Learning Quiz 35: Beautiful Soup

Due Nov 27 at 1pm **Points** 5 **Questions** 5 **Available** until Dec 4 at 11:59pm **Time Limit** None
Allowed Attempts Unlimited

Instructions

Prior to completing this quiz, be sure to skim the **Beautiful Soup Documentation (https://www.crummy.com/software/BeautifulSoup/bs4/doc/)** and download/install **Beautiful Soup (https://www.crummy.com/software/BeautifulSoup/)** .

There are a couple different ways to do this. If you are working on PyCharm, the easiest method is to look for "File" in the top left corner of the window and click on the following:

File >> Settings >> Project >> Project Interpreter >> "+" on the right side >> look for beautifulsoup4 >> install package

If you are not working on PyCharm, there is an installation guide is available on the documentation. In general, you may need to type one of the following on the "Console" at the bottom of PyCharm:

```
# Possibility 1
pip install beautifulsoup4

# Possibility 2
pip install bs4
```

There's a lot of information about Beautiful Soup available in the documentation, far more than we'll be able to cover in this class. While html.parser is easy to work with, Beautiful Soup is powerful in its web-scraping capabilities.

This quiz will help guide us through a glimpse of Beautiful Soup. You may attempt this quiz as many times as you would like. The highest score *prior to the deadline* will count towards the final course grade. No late submissions will be accepted.

Take the Quiz Again

Attempt History

	Attempt	Time	Score
KEPT	Attempt 1	15 minutes	4.42 out of 5
LATEST	Attempt 2	4 minutes	4.33 out of 5
	Attempt 1	15 minutes	4.42 out of 5

Score for this attempt: **4.33** out of 5
Submitted Nov 29 at 12:07pm
This attempt took 4 minutes.

Question 1

1 / 1 pts

Just printing the HTML output of a webpage can be quite ugly:

```
from urllib.request import urlopen

response = urlopen('https://americanliterature.com/author/hans-christian-andersen/short-story/the-little-m
atch-girl')
html = response.read()
html = html.decode() ## This converts the html to a string
print(html)
```

You can use Beautiful Soup's prettify() method to print the HTML output in a nice nested format:

```
from bs4 import BeautifulSoup
from urllib.request import urlopen

response = urlopen('https://americanliterature.com/author/hans-christian-andersen/short-story/the-little-m
atch-girl')
soup = BeautifulSoup(response, 'html.parser')
print(soup.prettify())
```

Let's try to get the text data that show up in between the tags. Try to get the text data that belongs to the <h3> tag by typing 'h3' (with quotes) in the blank below.

--------- START CODE ---------

from bs4 import BeautifulSoup
from urllib.request import urlopen

response = urlopen('https://americanliterature.com/author/hans-christian-andersen/short-story/the-little-match-girl')
soup = BeautifulSoup(response, 'html.parser')

story = soup.find('h3')

print(story.text)

--------- END CODE ---------

Note that find() will only give you the first instance of whatever is found behind h3.

Answer 1:

Correct!

'h3'

Question 2

1 / 1 pts

The find() method will only return the first instance. If you would like to collect the NEXT instance of a tag, you need to use the find_next method.

Let's print the text data associated with the FOURTH instance of the 'p'. Imagine HTML that might look like this (overly simplified):

```
<p> first instance text data </p>
<p> second instance text data </p>
<p> third instance text data </p>
<p> fourth instance text data </p>
```

Fill in the four blanks below with 'p':

---------- START CODE -----------

from bs4 import BeautifulSoup
from urllib.request import urlopen

response = urlopen('https://americanliterature.com/author/hans-christian-andersen/short-story/the-little-match-girl')
soup = BeautifulSoup(response, 'html.parser')
story = soup.find('p') ## find() will only give us the first instance

story = story.find_next('p')

story = story.find_next('p')

story = story.find_next('p')

print(story.text)

---------- END CODE ----------

NOTE, that in our first find, we used **soup**.find(). Thereafter, we used **story**.find_next(). If we use **soup**.find_next(), we'll be starting from the top again (unless we replace the soup variable).

If the search tag does not exist, then find() will return None.

Answer 1:

Correct! 'p'

Answer 2:

Correct! 'p'

Answer 3:

Correct! 'p'

Answer 4:

Correct! 'p'

Question 3

0.83 / 1 pts

You CAN use find() for attributes instead of tags, and you can use find() for both tags and attributes. Here's an example of attributes only (somewhere in the webpage there's the tag <cite class = "al-title">):

```
from bs4 import BeautifulSoup
from urllib.request import urlopen

response = urlopen('https://americanliterature.com/author/hans-christian-andersen/short-story/the-little-m
atch-girl')
soup = BeautifulSoup(response, 'html.parser')
story = soup.find(attrs={'class': "al-title"})
print(story.text)
```

Here's a example of a tag and specific attribute search, using the <cite class = "al-title"> tag:

```
from bs4 import BeautifulSoup
from urllib.request import urlopen

response = urlopen('https://americanliterature.com/author/hans-christian-andersen/short-story/the-little-m
atch-girl')
soup = BeautifulSoup(response, 'html.parser')
story = soup.find('cite', attrs={'class': "al-title"})
print(story.text)
```

Suppose we would like to get the text data associated with the tag <div class="afternote"> . Fill in the blanks to achieve this search:

---------- START CODE -----------

from bs4 import BeautifulSoup

from urllib.request import urlopen

response = urlopen('https://americanliterature.com/author/hans-christian-andersen/short-story/the-little-match-girl')
soup = BeautifulSoup(response, 'html.parser')
story = soup .find('div' , attrs={ 'class' :

'div' })

print(story.text)

----------- END CODE -----------

Answer 1:

Correct!

BeautifulSoup

Answer 2:

Correct!

urlopen

Answer 3:

Correct!

soup

Answer 4:

Correct!

'div'

Answer 5:

Correct!

'class'

Correct Answer

"class"

Answer 6:

You Answered

'div'

Correct Answer

"afternote"

Correct Answer 'afternote'

Question 4

1 / 1 pts

If you want to get text for ALL instances of a tag, you can use the find_all() method. Here's a example of the method in action:

```
from bs4 import BeautifulSoup
from urllib.request import urlopen

response = urlopen('https://americanliterature.com/author/hans-christian-andersen/short-story/the-little-m
atch-girl')
soup = BeautifulSoup(response, 'html.parser')
story = soup.find_all('div')
for tag in story:
    print(tag.text)
```

You can use find_all() for tags AND attributes similar to find().

Note that we looped through the output of find_all to print() each text data. Of course, you can do things other than just print(), such as store the information in a text file.

Add to the code above print(type(story)). What data type is the story variable above?

- 'dict'
- 'set'
- Correct! [✓ 'bs4.element.ResultSet']
- 'int'
- 'list'
- 'tuple'
- 'str'

Question 5

0.5 / 1 pts

In HTML Parser, we were able to get links listed in a webpage by looking at the <a href = '...'> tag. You can do something similar using Beautiful Soup using the combination of the find_all() and get() methods:

```
from bs4 import BeautifulSoup
from urllib.request import urlopen

response = urlopen('https://americanliterature.com/author/hans-christian-andersen/short-story/the-little-m
atch-girl')
soup = BeautifulSoup(response, 'html.parser')
```

```
for line in soup.find_all('a'):
    print(line.get('href'))
```

In the code above, find_add() looks for all instances of the tag 'a'. Thereafter, get() looks for the attribute 'href' and gets whatever follows the attribute (in this case, the links).

Fill in the blanks below to find all text information following the tag and attribute... <span class = ...>:

---------- START CODE ----------

from bs4 import BeautifulSoup
from urllib.request import urlopen

response = urlopen('https://americanliterature.com/author/hans-christian-andersen/short-story/the-little-match-girl')

soup = BeautifulSoup(response , 'html.parser')

for line in soup.find_all('span):

print(line .get('class))

---------- END CODE ----------

Answer 1:

Correct!

response

Answer 2:

You Answered

'span

Correct Answer

'span'

Correct Answer

"span"

Answer 3:

Correct!

line

Answer 4:

You Answered

'class

Correct Answer

'class'

Correct Answer

"class"

Quiz Score: **4.33** out of 5