In [ ]:
```
# FlipRobo
# NAME- Neha Somvanshi
# TITLE -Web Scraping Assignment-1
```

In [4]:
```
import requests
```

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

In [276…
```
#Q1- Write a python program to display all the header tags from wikipedia.org and make

url="https://en.wikipedia.org/wiki/Main_Page"
r=requests.get(url)
htmlcontent=r.content
soup=BeautifulSoup(htmlcontent,'html.parser')

headers= soup.find_all('span',class_="mw-headline")
df=pd.DataFrame({'headers':headers})
df
```

Out[276]:

| | headers |
|---|---|
| 0 | [Welcome to , [Wikipedia]] |
| 1 | [From today's featured article] |
| 2 | [Did you know ...] |
| 3 | [In the news] |
| 4 | [On this day] |
| 5 | [Today's featured picture] |
| 6 | [Other areas of Wikipedia] |
| 7 | [Wikipedia's sister projects] |
| 8 | [Wikipedia languages] |

In [277…
```
#Q2  Write s python program to display list of respected former presidents of India(i.
#from https://presidentofindia.nic.in/former-presidents.htm and make data frame.

url="https://presidentofindia.nic.in/former-presidents"
r=requests.get(url)
htmlcontent=r.content
soup=BeautifulSoup(htmlcontent,'html.parser')

president_names=soup.find_all('div',class_="desc-sec")
for names in president_names:
    print(names.text)
```

```
Shri Ram Nath Kovind
14th President of India


Shri Pranab Mukherjee
13th President of India


Smt Pratibha Devisingh Patil
12th President of India


DR. A.P.J. Abdul Kalam
11th President of India


Shri K. R. Narayanan
10th President of India


Dr Shankar Dayal Sharma
9th  President of India


Shri R Venkataraman
8th President of India


Giani Zail Singh
7th President of India


Shri Neelam Sanjiva Reddy
6th President of India


Dr. Fakhruddin Ali Ahmed
5th President of India


Shri Varahagiri Venkata Giri
4th President of India


Dr. Zakir Husain
3rd President of India


Dr. Sarvepalli Radhakrishnan
2nd President of India


Dr. Rajendra Prasad
1st President of India
```

In [7]:

```
#Q3  Write a python program to scrape cricket rankings from icc-cricket.com. You have
#a- Top 10 ODI teams in men's cricket along with the records for matches, points and r

url="https://www.icc-cricket.com/rankings/mens/team-rankings/odi"
```

```
r=requests.get(url)
htmlcontent=r.content
soup=BeautifulSoup(htmlcontent,'html.parser')

top_10_odi_teams= soup.find_all("span", class_="u-hide-phablet")[:10]
df=pd.DataFrame({'top_10_odi_teams':top_10_odi_teams})
df
```

Out[7]:

| | top_10_odi_teams |
|---|---|
| **0** | [Australia] |
| **1** | [Pakistan] |
| **2** | [India] |
| **3** | [New Zealand] |
| **4** | [England] |
| **5** | [South Africa] |
| **6** | [Bangladesh] |
| **7** | [Afghanistan] |
| **8** | [Sri Lanka] |
| **9** | [West Indies] |

In [5]:

```
# Q3
#b-Top 10 ODI Batsmen along with the records of their team and rating

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

top_10_odi_batsmen = soup.find_all("td", class_="table-body__cell name")[:10]
team=soup.find_all("span", class_="table-body__logo-text")[:10]
rating=soup.find_all("td", class_="table-body__cell u-text-right rating")[:10]

df=pd.DataFrame({'top_10_odi_batsmen':top_10_odi_batsmen,'team':team,'rating':rating})
df
```

Out[5]:

| | top_10_odi_batsmen | team | rating |
|---|---|---|---|
| 0 | [\n, [Rassie van der Dussen], \n] | [SA] | [777] |
| 1 | [\n, [Fakhar Zaman], \n] | [PAK] | [755] |
| 2 | [\n, [Imam-ul-Haq], \n] | [PAK] | [745] |
| 3 | [\n, [Shubman Gill], \n] | [IND] | [743] |
| 4 | [\n, [Harry Tector], \n] | [IRE] | [726] |
| 5 | [\n, [David Warner], \n] | [AUS] | [726] |
| 6 | [\n, [Quinton de Kock], \n] | [SA] | [718] |
| 7 | [\n, [Virat Kohli], \n] | [IND] | [705] |
| 8 | [\n, [Steve Smith], \n] | [AUS] | [702] |
| 9 | [\n, [Mitchell Starc], \n] | [AUS] | [686] |

