Для дальнейшего удобства необходимо дополнять имя файла ведующими нулями (например, 0000, 0001, ..., 0999). Для этого необходимо использовать один из методов класса **str**.

После загрузки всех изображений, необходимо их просмотреть на соответствие классу. В случае замеченных несоответствий необходимо будет дополнить набор данных до минимального размера. Для избежания подобных ситуаций рекомендуется загружать изображения с запасом.

Вариант подразумевает два уровня сложности:

1. Для первого уровня сложности достаточно загрузить лишь миниатюры изображений.
2. Для второго уровня сложности необходимо загрузить полноразмерные изображения.

**Вариант 6.**

С использованием страницы <https://yandex.ru/images/> сформировать запросы для поиска изображений, контент на которых соответствует классам \***tiger**\* и \***leopard***. Для каждого класса должно быть загружено не менее 1000 изображений. Изображения для каждого класса должны находиться в подпапке папки***dataset**\* с соответсвующим названием.

**Не допускается**:

1. Создание папок вручную. В коде должен быть отражен процесс создания папок и перемещения/загрузки в них файлов.
2. Дублирование изображений для класса.

**Примечания**

Каждое изображение должно иметь расширение *.jpg*

Именовать файлы необходимо порядковым номером (от 0 до 999).

Для дальнейшего удобства необходимо дополнять имя файла ведующими нулями (например, 0000, 0001, ..., 0999). Для этого необходимо использовать один из методов класса **str**.

После загрузки всех изображений, необходимо их просмотреть на соответствие классу. В случае замеченных несоответствий необходимо будет дополнить набор данных до минимального размера. Для избежания подобных ситуаций рекомендуется загружать изображения с запасом.

Вариант подразумевает два уровня сложности:

1. Для первого уровня сложности достаточно загрузить лишь миниатюры изображений.
2. Для второго уровня сложности необходимо загрузить полноразмерные изображения.

**Вариант 7.**

С использованием страницы <https://yandex.ru/images/> сформировать запросы для поиска изображений, контент на которых соответствует классам \***zebra**\* и \***bay horse***. Для каждого класса должно быть загружено не менее 1000 изображений. Изображения для каждого класса должны находиться в подпапке папки***dataset**\* с соответсвующим названием.

**Не допускается**:

1. Создание папок вручную. В коде должен быть отражен процесс создания папок и перемещения/загрузки в них файлов.
2. Дублирование изображений для класса.

**Примечания**

Каждое изображение должно иметь расширение *.jpg*

Именовать файлы необходимо порядковым номером (от 0 до 999).

Для дальнейшего удобства необходимо дополнять имя файла ведующими нулями (например, 0000, 0001, ..., 0999). Для этого необходимо использовать один из методов класса **str**.

После загрузки всех изображений, необходимо их просмотреть на соответствие классу. В случае замеченных несоответствий необходимо будет дополнить набор данных до минимального размера. Для избежания подобных ситуаций рекомендуется загружать изображения с запасом.

Вариант подразумевает два уровня сложности:

1. Для первого уровня сложности достаточно загрузить лишь миниатюры изображений.
2. Для второго уровня сложности необходимо загрузить полноразмерные изображения.

**Текстовые данные**

**Вариант 8**

С использованием сервиса **кинопоиск** выберите несколько фильмов с большим количеством рецензий. Таким образом, чтобы суммарно возможно было получить 1000 положительных и 1000 отрицательных рецензий. Затем реализуйте скрипт, который сохранит каждый отзыв в отдельный файл.

**Примечания**

Именовать файлы необходимо порядковым номером (от 0 до 999).

Для дальнейшего удобства необходимо дополнять имя файла ведующими нулями (например, 0000, 0001, ..., 0999). Для этого необходимо использовать один из методов класса **str**.

Каждую рецензию сохраните в отдельный текстовый файл в соответствующую подпапку папки **dataset**. (Пути должны быть dataset/bad/0001.txt или dataset/good/0001.txt)

Первая строка файла должна содержать название фильма.

Обратите внимание, на то что страницы с рецензиями необходимо обрабатывать в цикле.

Пример ссылки для получения рецензий: <https://www.kinopoisk.ru/film/535341/reviews/ord/rating/status/bad/perpage/10/page/1/>

**Вариант 9**

С использованием сервиса **otzovik** выберите объект с большим количеством отзывов, а именно, от 500 до 1000 отзывов для каждого количества звёзд. То есть суммарный объём датасета 2500-5000 отзывов. Затем реализуйте скрипт, который сохранит каждый отзыв в отдельный файл.

**Не допускается**:

1. Создание папок вручную. В коде должен быть отражен процесс создания папок и перемещения/загрузки в них файлов.
2. Дублирование данных для класса.

**Примечания**

Именовать файлы необходимо порядковым номером (от 0 до 999).

Для дальнейшего удобства необходимо дополнять имя файла ведующими нулями (например, 0000, 0001, ..., 0999). Для этого необходимо использовать один из методов класса **str**.

Каждую рецензию сохраните в отдельный текстовый файл в соответствующую подпапку папки **dataset**. (Пути должны быть dataset/0/0001.txt, dataset/1/0001.txt, и т. д. по количеству звёзд)

Обратите внимание, на то что страницы с отзывами необходимо обрабатывать в цикле.

Пример ссылки для получения отзывов: <https://otzovik.com/reviews/tv-kanal_netflix/>

Вариант подразумевает два уровня сложности:

1. Для первого уровня сложности достаточно сохранить начало отзыва, показываемое на странице.
2. Для второго уровня сложности необходимо сохранить отзыв полностью.

**Вариант 10**

С использованием сервиса **livelib** соберите по 1000 рецензий для каждого количества звёзд для различных книг. То есть суммарный объём датасета 5000 рецензий. Сохраните каждый отзыв в отдельный текстовый файл, где на первой строке будет указано название книги.

**Не допускается**:

1. Создание папок вручную. В коде должен быть отражен процесс создания папок и перемещения/загрузки в них файлов.
2. Дублирование данных для класса.

**Примечания**

Именовать файлы необходимо порядковым номером (от 0 до 999).

Для дальнейшего удобства необходимо дополнять имя файла ведующими нулями (например, 0000, 0001, ..., 0999). Для этого необходимо использовать один из методов класса **str**.

Каждую рецензию сохраните в отдельный текстовый файл в соответствующую подпапку папки **dataset**. (Пути должны быть dataset/0/0001.txt, dataset/1/0001.txt, и т. д. по количеству звёзд)

Обратите внимание, на то что страницы с отзывами необходимо обрабатывать в цикле.

Пример ссылки для получения отзывов: [https://www.livelib.ru/reviews/~2#reviews](https://www.livelib.ru/reviews/~2" \l "reviews)

Вариант подразумевает два уровня сложности:

1. Для первого уровня сложности достаточно сохранить начало отзыва, показываемое на странице.
2. Для второго уровня сложности необходимо сохранить отзыв полностью.

import cv2

image\_1 = cv2.imread("1.jpg")

image\_2 = cv2.imread("2.jpg")

def cmp(image\_1: cv2.Mat, image\_2: cv2.Mat) -> bool:

return image\_1 == image\_2