


TRƯỜNG: <b>ĐH Công thương TPHCM</b> KHOA: <b>CÔNG NGHỆ THÔNG TIN</b> BỘ MÔN: <b>KHDL&amp;TTNT</b> MH: <b>LẬP TRÌNH PYTHON</b>	<b>BUỔI 8B.</b> <b>Beautiful Soup</b>	
--	--	---

## A. MỤC TIÊU

- Làm việc với Beautiful Soup

## B. DỤNG CỤ - THIẾT BỊ THÍ NGHIỆM CHO MỘT SV

STT	Chủng loại – Quy cách vật tư	Số lượng	Đơn vị	Ghi chú
1	Computer	1	1	

## C. NỘI DUNG THỰC HÀNH

### I. Tóm tắt lý thuyết

#### 1. Giới thiệu về Beautiful Soup

Beautiful Soup là một thư viện Python được sử dụng để phân tích cú pháp và trích xuất dữ liệu từ các tài liệu HTML và XML. Được tạo ra để làm cho việc thao tác với dữ liệu web trở nên đơn giản và dễ dàng, Beautiful Soup cung cấp một giao diện Pythonic trực quan để đọc, tìm kiếm và trích xuất thông tin từ các trang web.

Với Beautiful Soup, việc lập trình trích xuất thông tin từ HTML trở nên dễ dàng hơn bao giờ hết. Thư viện này tự động chuyển đổi dữ liệu HTML hoặc XML thành một cây DOM (Document Object Model), cho phép người lập trình truy cập và tìm kiếm các phần tử, thuộc tính và nội dung của trang web một cách dễ dàng.

Một số đặc điểm chính của Beautiful Soup bao gồm:

- Dễ sử dụng: Beautiful Soup cung cấp một cú pháp đơn giản và giao diện lập trình dễ hiểu, giúp người lập trình dễ dàng thực hiện các tác vụ phân tích cú pháp và trích xuất dữ liệu.
- Hỗ trợ đa dạng định dạng: Beautiful Soup hỗ trợ cả HTML và XML, cho phép bạn làm việc với nhiều loại tài liệu khác nhau.

- Tích hợp linh hoạt: BeautifulSoup có thể được kết hợp với các thư viện khác trong hệ sinh thái Python như Requests để lấy dữ liệu từ web hoặc Pandas để phân tích và xử lý dữ liệu.

Tóm lại, BeautifulSoup là một công cụ mạnh mẽ và linh hoạt cho việc phân tích và trích xuất dữ liệu từ các trang web, giúp người lập trình thực hiện các tác vụ liên quan đến web scraping một cách dễ dàng và hiệu quả.

### **1.1. Mục đích**

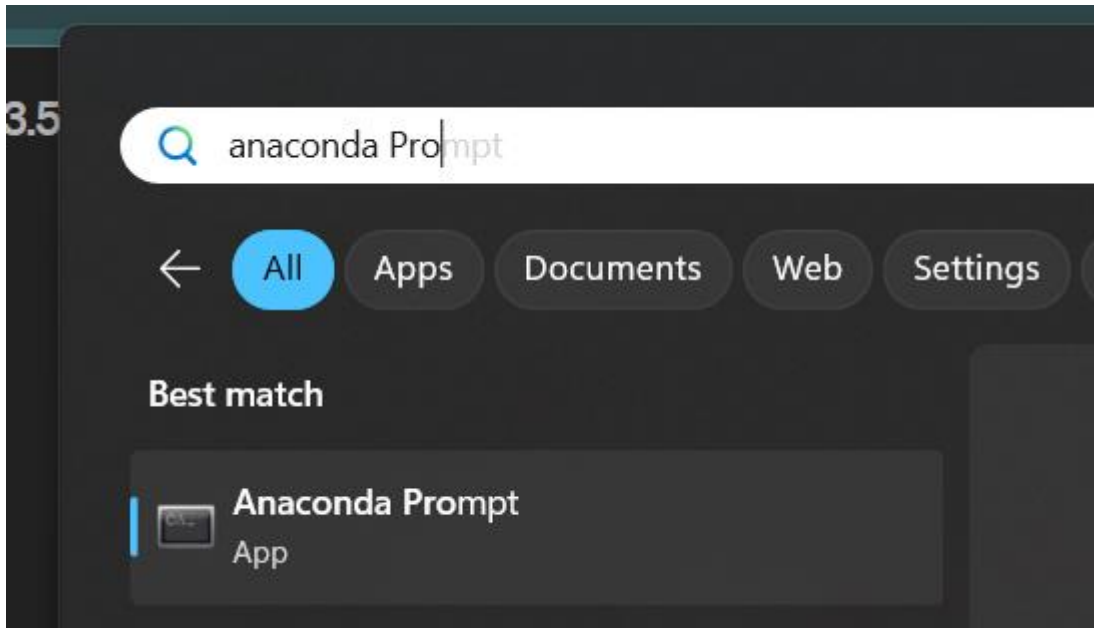
- Phân tích cú pháp HTML/XML: BeautifulSoup giúp phân tích cú pháp của các tài liệu HTML và XML một cách dễ dàng, cho phép bạn trích xuất thông tin từ các trang web.
- Trích xuất dữ liệu: BeautifulSoup cung cấp các công cụ để trích xuất thông tin từ các trang web, bao gồm văn bản, hình ảnh, siêu liên kết, và các thành phần khác của tài liệu HTML/XML.
- Tìm kiếm và lọc dữ liệu: Thư viện này cho phép bạn tìm kiếm và lọc dữ liệu dựa trên các tiêu chí như tên thẻ, thuộc tính, nội dung, v.v.

