In [1]:

```
import  requests
from  bs4 import BeautifulSoup

url = requests.get('https://wikipedia.org')
soup = BeautifulSoup(url.text, 'html.parser')
story = soup.find_all(['h1', 'h2','h3'])
for i in story:
    print(i)
```

```
<h1 class="central-textlogo-wrapper">
<span class="central-textlogo__image sprite svg-Wikipedia_wordmark">
Wikipedia
</span>
<strong class="jsl10n localized-slogan" data-jsl10n="portal.slogan">The Free
Encyclopedia</strong>
</h1>
<h2 class="bookshelf-container">
<span class="bookshelf">
<span class="text">
<bdi dir="ltr">
1 000 000+
</bdi>
<span class="jsl10n" data-jsl10n="entries">
articles
</span>
</span>
</span>
</h2>
<h2 class="bookshelf-container">
<span class="bookshelf">
<span class="text">
<bdi dir="ltr">
100 000+
</bdi>
<span class="jsl10n" data-jsl10n="portal.entries">
articles
</span>
</span>
</span>
</h2>
<h2 class="bookshelf-container">
<span class="bookshelf">
<span class="text">
<bdi dir="ltr">
10 000+
</bdi>
<span class="jsl10n" data-jsl10n="portal.entries">
articles
</span>
</span>
</span>
</h2>
<h2 class="bookshelf-container">
<span class="bookshelf">
<span class="text">
<bdi dir="ltr">
1 000+
</bdi>
<span class="jsl10n" data-jsl10n="portal.entries">
articles
</span>
</span>
</span>
</h2>
<h2 class="bookshelf-container">
<span class="bookshelf">
```

```
<span class="text">
<bdi dir="ltr">
100+
</bdi>
<span class="jsl10n" data-jsl10n="portal.entries">
articles
</span>
</span>
</span>
</h2>
<h3>Wikipedia is not for sale.</h3>
```

In [ ]:

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

# Send a GET request to the website
url = "https://presidentofindia.nic.in/former-presidents.htm"
response = requests.get(url)

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

# Find the table containing the information
table = soup.find("table")

# Create empty lists to store the data
names = []
terms = []

# Iterate over each row in the table
for row in table.find_all("tr")[1:]:
  # Extract the name and term of office from the columns
  columns = row.find_all("td")
  name = columns[0].text.strip()
  term = columns[1].text.strip()

  # Append the data to the respective lists
  names.append(name)
  terms.append(term)

# Create a data frame using the lists
data = {"Name": names, "Term of Office": terms}
df = pd.DataFrame(data)

# Display the data frame
print(df)
```

In [3]:

```
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.content, "html.parser")

team_data = []
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_data.append([team, matches, points, rating])

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

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

batsman_data = []
table = soup.find("table", class_="table")
rows = 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)

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

bowler_data = []
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_data.append([bowler, team, rating])

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

```
print(df)
```

```
              Team Matches Points Rating
0       India\nIND      55  6,640    121
1    Australia\nAUS      42  4,926    117
2  South Africa\nSA      34  3,750    110
3     Pakistan\nPAK      36  3,922    109
4   New Zealand\nNZ      43  4,399    102
5      England\nENG      38  3,777     99
6     Sri Lanka\nSL      47  4,134     88
7   Bangladesh\nBAN      44  3,836     87
8  Afghanistan\nAFG      30  2,533     84
9   West Indies\nWI      38  2,582     68
                Batsman Team Rating
0          Shubman Gill  IND    826
1            Babar Azam  PAK    824
2           Virat Kohli  IND    791
3          Rohit Sharma  IND    769
4       Quinton de Kock   SA    760
5        Daryl Mitchell   NZ    750
6          David Warner  AUS    745
7  Rassie van der Dussen   SA    735
8          Harry Tector  IRE    729
9           Dawid Malan  ENG    729
          Bowler Team Rating
0  Keshav Maharaj   SA    741
1  Josh Hazlewood  AUS    703
2  Mohammed Siraj  IND    699
3  Jasprit Bumrah  IND    685
4     Adam Zampa  AUS    675
5    Rashid Khan  AFG    667
6   Kuldeep Yadav  IND    667
7    Trent Boult   NZ    663
8  Shaheen Afridi  PAK    650
9  Mohammad Shami  IND    648
```

In [6]:

```
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/
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 na
  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/
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 na
  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", "Rati
df_batting = pd.DataFrame(batting_data, columns=["Player", "Team", "Rating"])
df_allrounders = pd.DataFrame(allrounders_data, columns=["Player", "Team", "Rat

# 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[6], line 31
     28 rows_batting = table_batting.find_all("tr")
     30 for row in rows_batting[1:11]:
---> 31   player_name = row.find("td", class_="table-body__cell rankings-tabl
e__name name").text.strip()
     32   team = row.find("span", class_="table-body__logo-text").text.strip
()
     33   rating = row.find("td", class_="table-body__cell rating").text.stri
p()

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

In [7]:

```
import requests
from bs4 import BeautifulSoup

url='https://www.bbc.com/news'
response = requests.get(url)

soup = BeautifulSoup(response.text, 'html.parser')
headlines = soup.find('body').find_all('h3')
for x in headlines:
    print(x.text.strip())
```

Israel says more hostages released from Gaza on sixth day of truce
US says it foiled alleged plot to kill Sikh activist
North Korea says it's got eyes on the White House - so what?
How 'rat-hole' miners freed Indian tunnel workers
'Rock star' pandas - not exactly a love story
Two Palestinian boys killed during West Bank raid
US military plane carrying six crashes off Japan
The story behind this heartwarming father-son kiss
Key Dutch party sees 'no basis' for talks with Wilders
Beyoncé's mum hits out at skin lightening comments
South Korea suspend Hwang over illegal filming claim
Beyoncé's mum hits out at skin lightening comments
South Korea suspend Hwang over illegal filming claim
Cristiano Ronaldo faces $1bn crypto ad lawsuit
Rare Sumatran rhino born in Indonesia
COP28 president denies using summit for oil deals
IDF investigating Hamas claims baby hostage was killed
Who are the released hostages?
UN fears for displaced Gazans as winter approaches
Israeli captives tell of fear, squalor and hunger
Boy, 14, and freelance reporter among Palestinians freed
Scobie book: How did the royal naming mistake happen?
BBC World News TV
BBC World Service Radio
Is Greta right? Is COP just ‘blah blah blah’?
'Every Greek felt offended' - how Athens sees the Elgin Marbles row
Spotify Wrapped: Are music genres irrelevant now?
'I was sold into marriage for £7 at the age of 12'
Finding happiness after the Clutha helicopter crash
World's 'saddest' elephant dies in Philippines zoo
Watch: Bride leaves wheelchair to walk down aisle
Teen steers runaway bus away from petrol pumps
Feminist site Jezebel to be revived after sale
Japan volcano spews ash and rock 200m into sky
In pictures: Bolivians care for animals affected by wildfires
Former US first ladies honour Rosalynn Carter
What wayfinding can teach us
How women could save the beer industry
The A-list retreat on a windswept rock
How a 1574 portrait was made 'Insta-fabulous'
The secret nursery of hammerhead sharks
The false promise of pay-range listings
The 2,000-year-old tamale few know
News daily newsletter
Mobile app
Get in touch

In [9]:

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

# Send a GET request to the URL
url = "https://www.journals.elsevier.com/artificial-intelligence/most-downloade
response = requests.get(url)

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

# Find the container that holds the article details
articles_container = soup.find("div", class_="pod-listing")

# Initialize empty lists to store the scraped data
titles = []
authors = []
dates = []
urls = []

# Iterate over each article in the container
for article in articles_container.find_all("li"):
  # Scrape the title
  title = article.find("h3").text.strip()
  titles.append(title)

  # Scrape the authors
  author = article.find("span", class_="text-xs").text.strip()
  authors.append(author)

  # Scrape the published date
  date = article.find("span", class_="text-xs").find_next_sibling("span").text.
  dates.append(date)

  # Scrape the paper URL
  url = article.find("a")["href"]
  urls.append(url)

# Create a dataframe with the scraped data
data = {
  "Paper Title": titles,
  "Authors": authors,
  "Published Date": dates,
  "Paper URL": urls
}
df = pd.DataFrame(data)

# Print the dataframe
print(df)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[9], line 22
     19 urls = []
     21 # Iterate over each article in the container
---> 22 for article in articles_container.find_all("li"):
     23   # Scrape the title
     24   title = article.find("h3").text.strip()
     25   titles.append(title)

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

In [10]:

```
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 [ ]: