Министерство науки и образования РФ

Федеральное государственное бюджетное учреждение

высшего образования

**«Тверской государственный технический университет»**

(ТвГТУ)

Кафедра программного обеспечения

**Отчет по лабораторной работе №1**

По дисциплине: «Новые технологии в РПС»

Тема: «Автоматизированный сбор данных. Работа со строками»

|  |
| --- |
| Выполнил:  студент группы  М.ПИН.РИС - 23.08  Тимофеев Александр Сергеевич |
| Проверила:  старший преподаватель  кафедры ПО  Корнеева Е.И. |

Тверь 2023

**Вариант №1. Числовые данные**

**Постановка задачи:** С использованием веб-сайта [https://www.cbr-xml-daily.ru](https://colab.research.google.com/corgiredirector?site=https%3A%2F%2Fwww.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.

**Код программы (Python):**

Библиотеки:

import os

import requests

import csv

from datetime import datetime, timedelta

Константы:

CURR\_DIR = os.path.dirname(os.path.abspath(\_\_file\_\_))

Функция записи CSV файла с полученным датасетом:

def write\_dataset(start\_date = datetime(2023, 9, 10), end\_date = datetime.now()):

    with open(CURR\_DIR + '\dataset.csv', 'w', newline='', encoding='utf-8') as file:

        writer = csv.writer(file)

        writer.writerow(['Date', 'USD'])

        current\_date = start\_date

        while current\_date <= end\_date:

            url = f'https://www.cbr-xml-daily.ru/archive/{current\_date.year}/{current\_date.strftime("%m")}/{current\_date.strftime("%d")}/daily\_json.js'

            response = requests.get(url, headers={'User-Agent':'Mozilla/5.0'})

            if response.status\_code == 200:

                data = response.json()

                today = current\_date.strftime('%Y-%m-%d')

                writer.writerow([today, data['Valute']['USD']['Value']])

            current\_date += timedelta(days=1)

Функция чтения записанного CSV файла:

def read\_dataset():

    csv\_file\_path = CURR\_DIR + '\dataset.csv'

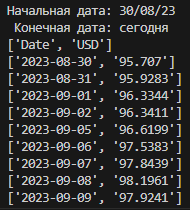
    with open(csv\_file\_path, 'r', newline='', encoding='utf-8') as csv\_file:

        csv\_reader = csv.reader(csv\_file)

        for row in csv\_reader:

            print(row)

Тестирование:



**Вариант №5. Изображения**

Постановка задачи: 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.

**Код программы (Python):**

Библиотеки:

import os

import requests

import re

from bs4 import BeautifulSoup

Константы:

CURR\_DIR = os.path.dirname(os.path.abspath(\_\_file\_\_))

Функции проверки созданных репозиториев:

def check\_dataset():

    dataset\_directory = os.path.join(CURR\_DIR, 'dataset')

    if not os.path.exists(dataset\_directory):

        os.makedirs(dataset\_directory)

def check\_repo\_dataset(class\_name):

    class\_folder = os.path.join(CURR\_DIR + '\dataset', class\_name)

    if not os.path.exists(class\_folder):

        os.makedirs(class\_folder)

        return class\_folder

    else:

        return class\_folder

Функция парсера ссылки на сурс картинки:

def parser\_url(url):

    pattern = r'img\_url=([^&]+)&text='

    match = re.search(pattern, url)

    if match:

        img\_url\_encoded = match.group(1)

        img\_url\_decoded = img\_url\_encoded.replace('%2F', '/').replace('%3A', ':')

        return img\_url\_decoded

    else:

        print('Ссылка после img\_url не найдена в URL')

Функция для расчета количества страниц HTML:

def calc\_pages(num\_images):

    return num\_images // 30 + (num\_images % 30 > 0) if num\_images > 30 else 1

Функция скачивания картинки:

def download\_image(url, save\_path):

    try:

        response = requests.get(url, headers={'User-Agent':'Mozilla/5.0'}, stream=True)

        if response.status\_code == 200:

            with open(save\_path, 'wb') as file:

                for chunk in response.iter\_content(1024):

                    file.write(chunk)

            return True

        else:

            print(f'Не удалось загрузить изображение: {url}')

            return False

    except Exception as e:

        print(f'Ошибка при загрузке изображения: {url}')

        return False

Функция парсинга HTML страницы Яндекс.Картинок:

def download\_images(query, num\_images, mini\_images = False):

    max\_pages = calc\_pages(num\_images)

    class\_folder = check\_repo\_dataset(query)

    downloaded\_count = 0

    base\_url = 'https:'

