

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

імені ІГОРЯ СІКОРСЬКОГО”

Факультет прикладної математики

Кафедра програмного забезпечення комп’ютерних систем

**Лабораторна робота №** **1**

з дисципліни “Бази даних. Частина 2”

|  |  |  |  |
| --- | --- | --- | --- |
| Виконав  студент III курсу  групи КП-82  Мельничук Олексій Геннадійович  (*прізвище, ім’я, по батькові*)  варіант № 12 |  |  | Зарахована  “\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 20\_\_\_ р.  викладачем  Петрашенко Андрій Васильович (*прізвище, ім’я, по батькові*) |

Київ 2021

**Мета роботи**

Здобуття практичних навичок створення програм, орієнтованих на обробку XML-документів засобами мови Python.

**Завдання**

1. На основі базової адреси [www.uartlib.org](http://www.uartlib.org) виконати обхід наявних сторінок сайту, відокремлюючи текстову та графічну інформацію від тегів HTML. Пошук вузлів виконувати засобами XPath. Наступну сторінку для аналізу цього ж сайту обрати як одне із гіперпосилань на даній сторінці (тег <a href=”url”/>). Обмежитись аналізом 20 сторінок сайту. Зберегти XML у вигляді файлу. Формат XML-документу:

<data>

<page url=”wwww.server.com/index.hml”>

<fragment type=”text”>

…. знайдений текст

</fragment>

<fragment type=”image”>

…. url зображення

</fragment>

</page>

<page url=”wwww.server.com/index1.hml”>

<fragment type=”text”>

…. знайдений текст

</fragment>

<fragment type=”image”>

…. url зображення

</fragment>

</page>

…

</data>

1. Вивести середню кількість текстових фрагментів на сторінку на ресурсі.
2. Проаналізувати вміст [www.hozmart.com.ua](http://www.hozmart.com.ua). Отримати ціну, опис та зображення для 20 товарів з нього за допомогою DOM-парсеру та мови XPath для пошуку відповідних вузлів. Результат записати в XML-файл.
3. Перетворити отриманий XML-файл у XHTML-сторінку за допомогою мови XSLT. Дані подати у вигляді XHTML-таблиці та записати його у файл.

**Код програми**

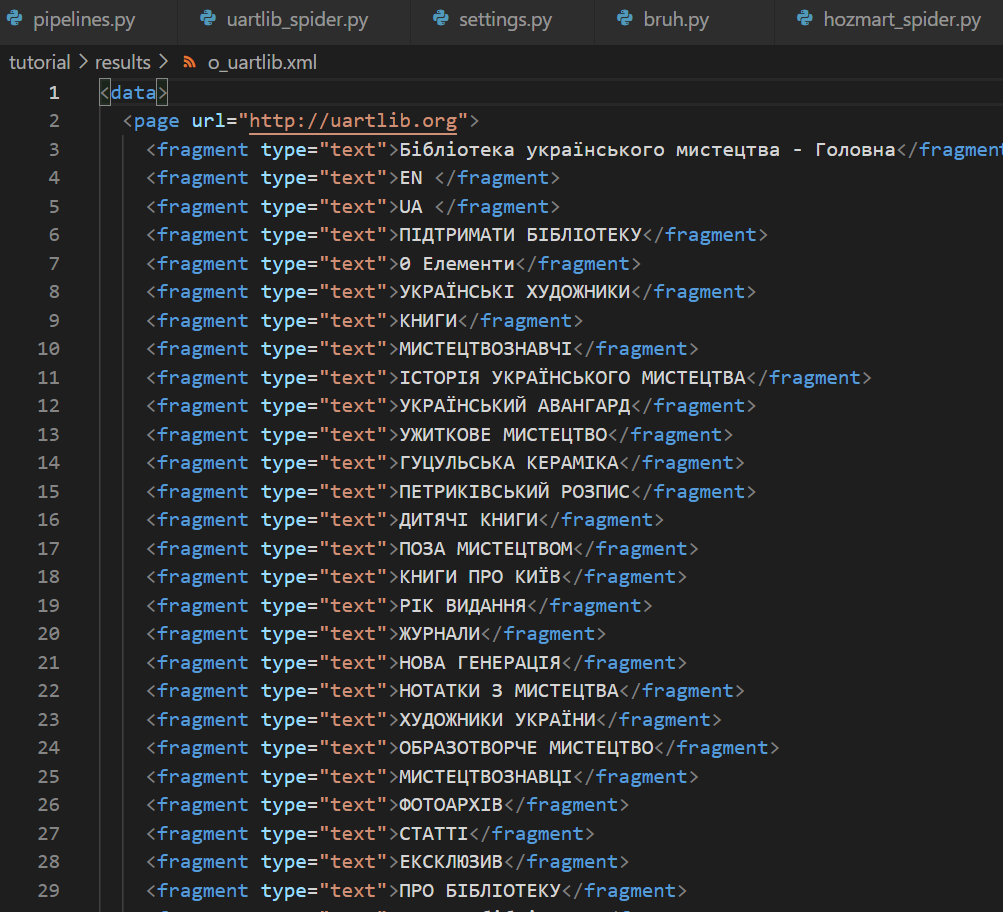
|  |
| --- |
| **Main.py** |
| import os  from scrapy import cmdline  from lxml import etree  mainpath = os.path.dirname(\_\_file\_\_) + '/tutorial'  root = None  # cmdline.execute("scrapy crawl uartlib".split())  # result = open(mainpath + '/results/o\_uartlib.xml', 'rb')  # root = etree.parse(result)  # result.close()  # pageCount = root.xpath('count(//page)')  # textCount = root.xpath('count(//fragment[@type="text"])')  # result = 'Average count of text fragments per page: %f' % (textCount / pageCount)  # f = open(mainpath + '/results/l\_artlib.txt', 'w')  # f.write(result)  # f.close()  def task\_1\_parse():  cmdline.execute("scrapy crawl uartlib".split())  def task\_1\_processing():  result = open(mainpath + '/results/o\_uartlib.xml', 'rb')  root = etree.parse(result)  result.close()  pageCount = root.xpath('count(//page)')  textCount = root.xpath('count(//fragment[@type="text"])')  result = 'Average count of text fragments per page: %f' % (textCount / pageCount)  f = open(mainpath + '/results/r\_uartlib.txt', 'w')  f.write(result)  f.close()  def task\_2\_parse():  cmdline.execute("scrapy crawl hozmart".split())  def task\_2\_processing():  dom = etree.parse(mainpath + '/results/o\_hozmart.xml')  xslt = etree.parse(mainpath + '/hozmart.xslt')  transform = etree.XSLT(xslt)  newdom = transform(dom)  with open(mainpath + '/results/r\_hozmart.html', 'wb') as f:  f.write(etree.tostring(newdom, pretty\_print=True))  task\_2\_processing() |

|  |
| --- |
| **Pipelines.py** |
| from lxml import etree  import os  class TutorialPipeline(object):  def open\_spider(self, spider):  self.root = etree.Element("data")  def close\_spider(self, spider):  f = open(os.path.dirname(\_\_file\_\_) + '/results/o\_' + spider.name + '.xml', 'wb')  f.write(etree.tostring(  self.root, encoding="UTF-8",  pretty\_print=True  ))  f.close()  def process\_item(self, item, spider):  if spider.name == "uartlib":  page = etree.SubElement(self.root, "page", url=item["url"])  for text in item['texts']:  etree.SubElement(page, 'fragment', type='text').text = text  for url in item['images']:  etree.SubElement(page, 'fragment', type='image').text = url  self.root.append(page)  else:  product: etree.Element = etree.Element("product")    name = etree.Element("name")  name.text = item["names"]  price = etree.Element("price")  price.text = item["prices"]  image = etree.Element("image")  image.text = item["images"]  avail = etree.Element("availability")  avail.text = item["avails"]  product.append(name)  product.append(price)  product.append(image)  product.append(avail)  self.root.append(product)  return item |

|  |
| --- |
| **Uartlib\_spyder.py** |
| import scrapy  class uartlibSpider(scrapy.Spider):  name = "uartlib"  start\_urls = [  'http://uartlib.org'  ]  selectors = {  'text': "//\*[not(self::script)][not(self::style)]//text()[normalize-space()][not(contains(.,'{'))][not(contains(.,';'))]",  'img': '//img/@src',  'url': "//a/@href[starts-with(., '" + start\_urls[0] + "')or starts-with(., '/')]"  }  def parse(self, response):  texts = response.xpath(self.selectors['text']).extract()  images = response.xpath(self.selectors['img']).extract()  urls = response.xpath(self.selectors['url']).extract()  yield {  'url': response.url,  'texts': texts,  'images': images  }  if response.url == self.start\_urls[0]:  links = [  link for link in urls if link != "/"  ]  for link in links[:19]:  if link.startswith("/"):  link = self.start\_urls[0] + link  yield response.follow(link, callback=self.parse) |

|  |
| --- |
| **hozmart\_spyder.py** |
| import scrapy  class hozmartSpider(scrapy.Spider):  name = "hozmart"  start\_urls = [  'https://hozmart.com.ua/uk/15-benzopili',  'https://hozmart.com.ua/uk/15-benzopili?p=2'  ]  selectors = {  'all\_items': "//ul[contains(@id, 'product\_list')]/li",  'names': ".//a[contains(@class, 'b1c-name-uk')]/text()",  'images': ".//img[contains(@class, 'b1c-img')]/@src",  'prices': ".//span[contains(@class, 'price')]/text()",  'available': ".//p[contains(@class, 'availability')]/span//text()[normalize-space()]"  }  def parse(self, response):  for result in response.xpath(self.selectors['all\_items'])[:20]:  yield {  'names': result.xpath(self.selectors['names']).extract\_first(),  'images': result.xpath(self.selectors['images']).extract\_first(),  'prices': result.xpath(self.selectors['prices']).extract\_first(),  'avails': result.xpath(self.selectors['available']).extract\_first().strip()  } |

**Результати роботи програм**

****

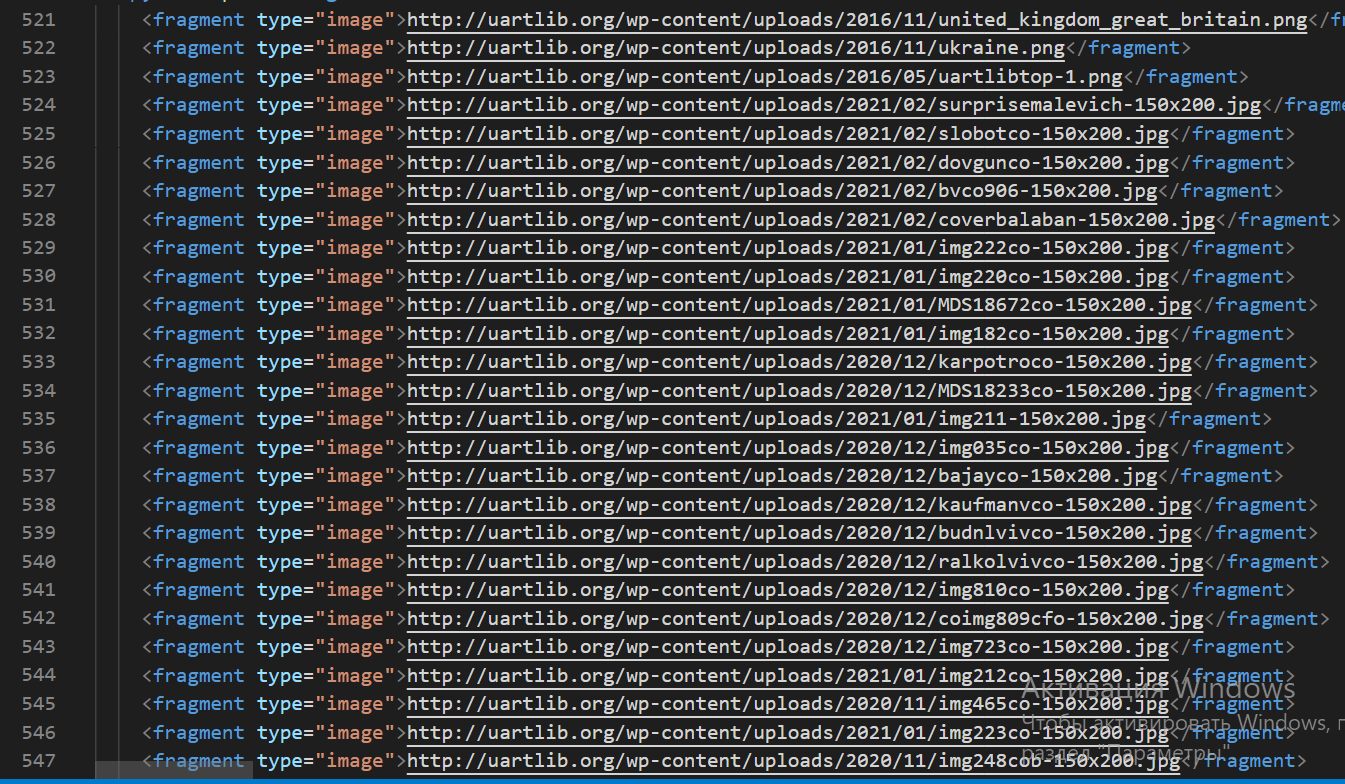
****

Рис.1.1-2. Скріншоти результатів завдання 1

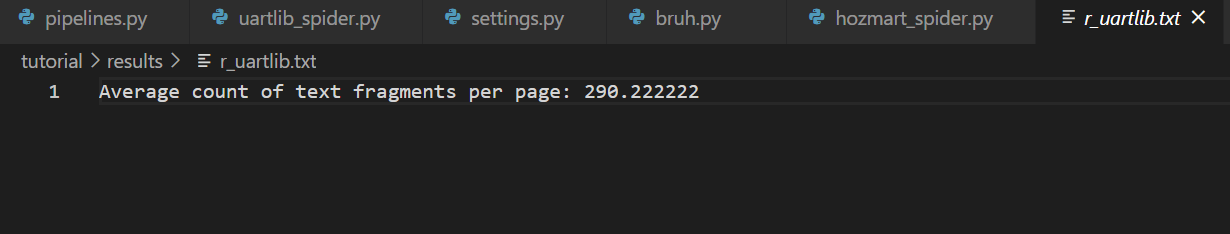


Рис.2. Скріншот результатів завдання 2

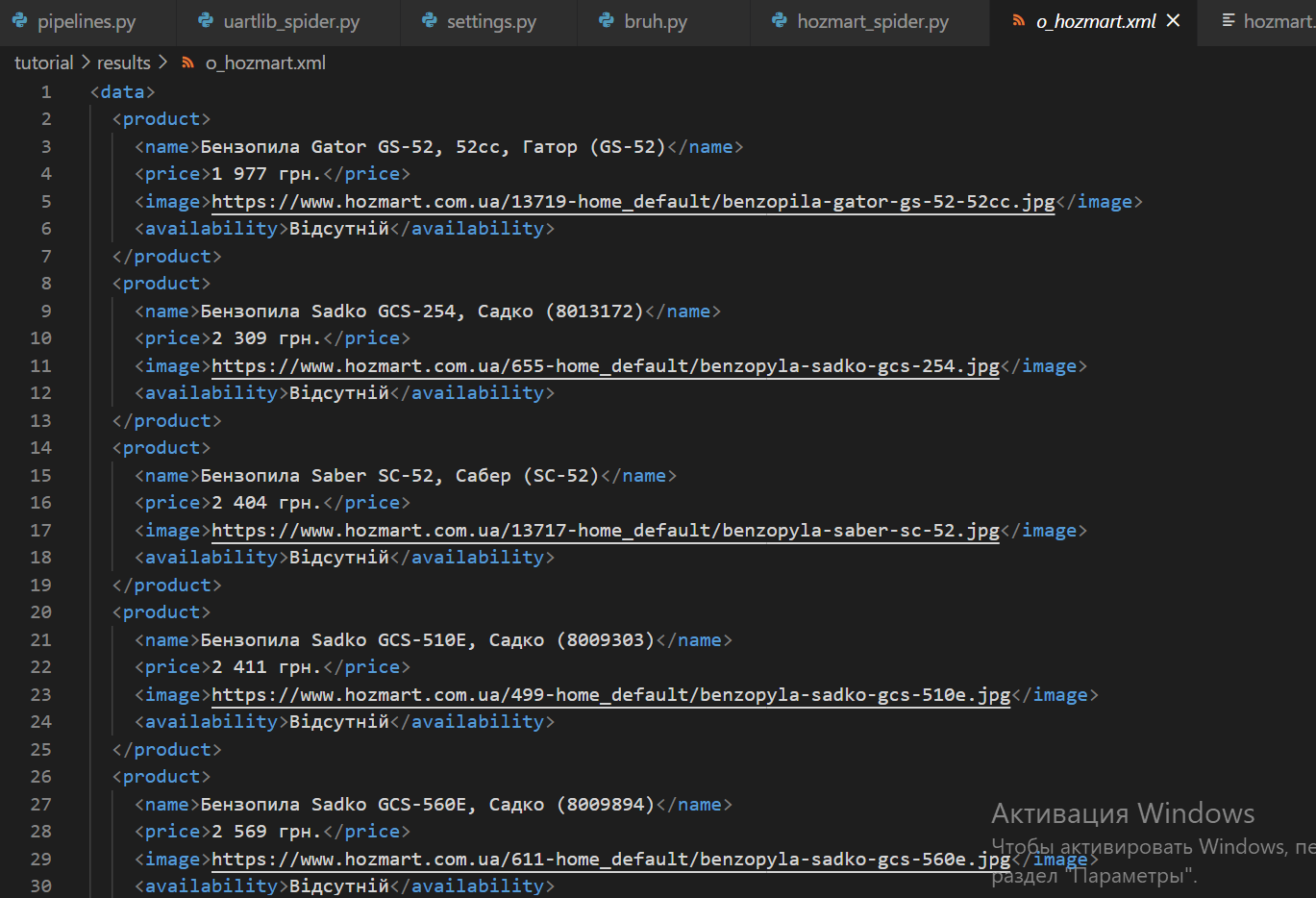
****

Рис.3. Скріншот результатів завдання 3

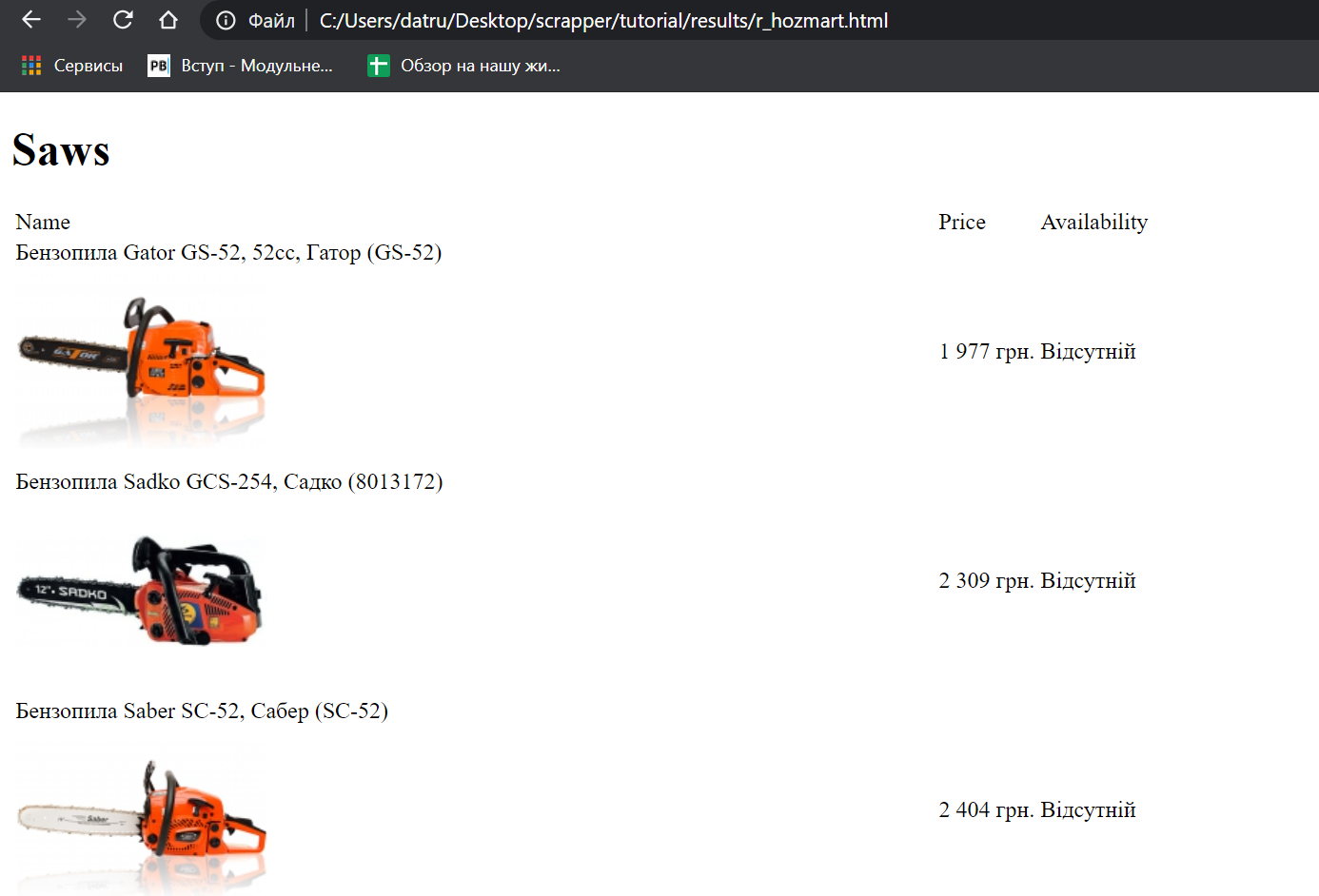


Рис.4.1. Скріншот результатів завдання 4

**Висновки**

Виконавши дану лабораторну роботу я ознайомився з синтаксисом та принципом роботи XPath та XSLT, використав здобуті знання для проектування та розробки програми для діставання та обробки великих об’ємів інформації з веб-сторінок.