Universitatea Tehnică a Moldovei

Facultatea Calculatoare Informatică şi Microelectronică

Departamentul Ingineria Software și Automatică

**RAPORT**

Lucrarea de laborator nr. 5

# La disciplina „Programarea Declarativa”

Tema: "Colectarea automată a datelor”

A efectuat: st. gr. SI-211 A. Chihai

A verificat: V. Rusu

**Chișinău – 2023**

1. Alegeți un subiect ce vă interesează pe *wikipedia.org* și îndepliniți următoarele sarcini:

* capturați titlul paginii;

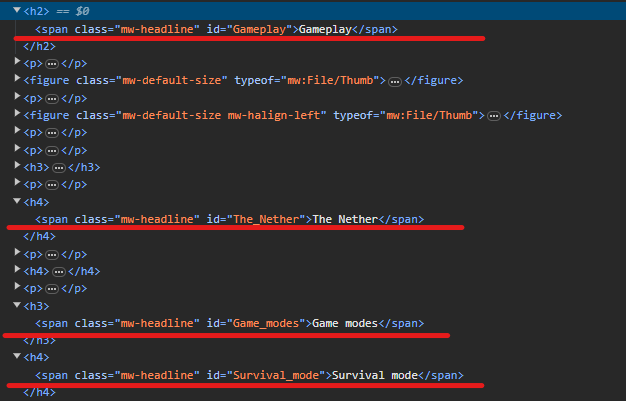


Observăm că titlul paginii se afla în interiorul tag-ului <title>

def wiki\_title(url):  
 res = requests.get(url)  
 soup = BeautifulSoup(res.text, "html.parser")  
  
 title = soup.find('title').text  
 print(f"Title:\n{title}\n")

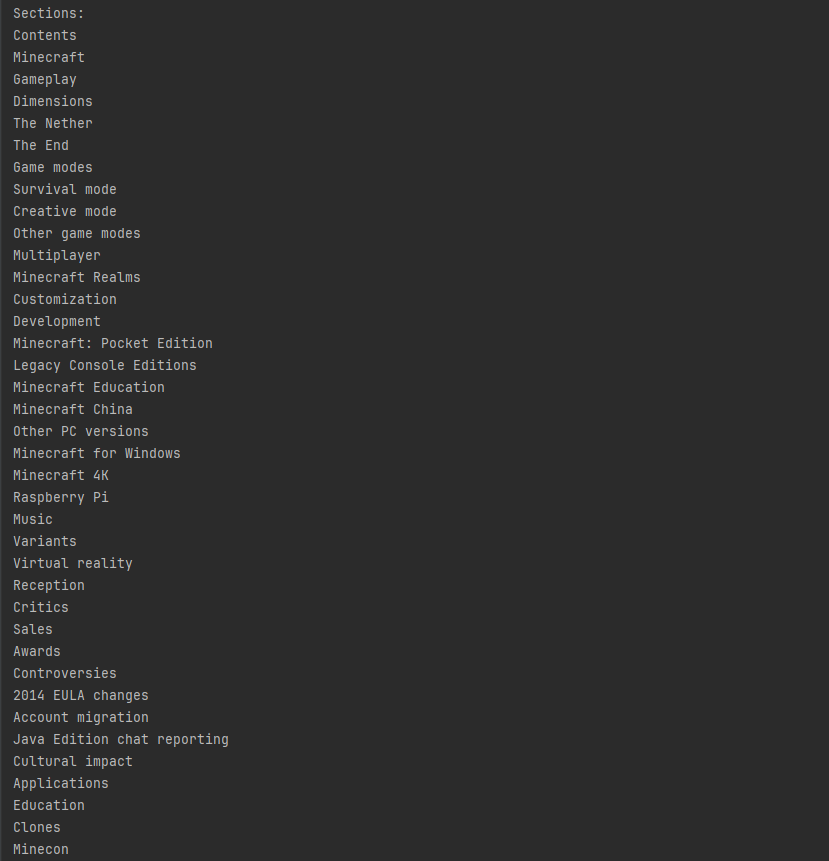


* capturați toate titlurile secțiunilor;

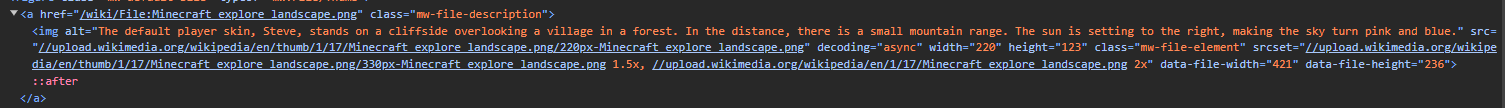


Titlurile secțiunilor le putem găsi ușor deoarece sunt cu tagul <h...>

def wiki\_sections(url):  
 res = requests.get(url)  
 soup = BeautifulSoup(res.text, 'html.parser')  
 sections = soup.find\_all(['h1', 'h2', 'h3', 'h4', 'h5', 'h6'])  
 print(f"Sections")  
 for section in sections:  
 print(f"{section.text.strip()}")

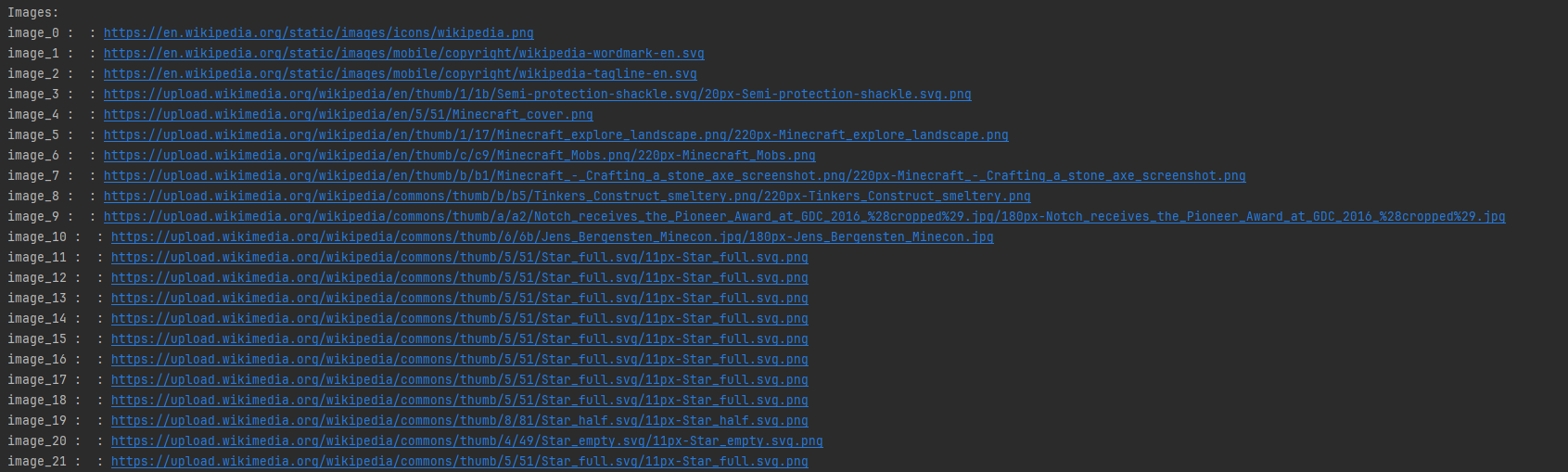


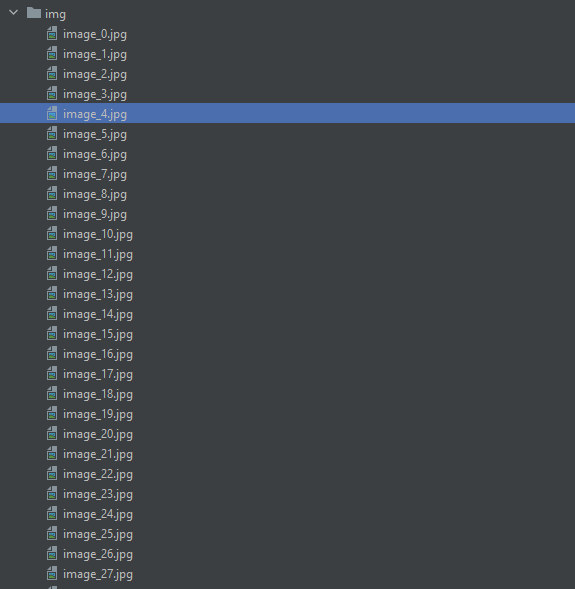
* obțineți minim o imagine de pe acel site.

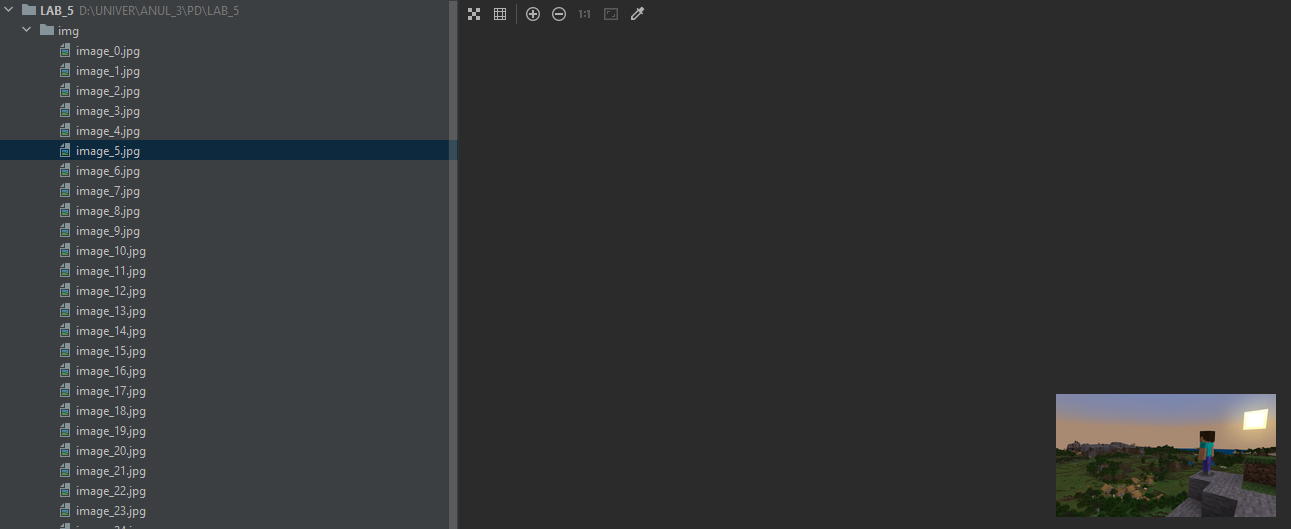


Imaginile le putem găsi cu ajutorul tag-ului <img> ‚ iar mai apoi le putem descărca direct de pe linkul atribuit lui src

def wiki\_photo(url):  
 res = requests.get(url)  
 soup = BeautifulSoup(res.text, 'html.parser')  
 images = soup.find\_all('img')  
 save\_dir = 'img'  
  
 counter = 0;  
 for image in images:  
 name = f"image\_{counter} : "  
 link = image['src']  
 absolute\_url = urljoin(url, link)  
 print(f"{name} : {absolute\_url}")  
 file\_name = f'image\_{counter}.jpg'  
 file\_path = os.path.join(save\_dir, file\_name)  
 with open(file\_path, 'wb') as f:  
 im = requests.get(absolute\_url)  
 f.write(im.content)  
 counter += 1







1. Accesați site-ul web: *http://books.toscrape.com/index.html* care este conceput special pentru testarea web scraping. Obțineți titlul fiecărei cărți care are o evaluare de 2 stele și, la sfârșit, să aveți doar o listă Python cu toate titlurile lor.

* **găsiți structura URL-ului pentru a parcurge fiecare pagină;**

Pentru a putea parcurge fiecare pagină vom avea nevoie de un URL care se va modifica conform normelor de căutare astfel putem folosi acest URL al catalogului

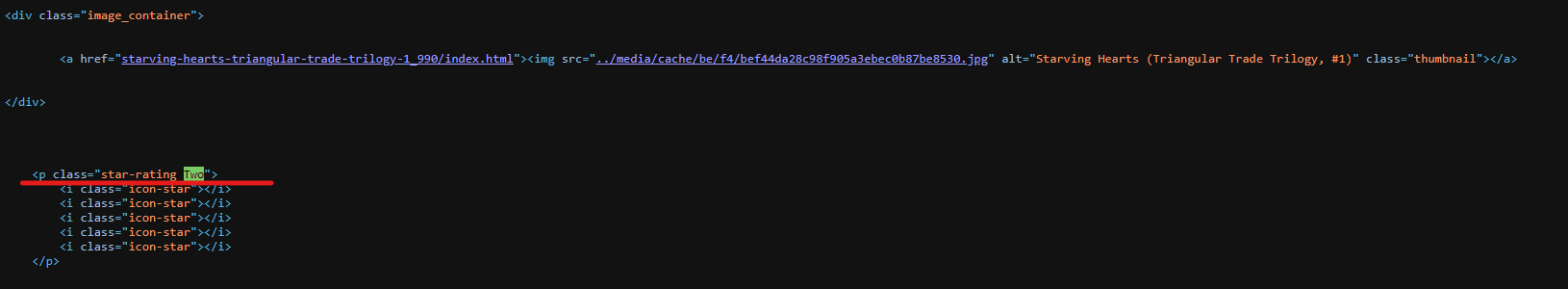
**<http://books.toscrape.com/catalogue/page-{page_nr}.html>**

* **parsați fiecare pagină din catalog;**

Astfel după ce am descoperit structura URL care ne va permite sa iterăm paginile vom putea folosi o variabila care se va incrementa după fiecare ciclu astfel atât timp cât request code-ul va fi 200 noi vom putea cerceta această pagină și selecta cărțile

* **găsiți ce etichetă/clasă reprezintă evaluarea cu stele;**

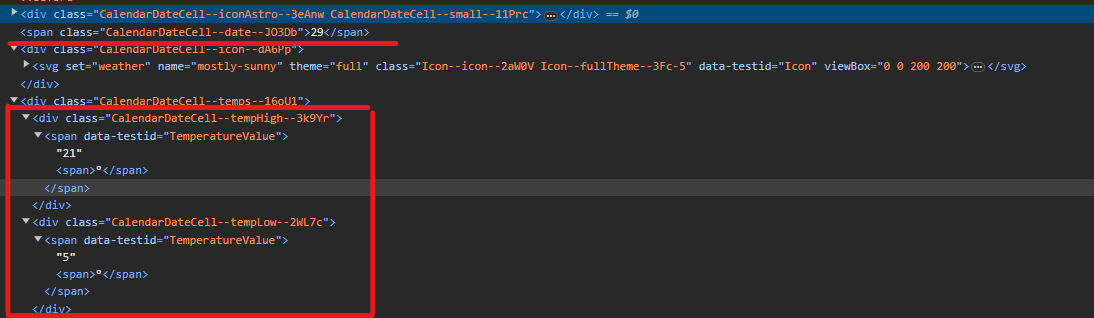
Clasa ce reprezintă valoarea cu stele este “star-rating nr\_of\_stars”

****

* **filtrați cu *if* evaluarea cu stele;**
* **stocați rezultatele într-o listă.**

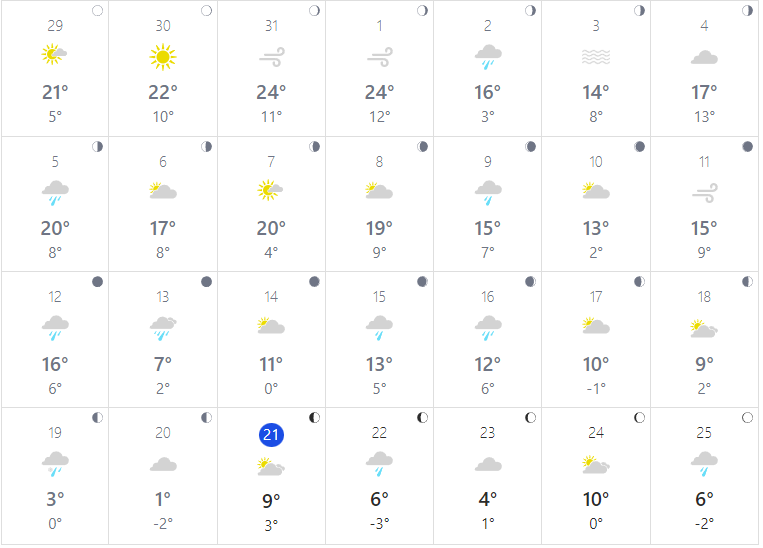
def find\_rating(default\_url):  
 books = []  
 page\_nr = 1  
  
 while True:  
 url = f"{default\_url}/catalogue/page-{page\_nr}.html"  
 res = requests.get(url)  
  
 if res.status\_code == 200:  
 soup = BeautifulSoup(res.text, 'html.parser')  
  
 book\_items = soup.find\_all('article', class\_='product\_pod')  
  
 for book\_item in book\_items:  
 title\_element = book\_item.find('h3')  
 if title\_element:  
 title = title\_element.a['title']  
  
 rating\_element = book\_item.find('p', class\_='star-rating Two')  
  
 if rating\_element:  
 books.append(title)  
  
 page\_nr += 1  
 else:  
 print(f"Can't go to page {page\_nr}\n")  
 break;  
  
 return books  
  
  
url = "http://books.toscrape.com"  
books = find\_rating(url)  
  
print(f"{len(books)} have been founded")  
for book in books:  
 print(book)

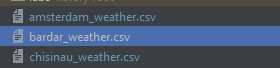
1. Faceți cereri către minim 3 site-uri pentru a obține informația dorită (la alegere: date meteo, curs valutar, preț produse, rating hotele, restaurante etc.). Ca rezultat trebuie să aveți 3 fișiere .csv cu date similare, de ex. date meteo pe luna octombrie de la 3 site-uri de prognoză meteo, cursul valutar pentru o lună de la 3 bănci diferite, ratingul la o listă de hotele de la 3 site-uri diferite etc.

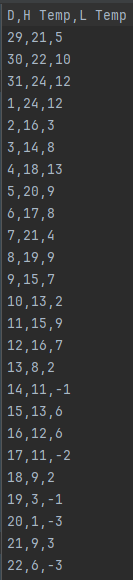
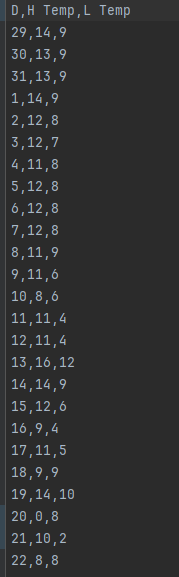
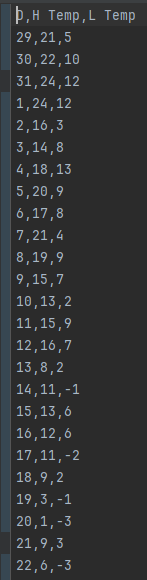


Cu ajutorul acestor clase am reușit să obțin informații despre dată, temperatura minimă și maximă

import requests  
from bs4 import BeautifulSoup  
import csv  
  
  
def fahrenheit\_to\_celsius(fahrenheit):  
 if '--' in fahrenheit:  
 return 0  
 if '°' in fahrenheit:  
 fahrenheit = fahrenheit.replace('°', '')  
 return round((float(fahrenheit) - 32) \* 5 / 9)  
  
  
def get\_weather(url, csv\_name):  
 response = requests.get(url)  
  
 if response.status\_code == 200:  
 soup = BeautifulSoup(response.text, 'html.parser')  
  
 buttons = soup.find\_all('button', class\_='Button--default--2gfm1')  
  
 with open(f'{csv\_name}.csv', 'w', newline='') as csvfile:  
 fieldnames = ['D', 'H Temp', 'L Temp']  
 csv\_writer = csv.DictWriter(csvfile, fieldnames=fieldnames)  
  
 csv\_writer.writeheader()  
  
 for button in buttons:  
 date\_element = button.find('span', class\_='CalendarDateCell--date--JO3Db')  
 if date\_element is None:  
 continue  
  
 date = date\_element.text.strip()  
  
 high\_temp\_element = button.find('div', class\_='CalendarDateCell--tempHigh--3k9Yr')  
 low\_temp\_element = button.find('div', class\_='CalendarDateCell--tempLow--2WL7c')  
  
 if high\_temp\_element is None or low\_temp\_element is None:  
 continue  
  
 high\_temp\_str = high\_temp\_element.text.strip()  
 low\_temp\_str = low\_temp\_element.text.strip()  
  
 high\_temp\_c = fahrenheit\_to\_celsius(high\_temp\_str)  
 low\_temp\_c = fahrenheit\_to\_celsius(low\_temp\_str)  
  
 csv\_writer.writerow(  
 {'D': date, 'H Temp': high\_temp\_c, 'L Temp': low\_temp\_c})  
  
 print(f"Date: {date}")  
 print(f"High Temperature: {high\_temp\_c}°C")  
 print(f"Low Temperature: {low\_temp\_c}°C")  
 print("\n")  
  
 print("Weather data saved to weather\_data.csv")  
  
 else:  
 print(f"Failed to retrieve the page. Status code: {response.status\_code}")  
  
  
url\_bardar = "https://weather.com/weather/monthly/l/c0150b8b0b198f915a47db72ba5d723deab5dfeedec7f099aed7f602e2f8d06d"  
get\_weather(url\_bardar, 'bardar\_weather')  
  
url\_chisinau = "https://weather.com/weather/monthly/l/c0150b8b0b198f915a47db72ba5d723deab5dfeedec7f099aed7f602e2f8d06d"  
get\_weather(url\_chisinau, 'chisinau\_weather')  
  
url\_amsterdam = "https://weather.com/weather/monthly/l/968d2f1a5509a2f71fca25929b7d83139ac5134f61611a9c6637c90354cd6da8"  
get\_weather(url\_amsterdam, 'amsterdam\_weather')







**Concluzie:**

În cadrul acestei lucrări de laborator, am explorat tehnici de web scraping utilizând Python și BeautifulSoup. Am capturat informații precum titlul paginii și secțiunile de pe Wikipedia, precum și imagini. De asemenea, am dezvoltat un script pentru extragerea titlurilor cărților cu evaluare de 2 stele de pe un site dedicat testării web scraping-ului. În paralel, am efectuat cereri către trei site-uri distincte pentru a obține date meteo, stocând rezultatele în fișiere CSV. Această experiență subliniază utilitatea și versatilitatea web scraping-ului în colectarea automată a datelor de pe diverse platforme online.