In [6]:
```
#Q3
#c- Top 10 ODI bowlers along with the records of their team andrating.

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

top_10_odi_bowlers = soup.find_all("td", class_="table-body__cell rankings-table__name
team=soup.find_all("span", class_="table-body__logo-text")[:10]
rating=soup.find_all("td", class_="table-body__cell rating")[:10]

df=pd.DataFrame({'top_10_odi_bowlers':top_10_odi_bowlers,'team':team,'rating':rating})
df
```

Out[6]:

| | top_10_odi_bowlers | team | rating |
|---|---|---|---|
| 0 | [\n, [Mitchell Starc], \n] | [AUS] | [686] |
| 1 | [\n, [Rashid Khan], \n] | [AFG] | [682] |
| 2 | [\n, [Mohammed Siraj], \n] | [IND] | [670] |
| 3 | [\n, [Matt Henry], \n] | [NZ] | [667] |
| 4 | [\n, [Mujeeb Ur Rahman], \n] | [AFG] | [661] |
| 5 | [\n, [Trent Boult], \n] | [NZ] | [660] |
| 6 | [\n, [Adam Zampa], \n] | [AUS] | [652] |
| 7 | [\n, [Shaheen Afridi], \n] | [PAK] | [630] |
| 8 | [\n, [Kuldeep Yadav], \n] | [IND] | [622] |
| 9 | [\n, [Shakib Al Hasan], \n] | [BAN] | [618] |

In [10]:
```
#Q4 rite a python program to scrape cricket rankings from icc-cricket.com. You have to
#a) Top 10 ODI teams in women's cricket along with the records for matches, points and

url="https://www.icc-cricket.com/rankings/womens/player-rankings/odi/all-rounder"
```

```
r=requests.get(url)
htmlcontent=r.content
soup=BeautifulSoup(htmlcontent,'html.parser')

ODI_teams= soup.find_all("span",class_="u-show-phablet")[:10]
matches=soup.find_all("td",class_="rankings-block__banner--matches")[:10]
points=soup.find_all("span",class_="rankings-block__banner--points")[:10]
rating=soup.find_all("td", class_="rankings-block__banner--rating u-text-right")[:10]

df=pd.DataFrame({'ODI_teams':ODI_teams,'matches':matches,'points':points,'rating':rati
df
```

Out[10]:

| | ODI_teams | matches | points | rating |
|---|---|---|---|---|

In [8]:

```
#Q4
#b) Top 10 women's ODI Batting players along with the records of their team and rating

url="https://www.icc-cricket.com/rankings/womens/player-rankings/odi"
r=requests.get(url)
htmlcontent=r.content
soup=BeautifulSoup(htmlcontent,'html.parser')

women_ODI_Batting= soup.find_all("td",class_="table-body__cell name")[:10]
team=soup.find_all("span", class_="table-body__logo-text")[:10]
rating=soup.find_all("td", class_="table-body__cell u-text-right rating")[:10]

df=pd.DataFrame({'women_ODI_Batting':women_ODI_Batting,'team':team,'rating':rating})
df
```

Out[8]:

| | women_ODI_Batting | team | rating |
|---|---|---|---|
| **0** | [\n, [Chamari Athapaththu], \n] | [SL] | [758] |
| **1** | [\n, [Beth Mooney], \n] | [AUS] | [751] |
| **2** | [\n, [Laura Wolvaardt], \n] | [SA] | [732] |
| **3** | [\n, [Smriti Mandhana], \n] | [IND] | [708] |
| **4** | [\n, [Alyssa Healy], \n] | [AUS] | [702] |
| **5** | [\n, [Harmanpreet Kaur], \n] | [IND] | [694] |
| **6** | [\n, [Ellyse Perry], \n] | [AUS] | [686] |
| **7** | [\n, [Meg Lanning], \n] | [AUS] | [682] |
| **8** | [\n, [Stafanie Taylor], \n] | [WI] | [618] |
| **9** | [\n, [Shabnim Ismail], \n] | [SA] | [722] |

In [9]:

```
#Q4
# c) Top 10 women's ODI all-rounder along with the records of their team and rating.

url="https://www.icc-cricket.com/rankings/womens/player-rankings/odi/all-rounder"
r=requests.get(url)
htmlcontent=r.content
soup=BeautifulSoup(htmlcontent,'html.parser')

women_ODI_all_rounder= soup.find_all("td",class_="table-body__cell rankings-table__nam
```

```
team=soup.find_all("span", class_="table-body__logo-text")[:10]
rating=soup.find_all("td", class_="table-body__cell rating")[:10]

df=pd.DataFrame({'women_ODI_all_rounder':women_ODI_all_rounder,'team':team,'rating':ra
df
```

Out[9]:

| | women_ODI_all_rounder | team | rating |
|---|---|---|---|
| **0** | [\n, [Ashleigh Gardner], \n] | [AUS] | [389] |
| **1** | [\n, [Hayley Matthews], \n] | [WI] | [382] |
| **2** | [\n, [Marizanne Kapp], \n] | [SA] | [349] |
| **3** | [\n, [Ellyse Perry], \n] | [AUS] | [329] |
| **4** | [\n, [Amelia Kerr], \n] | [NZ] | [328] |
| **5** | [\n, [Deepti Sharma], \n] | [IND] | [312] |
| **6** | [\n, [Jess Jonassen], \n] | [AUS] | [241] |
| **7** | [\n, [Sophie Devine], \n] | [NZ] | [233] |
| **8** | [\n, [Nida Dar], \n] | [PAK] | [232] |
| **9** | [\n, [Sophie Ecclestone], \n] | [ENG] | [200] |

In [281…

```
#Q5 Write a python program to scrape mentioned news details from https://www.cnbc.com/
#make data frame
#i) Headline
#ii) Time

url='https://www.cnbc.com/world/?region=world'
r=requests.get(url)
htmlcontent=r.content
soup=BeautifulSoup(htmlcontent,'html.parser')

headline= soup.find_all('div', class_="LatestNews-headlineWrapper")
time= soup.find_all('time', class_="LatestNews-timestamp")
df = pd.DataFrame({'Headline': headline,'time':time})
df = df.drop([28, 29])
df
```

Out[281]:

| | Headline | time |
|---|---|---|
| 0 | [[[25 Min Ago]], [Russia's inflation spike set... | [25 Min Ago] |
| 1 | [[[33 Min Ago]], [], [European markets set for... | [33 Min Ago] |
| 2 | [[[49 Min Ago]], [CNBC Daily Open: Inflation a... | [49 Min Ago] |
| 3 | [[[3 Hours Ago]], [Japan trade data dims growt... | [3 Hours Ago] |
| 4 | [[[3 Hours Ago]], [Walmart will report earning... | [3 Hours Ago] |
| 5 | [[[6 Hours Ago]], [Missiles aren't the only th... | [6 Hours Ago] |
| 6 | [[[6 Hours Ago]], [China's premier says countr... | [6 Hours Ago] |
| 7 | [[[6 Hours Ago]], [Asia promotes crypto clarit... | [6 Hours Ago] |
| 8 | [[[7 Hours Ago]], [[]], [Ark Invest is 'focuse... | [7 Hours Ago] |
| 9 | [[[7 Hours Ago]], [[]], [BofA names top out-of... | [7 Hours Ago] |
| 10 | [[[7 Hours Ago]], [[]], [China's economy is te... | [7 Hours Ago] |
| 11 | [[[7 Hours Ago]], [CNBC Daily Open: Inflation ... | [7 Hours Ago] |
| 12 | [[[7 Hours Ago]], [], [Asia markets mixed afte... | [7 Hours Ago] |
| 13 | [[[7 Hours Ago]], [Don't count out more rate h... | [7 Hours Ago] |
| 14 | [[[8 Hours Ago]], [Cramer's Lightning Round: R... | [8 Hours Ago] |
| 15 | [[[8 Hours Ago]], [5 market theories Cramer th... | [8 Hours Ago] |
| 16 | [[[8 Hours Ago]], [Cramer examines the potenti... | [8 Hours Ago] |
| 17 | [[[9 Hours Ago]], [], [Stock futures are littl... | [9 Hours Ago] |
| 18 | [[[9 Hours Ago]], [Stocks making the biggest m... | [9 Hours Ago] |
| 19 | [[[10 Hours Ago]], [New Covid boosters are on ... | [10 Hours Ago] |
| 20 | [[[10 Hours Ago]], [Record ocean temperatures:... | [10 Hours Ago] |
| 21 | [[[10 Hours Ago]], [Biden takes a victory lap ... | [10 Hours Ago] |
| 22 | [[[10 Hours Ago]], [[]], [Target shares pop ne... | [10 Hours Ago] |
| 23 | [[[10 Hours Ago]], [Inflation Reduction Act sp... | [10 Hours Ago] |
| 24 | [[[10 Hours Ago]], [Wegovy could prevent up to... | [10 Hours Ago] |
| 25 | [[[10 Hours Ago]], [[]], [Apple's ongoing pivo... | [10 Hours Ago] |
| 26 | [[[11 Hours Ago]], [A key part of success lots... | [11 Hours Ago] |
| 27 | [[[11 Hours Ago]], [Federal judge asks if Elon... | [11 Hours Ago] |

In [282…

```
#Q6 Write a python program to scrape the details of most downloaded articles from AI i
#days.https://www.journals.elsevier.com/artificial-intelligence/most-downloaded-articl
#Scrape below mentioned details and make data frame i) Paper Title ii) Authors iii) Pu

url='https://www.journals.elsevier.com/artificial-intelligence/most-downloaded-article
r=requests.get(url)
htmlcontent=r.content
soup=BeautifulSoup(htmlcontent,'html.parser')
```

```
Titles=soup.find_all('h2',class_="sc-1qrq3sd-1 gRGSUS sc-1nmom32-0 sc-1nmom32-1 btcbYu
authors=soup.find_all('span',class_="sc-1w3fpd7-0 dnCnAO")
date=soup.find_all('span',class_="sc-1thf9ly-2 dvggWt")


df=pd.DataFrame({'Titles':Titles,'authors':authors,'date':date})
df
```

Out[282]:

| | Titles | authors | date |
|---|---|---|---|
| **0** | [Reward is enough] | [David Silver, Satinder Singh, Doina Precup, R... | [[October 2021]] |
| **1** | [Explanation in artificial intelligence: Insig... | [Tim Miller ] | [[February 2019]] |
| **2** | [Creativity and artificial intelligence] | [Margaret A. Boden ] | [[August 1998]] |
| **3** | [Conflict-based search for optimal multi-agent... | [Guni Sharon, Roni Stern, Ariel Felner, Nathan... | [[February 2015]] |
| **4** | [Knowledge graphs as tools for explainable mac... | [Ilaria Tiddi, Stefan Schlobach ] | [[January 2022]] |
| **5** | [Law and logic: A review from an argumentation... | [Henry Prakken, Giovanni Sartor ] | [[October 2015]] |
| **6** | [Between MDPs and semi-MDPs: A framework for t... | [Richard S. Sutton, Doina Precup, Satinder Sin... | [[August 1999]] |
| **7** | [Explaining individual predictions when featur... | [Kjersti Aas, Martin Jullum, Anders Løland ] | [[September 2021]] |
| **8** | [Multiple object tracking: A literature review] | [Wenhan Luo, Junliang Xing and 4 more] | [[April 2021]] |
| **9** | [A survey of inverse reinforcement learning: C... | [Saurabh Arora, Prashant Doshi ] | [[August 2021]] |
| **10** | [Evaluating XAI: A comparison of rule-based an... | [Jasper van der Waa, Elisabeth Nieuwburg, Anit... | [[February 2021]] |
| **11** | [Explainable AI tools for legal reasoning abou... | [Joe Collenette, Katie Atkinson, Trevor Bench-... | [[April 2023]] |
| **12** | [Hard choices in artificial intelligence] | [Roel Dobbe, Thomas Krendl Gilbert, Yonatan Mi... | [[November 2021]] |
| **13** | [Assessing the communication gap between AI mo... | [Oskar Wysocki, Jessica Katharine Davies and 5... | [[March 2023]] |
| **14** | [Explaining black-box classifiers using post-h... | [Eoin M. Kenny, Courtney Ford, Molly Quinn, Ma... | [[May 2021]] |
| **15** | [The Hanabi challenge: A new frontier for AI r... | [Nolan Bard, Jakob N. Foerster and 13 more] | [[March 2020]] |
| **16** | [Wrappers for feature subset selection] | [Ron Kohavi, George H. John ] | [[December 1997]] |
| **17** | [Artificial cognition for social human–robot i... | [Séverin Lemaignan, Mathieu Warnier and 3 more] | [[June 2017]] |
| **18** | [A review of possible effects of cognitive bia... | [Tomáš Kliegr, Štěpán Bahník, Johannes Fürnkra... | [[June 2021]] |
| **19** | [The multifaceted impact of Ada Lovelace in th... | [Luigia Carlucci Aiello ] | [[June 2016]] |
| **20** | [Robot ethics: Mapping the issues for a mechan... | [Patrick Lin, Keith Abney, George Bekey ] | [[April 2011]] |
| **21** | [Reward (Mis)design for autonomous driving] | [W. Bradley Knox, Alessandro Allievi and 3 more] | [[March 2023]] |

| | Titles | authors | date |
|---|---|---|---|
| **22** | [Planning and acting in partially observable s... | [Leslie Pack Kaelbling, Michael L. Littman, An... | [[May 1998]] |
| **23** | [What do we want from Explainable Artificial I... | [Markus Langer, Daniel Oster and 6 more] | [[July 2021]] |

In [286...

```
#Q7 Write a python program to scrape mentioned details from dineout.co.inand make data

url='https://www.dineout.co.in/delhi-restaurants/buffet-special'
r=requests.get(url)
htmlcontent=r.content
soup=BeautifulSoup(htmlcontent,'html.parser')

restaurant_name = soup.find_all('div', class_='restnt-info cursor')
rating=soup.find_all('div',class_="restnt-rating rating-4")
image=soup.find_all('img',class_="no-img")

data_srcs = [img.get('data-src') for img in image]


df = pd.DataFrame({'Restaurant_name': restaurant_name,'rating':rating,'image_url':data
df
```

Out[286]:

| | Restaurant_name | rating | image_url |
|---|---|---|---|
| **0** | [[Castle Barbeque], [[Connaught Place], , , [C... | [4] | https://im1.dineout.co.in/images/uploads/resta... |
| **1** | [[Cafe Knosh], [[The Leela Ambience Convention... | [4.3] | https://im1.dineout.co.in/images/uploads/resta... |
| **2** | [[Castle's Barbeque], [[Pacific Mall,], [Tagor... | [3.9] | https://im1.dineout.co.in/images/uploads/resta... |
| **3** | [[The Barbeque Company], [[Gardens Galleria,],... | [3.9] | https://im1.dineout.co.in/images/uploads/resta... |
| **4** | [[India Grill], [[Hilton Garden Inn,], [Saket]... | [3.9] | https://im1.dineout.co.in/images/uploads/resta... |
| **5** | [[Delhi Barbeque], [[Taurus Sarovar Portico,],... | [3.7] | https://im1.dineout.co.in/images/uploads/resta... |
| **6** | [[The Monarch - Bar Be Que Village], [[Indirap... | [3.8] | https://im1.dineout.co.in/images/uploads/resta... |
| **7** | [[The Barbeque Times], [[M2K Corporate Park,],... | [4.1] | https://im1.dineout.co.in/images/uploads/resta... |
| **8** | [[Indian Grill Room], [[Suncity Business Tower... | [4.3] | https://im1.dineout.co.in/images/uploads/resta... |

In [ ]:

```
#END
```