In [1]:
```
!pip install bs4
!pip install requests
```

```
Requirement already satisfied: bs4 in c:\users\ebele okonkwo\anaconda3\lib\site-
packages (0.0.1)
Requirement already satisfied: beautifulsoup4 in c:\users\ebele okonkwo\anaconda
3\lib\site-packages (from bs4) (4.12.2)
Requirement already satisfied: soupsieve>1.2 in c:\users\ebele okonkwo\anaconda3
\lib\site-packages (from beautifulsoup4->bs4) (2.4)
Requirement already satisfied: requests in c:\users\ebele okonkwo\anaconda3\lib
\site-packages (2.31.0)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\ebele okonkw
o\anaconda3\lib\site-packages (from requests) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\ebele okonkwo\anaconda3
\lib\site-packages (from requests) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\ebele okonkwo\anac
onda3\lib\site-packages (from requests) (1.26.16)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\ebele okonkwo\anac
onda3\lib\site-packages (from requests) (2023.7.22)
```

In [2]:
```
import requests
from bs4 import BeautifulSoup
import pandas as pd
```

In [3]:
```
url = "https://en.wikipedia.org/wiki/Main_Page"
response = requests.get(url)
soup = BeautifulSoup(response.content, "html.parser")
header_tags = soup.find_all(["h1", "h2", "h3", "h4", "h5", "h6"])
header_texts = [tag.get_text() for tag in header_tags]

df = pd.DataFrame(header_texts, columns=["Header"])

print(df)
```

```
                          Header
0                      Main Page
1           Welcome to Wikipedia
2   From today's featured article
3                Did you know ...
4                     In the news
5                     On this day
6      From today's featured list
7        Today's featured picture
8        Other areas of Wikipedia
9     Wikipedia's sister projects
10            Wikipedia languages
```

In [4]:
```
page = requests.get("https://presidentofindia.nic.in/former-presidents.htm")
```

In [5]:
```
page
```

Out[5]:
```
<Response [404]>
```

In [6]:
```
page = requests.get("https://www.icc-cricket.com/rankings/team-rankings/mens/odi"
```

In [7]:
```
page
```

Out[7]:
```
<Response [200]>
```

In [8]:
```
soup = BeautifulSoup(page.content)
soup
```

```
(i.parentNode,"boomr-async");a=document.createElement("IFRAME"),a.src="about:
blank",a.title="",a.role="presentation",a.loading="eager",r=(a.frameElement||
a).style,r.width=0,r.height=0,r.border=0,r.display="none",i.parentNode.append
Child(a);try{O=a.contentWindow,d=O.document.open()}catch(_){n=document.domai
n,a.src="javascript:var d=document.open();d.domain='"+n+"';void(0);",O=a.cont
entWindow,d=O.document.open()}if(n)d._boomrl=function(){this.domain=n,t()},d.
write("<bo"+"dy onload='document._boomrl();'>");else if(O._boomrl=function()
{t()},O.addEventListener)O.addEventListener("load",O._boomrl,!1);else if(O.at
tachEvent)O.attachEvent("onload",O._boomrl);d.close()}function a(e){window.BO
OMR_onload=e&&e.timeStamp||(new Date).getTime()}if(!window.BOOMR||!window.BOO
MR.version&&!window.BOOMR.snippetExecuted){window.BOOMR=window.BOOMR||{},wind
ow.BOOMR.snippetStart=(new Date).getTime(),window.BOOMR.snippetExecuted=!0,wi
ndow.BOOMR.snippetVersion=12,window.BOOMR.url=n+"Q298E-BXCRP-TNXXB-UWJ6T-DH7Z
L";var i=document.currentScript||document.getElementsByTagName("script")[0],o
=!1,r=document.createElement("link");if(r.relList&&"function"==typeof r.relLi
st.supports&&r.relList.supports("preload")&&"as"in r)window.BOOMR.snippetMeth
od="p",r.href=window.BOOMR.url,r.rel="preload",r.as="script",r.addEventListen
er("load",e),r.addEventListener("error",function(){t(!0)}),setTimeout(functio
n(){if(!o)t(!0)},3e3),BOOMR_lstart=(new Date).getTime(),i.parentNode.appendCh
ild(r);else t(!1);if(window.addEventListener)window.addEventListener("load",
```

In [9]:
```
team = []
table = soup.find("table", class_="table")
rows = table.find_all("tr")

for row in rows[1:11]:
    cells = row.find_all("td")
    team = cells[1].text.strip()
    matches = cells[2].text.strip()
    points = cells[3].text.strip()
    rating = cells[4].text.strip()
    team.append([team, matches, points, rating])

df = pd.DataFrame(team, columns=["Team", "Matches", "Points", "Rating"])
print(df)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[9], line 3
      1 team = []
      2 table = soup.find("table", class_="table")
----> 3 rows = table.find_all("tr")
      5 for row in rows[1:11]:
      6     cells = row.find_all("td")

AttributeError: 'NoneType' object has no attribute 'find_all'
```

In [11]:
```
import requests
from bs4 import BeautifulSoup
import pandas as pd

url = "https://www.icc-cricket.com/rankings/mens/team-rankings/odi"
response = requests.get(url)
soup = BeautifulSoup(response.text, "html.parser")

teams = []

for team in soup.select("tbody tr"):
    name = team.select("td")[1].text.strip()
    matches = team.select("td")[2].text.strip()
    points = team.select("td")[3].text.strip()
    rating = team.select("td")[4].text.strip()
    teams.append((name, matches, points, rating))

df = pd.DataFrame(teams, columns=["Team", "Matches", "Points", "Rating"])
print(df)
```

```
Empty DataFrame
Columns: [Team, Matches, Points, Rating]
Index: []
```

In [12]:
```
url = "https://www.icc-cricket.com/rankings/mens/player-rankings/odi/batting"
response = requests.get(url)
soup = BeautifulSoup(response.content, "html.parser")

batsman = []
table = soup.find("table", class_="table")
row = table.find_all("tr")

for row in rows[1:11]:
    cells = row.find_all("td")
    batsman = cells[1].text.strip()
    team = cells[2].text.strip()
    rating = cells[3].text.strip()
    batsman_data.append([batsman, team, rating])
df = pd.DataFrame(batsman_data, columns=["Batsman", "Team", "Rating"])
print(df)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[12], line 7
      5 batsman = []
      6 table = soup.find("table", class_="table")
----> 7 row = table.find_all("tr")
      9 for row in rows[1:11]:
     10     cells = row.find_all("td")

AttributeError: 'NoneType' object has no attribute 'find_all'
```

In [13]:
```
url = "https://www.icc-cricket.com/rankings/mens/player-rankings/odi/bowling"
response = requests.get(url)
soup = BeautifulSoup(response.content, "html.parser")

bowler = []
table = soup.find("table", class_="table")
rows = table.find_all("tr")

for row in rows[1:11]:
    cells = row.find_all("td")
    bowler = cells[1].text.strip()
    team = cells[2].text.strip()
    rating = cells[3].text.strip()
    bowler.append([bowler, team, rating])

df = pd.DataFrame(bowler_data, columns=["Bowler", "Team", "Rating"])
print(df)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[13], line 7
      5 bowler = []
      6 table = soup.find("table", class_="table")
----> 7 rows = table.find_all("tr")
      9 for row in rows[1:11]:
     10     cells = row.find_all("td")

AttributeError: 'NoneType' object has no attribute 'find_all'
```

In [14]:

```
import requests
from bs4 import BeautifulSoup
import pandas as pd

# Scrape Top 10 ODI teams in women's cricket
url_teams = "https://www.icc-cricket.com/rankings/womens/team-rankings/odi"
response_teams = requests.get(url_teams)
soup_teams = BeautifulSoup(response_teams.content, "html.parser")

teams_data = []
table_teams = soup_teams.find("table", class_="table")
rows_teams = table_teams.find_all("tr")

for row in rows_teams[1:11]:
  team_name = row.find("span", class_="u-hide-phablet").text.strip()
  matches = row.find_all("td")[2].text.strip()
  points = row.find_all("td")[3].text.strip()
  rating = row.find_all("td")[4].text.strip()
  teams_data.append([team_name, matches, points, rating])

# Scrape Top 10 women's ODI Batting players
url_batting = "https://www.icc-cricket.com/rankings/womens/player-rankings/odi/bat
response_batting = requests.get(url_batting)
soup_batting = BeautifulSoup(response_batting.content, "html.parser")

batting_data = []
table_batting = soup_batting.find("table", class_="table")
rows_batting = table_batting.find_all("tr")

for row in rows_batting[1:11]:
  player_name = row.find("td", class_="table-body__cell rankings-table__name name
  team = row.find("span", class_="table-body__logo-text").text.strip()
  rating = row.find("td", class_="table-body__cell rating").text.strip()
  batting_data.append([player_name, team, rating])

# Scrape Top 10 women's ODI all-rounders
url_allrounders = "https://www.icc-cricket.com/rankings/womens/player-rankings/odi
response_allrounders = requests.get(url_allrounders)
soup_allrounders = BeautifulSoup(response_allrounders.content, "html.parser")

allrounders_data = []
table_allrounders = soup_allrounders.find("table", class_="table")
rows_allrounders = table_allrounders.find_all("tr")

for row in rows_allrounders[1:11]:
  player_name = row.find("td", class_="table-body__cell rankings-table__name name
  team = row.find("span", class_="table-body__logo-text").text.strip()
  rating = row.find("td", class_="table-body__cell rating").text.strip()
  allrounders_data.append([player_name, team, rating])

# Create data frames
df_teams = pd.DataFrame(teams_data, columns=["Team", "Matches", "Points", "Rating
df_batting = pd.DataFrame(batting_data, columns=["Player", "Team", "Rating"])
df_allrounders = pd.DataFrame(allrounders_data, columns=["Player", "Team", "Ratin

# Print the data frames
print("Top 10 ODI teams in women's cricket:")
print(df_teams)
print("\nTop 10 women's ODI Batting players:")
print(df_batting)
print("\nTop 10 women's ODI all-rounders:")
```

```
print(df_allrounders)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[14], line 12
     10 teams_data = []
     11 table_teams = soup_teams.find("table", class_="table")
---> 12 rows_teams = table_teams.find_all("tr")
     14 for row in rows_teams[1:11]:
     15   team_name = row.find("span", class_="u-hide-phablet").text.strip()

AttributeError: 'NoneType' object has no attribute 'find_all'
```

In [16]:
```
page = requests.get("https://www.cnbc.com/world/?region=world")
```

In [17]:
```
page
```

Out[17]:
```
<Response [200]>
```

In [18]:
```
articles = soup.find_all("div", class_="Card-titleContainer")
```

In [23]:

```
import requests
from bs4 import BeautifulSoup
import pandas as pd

url = "https://www.cnbc.com/world/?region=world"
response = requests.get(url)

soup = BeautifulSoup(response.content, "html.parser")

articles = soup.find_all("div", class_="Card-titleContainer")

headlines = []
times = []
links = []

for article in articles:
    headline = article.find("a").text.strip()
    headlines.append(headline)

    time = article.find("time").text.strip()
    times.append(time)

    link = article.find("a")["href"]
    links.append(link)

data = {"Headline": headlines,"Time": times,"News Link": links}
df = pd.DataFrame(data)

print(df)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[23], line 27
     24 headlines.append(headline)
     26 # Extract the time
---> 27 time = article.find("time").text.strip()
     28 times.append(time)
     30 # Extract the news link

AttributeError: 'NoneType' object has no attribute 'text'
```

In [24]:
```
import requests
from bs4 import BeautifulSoup
import pandas as pd

url = "https://www.journals.elsevier.com/artificial-intelligence/most-downloaded-a
response = requests.get(url)

soup = BeautifulSoup(response.content, "html.parser")

articles_container = soup.find("div", class_="pod-listing")

titles = []
authors = []
dates = []
urls = []
for article in articles_container.find_all("li"):

    title = article.find("h2").text.strip()
    author = article.find("span", class_="author-list").text.strip()
    date = article.find("span", class_="pod-listing-date").text.strip()
    url = article.find("a")["href"]

titles.append(title)
authors.append(author)
dates.append(date)
urls.append(url)

data = {"Paper Title": titles,"Authors": authors,"Published Date": dates,"Paper UI
df = pd.DataFrame(data)

# Print the dataframe
print(df)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[24], line 16
     14 dates = []
     15 urls = []
---> 16 for article in articles_container.find_all("li"):
     18     title = article.find("h2").text.strip()
     19     author = article.find("span", class_="author-list").text.strip()

AttributeError: 'NoneType' object has no attribute 'find_all'
```

In [25]:
```
import requests
from bs4 import BeautifulSoup
import pandas as pd
```

In [28]:
```
page = requests.get("https://www.dineout.co.in")
```

In [29]:
```
page
```

Out[29]:
```
<Response [200]>
```

In [30]:
```
loc = soup.find('div',class_="restnt-loc ellipsis")
```

In [31]:
```
loc
```

In [32]:
```
loc.text
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[32], line 1
----> 1 loc.text

AttributeError: 'NoneType' object has no attribute 'text'
```

In [33]:

```
import requests
from bs4 import BeautifulSoup
import pandas as pd

# Send a GET request to the website
url = "https://www.dineout.co.in"
response = requests.get(url)

# Create a BeautifulSoup object to parse the HTML content
soup = BeautifulSoup(response.content, 'html.parser')

# Find the elements containing the details you want to scrape
restaurant_names = soup.find_all('h2', class_='restnt-name ellipsis')
cuisines = soup.find_all('span', class_='double-line-ellipsis')
locations = soup.find_all('span', class_='double-line-ellipsis')
ratings = soup.find_all('span', class_='rating-value')
image_urls = soup.find_all('img', class_='img-responsive')

# Create empty lists to store the scraped data
restaurant_list = []
cuisine_list = []
location_list = []
rating_list = []
image_url_list = []

# Extract the data from the elements and append them to the respective lists
for name in restaurant_names:
  restaurant_list.append(name.text.strip())

for cuisine in cuisines:
  cuisine_list.append(cuisine.text.strip())

for location in locations:
  location_list.append(location.text.strip())

for rating in ratings:
  rating_list.append(rating.text.strip())

for image in image_urls:
  image_url_list.append(image['src'])

# Create a dictionary from the lists
data = {
  'Restaurant Name': restaurant_list,
  'Cuisine': cuisine_list,
  'Location': location_list,
  'Ratings': rating_list,
  'Image URL': image_url_list
}

# Create a dataframe from the dictionary
df = pd.DataFrame(data)

# Print the dataframe
print(df)
```

```
Empty DataFrame
Columns: [Restaurant Name, Cuisine, Location, Ratings, Image URL]
Index: []
```

In [ ]: