|  |  |  |
| --- | --- | --- |
| ДонГТУ | Методы анализа данных | СКС-23 |
| Кафедра СКС | Лабораторная работа №3 | Кукарин А.А. |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Выполнение | | | | Защита | | | |
| Подпись  студента | Фамилия  преподавателя | Дата  выполн. | Подпись  препод. | Фамилия  преподавателя | Оценка | Дата  защиты | Подпись  препод. |
|  | Самойлов Д.В. |  |  | Самойлов Д.В. |  |  |  |

**Тема работы:** : Извлечение данных из web-ресурсов.

**Цель работы:** научиться использовать библиотеки Python для извлечения данных из web-реусурсов.

Задание к лабораторной работе:

1. Выберите на сайте объявлений интересующую вас рубрику самого нижнего уровня (например, Для дома и дачи – Мебель и интерьер – Кровати, диваны и кресла – Диваны).

2. Убедитесь в том, что сайт учел в выдаче ваше местоположение (если нет – укажите его).

3. Обратите внимание на адрес полученной вами страницы и на то, как он изменяется при переходе к следующей странице выдачи (где и как указывается номер страницы), зафиксируйте это.

4. Откройте код страницы выдачи (F12 или правой кнопкой мыши). Выберите ключевые элементы блока записи об одном объявлении так, чтобы вы могли выделить этот блок из всего текста: например, это может быть определенный тег, с определенным набором атрибутов.

5. Напишите скрипт, который перебирает все страницы выдачи, и собирает в список следующие параметры: id объявления, название, цена, срок публикации.

6. Проверьте полученный список на корректность – в нем не должно быть посторонних записей и выбросов (слишком малых, слишком больших и отсутствующих цен). При необходимости, устраните ненужные записи программно.

7. Отфильтруйте полученный список по заданному вами критерию: например, оставьте только цены между заданными вами значениями, или оставьте только объявления, находящие в просмотре не больше суток.

8. Сохраните результат вашей работы так, чтобы он был пригоден для машинного анализа.

Исходный код:

import time

import random

import pandas as pd

from bs4 import BeautifulSoup

from selenium import webdriver

from selenium.webdriver.edge.service import Service

from selenium.webdriver.edge.options import Options

from webdriver\_manager.microsoft import EdgeChromiumDriverManager

# ====================

# НАСТРОЙКИ

# ====================

MAX\_PAGES = 3 # Кол-во страниц для парсинга

MIN\_PRICE = 500000 # Минимальная цена (фильтр)

MAX\_PRICE = 5000000 # Максимальная цена (фильтр)

DELAY\_RANGE = (3, 7) # Случайная задержка между запросами

USER\_AGENT = "Chromium";v="92", "Microsoft Edge";v="92", "Placeholder;Browser Brand";v="99"

# ====================

# ФУНКЦИЯ ЗАГРУЗКИ HTML

# ====================

def get\_html\_selenium(url):

options = Options()

options.add\_argument(f"user-agent={USER\_AGENT}")

options.add\_argument("--disable-blink-features=AutomationControlled")

options.add\_experimental\_option("excludeSwitches", ["enable-automation"])

options.add\_argument("--headless")

options.add\_argument("--window-size=1920,1080")

driver = webdriver.Edge(

service=Service(EdgeChromiumDriverManager().install()),

options=options

)

# Маскировка WebDriver

driver.execute\_cdp\_cmd("Page.addScriptToEvaluateOnNewDocument", {

"source": """

Object.defineProperty(navigator, 'webdriver', { get: () => undefined });

window.chrome = { runtime: {} };

"""

})

try:

driver.get(url)

time.sleep(random.uniform(\*DELAY\_RANGE))

return driver.page\_source

# Скролл страницы

for \_ in range(2):

driver.execute\_script("window.scrollTo(0, document.body.scrollHeight \* 0.8);")

time.sleep(random.uniform(0.5, 1.5))

return driver.page\_source

except Exception as e:

print(f"Ошибка при загрузке страницы: {str(e)[:200]}")

return None

finally:

driver.quit()

# ====================

# ФУНКЦИЯ ПАРСИНГА

# ====================

def parse\_page(html):

soup = BeautifulSoup(html, "html.parser")

ads = []

for item in soup.find\_all("div", class\_="ListingItem"):

try:

# Название

title\_tag = item.find("a", class\_="Link ListingItemTitle\_\_link")

title = title\_tag.text.strip() if title\_tag else "Название не указано"

# Цена

price\_tag = item.find("div", class\_="ListingItemPrice\_\_content")

price = price\_tag.text.strip() if price\_tag else "0"

price = int(''.join([c for c in price if c.isdigit()])) if price else 0

# Фильтр по цене

if not (MIN\_PRICE <= price <= MAX\_PRICE):

continue

# Характеристики

year = mileage = "Нет данных"

details\_tag = item.find("div", class\_="ListingItemTechSummaryDesktop\_\_column")

if details\_tag:

details = details\_tag.find\_all("div")

year = details[0].text.strip() if len(details) > 0 else year

mileage = details[1].text.strip() if len(details) > 1 else mileage

# Ссылка

link\_tag = item.find("a", class\_="Link ListingItemTitle\_\_link")

link = link\_tag.get("href", "#") if link\_tag else "#"

if link.startswith("/"):

link = f"https://auto.ru{link}"

# Дата

date\_tag = item.find("div", class\_="ListingItem\_\_date")

date = date\_tag.text.strip() if date\_tag else "Дата не указана"

ads.append({

"Модель": title,

"Цена": f"{price:,}".replace(",", " "),

"Год": year,

"Пробег": mileage,

"Дата": date,

"Ссылка": link

})

except Exception as e:

print(f"Ошибка парсинга: {str(e)[:100]}")

continue

return ads

# ====================

# ОСНОВНАЯ ФУНКЦИЯ

# ====================

def main():

base\_url = "https://auto.ru/cars/bmw/all/"

results = []

for page in range(1, MAX\_PAGES + 1):

print(f"[+] Обработка страницы {page}/{MAX\_PAGES}")

url = f"{base\_url}?page={page}"

# Загрузка HTML

html = get\_html\_selenium(url)

if not html:

print(f"[!] Страница {page} не загружена")

continue

# Парсинг данных

ads = parse\_page(html)

if ads:

results.extend(ads)

print(f"Найдено объявлений: {len(ads)}")

else:

print("[!] Объявления не найдены")

# Случайная задержка

delay = random.randint(\*DELAY\_RANGE)

print(f"Пауза: {delay} сек.")

time.sleep(delay)

# Сохранение результатов

if results:

df = pd.DataFrame(results)

filename = "auto\_ru\_bmw.csv"

df.to\_csv(filename, index=False, encoding="utf-8-sig")

print(f"\n[✓] Данные сохранены в файл: {filename}")

print(f"Всего объявлений: {len(df)}")

else:

print("\n[!] Нет данных для сохранения")

if \_\_name\_\_ == "\_\_main\_\_":

main()

Результат:

