HTML Parser - Part 1

87/115 challenges solved

Rank: 34240 | Points: 1515

Problem Submissions Leaderboard Editorial

HTML

Hypertext Markup Language is a standard markup language used for creating World Wide Web pages.

Parsing

Parsing is the process of syntactic analysis of a string of symbols. It involves resolving a string into its component parts and describing their syntactic roles.

HTMLParser

An HTMLParser instance is fed HTML data and calls handler methods when start tags, end tags, text, comments, and other markup elements are encountered.

Example (from the Python 3 documentation):

Code

```
from html.parser import HTMLParser

class MyHTMLParser(HTMLParser):
   def handle_starttag(self, tag, attrs):
      print("Encountered a start tag:", tag)

   def handle_endtag(self, tag):
      print("Encountered an end tag :", tag)

   def handle_data(self, data):
      print("Encountered some data  :", data)

parser = MyHTMLParser()
parser.feed('<html><head><title>Test</title></head>'
         '<body><h1>Parse me!</h1></body></html>')
```

Output

```
Encountered a start tag: html
Encountered a start tag: head
Encountered a start tag: title
Encountered some data  : Test
Encountered an end tag : title
Encountered an end tag : head
Encountered a start tag: body
Encountered a start tag: h1
Encountered some data  : Parse me!
Encountered an end tag : h1
Encountered an end tag : body
Encountered an end tag : html
```

.handle_starttag(tag, attrs)

This method is called to handle the start tag of an element. (For example:)

The tag argument is the name of the tag converted to lowercase.

The attrs argument is a list of (name, value) pairs containing the attributes found inside the tag's <> brackets.

.handle_endtag(tag)

This method is called to handle the end tag of an element. (For example:)

The tag argument is the name of the tag converted to lowercase.

.handle_startendtag(tag,attrs)

This method is called to handle the empty tag of an element. (For example:
)

The tag argument is the name of the tag converted to lowercase.

The attrs argument is a list of (name, value) pairs containing the attributes found inside the tag's <> brackets.

Task

You are given an HTML code snippet of N lines.

Your task is to print start tags, end tags and empty tags separately.

Format your results in the following way:

```
Start : Tag1
End   : Tag1
Start : Tag2
-> Attribute2[0] > Attribute_value2[0]
-> Attribute2[1] > Attribute_value2[1]
-> Attribute2[2] > Attribute_value2[2]
Start : Tag3
-> Attribute3[0] > None
Empty : Tag4
-> Attribute4[0] > Attribute_value4[0]
End   : Tag3
End   : Tag2
```

Here, the -> symbol indicates that the tag contains an attribute. It is immediately followed by the name of the attribute and the attribute value.

The > symbol acts as a separator of the attribute and the attribute value.

If an HTML tag has no attribute then simply print the name of the tag.

If an attribute has no attribute value then simply print the name of the attribute value as `None`.

Note: Do not detect any HTML tag, attribute or attribute value inside the HTML comment tags (`<!-- Comments -->`).Comments can be multiline as well.

Input Format

The first line contains integer N, the number of lines in a HTML code snippet.

The next N lines contain HTML code.

Constraints

- $0 < N < 100$

Output Format

Print the HTML tags, attributes and attribute values in order of their occurrence from top to bottom in the given snippet.

Use proper formatting as explained in the problem statement.

Sample Input

```
2
<html><head><title>HTML Parser - I</title></head>
<body data-modal-target class='1'><h1>HackerRank</h1><br /></body></html>
```

Sample Output

```
Start : html
Start : head
Start : title
End   : title
End   : head
Start : body
-> data-modal-target > None
-> class > 1
Start : h1
End   : h1
Empty : br
End   : body
End   : html
```

Change Theme Language Python 3

```python
# Enter your code here. Read input from STDIN. Print output to STDOUT
from html.parser import HTMLParser

class MyHTMLParser(HTMLParser):
    def handle_starttag(self, tag, attrs):
        print(f"Start : {tag}")
        for attr in attrs:
            print(f"-> {attr[0]} > {attr[1]}")

    def handle_endtag(self, tag):
        print(f"End   : {tag}")

    def handle_startendtag(self, tag, attrs):
        print(f"Empty : {tag}")
        for attr in attrs:
            print(f"-> {attr[0]} > {attr[1]}")

parser = MyHTMLParser()
for _ in range(int(input())):
    parser.feed(input())

```

EMACS Line: 21 Col: 1

Upload Code as File Test against custom input Run Code Submit Code

Test case 0

Test case 1

Test case 2

Test case 3

Test case 4

Test case 5

Compiler Message

Success

Input (stdin) Download

```
2
<html><head><title>HTML Parser - I</title></head>
<body data-modal-target class='1'><h1>HackerRank</h1><br /></body></html>
```

Expected Output Download

```
Start : html
Start : head
```

```
Start : title
End   : title
```