**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

Федеральное государственное бюджетное образовательное учреждение высшего образования

Воронежский государственный технический университет

(ФГБОУ ВО «ВГТУ», ВГТУ)

**Факультет** строительный

**Кафедра** [инноватики и строительной физики имени профессора И.С. Суровцева](https://cchgeu.ru/education/cafedras/kafiisf/)

**Отчет по лабораторной работе 1**

**По дисциплине**: Модели и методы анализа больших данных

**Тема**: «Парсинг сайтов»

Выполнила студент: Саввин Н.В.

Группа: мТИИ-211

Руководитель: к. ф.-м. н,

Дробышев А.А.

Работа защищена « » \_\_\_\_\_ 2022 г.

С оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_

(подпись)

Воронеж 2022

# Задание

Получить данные с тестового интернет-магазина https://parsemachine.com.

# Выполнение работы

Полный код программы приведен в Приложении 1.

Парсинг используется для анализа конкурентов, исследования рынка, поиска и устранения ошибок на собственной площадке, создания контента. Интернет-магазины используют его, чтобы переводить описания товаров с иностранных площадок. Парсинг вполне законен, если вы собираете информацию, которая есть в открытом доступе.

Используемые библиотеки:

Beautiful Soup — это библиотека Python для извлечения данных из файлов HTML и XML.

Используется драйвер undetected-chromedriver для Selenium. Он содержит в себе алгоритмы, позволяющие скрывать средства автоматизации браузера, тем самым становясь менее подозрительным для веб-приложения и всевозможных WAF. Selenium WebDriver – это программная библиотека для управления браузерами. WebDriver представляет собой драйверы для различных браузеров и клиентские библиотеки на разных языках программирования, предназначенные для управления этими драйверами.  
По сути своей использование такого веб-драйвера сводится к созданию бота, выполняющего всю ручную работу с браузером автоматизированно.

import json для работы с файлами json.

после парсинга данные хранятся в файле json. Стандартный текстовый формат для представления структурированных данных на основе синтаксиса объекта JavaScript. Он обычно используется для передачи данных в веб-приложениях (например, отправка некоторых данных с сервера клиенту,таким образом чтобы это могло отображаться на веб-странице или наоборот).

В Python есть модуль time, который используется для решения задач, связанных со временем. Для использования определенных в нем функций необходимо сначала его импортировать:import time.

На рисунке 1 представлен сайт для парсинга.

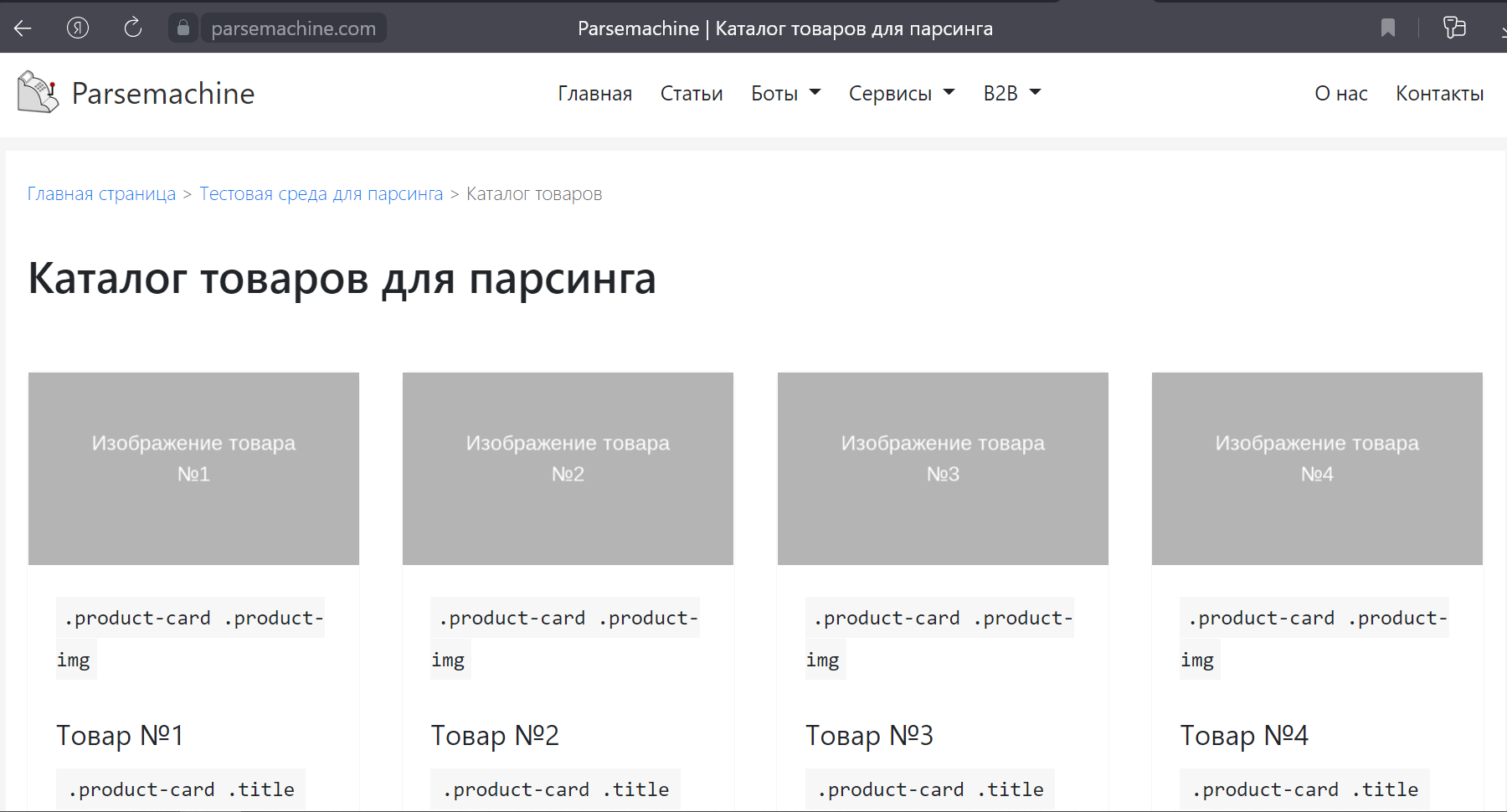


Рисунок 1 – Сайт для парсинга

При запуске отладки программы происходит открытие браузера Chrome и переход на страницу интернет магазина. После чего считывается информация о бренде, названии, цене и ссылка, записывается в массив. После чего создается файл json и записывается массив – рисунок 2.

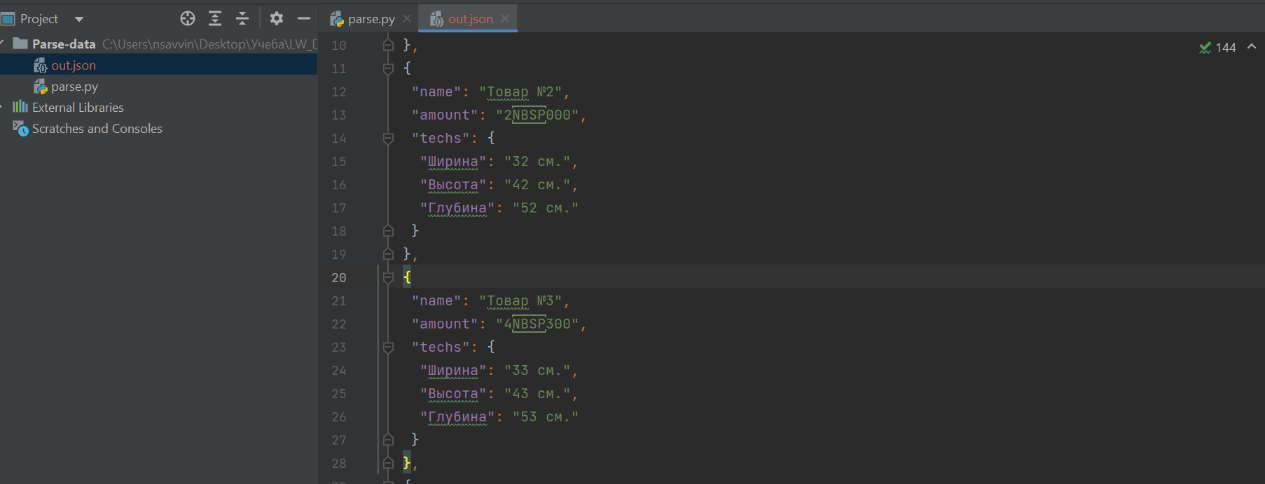


Рисунок 2 – Результат парсинга

Приложение 1

import json  
  
import requests  
from bs4 import BeautifulSoup  
  
  
PAGES\_COUNT = 3  
OUT\_FILENAME = 'out.json'  
  
  
def get\_soup(url, \*\*kwargs):  
 response = requests.get(url, \*\*kwargs)  
 if response.status\_code == 200:  
 soup = BeautifulSoup(response.text, features='html.parser')  
 else:  
 soup = None  
 return soup  
  
  
def crawl\_products(pages\_count):  
 urls = []  
 fmt = 'https://parsemachine.com/sandbox/catalog/?page={page}'  
  
 for page\_n in range(1, 1 + pages\_count):  
 print('page: {}'.format(page\_n))  
  
 page\_url = fmt.format(page=page\_n)  
 soup = get\_soup(page\_url)  
 if soup is None:  
 break  
  
 for tag in soup.select('.product-card .title'):  
 href = tag.attrs['href']  
 url = 'https://parsemachine.com{}'.format(href)  
 urls.append(url)  
  
 return urls  
  
  
def parse\_products(urls):  
 data = []  
  
 for url in urls:  
 print('product: {}'.format(url))  
  
 soup = get\_soup(url)  
 if soup is None:  
 break  
  
 name = soup.select\_one('#product\_name').text.strip()  
 amount = soup.select\_one('#product\_amount').text.strip()  
 techs = {}  
 for row in soup.select('#characteristics tbody tr'):  
 cols = row.select('td')  
 cols = [c.text.strip() for c in cols]  
 techs[cols[0]] = cols[1]  
  
 item = {  
 'name': name,  
 'amount': amount,  
 'techs': techs,  
 }  
 data.append(item)  
  
 return data  
  
  
def main():  
 urls = crawl\_products(PAGES\_COUNT)  
 data = parse\_products(urls)  
  
 with open(OUT\_FILENAME, 'w') as f:  
 json.dump(data, f, ensure\_ascii=False, indent=1)  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 main()