### **1.2. Ứng dụng**

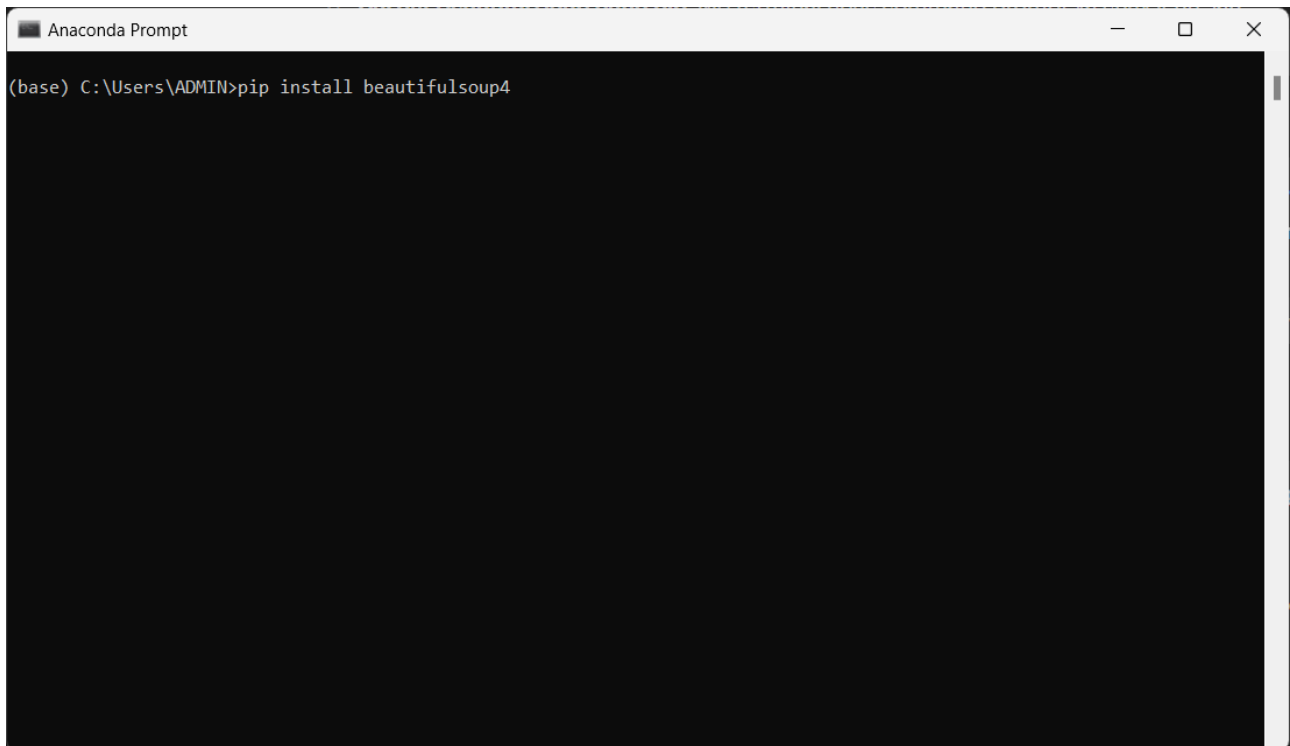
- Web scraping: BeautifulSoup là một công cụ mạnh mẽ cho việc trích xuất dữ liệu từ các trang web. Bạn có thể sử dụng nó để thu thập thông tin từ các trang web, như tin tức, giá sản phẩm, dữ liệu thống kê, v.v.
- Phân tích dữ liệu web: BeautifulSoup có thể được sử dụng để phân tích và hiểu cấu trúc của các trang web, giúp bạn xây dựng các ứng dụng hoặc dịch vụ dựa trên dữ liệu web.
- Kiểm tra và gỡ lỗi: Bạn có thể sử dụng BeautifulSoup để kiểm tra cấu trúc của các trang web và gỡ lỗi khi phát triển ứng dụng hoặc bot trích xuất dữ liệu.
- Xây dựng công cụ tự động hóa: BeautifulSoup cung cấp một cách tiếp cận dễ dàng cho việc xây dựng các công cụ tự động hóa hoặc bot để tự động cập nhật thông tin, tạo nội dung, v.v.
- Xử lý dữ liệu không cấu trúc: BeautifulSoup cũng có thể được sử dụng để xử lý dữ liệu không cấu trúc từ các nguồn như các tệp tin HTML không đồng nhất, dữ liệu thu thập từ các nguồn khác nhau trên web, v.v.

