Due Nov 27 at 1pm

Allowed Attempts Unlimited

Points 5

Questions 5

Available until Dec 4 at 11:59pm

Time Limit None

Instructions

Prior to completing this quiz, be sure to read:

• Section 11.2: Python WWW API (p. 379-386)

Please also go over Practice Problems 11.1-11.3 in the textbook (solutions at the end of the chapter) before attempting this quiz.

This quiz was created for learning purposes. 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 2	3 minutes	5 out of 5	
LATEST	Attempt 2	3 minutes	5 out of 5	
	Attempt 1	7 minutes	4 out of 5	

Score for this attempt: **5** out of 5 Submitted Nov 29 at 10:42am This attempt took 3 minutes.

Question 1

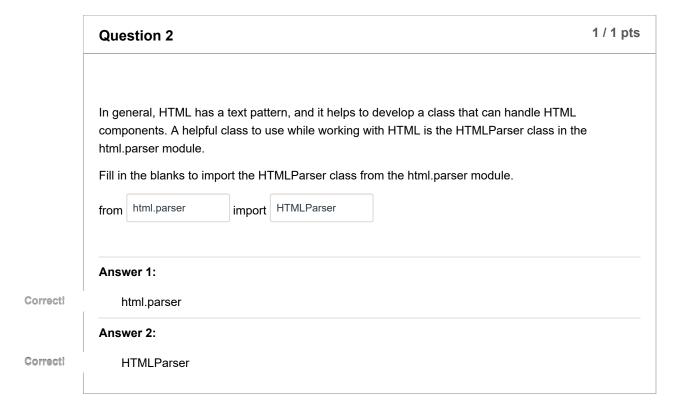
1 / 1 pts

We can access webpages similar to how we read in text files in Chapter 4. We'll need to use the urlopen() function from the urllib.request module. While urlopen() is similar to the built-in open() function that we saw in Chapter 4, it differs in a couple of ways:

In the following lines of code, we import the urlopen function from urllib.request. We then open the Wikipedia page on Los Angeles, and read the html file. Notice that our read in url (myhtml) is of type "byte" which is not human-readable. However when we print myhtml, we are able to read the content. That is because Python does the converting for us.

```
from urllib.request import urlopen
response = urlopen('https://en.wikipedia.org/wiki/Los_Angeles')
myhtml = response.read()
print(type(myhtml))
print(myhtml))
```

Let's decode myhtml so that we can actually work with it. Notice that after using the decode() function, type(myhtml) tells us that it's a string. We can now work with the string. myhtml = myhtml.decode() print(type(myhtml)) print(myhtml) Consider the following code: from urllib.request import urlopen response = urlopen('https://en.wikipedia.org/wiki/Python_(programming_language)') read1 = response.read() read2 = read1.decode() What data type is read1? read2? Correct! read1 byte Correct! read2 str Other Incorrect Match Options: float bool • int



Question 3 1 / 1 pts

The HTMLParser class was written with functions that were intended to be overwritten. That means the overall class will handle the HTML for you, but what you want to do with the elements needs to be coded by you. Functions like handle_starttag(), handle_endtag(), and handle_data() can be though of as functions with 'pass' as the body. They do nothing.

In order to inherit the attributes of HTMLParser, but have functions behave differently, we'll need to use the concept of inheritance.

Fill in the blank below to create a class MyParser that prints 'anchor encountered' whenever it encounters an 'a' tag.

	3	
from html.parser	import HTMLParser	
class	MyParser(HTMLParser):
	starttag(self, tag, attrs):	
if tag ≕ pr	= 'a': int('Anchor Encountered')	
Answer 1:		
html.parser		
Answer 2:		
class		
Answer 3:		
HTMLParser		

Correct!

Correct!

Correct!

	Question 4	1 / 1 pts
	If you enter the following lines of code in Python, you will find list of functions available to th HTMLParser class.	e
	from html.parser import HTMLParser help(HTMLParser)	
	Which of the following functions are intended to be overwritten?	
Correct!	handle_comment(self, data)	
Correct!	<pre>Mandle_startendtag(self, tag, attrs)</pre>	
	parse_endtag(self, i)	

	goahead(self, end)
Correct!	<pre>handle_charref(self, name)</pre>
	parse_pi(self, i)

Question 5		1 / 1 pt
relative URLs determine the	_	-
from urllib.par	import urljoin www3.org/Consortium/mission.html'	
relative urljoin(url, rela	= '/Consortium/contact.html'	##Hint, see urljoin() below
Answer 1:		
urllib.pars	е	
Answer 2:		
urljoin		
Answer 3:		

Quiz Score: 5 out of 5