In [9]:

```
!pip install bs4
!pip install requests
```

```
Requirement already satisfied: bs4 in c:\users\hp\anaconda3\lib\site-packages (0.0.1)
Requirement already satisfied: beautifulsoup4 in c:\users\hp\anaconda3\lib\site-packages (from bs4) (4.10.0)
Requirement already satisfied: soupsieve>1.2 in c:\users\hp\anaconda3\lib\site-packages (from beautifulsoup4->bs4) (2.2.1)
Requirement already satisfied: requests in c:\users\hp\anaconda3\lib\site-packages (2.26.0)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\hp\anaconda3\lib\site-packages (from requests) (2021.10.8)
Requirement already satisfied: idna<4,>=2.5 in c:\users\hp\anaconda3\lib\site-packages (from requests) (3.2)
Requirement already satisfied: charset-normalizer~=2.0.0 in c:\users\hp\anaconda3\lib\site-packages (from requests) (2.0.4)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\hp\anaconda3\lib\site-packages (from requests) (1.26.7)
```

In [1]:

```
from bs4 import BeautifulSoup
import requests
```

In [2]:

```
#Write a python program to display all the header tags from wikipedia.org.
wikipage = requests.get("https://en.wikipedia.org/wiki/Main_Page")
```

In [3]:

```
#wikipage
```

In [4]:

```
wikicontent = BeautifulSoup(wikipage.content)
```

In [5]:

```
wikititles = wikicontent.find_all(['h1','h2','h3','h3','h4','h5','h6'])
wikititles
```

Out[5]:

```
[<h1 class="firstHeading mw-first-heading" id="firstHeading" style="display: none">Main Page</h1>,
 <h1><span class="mw-headline" id="Welcome_to_Wikipedia">Welcome to <a href="/wiki/Wikipedia" title="Wikipedia">Wikipedia</a></span></h1>,
 <h2 class="mp-h2" id="mp-tfa-h2"><span id="From_today.27s_featured_article"></span><span class="mw-headline" id="From_today's_featured_article">From today's featured article</span></h2>,
 <h2 class="mp-h2" id="mp-dyk-h2"><span class="mw-headline" id="Did_you_know_...">Did you know ...</span></h2>,
 <h2 class="mp-h2" id="mp-itn-h2"><span class="mw-headline" id="In_the_news">In the news</span></h2>,
 <h2 class="mp-h2" id="mp-otd-h2"><span class="mw-headline" id="On_this_day">On this day</span></h2>,
 <h2 class="mp-h2" id="mp-tfp-h2"><span id="Today.27s_featured_picture"></span><span class="mw-headline" id="Today's_featured_picture">Today's featured picture</span></h2>,
 <h2 class="mp-h2" id="mp-other"><span class="mw-headline" id="Other_areas_of_Wikipedia">Other areas of Wikipedia</span></h2>,
 <h2 class="mp-h2" id="mp-sister"><span id="Wikipedia.27s_sister_projects"></span><span class="mw-headline" id="Wikipedia's_sister_projects">Wikipedia's sister projects</span></h2>,
 <h2 class="mp-h2" id="mp-lang"><span class="mw-headline" id="Wikipedia_languages">Wikipedia languages</span></h2>,
 <h2>Navigation menu</h2>,
 <h3 class="vector-menu-heading" id="p-personal-label">
 <span class="vector-menu-heading-label">Personal tools</span>
 </h3>,
 <h3 class="vector-menu-heading" id="p-namespaces-label">
 <span class="vector-menu-heading-label">Namespaces</span>
 </h3>,
 <h3 class="vector-menu-heading" id="p-views-label">
 <span class="vector-menu-heading-label">Views</span>
 </h3>,
 <h3>
 <label for="searchInput">Search</label>
 </h3>,
 <h3 class="vector-menu-heading" id="p-navigation-label">
 <span class="vector-menu-heading-label">Navigation</span>
 </h3>,
 <h3 class="vector-menu-heading" id="p-interaction-label">
 <span class="vector-menu-heading-label">Contribute</span>
 </h3>,
 <h3 class="vector-menu-heading" id="p-tb-label">
 <span class="vector-menu-heading-label">Tools</span>
 </h3>,
 <h3 class="vector-menu-heading" id="p-coll-print_export-label">
 <span class="vector-menu-heading-label">Print/export</span>
 </h3>,
 <h3 class="vector-menu-heading" id="p-wikibase-otherprojects-label">
 <span class="vector-menu-heading-label">In other projects</span>
 </h3>,
 <h3 class="vector-menu-heading" id="p-lang-label">
 <span class="vector-menu-heading-label">Languages</span>
 </h3>]
```

In [6]:

```
#Write a python program to display IMDB's Top rated 100 movies' data
#(i.e. name, rating, year of release) and make data frame
IMDB1 = requests.get('https://www.imdb.com/search/title/?groups=top_100&sort=user_rating,desc')
#IMDB1
```

In [7]:

```
IMDB100 = BeautifulSoup(IMDB1.content)
IMDBtitles = []
for i in IMDB100.find_all('h3',class_="lister-item-header"):
    movname = i.findChildren('a')
    for j in movname:
        IMDBtitles.append(j.text)
#IMDBtitles
```

In [8]:

```
IMDByears = []
for i in IMDB100.find_all('span',class_="lister-item-year"):
    IMDByears.append(i.text)
#IMDByears
```

In [9]:

```
IMDBrating = []
for i in IMDB100.find_all('div',class_= "ratings-imdb-rating"):
    IMDBrating.append(i.text.split()[0])
#IMBDrating
```

In [10]:

```
import pandas as pd
df = pd.DataFrame({'Movie name':IMDBtitles, 'Years':IMDByears, 'Rating':IMDBrating})
df
```

Out[10]:

	Movie name	Years	Rating
0	The Shawshank Redemption	(1994)	9.3
1	The Godfather	(1972)	9.2
2	The Dark Knight	(2008)	9.0
3	The Lord of the Rings: The Return of the King	(2003)	9.0
4	Schindler's List	(1993)	9.0
5	The Godfather Part II	(1974)	9.0
6	12 Angry Men	(1957)	9.0
7	Pulp Fiction	(1994)	8.9
8	Inception	(2010)	8.8
9	The Lord of the Rings: The Two Towers	(2002)	8.8
10	Fight Club	(1999)	8.8
11	The Lord of the Rings: The Fellowship of the Ring	(2001)	8.8
12	Forrest Gump	(1994)	8.8
13	Il buono, il brutto, il cattivo	(1966)	8.8
14	The Matrix	(1999)	8.7
15	Goodfellas	(1990)	8.7
16	The Empire Strikes Back	(1980)	8.7
17	One Flew Over the Cuckoo's Nest	(1975)	8.7
18	Top Gun: Maverick	(2022)	8.6
19	Interstellar	(2014)	8.6
20	Cidade de Deus	(2002)	8.6
21	Sen to Chihiro no kamikakushi	(2001)	8.6
22	Saving Private Ryan	(1998)	8.6
23	The Green Mile	(1999)	8.6
24	La vita è bella	(1997)	8.6
25	Se7en	(1995)	8.6
26	Terminator 2: Judgment Day	(1991)	8.6
27	The Silence of the Lambs	(1991)	8.6
28	Star Wars	(1977)	8.6
29	Seppuku	(1962)	8.6
30	Shichinin no samurai	(1954)	8.6
31	It's a Wonderful Life	(1946)	8.6
32	Gisaengchung	(2019)	8.5
33	Whiplash	(2014)	8.5
34	The Intouchables	(2011)	8.5
35	The Prestige	(2006)	8.5
36	The Departed	(2006)	8.5
37	The Pianist	(2002)	8.5
38	Gladiator	(2000)	8.5
39	American History X	(1998)	8.5
40	The Usual Suspects	(1995)	8.5
41	Léon	(1994)	8.5
42	The Lion King	(1994)	8.5
43	Nuovo Cinema Paradiso	(1988)	8.5
44	Hotaru no haka	(1988)	8.5
45	Back to the Future	(1985)	8.5
46	Apocalypse Now	(1979)	8.5
47	Alien	(1979)	8.5
48	Once Upon a Time in the West	(1968)	8.5
49	Psycho	(1960)	8.5

In [11]:

```
#Write a python program to display IMDB's Top rated 100 Indian movies' data
#(i.e. name, rating, year of release) and make data frame
IMDBH1 = requests.get('https://www.imdb.com/list/ls009997493/')
#IMDBH1
```

In [12]:

```
IMDBH100 = BeautifulSoup(IMDBH1.content)
IMDBHtitles = []
for i in IMDBH100.find_all('h3',class_="lister-item-header"):
    movname = i.findChildren('a')
    for j in movname:
        IMDBHtitles.append(j.text)
#IMDBHtitles
```

In [13]:

```
IMDBHyears = []
for i in IMDBH100.find_all('span',class_="lister-item-year"):
    IMDBHyears.append(i.text)
#IMDBHyears
```

In [14]:

```
IMDBHrating = []
for i in IMDBH100.find_all('div',class_= "ipl-rating-star small"):
    IMDBHrating.append(i.text.split()[0])
#IMDBHrating
```

In [15]:

```
HF = pd.DataFrame({'Movie name':IMDBHtitles, 'Years':IMDBHyears, 'Ratings':IMDBHrating})
HF
```

Out[15]:

	Movie name	Years	Ratings
0	Rang De Basanti	(2006)	8.1
1	3 Idiots	(2009)	8.4
2	Taare Zameen Par	(2007)	8.3
3	Dil Chahta Hai	(2001)	8.1
4	Swades: We, the People	(2004)	8.1
...	...	...	...
95	Wake Up Sid	(2009)	7.6
96	Rangeela	(1995)	7.4
97	Shatranj Ke Khilari	(1977)	7.5
98	Pyaar Ka Punchnama	(2011)	7.6
99	Ek Hasina Thi	(2004)	7.5

100 rows × 3 columns

In [16]:

```
#Write s python program to display list of respected former presidents of India(i.e. Name , Term of office)
#from https://presidentofindia.nic.in/former-presidents.htm
PROI = requests.get('https://presidentofindia.nic.in/former-presidents.htm')
#PROI
PROICON = BeautifulSoup(PROI.content)
PROINAME = []
for i in PROICON.find_all('div',class_="presidentListing"):
    PRODATA = i.findChildren('h3')
    for j in PRODATA:
        PROINAME.append(j.text)
PROINAME
```

Out[16]:

```
['Shri Ram Nath Kovind (birth - 1945)',
 'Shri Pranab Mukherjee (1935-2020)',
 'Smt Pratibha Devisingh Patil (birth - 1934)',
 'DR. A.P.J. Abdul Kalam (1931-2015)',
 'Shri K. R. Narayanan (1920 - 2005)',
 'Dr Shankar Dayal Sharma (1918-1999)',
 'Shri R Venkataraman (1910-2009)',
 'Giani Zail Singh (1916-1994)',
 'Shri Neelam Sanjiva Reddy (1913-1996)',
 'Dr. Fakhruddin Ali Ahmed (1905-1977)',
 'Shri Varahagiri Venkata Giri (1894-1980)',
 'Dr. Zakir Husain (1897-1969)',
 'Dr. Sarvepalli Radhakrishnan (1888-1975)',
 'Dr. Rajendra Prasad (1884-1963) ']
```

In [17]:

```
PROTERM = []
for i in PROICON.find_all('div',class_="presidentListing"):
    PRODATA2 = i.findChildren('p')[0]
    for j in PRODATA2:
        PROTERM.append(j.text)
PROTERM
```

Out[17]:

```
['Term of Office:',
 ' 25 July, 2017 to 25 July, 2022 ',
 'Term of Office:',
 ' 25 July, 2012 to 25 July, 2017 ',
 'Term of Office:',
 ' 25 July, 2007 to 25 July, 2012 ',
 'Term of Office:',
 ' 25 July, 2002 to 25 July, 2007 ',
 'Term of Office:',
 ' 25 July, 1997 to 25 July, 2002 ',
 'Term of Office:',
 ' 25 July, 1992 to 25 July, 1997 ',
 'Term of Office:',
 ' 25 July, 1987 to 25 July, 1992 ',
 'Term of Office:',
 ' 25 July, 1982 to 25 July, 1987 ',
 'Term of Office:',
 ' 25 July, 1977 to 25 July, 1982 ',
 'Term of Office:',
 ' 24 August, 1974 to 11 February, 1977',
 'Term of Office:',
 ' 3 May, 1969 to 20 July, 1969 and 24 August, 1969 to 24 August, 1974',
 'Term of Office:',
 ' 13 May, 1967 to 3 May, 1969',
 'Term of Office:',
 ' 13 May, 1962 to 13 May, 1967',
 'Term of Office:',
 ' 26 January, 1950 to 13 May, 1962']
```

In [18]:

```
# 5) Write a python program to scrape cricket rankings from icc-cricket.com. You have to scrape:
# a) Top 10 ODI teams in men's cricket along with the records for matches, points and rating
menODITR = requests.get('https://www.icc-cricket.com/rankings/mens/team-rankings/odi')
menODIT = BeautifulSoup(menODITR.content)
```

In [19]:

```
menODITeam = []
menODIMatches = []
menODIPoints = []
menODIRating = []
for i in menODIT.find_all('tr',class_="rankings-block__banner"):
    thisTeamData = i.text.split('\n')
    menODITeam.append(thisTeamData[4])
    menODIMatches.append(thisTeamData[7])
    menODIPoints.append(thisTeamData[8])
    menODIRating.append(thisTeamData[10])
j = 0
for i in menODIT.find_all('tr',class_="table-body"):
    j=j+1
    thisTeamData = i.text.split('\n')
    menODITeam.append(thisTeamData[4])
    menODIMatches.append(thisTeamData[7])
    menODIPoints.append(thisTeamData[8])
    menODIRating.append(thisTeamData[9])
    if j == 9:
        break
```

In [20]:

```
menODITeamF = pd.DataFrame({'Team':menODITeam, 'Matches':menODIMatches, 'Points':menODIPoints, 'Rating':menODIRating})
menODITeamF
```

Out[20]:

	Team	Matches	Points	Rating
0	New Zealand	16	2,051	128
1	England	27	3,226	119
2	India	28	3,085	110
3	Pakistan	19	2,005	106
4	Australia	23	2,325	101
5	South Africa	21	2,111	101
6	Bangladesh	27	2,639	98
7	Sri Lanka	29	2,658	92
8	West Indies	38	2,621	69
9	Afghanistan	18	1,238	69

In [21]:

```
# b) Top 10 ODI teams in men's cricket along with the records for matches, points and rating
menODIBR = requests.get('https://www.icc-cricket.com/rankings/mens/player-rankings/odi/batting')
menODIB = BeautifulSoup(menODIBR.content)
```

In [22]:

```
menODIBats = []
menODIBTeam = []
menODIBRating = []
for i in menODIB.find_all('tr',class_="rankings-block__banner"):
    thisTeamData = i.text.split('\n')
    menODIBats.append(thisTeamData[20])
    menODIBTeam.append(thisTeamData[27])
    menODIBRating.append(thisTeamData[31])
j = 0
for i in menODIB.find_all('tr',class_="table-body"):
    j=j+1
    thisTeamData = i.text.split('\n')
    menODIBats.append(thisTeamData[13])
    menODIBTeam.append(thisTeamData[17])
    menODIBRating.append(thisTeamData[19])
    if j == 9:
        break
```

In [23]:

```
menODIBatF = pd.DataFrame({'Player':menODIBats, 'Team':menODIBTeam, 'Rating':menODIBRating})
menODIBatF
```

Out[23]:

	Player	Team	Rating
0	Babar Azam	PAK	892
1	Imam-ul-Haq	PAK	815
2	Rassie van der Dussen	SA	789
3	Quinton de Kock	SA	784
4	Virat Kohli	IND	767
5	Rohit Sharma	IND	763
6	Ross Taylor	NZ	744
7	David Warner	AUS	737
8	Jonny Bairstow	ENG	732
9	Aaron Finch	AUS	715

In [24]:

```
# c) Top 10 ODI bowlers along with the records of their team and rating.
menODIBlR = requests.get('https://www.icc-cricket.com/rankings/mens/player-rankings/odi/bowling')
menODIBl = BeautifulSoup(menODIBlR.content)
```

In [25]:

```
menODIBowls = []
menODIBlTeam = []
menODIBlRating = []
for i in menODIBl.find_all('tr',class_="rankings-block__banner"):
    thisTeamData = i.text.split('\n')
    menODIBowls.append(thisTeamData[20])
    menODIBlTeam.append(thisTeamData[27])
    menODIBlRating.append(thisTeamData[31])
j = 0
for i in menODIBl.find_all('tr',class_="table-body"):
    j=j+1
    thisTeamData = i.text.split('\n')
    menODIBowls.append(thisTeamData[13])
    menODIBlTeam.append(thisTeamData[17])
    menODIBlRating.append(thisTeamData[19])
    if j == 9:
        break
```

In [26]:

```
menODIBlF = pd.DataFrame({'Player':menODIBowls, 'Team':menODIBlTeam, 'Rating':menODIBlRating})
menODIBlF
```

Out[26]:

	Player	Team	Rating
0	Trent Boult	NZ	697
1	Jasprit Bumrah	IND	682
2	Shaheen Afridi	PAK	681
3	Josh Hazlewood	AUS	679
4	Mujeeb Ur Rahman	AFG	676
5	Mehedi Hasan	BAN	672
6	Matt Henry	NZ	663
7	Mohammad Nabi	AFG	657
8	Rashid Khan	AFG	651
9	Chris Woakes	ENG	640

In [27]:

```
# 6) Write a python program to scrape cricket rankings from icc-cricket.com. You have to scrape:
# a) Top 10 ODI teams in women's cricket along with the records for matches, points and rating.
wmenODITR = requests.get('https://www.icc-cricket.com/rankings/womens/team-rankings/odi')
wmenODIT = BeautifulSoup(wmenODITR.content)
```

In [28]:

```
wmenODITeam = []
wmenODIMatches = []
wmenODIPoints = []
wmenODIRating = []
for i in wmenODIT.find_all('tr',class_="rankings-block__banner"):
    thisTeamData = i.text.split('\n')
    wmenODITeam.append(thisTeamData[4])
    wmenODIMatches.append(thisTeamData[7])
    wmenODIPoints.append(thisTeamData[8])
    wmenODIRating.append(thisTeamData[10])
j = 0
for i in wmenODIT.find_all('tr',class_="table-body"):
    j=j+1
    thisTeamData = i.text.split('\n')
    wmenODITeam.append(thisTeamData[4])
    wmenODIMatches.append(thisTeamData[7])
    wmenODIPoints.append(thisTeamData[8])
    wmenODIRating.append(thisTeamData[9])
    if j == 9:
        break
```

In [30]:

```
wmenODITeamF = pd.DataFrame({'Team':wmenODITeam, 'Matches':wmenODIMatches, 'Points':wmenODIPoints, 'Rating':wmenODIRating})
wmenODITeamF
```

Out[30]:

	Team	Matches	Points	Rating
0	Australia	29	4,837	167
1	England	33	4,046	123
2	South Africa	35	4,157	119
3	India	32	3,219	101
4	New Zealand	31	3,019	97
5	West Indies	30	2,768	92
6	Bangladesh	12	930	78
7	Pakistan	30	1,962	65
8	Sri Lanka	11	495	45
9	Ireland	8	351	44

In [31]:

```
# b) Top 10 ODI teams in women's cricket along with the records for matches, points and rating
wmenODIBR = requests.get('https://www.icc-cricket.com/rankings/womens/player-rankings/odi/batting')
wmenODIB = BeautifulSoup(wmenODIBR.content)
```

In [32]:

```
wmenODIBats = []
wmenODIBTeam = []
wmenODIBRating = []
for i in wmenODIB.find_all('tr',class_="rankings-block__banner"):
    thisTeamData = i.text.split('\n')
    wmenODIBats.append(thisTeamData[20])
    wmenODIBTeam.append(thisTeamData[27])
    wmenODIBRating.append(thisTeamData[31])
j = 0
for i in wmenODIB.find_all('tr',class_="table-body"):
    j=j+1
    thisTeamData = i.text.split('\n')
    wmenODIBats.append(thisTeamData[13])
    wmenODIBTeam.append(thisTeamData[17])
    wmenODIBRating.append(thisTeamData[19])
    if j == 9:
        break
```

In [34]:

```
wmenODIBatF = pd.DataFrame({'Player':wmenODIBats, 'Team':wmenODIBTeam, 'Rating':wmenODIBRating})
wmenODIBatF
```

Out[34]:

	Player	Team	Rating
0	Alyssa Healy	AUS	785
1			
2			
3	Laura Wolvaardt	SA	732
4	Meg Lanning	AUS	710
5	Rachael Haynes	AUS	701
6	Amy Satterthwaite	NZ	681
7			
8			
9			

In [35]:

```
# c) Top 10 women's ODI all-rounder along with the records of their team and rating.
wmenODIBlR = requests.get('https://www.icc-cricket.com/rankings/womens/player-rankings/odi/all-rounder')
wmenODIBl = BeautifulSoup(wmenODIBlR.content)
```

In [36]:

```
wmenODIBowls = []
wmenODIBlTeam = []
wmenODIBlRating = []
for i in wmenODIBl.find_all('tr',class_="rankings-block__banner"):
    thisTeamData = i.text.split('\n')
    wmenODIBowls.append(thisTeamData[20])
    wmenODIBlTeam.append(thisTeamData[27])
    wmenODIBlRating.append(thisTeamData[31])
j = 0
for i in wmenODIBl.find_all('tr',class_="table-body"):
    j=j+1
    thisTeamData = i.text.split('\n')
    wmenODIBowls.append(thisTeamData[13])
    wmenODIBlTeam.append(thisTeamData[17])
    wmenODIBlRating.append(thisTeamData[19])
    if j == 9:
        break
```

In [37]:

```
wmenODIBlF = pd.DataFrame({'Player':wmenODIBowls, 'Team':wmenODIBlTeam, 'Rating':wmenODIBlRating})
wmenODIBlF
```

Out[37]:

	Player	Team	Rating
0	Natalie Sciver	ENG	379
1	Ellyse Perry	AUS	374
2	Marizanne Kapp	SA	349
3	Hayley Matthews	WI	339
4	Amelia Kerr	NZ	336
5	Ashleigh Gardner	AUS	270
6	Deepti Sharma	IND	252
7	Jess Jonassen	AUS	246
8	Katherine Brunt	ENG	220
9	Stafanie Taylor	WI	207

In [38]:

```
# 7) Write a python program to scrape mentioned news details from https://www.cnbc.com/world/?region=world :
cnbcR = requests.get('https://www.cnbc.com/world/?region=world')
cnbcC = BeautifulSoup(cnbcR.content)
```

In [39]:

```
cnbcHL = []
cnbcLink = []
for i in cnbcC.find_all('a',class_="LatestNews-headline"):
    cnbcHL.append(i.text)
    cnbcLink.append(i["href"])
cnbcTime = []
for i in cnbcC.find_all('time',class_="LatestNews-timestamp"):
    cnbcTime.append(i.text)
```

In [40]:

```
cnbcF = pd.DataFrame({'Headline':cnbcHL, 'Time':cnbcTime, 'Link':cnbcLink})
cnbcF
```

Out[40]:

	Headline	Time	Link
0	Deutsche Bank downgrades Palantir, citing a sl...	14 Min Ago	https://www.cnbc.com/2022/08/09/deutsche-bank-...
1	Britain is becoming an 'emerging market countr...	31 Min Ago	https://www.cnbc.com/2022/08/09/britain-is-bec...
2	WhatsApp will soon let you slip out of group c...	41 Min Ago	https://www.cnbc.com/2022/08/09/whatsapp-will-...
3	Spirit Airlines posts loss on surge in costs, ...	47 Min Ago	https://www.cnbc.com/2022/08/09/spirit-airline...
4	Family offices invested a record $120 billion ...	51 Min Ago	https://www.cnbc.com/2022/08/09/family-offices...
5	Entrepreneurs aren't usually great investors, ...	57 Min Ago	https://www.cnbc.com/2022/08/09/entrepreneurs-...
6	Bed Bath & Beyond shares downgraded by Baird o...	1 Hour Ago	https://www.cnbc.com/2022/08/09/sell-bed-bath-...
7	Russia strips jetliners for parts amid sanctio...	2 Hours Ago	https://www.cnbc.com/2022/08/09/russia-ukraine...
8	U.S. Treasury yields tick up as investors look...	2 Hours Ago	https://www.cnbc.com/2022/08/09/us-bonds-treas...
9	European markets retreat slightly as focus tur...	4 Hours Ago	https://www.cnbc.com/2022/08/09/europe-markets...
10	Don't chase the rally in stocks and bonds righ...	5 Hours Ago	https://www.cnbc.com/2022/08/09/market-outlook...
11	CNBC Pro Talks: Fund manager Paul Meeks on wha...	6 Hours Ago	https://www.cnbc.com/2022/08/09/cnbc-pro-talks...
12	Alibaba gets Hong Kong's approval for a primar...	6 Hours Ago	https://www.cnbc.com/2022/08/09/alibaba-gets-h...
13	Asian exporters could face a trade recession a...	8 Hours Ago	https://www.cnbc.com/2022/08/09/us-europe-chin...
14	Google is back online after users around the w...	9 Hours Ago	https://www.cnbc.com/2022/08/09/google-down-ou...
15	Tornado Cash crackdown puts honest crypto inve...	9 Hours Ago	https://www.cnbc.com/2022/08/08/tornado-cash-c...
16	Wage inflation, used car prices could jump hig...	10 Hours Ago	https://www.cnbc.com/2022/08/08/wage-inflation...
17	Nearly half of Singaporeans want to travel to ...	11 Hours Ago	https://www.cnbc.com/2022/08/09/is-japan-open-...
18	'Pretty dang good': Analyst bullish on energy ...	11 Hours Ago	https://www.cnbc.com/2022/08/09/analyst-picks-...
19	Jim Cramer says these 7 Covid-era winning stoc...	11 Hours Ago	https://www.cnbc.com/2022/08/08/jim-cramer-say...
20	Morningstar strategist sees headwinds fading; ...	11 Hours Ago	https://www.cnbc.com/2022/08/09/morningstar-ov...
21	Japan's Nikkei leads losses in mixed Asia mark...	11 Hours Ago	https://www.cnbc.com/2022/08/09/asia-markets-e...
22	Food prices fell sharply in July — but the res...	12 Hours Ago	https://www.cnbc.com/2022/08/09/fao-food-price...
23	Cramer's lightning round: Sunrun is too specul...	12 Hours Ago	https://www.cnbc.com/2022/08/08/cramers-lightn...
24	Trump says the FBI raided his Mar-a-Lago home	12 Hours Ago	https://www.cnbc.com/2022/08/08/trump-says-fbi...
25	Ezra Miller charged with felony burglary days ...	12 Hours Ago	https://www.cnbc.com/2022/08/08/ezra-miller-ch...
26	Cramer's week ahead: Hot inflation data could ...	13 Hours Ago	https://www.cnbc.com/2022/08/08/cramers-week-a...
27	S&P 500, Nasdaq futures fall on Tuesday as chi...	13 Hours Ago	https://www.cnbc.com/2022/08/08/stock-market-f...
28	CNBC in 5 minutes: Here are all the stock call...	14 Hours Ago	https://www.cnbc.com/2022/08/08/cnbc-in-5-minu...
29	Stocks making the biggest moves after hours: N...	14 Hours Ago	https://www.cnbc.com/2022/08/08/stocks-making-...

In [41]:

```
# 8) Write a python program to scrape the details of most downloaded articles from AI in last 90 days.
#https://www.journals.elsevier.com/artificial-intelligence/most-downloaded-articles
aiR = requests.get('https://www.journals.elsevier.com/artificial-intelligence/most-downloaded-articles')
aiC = BeautifulSoup(aiR.content)
```

In [42]:

```
aiTitle = []
for i in aiC.find_all('h2',class_="sc-1qrq3sd-1 MKjKb sc-1nmom32-0 sc-1nmom32-1 hqhUYH ebTA-dR"):
    aiTitle.append(i.text)
airAuth = []
for i in aiC.find_all('span',class_="sc-1w3fpd7-0 pgLAT"):
    airAuth.append(i.text)
aiDate = []
for i in aiC.find_all('span',class_="sc-1thf9ly-2 bKddwo"):
    aiDate.append(i.text)
aiURL = []
for i in aiC.find_all('a',class_="sc-5smygv-0 nrDZj"):
    aiURL.append(i["href"])
```

In [43]:

```
aiF = pd.DataFrame({'Title':aiTitle, 'Auther':airAuth, 'Date':aiDate, 'Link':aiURL})
aiF
```

Out[43]:

	Title	Auther	Date	Link
0	Reward is enough	Silver, David, Singh, Satinder, Precup, Doina,...	October 2021	https://www.sciencedirect.com/science/article/...
1	Making sense of raw input	Evans, Richard, Bošnjak, Matko and 5 more	October 2021	https://www.sciencedirect.com/science/article/...
2	Law and logic: A review from an argumentation ...	Prakken, Henry, Sartor, Giovanni	October 2015	https://www.sciencedirect.com/science/article/...
3	Creativity and artificial intelligence	Boden, Margaret A.	August 1998	https://www.sciencedirect.com/science/article/...
4	Artificial cognition for social human–robot in...	Lemaignan, Séverin, Warnier, Mathieu and 3 more	June 2017	https://www.sciencedirect.com/science/article/...
5	Explanation in artificial intelligence: Insigh...	Miller, Tim	February 2019	https://www.sciencedirect.com/science/article/...
6	Making sense of sensory input	Evans, Richard, Hernández-Orallo, José and 3 more	April 2021	https://www.sciencedirect.com/science/article/...
7	Conflict-based search for optimal multi-agent ...	Sharon, Guni, Stern, Roni, Felner, Ariel, Stur...	February 2015	https://www.sciencedirect.com/science/article/...
8	Between MDPs and semi-MDPs: A framework for te...	Sutton, Richard S., Precup, Doina, Singh, Sati...	August 1999	https://www.sciencedirect.com/science/article/...
9	The Hanabi challenge: A new frontier for AI re...	Bard, Nolan, Foerster, Jakob N. and 13 more	March 2020	https://www.sciencedirect.com/science/article/...
10	Evaluating XAI: A comparison of rule-based and...	van der Waa, Jasper, Nieuwburg, Elisabeth, Cre...	February 2021	https://www.sciencedirect.com/science/article/...
11	Argumentation in artificial intelligence	Bench-Capon, T.J.M., Dunne, Paul E.	October 2007	https://www.sciencedirect.com/science/article/...
12	Algorithms for computing strategies in two-pla...	Bošanský, Branislav, Lisý, Viliam and 3 more	August 2016	https://www.sciencedirect.com/science/article/...
13	Multiple object tracking: A literature review	Luo, Wenhan, Xing, Junliang and 4 more	April 2021	https://www.sciencedirect.com/science/article/...
14	Selection of relevant features and examples in...	Blum, Avrim L., Langley, Pat	December 1997	https://www.sciencedirect.com/science/article/...
15	A survey of inverse reinforcement learning: Ch...	Arora, Saurabh, Doshi, Prashant	August 2021	https://www.sciencedirect.com/science/article/...
16	Explaining individual predictions when feature...	Aas, Kjersti, Jullum, Martin, Løland, Anders	September 2021	https://www.sciencedirect.com/science/article/...
17	A review of possible effects of cognitive bias...	Kliegr, Tomáš, Bahník, Štěpán, Fürnkranz, Joha...	June 2021	https://www.sciencedirect.com/science/article/...
18	Integrating social power into the decision-mak...	Pereira, Gonçalo, Prada, Rui, Santos, Pedro A.	December 2016	https://www.sciencedirect.com/science/article/...
19	"That's (not) the output I expected!" On the r...	Riveiro, Maria, Thill, Serge	September 2021	https://www.sciencedirect.com/science/article/...
20	Explaining black-box classifiers using post-ho...	Kenny, Eoin M., Ford, Courtney, Quinn, Molly, ...	May 2021	https://www.sciencedirect.com/science/article/...
21	Algorithm runtime prediction: Methods & evalua...	Hutter, Frank, Xu, Lin, Hoos, Holger H., Leyto...	January 2014	https://www.sciencedirect.com/science/article/...
22	Wrappers for feature subset selection	Kohavi, Ron, John, George H.	December 1997	https://www.sciencedirect.com/science/article/...
23	Commonsense visual sensemaking for autonomous ...	Suchan, Jakob, Bhatt, Mehul, Varadarajan, Srik...	October 2021	https://www.sciencedirect.com/science/article/...
24	Quantum computation, quantum theory and AI	Ying, Mingsheng	February 2010	https://www.sciencedirect.com/science/article/...