    #а вот это чтобы без движков было, грузим странички

    for page in range(0, max\_pages):

        search\_url = f'https://yandex.ru/images/search?text={query}&p={page}'

        response = requests.get(search\_url, headers={'User-Agent':'Mozilla/5.0'})

        soup = BeautifulSoup(response.text, 'html.parser')

        if (mini\_images == False):

            for a in soup.find\_all('a', class\_='serp-item\_\_link'):

                img\_url = a['href']

                # получаем полный URL изображения

                img\_url = parser\_url(img\_url)

                image\_filename = f"{downloaded\_count:04d}.jpg"

                image\_path = os.path.join(class\_folder, image\_filename)

                if(download\_image(img\_url, image\_path)):

                    downloaded\_count += 1

                    print(f"Загружено изображений для {query}: {downloaded\_count}/{num\_images}")

                if(downloaded\_count >= num\_images):

                    break

        else:

            for a in soup.find\_all('img', class\_='serp-item\_\_thumb'):

                img\_url = a['src']

                # из за получение //avatar, надо бы добавить https:// чтобы ссылка стала полной

                if(not img\_url.startswith('http')):

                    img\_url = base\_url + img\_url

                    image\_filename = f'{downloaded\_count:04d}.jpg'

                    image\_path = os.path.join(class\_folder, image\_filename)

                    if(download\_image(img\_url, image\_path)):

                        downloaded\_count += 1

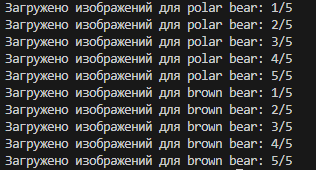
                        print(f'Загружено изображений для {query}: {downloaded\_count}/{num\_images}')

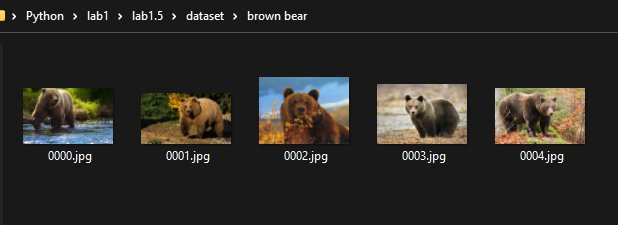
                    if(downloaded\_count >= num\_images):

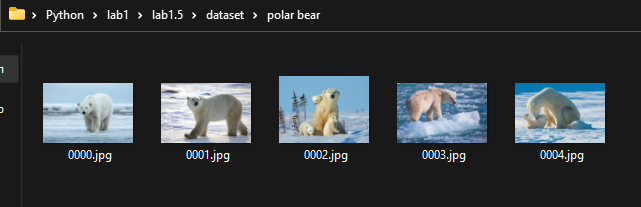
                        break

**Тестирование:**

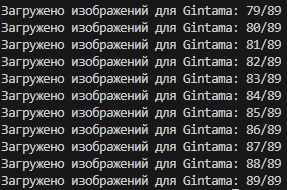
Тест №1. Миниатюры для ‘polar bear’ и ‘brown bear’, 5 штук

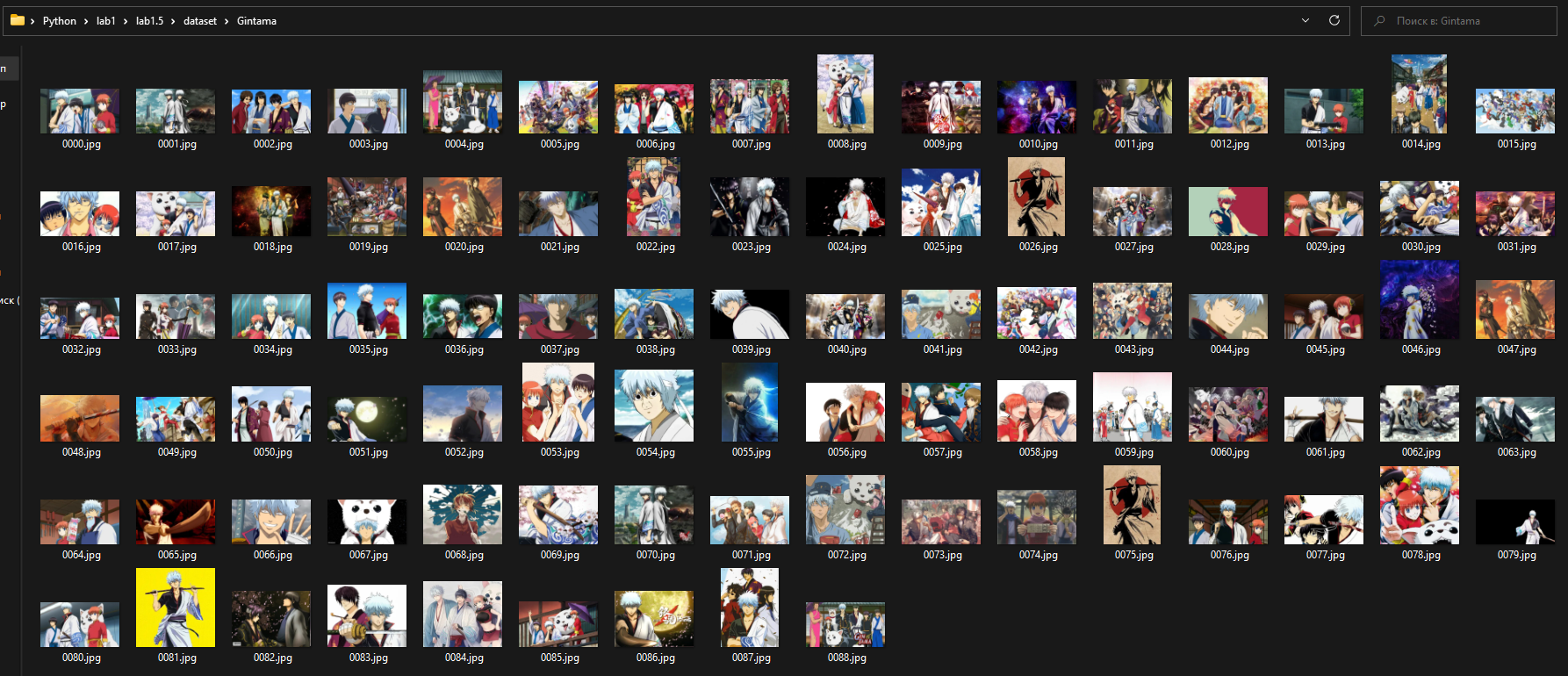
****

****

****

Тест №2. Миниатюры для ‘Gintama’, 89 штук

****

****

**Вариант №10. Текстовые данные (50% выполнена, без рефакторинга)**

**Постановка задачи:** 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.

**Код программы (Python):**

Библиотеки:

import os

import requests

import re

from bs4 import BeautifulSoup

Константы:

CURR\_DIR = os.path.dirname(os.path.abspath(\_\_file\_\_))

Функция расчета страниц для рецензий:

def calc\_pages(num\_reviews):

    return num\_reviews // 25 + (num\_reviews % 25 > 0) if num\_reviews > 25 else 2

Функция парсинга строки рейтинга рецензии:

def parse\_rating(str):

    str = re.search(r'\d+', str)

    if str:

        number = str.group()

        return number

    else:

        return 0

Функция парсинга HTML страницы и сохранения рецензии:

def download\_reviews(num\_reviews, full\_mode = False, pages = 0):

    if(pages == 0):

        pages = calc\_pages(num\_reviews)

    for rate in range(1, 5 + 1):

        downloaded\_count = 0

        rate\_folder = check\_repo\_dataset(str(rate))

        for page in range(2, pages + 1):

            search\_url = f'https://www.livelib.ru/reviews/~{page}#reviews'

            response = requests.get(search\_url, headers={'User-Agent':'Mozilla/5.0'})

            response.encoding = 'utf-8' #чтобы были русские символы а то кряки будут без форса кодировки

            soup = BeautifulSoup(response.text, 'html.parser')

            if(response.status\_code == 200):

                if(full\_mode == False):

                    for review in soup.find\_all('div', class\_='lenta-card'):

                        rating\_tag = review.find('span', class\_='lenta-card\_\_mymark')

                        #title\_tag = review.find('h3', class\_='lenta-card\_\_title')

                        #link = title\_tag.find('a')['href']

                        #title = title\_tag.find('a').text

                        if rating\_tag: #нехорошие люди не ставят рейтинг книги

                            rating = rating\_tag.text #не забыть про проверку рейтинга а то люди некоторые не ставят цифру

                            rating = parse\_rating(rating)

                            if(int(rating) == rate):

                                title\_book\_tag = review.find('a', class\_='lenta-card\_\_book-title')

                                title\_book = title\_book\_tag.text

                                text\_escaped\_tag = review.find('div', id='lenta-card\_\_text-review-escaped')

                                text = text\_escaped\_tag.text

                                review\_filename = f'{downloaded\_count:04d}.txt'

                                review\_path = os.path.join(rate\_folder, review\_filename)

                                with open(review\_path, "w", encoding="utf-8") as file: #кодировку для неизвестных символов charmap

                                    file.write(title\_book + "\n")

                                    file.write(text)

                                downloaded\_count += 1

                                print(f"Загружено рецензий для {rate}: {downloaded\_count}/{num\_reviews}")

                        if(downloaded\_count >= num\_reviews):

                            break

                else:

                    print('test')

                if(downloaded\_count >= num\_reviews):

                    break

**Тестирование:**