## 2. Cài đặt Beautiful Soup

- Mở Start -> Search: Anaconda Prompt

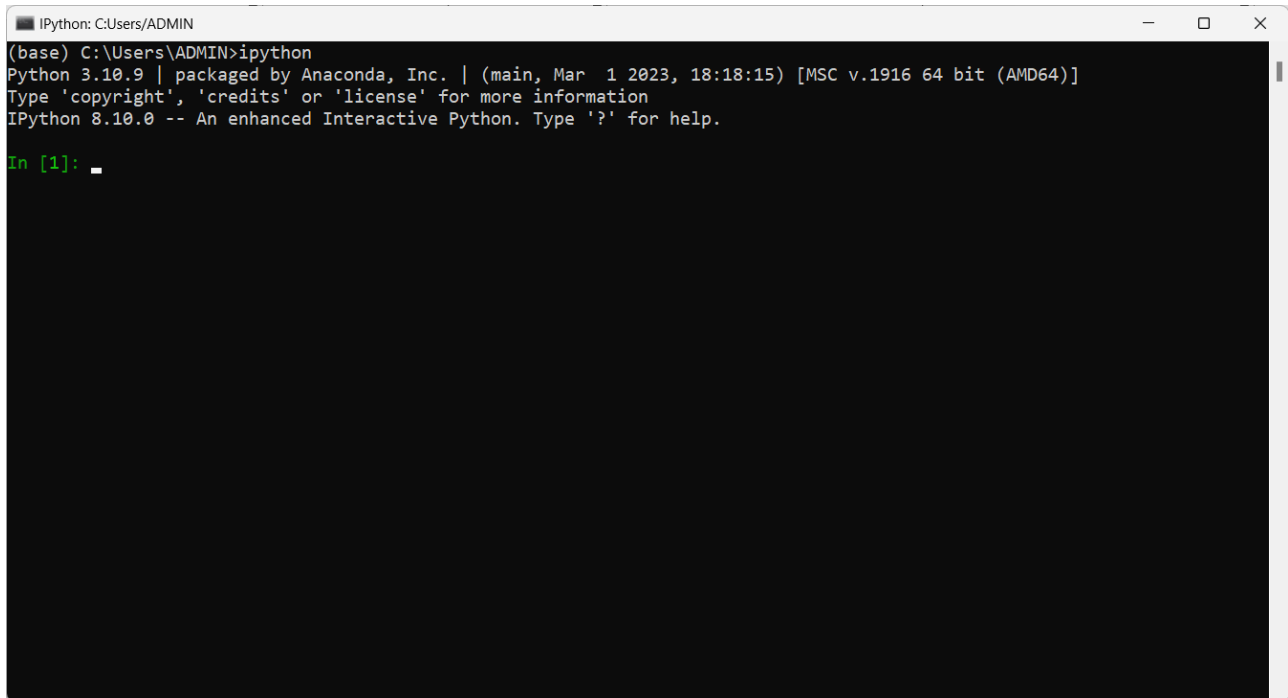


- Chạy Anaconda Prompt. Màn hình Prompt (tương tự cmd) xuất hiện. Gõ lệnh  
`pip install beautifulsoup4`



- Thông thường, gói Beautiful Soup được cài đặt sẵn cùng với Anaconda. Do đó, bạn có thể test gói này trước khi thực hiện cài đặt

- Gõ iPython trong Anaconda Prompt để mở Python interactive shell



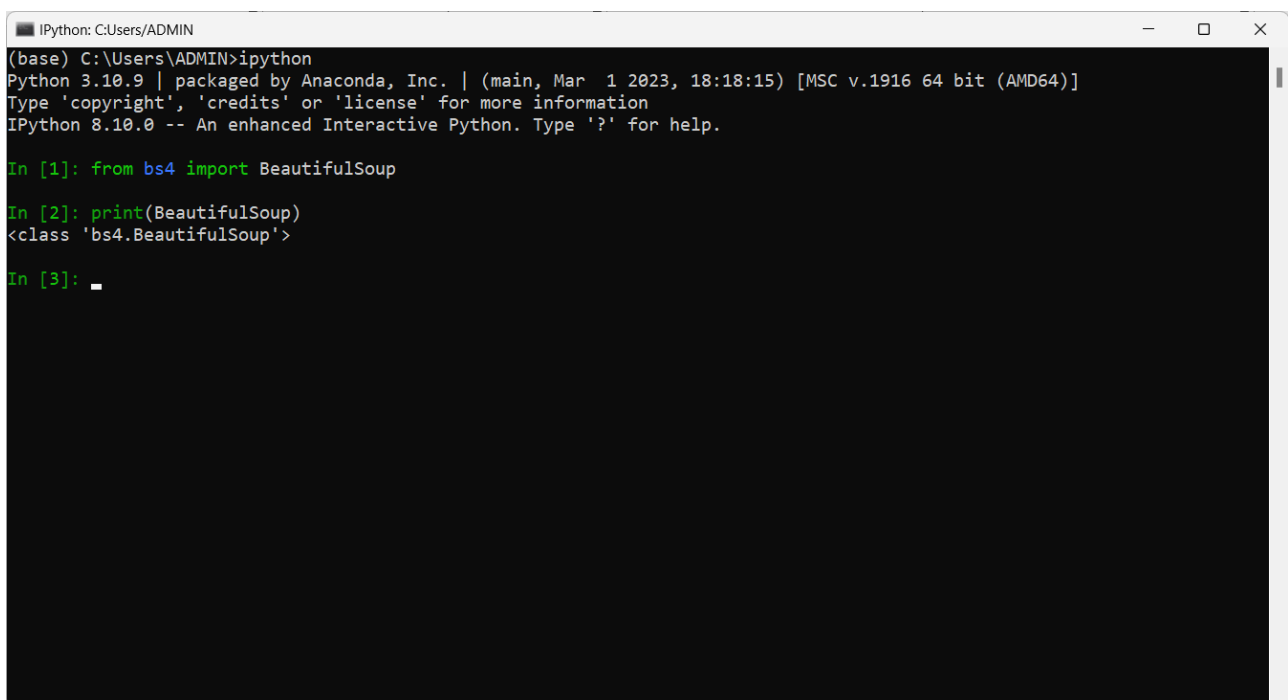
```
IPython: C:\Users\ADMIN
(base) C:\Users\ADMIN>ipython
Python 3.10.9 | packaged by Anaconda, Inc. | (main, Mar 1 2023, 18:18:15) [MSC v.1916 64 bit (AMD64)]
Type 'copyright', 'credits' or 'license' for more information
IPython 8.10.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: _
```

- Gõ lệnh sau:

```
from bs4 import BeautifulSoup
```

```
print(BeautifulSoup)
```



```
IPython: C:\Users\ADMIN
(base) C:\Users\ADMIN>ipython
Python 3.10.9 | packaged by Anaconda, Inc. | (main, Mar 1 2023, 18:18:15) [MSC v.1916 64 bit (AMD64)]
Type 'copyright', 'credits' or 'license' for more information
IPython 8.10.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: from bs4 import BeautifulSoup

In [2]: print(BeautifulSoup)
<class 'bs4.BeautifulSoup'>

In [3]: _
```

- Nếu không xuất hiện thông báo lỗi, có nghĩa là gói Beautiful Soup đã được cài đặt thành công trên máy tính.

### **3. Cơ bản về HTML và XML**

#### **3.1. HTML là gì?**

HTML là viết tắt của HyperText Markup Language (Ngôn ngữ Đánh dấu Siêu văn bản), là ngôn ngữ lập trình được sử dụng để tạo ra các trang web. HTML mô tả cấu trúc của một trang web bằng cách sử dụng các thẻ và các thuộc tính. Dưới đây là một số điểm cơ bản về HTML.

##### **3.1.1. Ngôn ngữ siêu văn bản (HyperText Markup Language)**

HTML là một ngôn ngữ đánh dấu được sử dụng để tạo ra các trang web.

Nó được sử dụng để xác định cấu trúc của một trang web bằng cách sử dụng các thẻ và các thuộc tính.

##### **3.1.2. Cấu trúc cơ bản của HTML**

Mỗi tài liệu HTML bắt đầu với thẻ `<html>` và kết thúc với thẻ `</html>`.

Phần chính của một trang web thường được đặt trong thẻ `<body>`.

Các tiêu đề và đoạn văn bản được định nghĩa trong các thẻ như `<h1>`, `<p>`, `<div>`, v.v.