In [72]:

```
#Write a python program to scrape mentioned details from dineout.co.in :
page = requests.get("https://www.dineout.co.in/delhi-restaurants/buffet-special")
soup = BeautifulSoup(page.content)
```

In [73]:

```
titles = []
for i in soup.find_all('div',class_='restnt-info cursor'):
    titles.append(i.text)
#titles
```

In [74]:

```
location = []
for i in soup.find_all('div',class_='restnt-loc ellipsis'):
    location.append(i.text)
#location
```

In [75]:

```
ratings = []
for i in soup.find_all('div',class_='restnt-rating rating-4'):
    ratings.append(i.text)
#ratings
```

In [76]:

```
imgUrl = []
for i in soup.find_all('img',class_='no-img'):
    imgUrl.append(i["data-src"])
#imgUrl
```

In [77]:

```
restF = pd.DataFrame({'Title':titles, 'Location':location, 'Rating':ratings, 'Image':imgUrl})
restF
```

Out[77]:

	Title	Location	Rating	Image
0	Castle BarbequeConnaught Place, Central Delhi	Connaught Place, Central Delhi	4.1	https://im1.dineout.co.in/images/uploads/resta...
1	Jungle Jamboree3CS Mall,Lajpat Nagar - 3, Sout...	3CS Mall,Lajpat Nagar - 3, South Delhi	3.9	https://im1.dineout.co.in/images/uploads/resta...
2	Castle BarbequePacific Mall,Tagore Garden, Wes...	Pacific Mall,Tagore Garden, West Delhi	3.9	https://im1.dineout.co.in/images/uploads/resta...
3	Cafe KnoshThe Leela Ambience Convention Hotel,...	The Leela Ambience Convention Hotel,Shahdara, ...	4.3	https://im1.dineout.co.in/images/uploads/resta...
4	The Barbeque CompanyGardens Galleria,Sector 38...	Gardens Galleria,Sector 38A, Noida	4	https://im1.dineout.co.in/images/uploads/resta...
5	India GrillHilton Garden Inn,Saket, South Delhi	Hilton Garden Inn,Saket, South Delhi	3.9	https://im1.dineout.co.in/images/uploads/resta...
6	Delhi BarbequeTaurus Sarovar Portico,Mahipalpu...	Taurus Sarovar Portico,Mahipalpur, South Delhi	3.7	https://im1.dineout.co.in/images/uploads/resta...
7	The Monarch - Bar Be Que VillageIndirapuram Ha...	Indirapuram Habitat Centre,Indirapuram, Ghaziabad	3.8	https://im1.dineout.co.in/images/uploads/resta...
8	Indian Grill RoomSuncity Business Tower,Golf C...	Suncity Business Tower,Golf Course Road, Gurgaon	4.3	https://im1.dineout.co.in/images/uploads/resta...

In [78]:

```
# 10) Write a python program to scrape the details of top publications from Google Scholar from
# https://scholar.google.com/citations?view_op=top_venues&hl=en
pubSchR = requests.get('https://scholar.google.com/citations?view_op=top_venues&hl=en')
pubSchC = BeautifulSoup(pubSchR.content)
```

In [79]:

```
pubRank = []
for i in pubSchC.find_all('td',class_="gsc_mvt_p"):
    pubRank.append(i.text)
pubSch = []
for i in pubSchC.find_all('td',class_="gsc_mvt_t"):
    pubSch.append(i.text)
pubh5ind = []
for i in pubSchC.find_all('a',class_="gs_ibl gsc_mp_anchor"):
    pubh5ind.append(i.text)
pubh5med = []
for i in pubSchC.find_all('span',class_="gs_ibl gsc_mp_anchor"):
    pubh5med.append(i.text)
```

In [80]:

```
pubSchF = pd.DataFrame({'Rank':pubRank, 'Publication':pubSch, 'H5 Index':pubh5ind, 'H5 Median':pubh5med})
pubSchF
```

Out[80]:

	Rank	Publication	H5 Index	H5 Median
0	1.	Nature	444	667
1	2.	The New England Journal of Medicine	432	780
2	3.	Science	401	614
3	4.	IEEE/CVF Conference on Computer Vision and Pat...	389	627
4	5.	The Lancet	354	635
...	...	...	...	...
95	96.	Journal of Business Research	145	233
96	97.	Molecular Cancer	145	209
97	98.	Sensors	145	201
98	99.	Nature Climate Change	144	228
99	100.	IEEE Internet of Things Journal	144	212

100 rows × 4 columns

In [ ]: