In [1]:

```
!pip install selenium
!pip install requests
!pip install urllib3
!pip install bs4
```

```
Requirement already satisfied: selenium in d:\anaconda\lib\site-packages (3.
141.0)
Requirement already satisfied: urllib3 in d:\anaconda\lib\site-packages (fro
m selenium) (1.25.11)
Requirement already satisfied: requests in d:\anaconda\lib\site-packages (2.
24.0)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in
d:\anaconda\lib\site-packages (from requests) (1.25.11)
Requirement already satisfied: idna<3,>=2.5 in d:\anaconda\lib\site-packages
(from requests) (2.10)
Requirement already satisfied: certifi>=2017.4.17 in d:\anaconda\lib\site-pa
ckages (from requests) (2020.6.20)
Requirement already satisfied: chardet<4,>=3.0.2 in d:\anaconda\lib\site-pac
kages (from requests) (3.0.4)
Requirement already satisfied: urllib3 in d:\anaconda\lib\site-packages (1.2
5.11)
Requirement already satisfied: bs4 in d:\anaconda\lib\site-packages (0.0.1)
Requirement already satisfied: beautifulsoup4 in d:\anaconda\lib\site-packag
es (from bs4) (4.9.3)
Requirement already satisfied: soupsieve>1.2; python_version >= "3.0" in
d:\anaconda\lib\site-packages (from beautifulsoup4->bs4) (2.0.1)
```

In [2]:

```python
# import required modules
import requests

# get URL
page = requests.get("https://en.wikipedia.org/wiki/Paragraph")

# display status code
print(page.status_code)

# display scrapped data
print(page.content)
```

```
200
b'<!DOCTYPE html>\n<html class="client-nojs" lang="en" dir="ltr">\n<head>
\n<meta charset="UTF-8"/>\n<title>Paragraph - Wikipedia</title>\n<script>
document.documentElement.className="client-js";RLCONF={"wgBreakFrames":!
1,"wgSeparatorTransformTable":["",""],"wgDigitTransformTable":["",""],"wg
DefaultDateFormat":"dmy","wgMonthNames":["","January","February","Marc
h","April","May","June","July","August","September","October","Novembe
r","December"],"wgRequestId":"88b2811d-7640-4718-97d3-5fdcf9cd644e","wgCS
PNonce":!1,"wgCanonicalNamespace":"","wgCanonicalSpecialPageName":!1,"wgN
amespaceNumber":0,"wgPageName":"Paragraph","wgTitle":"Paragraph","wgCurRe
visionId":1039583963,"wgRevisionId":1039583963,"wgArticleId":230752,"wgIs
Article":!0,"wgIsRedirect":!1,"wgAction":"view","wgUserName":null,"wgUser
Groups":["*"],"wgCategories":["Articles with limited geographic scope fro
m June 2013","Articles containing Ancient Greek (to 1453)-language tex
t","All articles with unsourced statements","Articles with unsourced stat
ements from January 2020","Typography","Writing"],"wgPageContentLanguag
e":"en","wgPageContentModel":"wikitext",\n"wgRelevantPageName":"Paragrap
h","wgRelevantArticleId":230752,"wgIsProbablyEditable":!0,"wgRelevantPage
IsProbablyEditable":!0,"wgRestrictionEdit":[],"wgRestrictionMove":[],"wgF
```

In [3]:

```python
# import required modules
from bs4 import BeautifulSoup

# scrape webpage
soup = BeautifulSoup(page.content, 'html.parser')

# display scrapped data
print(soup.prettify())
```

```
<!DOCTYPE html>
<html class="client-nojs" dir="ltr" lang="en">
 <head>
  <meta charset="utf-8"/>
  <title>
   Paragraph - Wikipedia
  </title>
  <script>
   document.documentElement.className="client-js";RLCONF={"wgBreakFrame
s":!1,"wgSeparatorTransformTable":["",""],"wgDigitTransformTable":
["",""],"wgDefaultDateFormat":"dmy","wgMonthNames":["","January","Februar
y","March","April","May","June","July","August","September","October","No
vember","December"],"wgRequestId":"88b2811d-7640-4718-97d3-5fdcf9cd644
e","wgCSPNonce":!1,"wgCanonicalNamespace":"","wgCanonicalSpecialPageNam
e":!1,"wgNamespaceNumber":0,"wgPageName":"Paragraph","wgTitle":"Paragrap
h","wgCurRevisionId":1039583963,"wgRevisionId":1039583963,"wgArticleId":2
30752,"wgIsArticle":!0,"wgIsRedirect":!1,"wgAction":"view","wgUserName":n
ull,"wgUserGroups":["*"],"wgCategories":["Articles with limited geographi
c scope from June 2013","Articles containing Ancient Greek (to 1453)-lang
```

In [4]:

```
# EXTRACTING PARAGRAPH

list(soup.children)

# find all occurance of p in HTML
# includes HTML tags
# Why p? Because we want paragraphs
print(soup.find_all('p'))

print('\n\n')
print("===============================================================================
# return only text
# does not include HTML tags
print(soup.find_all('p')[0].get_text())
print("===============================================================================
```

```
[<p>A <b>paragraph</b> (from the <a href="/wiki/Ancient_Greek" title="Anc
ient Greek">Ancient Greek</a> <span lang="grc" title="Ancient Greek (to 1
453)-language text">παράγραφος</span>, <i lang="grc-Latn" title="Ancient
Greek (to 1453)-language text">parágraphos</i>, "<i>to write beside</i>")
is a self-contained unit of discourse in <a href="/wiki/Writing" title="W
riting">writing</a> dealing with a particular point or <a href="/wiki/Ide
a" title="Idea">idea</a>. A paragraph consists of one or more <a href="/w
iki/Sentence_(linguistics)" title="Sentence (linguistics)">sentences</a>.
<sup class="reference" id="cite_ref-UNC_1-0"><a href="#cite_note-UNC-1">
[1]</a></sup> Though not required by the syntax of any language, paragrap
hs are usually an expected part of formal writing, used to organize longe
r <a href="/wiki/Prose" title="Prose">prose</a>.<sup class="noprint Inlin
e-Template Template-Fact" style="white-space:nowrap;">[<i><a href="/wiki/
Wikipedia:Citation_needed" title="Wikipedia:Citation needed"><span title
="This claim needs references to reliable sources. (January 2020)">citati
on needed</span></a></i>]</sup>
</p>, <p>The oldest classical Greek and Latin writing had little or no sp
ace between words and could be written in <a href="/wiki/Boustrophedon" t
itle="Boustrophedon">boustrophedon</a> (alternating directions). Over tim
```

In [5]:

```
# EXTRACTING CUSTOMIZED TAG CONTENTS

object = soup.find(id="Numbering")

# find tags
items = object.find_all(class_="anchor")
result = items[0]

# display tags
print(result.prettify())
```

```
<span class="anchor" id="Decimal_numbering">
</span>
```

In [6]:

```
# EXTRACTING IMAGE

image_tags = soup.findAll('img')
# print out image urls
for image_tag in image_tags:
    print(image_tag.get('src'))
```

```
//upload.wikimedia.org/wikipedia/commons/thumb/b/bd/Ambox_globe_content.svg/
48px-Ambox_globe_content.svg.png
//upload.wikimedia.org/wikipedia/commons/thumb/5/59/United_States_Constituti
on.jpg/220px-United_States_Constitution.jpg
//upload.wikimedia.org/wikipedia/commons/thumb/9/99/Wiktionary-logo-en-v2.sv
g/16px-Wiktionary-logo-en-v2.svg.png
//upload.wikimedia.org/wikipedia/en/thumb/9/96/Symbol_category_class.svg/16p
x-Symbol_category_class.svg.png
//en.wikipedia.org/wiki/Special:CentralAutoLogin/start?type=1x1
/static/images/footer/wikimedia-button.png
/static/images/footer/poweredby_mediawiki_88x31.png
```

In [7]:

```
#getting jpg images
import re
image_tags = soup.findAll('img', {'src':re.compile('.jpg')})
# print out image urls
for image_tag in image_tags:
    print(image_tag['src']+'\n')
```

```
//upload.wikimedia.org/wikipedia/commons/thumb/5/59/United_States_Constituti
on.jpg/220px-United_States_Constitution.jpg
```

In [8]:

```
import urllib.request
urllib.request.urlretrieve("https://upload.wikimedia.org/wikipedia/commons/thumb/5/59/U
```

Out[8]:

```
('image1.jpg', <http.client.HTTPMessage at 0x18abef144f0>)
```

In [9]:

```
from PIL import Image

#read the image
im = Image.open("image1.jpg")

#show image
im.show()
```

In [10]:

```
# EXTRACTING TITLE
title = soup.find_all('title')
print(title[0].get_text())
```

```
Paragraph - Wikipedia
```

In [ ]:

```

```