##### **3.1.3. Thẻ và Thuộc tính**

**Thẻ:** Mỗi phần tử trong HTML được định nghĩa bằng cặp thẻ mở và thẻ đóng, ví dụ: `<p>` và `</p>`.

**Thuộc tính:** Các phần tử HTML có thể chứa các thuộc tính. Ví dụ, thuộc tính `class`, `id`, `src`, v.v.

##### **3.1.4. Các phần tử cơ bản trong HTML**

**Tiêu đề:** Được định nghĩa bằng các thẻ `<h1>` đến `<h6>`.

**Đoạn văn bản:** Được định nghĩa bằng thẻ `<p>`.

**Liên kết (Link):** Được định nghĩa bằng thẻ `<a>` và có thể chứa thuộc tính `href`.

**Hình ảnh:** Được định nghĩa bằng thẻ `<img>` và chứa thuộc tính `src` để chỉ định đường dẫn đến hình ảnh.

### 3.2. XML là gì?

XML là viết tắt của Extensible Markup Language (Ngôn ngữ Đánh dấu Mở rộng), là một ngôn ngữ đánh dấu dùng để lưu trữ và truyền tải dữ liệu dưới dạng văn bản có cấu trúc. XML được thiết kế để làm cho dữ liệu trở nên dễ đọc và dễ hiểu cho cả con người và máy tính. Dưới đây là một số điểm cơ bản về XML.

#### 3.2.1. Ngôn ngữ Đánh dấu Mở rộng (Extensible Markup Language)

XML là một ngôn ngữ đánh dấu dùng để định nghĩa các tập tin có cấu trúc và dễ đọc.

Nó không giới hạn trong việc định nghĩa cấu trúc dữ liệu và có thể được mở rộng theo nhu cầu cụ thể của ứng dụng.

#### 3.2.2. Cấu trúc của XML

Mỗi tập tin XML bắt đầu với một phân khai báo XML (XML declaration) để xác định phiên bản của XML sử dụng.

Tiếp theo là phần tử gốc (root element) của tài liệu XML, được đặt trong cặp thẻ mở và đóng.

Dữ liệu trong XML được tổ chức trong các phần tử và thuộc tính, được định nghĩa bằng cặp thẻ mở và đóng, và có thể chứa nội dung văn bản hoặc các phần tử con.

#### 3.2.3. Thẻ và Thuộc tính trong XML

Thẻ: Mỗi phần tử XML được định nghĩa bằng cặp thẻ mở và thẻ đóng, ví dụ: <element> và </element>.

Thuộc tính: Các phần tử XML có thể chứa các thuộc tính, được đặt trong cặp dấu ngoặc kép và có giá trị cụ thể.

#### 3.2.3. Ví dụ một tài liệu XML

```
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book category="fiction">
    <title lang="en">Harry Potter</title>
    <author>J.K. Rowling</author>
    <year>2005</year>
```

```
        <price>29.99</price>
    </book>
    <book category="non-fiction">
        <title lang="en">Introduction to XML</title>
        <author>John Doe</author>
        <year>2010</year>
        <price>19.99</price>
    </book>
</bookstore>
```

### 3.3. Sự khác biệt giữa HTML và XML

HTML (HyperText Markup Language) và XML (Extensible Markup Language) là hai ngôn ngữ đánh dấu phổ biến được sử dụng trong lập trình web và lưu trữ dữ liệu. Dưới đây là sự khác biệt chính giữa HTML và XML:

#### \* Mục đích sử dụng

HTML: Được sử dụng để tạo ra các trang web và hiển thị nội dung trên trình duyệt web. HTML mô tả cấu trúc và nội dung của trang web, bao gồm văn bản, hình ảnh, liên kết, và các phần tử tương tác.

XML: Được sử dụng để lưu trữ và truyền tải dữ liệu có cấu trúc. XML không chỉ dành cho trình duyệt web mà còn được sử dụng trong các ứng dụng khác như trao đổi dữ liệu giữa các hệ thống, lưu trữ cấu hình, và truyền tải dữ liệu qua mạng.

#### \* Cấu trúc

HTML: Có cấu trúc cụ thể và cố định được xác định bởi các chuẩn của HTML. Mỗi tài liệu HTML phải có một cấu trúc cụ thể, bao gồm các thẻ như <html>, <head>, <body>, v.v.

XML: Không có cấu trúc cố định và có thể được tùy chỉnh tùy theo nhu cầu cụ thể của ứng dụng. Một tài liệu XML có thể chứa bất kỳ phần tử nào, và cấu trúc của nó phụ thuộc hoàn toàn vào việc thiết kế.

### *\* Mục đích của các phần tử*

HTML: Các phần tử trong HTML thường mô tả cấu trúc và nội dung của trang web, bao gồm các thẻ như <h1>, <p>, <div>, <img>, <a>, v.v.

XML: Các phần tử trong XML thường đại diện cho dữ liệu cụ thể và có thể được tự do định nghĩa để phù hợp với nhu cầu cụ thể của ứng dụng.

### *\* Tính linh hoạt*

HTML: Thường có cấu trúc cố định và được sử dụng chủ yếu để hiển thị nội dung trên trình duyệt web. Cấu trúc của HTML thường không linh hoạt và được thiết kế để tuân thủ các quy tắc và chuẩn của HTML.

XML: Có tính linh hoạt cao và có thể được tùy chỉnh theo nhu cầu cụ thể của ứng dụng. XML không giới hạn trong việc định nghĩa cấu trúc dữ liệu và có thể được mở rộng dễ dàng.

Tóm lại, mặc dù HTML và XML đều là ngôn ngữ đánh dấu, nhưng chúng có mục đích và cấu trúc khác nhau và thường được sử dụng trong các ngữ cảnh khác nhau. HTML thường được sử dụng cho việc tạo ra các trang web, trong khi XML thường được sử dụng cho việc lưu trữ và truyền tải dữ liệu có cấu trúc.

## **4. Làm việc với BeautifulSoup**

### *4.1. Khởi tạo đối tượng soup*

#### *4.1.1. Từ một chuỗi HTML*

```
from bs4 import BeautifulSoup

# Chuỗi HTML cần phân tích
html_content = """
<html>
<head>
<title>Example</title>
</head>
<body>
<h1>Hello, world!</h1>
<p>This is an example paragraph.</p>

```



```
</body>
</html>
"""

# Tạo đối tượng BeautifulSoup từ chuỗi HTML
soup = BeautifulSoup(html_content, 'html.parser')
# Bây giờ bạn có thể sử dụng đối tượng soup để tìm kiếm và
trích xuất dữ liệu từ chuỗi HTML
```

#### 4.1.2. Khởi tạo đối tượng soup từ một trang web

```
import requests
from bs4 import BeautifulSoup

# URL của trang web cần phân tích
url = 'https://example.com'

# Sử dụng requests để tải nội dung của trang web
response = requests.get(url)

# Kiểm tra xem việc tải trang web thành công hay không
if response.status_code == 200:
    # Tạo đối tượng BeautifulSoup từ nội dung HTML của
    trang web
    soup = BeautifulSoup(response.text, 'html.parser')
    # Bây giờ bạn có thể sử dụng đối tượng soup để tìm
    kiếm và trích xuất dữ liệu từ trang web
else:
    print('Failed to load the website')
```

#### 4.2. Tìm kiếm và trích xuất dữ liệu

Khi sử dụng BeautifulSoup, bạn có thể sử dụng các phương thức như `find()`, `find_all()`, `select()` để tìm kiếm và trích xuất dữ liệu từ tài liệu HTML/XML. Dưới đây là một số ví dụ minh họa về cách sử dụng các phương thức này:

#### 4.2.1. Tìm kiếm một phần tử cụ thể

```
import requests
from bs4 import BeautifulSoup

# URL của trang web cần phân tích
url = 'https://thongthai.work'

response = requests.get(url)

if response.status_code == 200:
    soup = BeautifulSoup(response.text, 'html.parser')
else:
    print('Failed to load the website')

# Tìm phần tử đầu tiên có thẻ <h1>
h2_tag = soup.find('h2')

# Trích xuất nội dung của phần tử <h1>
h2_content = h2_tag.text
print(h2_content)
```

Trong ví dụ này, chúng ta tìm phần tử <h2> đầu tiên của trang web.

#### 4.2.2. Tìm kiếm tất cả các phần tử cùng loại

```
import requests
from bs4 import BeautifulSoup

# URL của trang web cần phân tích
url = 'https://thongthai.work'

response = requests.get(url)

if response.status_code == 200:
    soup = BeautifulSoup(response.text, 'html.parser')
else:
```

```

    print('Failed to load the website')

# Tìm tất cả các phần tử có thẻ <a>
a_tags = soup.find_all('a')

# Trích xuất liên kết và nội dung của các phần tử <a>
for a_tag in a_tags:
    link = a_tag.get('href')
    text = a_tag.text
    print(link, text)

```

Trong ví dụ này, chúng ta tìm tất cả các phần tử thẻ <a>, in ra link và text tương ứng.

#### 4.2.3. Tìm kiếm các phần tử theo các thuộc tính

```

import requests
from bs4 import BeautifulSoup

# URL của trang web cần phân tích
url = 'https://thongthai.work'

response = requests.get(url)

if response.status_code == 200:
    soup = BeautifulSoup(response.text, 'html.parser')
else:
    print('Failed to load the website')

# Tìm tất cả các phần tử có thuộc tính class là "entry-
title"
content_tags = soup.find_all(class_='entry-title')

# Trích xuất nội dung của các phần tử có class "entry-
title"
for content_tag in content_tags:
    print(content_tag.text)

```

Trong ví dụ này, ta tìm tất cả các phần tử có class là entry-title.

#### 4.2.4. Tìm kiếm sử dụng các Selector CSS

```
import requests
from bs4 import BeautifulSoup

# URL của trang web cần phân tích
url = 'https://thongthai.work'

response = requests.get(url)

if response.status_code == 200:
    soup = BeautifulSoup(response.text, 'html.parser')
else:
    print('Failed to load the website')

# Sử dụng selector CSS để tìm tất cả các phần tử có class
"article" bên trong một phần tử có id "main"
article_tags = soup.select('#main .article')

# Trích xuất nội dung của các phần tử có class "article"
for article_tag in article_tags:
    print(article_tag.text)
```

Trong ví dụ này, ta tìm phần tử có class là article với id là main.

#### 4.2.5. Các điều kiện kết hợp

Giả sử ta muốn phân tích các phần tử có thẻ <a> là con của phần tử thẻ <p> với class là "article"

```
# Giả sử `soup` là đối tượng BeautifulSoup chứa nội dung
HTML bạn muốn phân tích

# Tìm tất cả các phần tử <p> có class là "article"
```

```
p_tags = soup.find_all('p', class_='article')

# Duyệt qua từng phần tử <p> để tìm các phần tử <a> là con
của nó
for p_tag in p_tags:
    # Tìm tất cả các phần tử <a> là con của phần tử <p>
    hiện tại
    a_tags = p_tag.find_all('a')

    # In ra các phần tử <a> tìm thấy (nếu có)
    if a_tags:
        print(f"Found <a> tags inside <p> with class
'<strong>article'")
        for a_tag in a_tags:
            print(a_tag)
    else:
        print("No <a> tags found inside <p> with class
'<strong>article'")
```

## 5. Ví dụ minh họa

### 5.1. Lấy dữ liệu những câu phát biểu của Trump

Dữ liệu được trích xuất từ địa chỉ:

<https://www.nytimes.com/interactive/2017/06/23/opinion/trumps-lies.html>

Khi truy cập vào trang web, ta sẽ thấy được nội dung như hình bên dưới

**JAN. 21** “I wasn't a fan of Iraq. I didn't want to go into Iraq.” (*He was for an invasion before he was against it.*)      **JAN. 21** “A reporter for Time magazine — and I have been on their cover 14 or 15 times. I think we have the all-time record in the history of Time magazine.” (*Trump was on the cover 11 times and Nixon appeared 55 times.*)      **JAN. 23** “Between 3 million and 5 million illegal votes caused me to lose the popular vote.” (*There's no evidence of illegal voting.*)      **JAN. 25** “Now, the audience was the biggest ever. But this crowd was massive. Look how far back it goes. This crowd was massive.” (*Official aerial photos show Obama's 2009 inauguration was much more heavily*

Kiểm tra code HTML tương ứng

```
1094 <span class="short-desc"><strong>Jan. 21&nbsp;</strong>“I wasn't a
fan of Iraq. I didn't want to go into Iraq.” <span class="short-
truth"><a href="https://www.buzzfeed.com/andrewkaczynski/in-2002-
donald-trump-said-he-supported-invading-iraq-on-the"
target="_blank">(He was for an invasion before he was against it.)
</a></span></span>&nbsp;&nbsp;&nbsp;<span class="short-desc"><strong>Jan.
21&nbsp;</strong>“A reporter for Time magazine – and I have been on
their cover 14 or 15 times. I think we have the all-time record in
the history of Time magazine.” <span class="short-truth"><a
href="http://nation.time.com/2013/11/06/10-things-you-didnt-know-
about-time/" target="_blank">(Trump was on the cover 11 times and
Nixon appeared 55 times.)</a></span></span>&nbsp;&nbsp;&nbsp;<span
```

Như vậy, mỗi câu phát biểu được trình bày dưới dạng code HTML sau

```
<span class="short-desc"><strong> DATE </strong> LIE <span
class="short-truth"><a href="URL"> EXPLANATION
</a></span></span>
```

Ta thấy có một thẻ <span> bên ngoài với class “short-desc”, bên trong ngày tháng được đưa vào thẻ <strong>. Sau đó đến lời phát biểu (LIE) và một thẻ <span> khác, bên trong lại có một thẻ <a> chứa URL của lời giải thích.

Từ phân tích cấu trúc HTML, ta có thể code python để thực hiện lấy toàn bộ dữ liệu trang web này như sau (kết quả được lưu vào file csv)

```

import requests
from bs4 import BeautifulSoup
import pandas as pd

r =
requests.get('https://www.nytimes.com/interactive/2017/06/
23/opinion/trumps-lies.html')

soup = BeautifulSoup(r.text, 'html.parser')
results = soup.find_all('span', attrs={'class':'short-
desc'})

records = []
for result in results:
    date = result.find('strong').text[0:-1] + ', 2017'
    lie = result.contents[1][1:-2]
    explanation = result.find('a').text[1:-1]
    url = result.find('a')['href']
    records.append((date, lie, explanation, url))

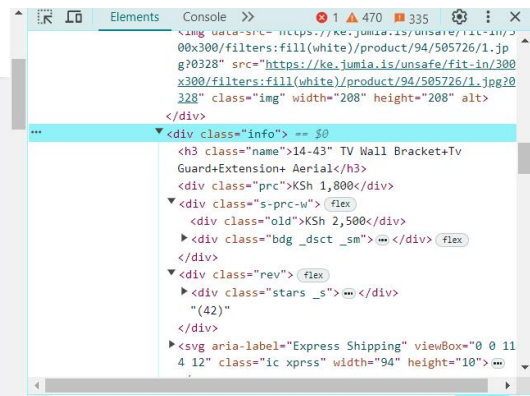
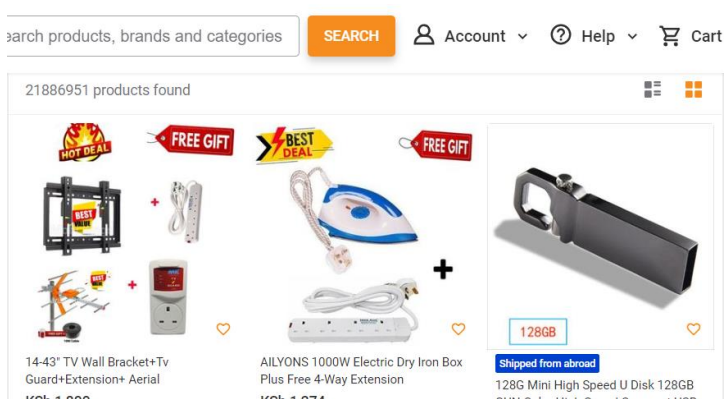
df = pd.DataFrame(records, columns=['date', 'lie',
'explanation', 'url'])
df['date'] = pd.to_datetime(df['date'])
df.to_csv('trump_lies.csv', index=False, encoding='utf-8')

```

## 5.2. Lấy thông tin sản phẩm từ một trang thương mại điện tử

Truy cập trang web: <https://www.jumia.co.ke/all-products/>

Phân tích cấu trúc HTML



Ta sẽ thấy mỗi thông tin sản phẩm bắt đầu với <div class="info"> ...

Code hoàn chỉnh

```
from bs4 import BeautifulSoup
import requests

url = "https://www.jumia.co.ke/all-products/"
furl = requests.get(url)
jsoup = BeautifulSoup(furl.content , 'html.parser')
products = jsoup.find_all('div' , class_ = 'info')

for product in products:
    Name = product.find('h3' ,
class_="name").text.replace('\n', '')
    Price = product.find('div' , class_=
"prc").text.replace('\n', '')
    try:
        Rating = product.find('div', class_='stars
_s').text.replace('\n', '')
    except:
        Rating = 'None'

    info = [ Name, Price,Rating]
    print(info)
```



## 6. Bài tập

**Bài 1.** Xây dựng một chương trình python trích xuất tất cả thông báo trong 5 trang đầu tiên từ trang web khoa CNTT. Thông tin trích xuất bao gồm: Tiêu đề, Ngày đăng và URL.

**Bài 2.** Xây dựng một chương trình python trích xuất tên và email của BCN Khoa CNTT từ trang <https://fit.huit.edu.vn/gioi-thieu/ban-chu-nhiem-khoa>

**Bài 3.** Xây dựng một chương trình python hiển thị danh sách 10 việc làm IT mới nhất làm việc ở TPHCM từ trang <https://topdev.vn/>

**Bài 4.** Xây dựng một chương trình python trích xuất ngày tháng, tên và URL từ mục Latest News của trang <https://www.python.org/>

**Bài 5.** Tìm hiểu về dịch vụ RSS và xây dựng một chương trình python trích xuất 10 mẫu tin giáo dục từ Dịch vụ Báo thanh niên RSS: <https://thanhnien.vn/rss/giao-duc.rss>

Chú ý rằng dữ liệu RSS Feed là XML nên khi tạo đối tượng soup, thêm vào thông số xml

```
soup = BeautifulSoup(response.text, 'xml')
```