Beautiful soup assingmnet 1 :- Answer 1 to 9

In [2]:

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

```
Defaulting to user installation because normal site-packages is not writea
ble
Requirement already satisfied: bs4 in c:\users\shubh\appdata\roaming\pytho
n\python39\site-packages (0.0.1)
Requirement already satisfied: beautifulsoup4 in c:\programdata\anaconda3
\lib\site-packages (from bs4) (4.11.1)
Requirement already satisfied: soupsieve>1.2 in c:\programdata\anaconda3\l
ib\site-packages (from beautifulsoup4->bs4) (2.3.1)
Defaulting to user installation because normal site-packages is not writea
ble
Requirement already satisfied: requests in c:\programdata\anaconda3\lib\si
te-packages (2.27.1)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\programdata\ana
conda3\lib\site-packages (from requests) (1.26.9)
Requirement already satisfied: certifi>=2017.4.17 in c:\programdata\anacon
da3\lib\site-packages (from requests) (2021.10.8)
Requirement already satisfied: idna<4,>=2.5 in c:\programdata\anaconda3\li
b\site-packages (from requests) (3.3)
Requirement already satisfied: charset-normalizer~=2.0.0 in c:\programdata
\anaconda3\lib\site-packages (from requests) (2.0.4)
```

In [3]:

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

Question no 1:-Write a python program to display all the header tags from wikipedia.org and make data frame

In [4]:

```
wikipedia_apge=requests.get('https://en.wikipedia.org/wiki/Main_Page')
```

In [7]:

```
soup1=BeautifulSoup(wikipedia_apge.content)
soup1
```

Out[7]:

```
<!DOCTYPE html>
<html class="client-nojs vector-feature-language-in-header-enabled vect
or-feature-language-in-main-page-header-disabled vector-feature-languag
e-alert-in-sidebar-enabled vector-feature-sticky-header-disabled vector
-feature-page-tools-pinned-disabled vector-feature-toc-pinned-enabled v
ector-feature-main-menu-pinned-disabled vector-feature-limited-width-en
abled vector-feature-limited-width-content-enabled vector-feature-zebra
-design-disabled" dir="ltr" lang="en">
<head>
<meta charset="utf-8"/>
<title>Wikipedia, the free encyclopedia</title>
<script>document.documentElement.className="client-js vector-feature-la
nguage-in-header-enabled vector-feature-language-in-main-page-header-di
sabled vector-feature-language-alert-in-sidebar-enabled vector-feature-
sticky-header-disabled vector-feature-page-tools-pinned-disabled vector
-feature-toc-pinned-enabled vector-feature-main-menu-pinned-disabled ve
ctor-feature-limited-width-enabled vector-feature-limited-width-content
-enabled vector-feature-zebra-design-disabled";(function(){var cookie=d
```

In [9]:

```
wikipedia_header=[]
for i in soup1.find_all('h2',class_="mp-h2"):
    wikipedia_header.append(i.text)
```

In [10]:

```
wikipedia_header
```

Out[10]:

```
["From today's featured article",
 'Did you know\xa0...',
 'In the news',
 'On this day',
 "Today's featured picture",
 'Other areas of Wikipedia',
 "Wikipedia's sister projects",
 'Wikipedia languages']
```

In [12]:

```
import pandas as pd
df=pd.DataFrame({'header':wikipedia_header})
df
```

Out[12]:

	header
0	From today's featured article
1	Did you know ...
2	In the news
3	On this day
4	Today's featured picture
5	Other areas of Wikipedia
6	Wikipedia's sister projects
7	Wikipedia languages

Questionno 2:- Write a python program to display IMDB’s Top rated 50 movies’ data (i.e. name, rating, year of release) and make data frame

In [35]:

```
movie_page=requests.get("https://www.imdb.com/list/ls053181721/")
```

In [36]:

```
movie_page
```

Out[36]:

```
<Response [200]>
```

In [38]:

```
soup2=BeautifulSoup(movie_page.content)
```

In [39]:

```
soup2
```

```
          "url": "/title/tt0088847/"
        },
        {
          "@type": "ListItem",
          "position": "11",
          "url": "/title/tt1853728/"
        },
        {
          "@type": "ListItem",
          "position": "12",
          "url": "/title/tt1045658/"
        },
        {
          "@type": "ListItem",
          "position": "13",
          "url": "/title/tt0081505/"
        },
        {
          "@type": "ListItem",
          "position": "14",
```

In [40]:

```
name=[]
for i in  soup2.find_all('h3',class_="lister-item-header"):
    star=i.find('a').text
    name.append(star)
```

In [41]:

```
name
```

Out[41]:

```
['Forrest Gump',
 'The Shawshank Redemption',
 'The Perks of Being a Wallflower',
 'The Dark Knight',
 'Changeling',
 "This Boy's Life",
 "It's a Wonderful Life",
 'The Silence of the Lambs',
 '8 Mile',
 'The Breakfast Club',
 'Django Unchained',
 'Silver Linings Playbook',
 'The Shining',
 'Se7en',
 'American Beauty',
 'Pulp Fiction',
 'Zero Dark Thirty',
 'Argo',
 'The Hurt Locker',
 'The Godfather',
 'The Town',
 'The Departed',
 'Scream',
 'Up in the Air',
 "What's Eating Gilbert Grape",
 'Lost in Translation',
 'The Conjuring',
 'Juno',
 'Stand by Me',
 'The Green Mile',
 'Super 8',
 'Jarhead',
 'Misery',
 'Fight Club',
 'Shutter Island',
 'Lawless',
 "Winter's Bone",
 'Taxi Driver',
 'Saving Private Ryan',
 'Black Swan',
 'Inception',
 'Boogie Nights',
 '50/50',
 'Brothers',
 'Blood Diamond',
 'A Few Good Men',
 'Gladiator',
 'Law Abiding Citizen',
 'Lakeview Terrace',
 'Glory Road']
```

In [43]:

```
years=[]
for i in soup2.find_all('span',class_="lister-item-year text-muted unbold"):
    years.append(i.text.split('|'))
years
```

Out[43]:

```
[['(1994)'],
 ['(1994)'],
 ['(2012)'],
 ['(2008)'],
 ['(2008)'],
 ['(1993)'],
 ['(1946)'],
 ['(1991)'],
 ['(2002)'],
 ['(1985)'],
 ['(2012)'],
 ['(2012)'],
 ['(1980)'],
 ['(1995)'],
 ['(1999)'],
 ['(1994)'],
 ['(2012)'],
 ['(2012)'],
 ['(2008)'],
 ['(1972)'],
 ['(2010)'],
 ['(2006)'],
 ['(1996)'],
 ['(I) (2009)'],
 ['(1993)'],
 ['(2003)'],
 ['(2013)'],
 ['(2007)'],
 ['(1986)'],
 ['(1999)'],
 ['(2011)'],
 ['(2005)'],
 ['(1990)'],
 ['(1999)'],
 ['(2010)'],
 ['(2012)'],
 ['(2010)'],
 ['(1976)'],
 ['(1998)'],
 ['(2010)'],
 ['(2010)'],
 ['(1997)'],
 ['(2011)'],
 ['(I) (2009)'],
 ['(2006)'],
 ['(1992)'],
 ['(2000)'],
 ['(2009)'],
 ['(2008)'],
 ['(2006)']]
```

In [46]:

```
scarepd_ratings=soup2.find_all('div',class_="ipl-rating-star small")
scarepd_ratings
```

Out[46]:

```
[<div class="ipl-rating-star small">
 <span class="ipl-rating-star__star">
 <svg class="ipl-icon ipl-star-icon" fill="#000000" height="24" viewbox
="0 0 24 24" width="24" xmlns="http://www.w3.org/2000/svg">
 <path d="M0 0h24v24H0z" fill="none"></path>
 <path d="M12 17.27L18.18 21l-1.64-7.03L22 9.24l-7.19-.61L12 2 9.19 8.6
3 2 9.24l5.46 4.73L5.82 21z"></path>
 <path d="M0 0h24v24H0z" fill="none"></path>
 </svg>
 </span>
 <span class="ipl-rating-star__rating">8.8</span>
 </div>,
 <div class="ipl-rating-star small">
 <span class="ipl-rating-star__star">
 <svg class="ipl-icon ipl-star-icon" fill="#000000" height="24" viewbox
="0 0 24 24" width="24" xmlns="http://www.w3.org/2000/svg">
 <path d="M0 0h24v24H0z" fill="none"></path>
 <path d="M12 17.27L18.18 21l-1.64-7.03L22 9.24l-7.19-.61L12 2 9.19 8.6
```

In [47]:

```
ratings=[]
for rating in scarepd_ratings:
    rating =rating.get_text().replace("\n","")
    ratings.append(rating)
ratings
```

Out[47]:

```
['8.8',
 '9.3',
 '7.9',
 '9',
 '7.8',
 '7.3',
 '8.6',
 '8.6',
 '7.2',
 '7.8',
 '8.4',
 '7.7',
 '8.4',
 '8.6',
 '8.3',
 '8.9',
 '7.4',
 '7.7',
 '7.5',
 '9.2',
 '7.5',
 '8.5',
 '7.4',
 '7.4',
 '7.7',
 '7.7',
 '7.5',
 '7.5',
 '8.1',
 '8.6',
 '7',
 '7',
 '7.8',
 '8.8',
 '8.2',
 '7.2',
 '7.1',
 '8.2',
 '8.6',
 '8',
 '8.8',
 '7.9',
 '7.6',
 '7.1',
 '8',
 '7.7',
 '8.5',
 '7.4',
 '6.2',
 '7.2']
```

In [49]:

```
import pandas as pd
df=pd.DataFrame({'Name of movie':name,'released year':years,'IMBD rating':ratings})
df
```

Out[49]:

	Name of movie	released year	IMBD rating
0	Forrest Gump	[(1994)]	8.8
1	The Shawshank Redemption	[(1994)]	9.3
2	The Perks of Being a Wallflower	[(2012)]	7.9
3	The Dark Knight	[(2008)]	9
4	Changeling	[(2008)]	7.8
5	This Boy's Life	[(1993)]	7.3
6	It's a Wonderful Life	[(1946)]	8.6
7	The Silence of the Lambs	[(1991)]	8.6
8	8 Mile	[(2002)]	7.2
9	The Breakfast Club	[(1985)]	7.8
10	Django Unchained	[(2012)]	8.4
11	Silver Linings Playbook	[(2012)]	7.7
12	The Shining	[(1980)]	8.4
13	Se7en	[(1995)]	8.6
14	American Beauty	[(1999)]	8.3
15	Pulp Fiction	[(1994)]	8.9
16	Zero Dark Thirty	[(2012)]	7.4
17	Argo	[(2012)]	7.7
18	The Hurt Locker	[(2008)]	7.5
19	The Godfather	[(1972)]	9.2
20	The Town	[(2010)]	7.5
21	The Departed	[(2006)]	8.5
22	Scream	[(1996)]	7.4
23	Up in the Air	[(I) (2009)]	7.4
24	What's Eating Gilbert Grape	[(1993)]	7.7
25	Lost in Translation	[(2003)]	7.7
26	The Conjuring	[(2013)]	7.5
27	Juno	[(2007)]	7.5
28	Stand by Me	[(1986)]	8.1
29	The Green Mile	[(1999)]	8.6
30	Super 8	[(2011)]	7
31	Jarhead	[(2005)]	7
32	Misery	[(1990)]	7.8
33	Fight Club	[(1999)]	8.8
34	Shutter Island	[(2010)]	8.2
35	Lawless	[(2012)]	7.2
36	Winter's Bone	[(2010)]	7.1

	Name of movie	released year	IMBD rating
37	Taxi Driver	[(1976)]	8.2
38	Saving Private Ryan	[(1998)]	8.6
39	Black Swan	[(2010)]	8
40	Inception	[(2010)]	8.8
41	Boogie Nights	[(1997)]	7.9
42	50/50	[(2011)]	7.6
43	Brothers	[(I) (2009)]	7.1
44	Blood Diamond	[(2006)]	8
45	A Few Good Men	[(1992)]	7.7
46	Gladiator	[(2000)]	8.5
47	Law Abiding Citizen	[(2009)]	7.4
48	Lakeview Terrace	[(2008)]	6.2
49	Glory Road	[(2006)]	7.2

Question 3:- write a programmme to scrape IMBD top rated 50 indian movies (name,year of release,ratings)

In [50]:

```
indian_movies_page=requests.get("https://www.imdb.com/list/ls023325613/")
```

In [52]:

```
indian_movies_page
```

Out[52]:

```
<Response [200]>
```

In [54]:

```
soup3=BeautifulSoup(indian_movies_page.content)
soup3
```

```
<script type="application/ld+json">{
  "@context": "http://schema.org",
  "@type": "CreativeWork",
  "about": {
    "@type": "ItemList",
    "itemListElement": [
      {
        "@type": "ListItem",
        "position": "1",
        "url": "/title/tt0995740/"
      },
      {
        "@type": "ListItem",
        "position": "2",
        "url": "/title/tt6484982/"
      },
      {
        "@type": "ListItem",
        "position": "3",
```

In [55]:

```
name=[]
for i in soup3.find_all('h3',class_="lister-item-header"):
    names=i.find('a').text
    name.append(names)
name
```

Out[55]:

```
['No Smoking',
 'Newton',
 'Shahid',
 'Trapped',
 'City Lights',
 'Aligarh',
 'Paanch',
 'Black Friday',
 'Gulaal',
 'Gangs of Wasseypur',
 'Ugly',
 'Raman Raghav 2.0',
 'Mukkabaaz',
 'Masaan',
 'Firaaq',
 'Peepli [Live]',
 'The Lunchbox',
 'Manjhi: The Mountain Man',
 'Maqbool',
 'Titli',
 'Paan Singh Tomar',
 'Yeh Saali Zindagi',
 'Life in a Metro',
 'A Wednesday',
 'Shaurya: It Takes Courage to Make Right... Right',
 'Antardwand',
 'Ankhon Dekhi',
 'Sehar',
 'Manjunath',
 'Amu',
 'The Ghazi Attack',
 'Poorna',
 'Haraamkhor',
 'A Death in the Gunj',
 'Mukti Bhawan',
 'Aamir',
 'Talvar',
 'Drishyam',
 'Kaun?',
 'Manorama: Six Feet Under',
 'Udaan',
 'Shanghai',
 'Hazaaron Khwaishein Aisi',
 'Saare Jahaan Se Mehnga...',
 'Rang De Basanti',
 'Special Chabbis',
 'Baby',
 'Omkara',
 'Nil Battey Sannata',
 'Pink']
```

In [56]:

```
years=[]
for i in soup3.find_all('span',class_="lister-item-year text-muted unbold"):
    years.append(i.text.split('|'))
years
```

Out[56]:

```
[['(2007)'],
 ['(2017)'],
 ['(2012)'],
 ['(XVII) (2016)'],
 ['(2014)'],
 ['(2015)'],
 ['(2003)'],
 ['(2004)'],
 ['(2009)'],
 ['(2012)'],
 ['(2013)'],
 ['(2016)'],
 ['(2017)'],
 ['(2015)'],
 ['(2008)'],
 ['(2010)'],
 ['(2013)'],
 ['(2015)'],
 ['(2003)'],
 ['(2014)'],
 ['(2012)'],
 ['(2011)'],
 ['(2007)'],
 ['(2008)'],
 ['(2008)'],
 ['(2008)'],
 ['(2013)'],
 ['(2005)'],
 ['(2014)'],
 ['(2005)'],
 ['(2017)'],
 ['(2017)'],
 ['(2015)'],
 ['(2016)'],
 ['(2016)'],
 ['(2008)'],
 ['(2015)'],
 ['(2015)'],
 ['(1999)'],
 ['(2007)'],
 ['(2010)'],
 ['(2012)'],
 ['(2003)'],
 ['(2013)'],
 ['(2006)'],
 ['(2013)'],
 ['(I) (2015)'],
 ['(2006)'],
 ['(2015)'],
 ['(III) (2016)']]
```

In [57]:

```
scraped_ratings=soup3.find_all('div',class_="ipl-rating-star small")
scraped_ratings
```

Out[57]:

```
[<div class="ipl-rating-star small">
 <span class="ipl-rating-star__star">
 <svg class="ipl-icon ipl-star-icon" fill="#000000" height="24" viewbox
="0 0 24 24" width="24" xmlns="http://www.w3.org/2000/svg">
 <path d="M0 0h24v24H0z" fill="none"></path>
 <path d="M12 17.27L18.18 21l-1.64-7.03L22 9.24l-7.19-.61L12 2 9.19 8.6
3 2 9.24l5.46 4.73L5.82 21z"></path>
 <path d="M0 0h24v24H0z" fill="none"></path>
 </svg>
 </span>
 <span class="ipl-rating-star__rating">7.3</span>
 </div>,
 <div class="ipl-rating-star small">
 <span class="ipl-rating-star__star">
 <svg class="ipl-icon ipl-star-icon" fill="#000000" height="24" viewbox
="0 0 24 24" width="24" xmlns="http://www.w3.org/2000/svg">
 <path d="M0 0h24v24H0z" fill="none"></path>
 <path d="M12 17.27L18.18 21l-1.64-7.03L22 9.24l-7.19-.61L12 2 9.19 8.6
```

In [58]:

```
IMBD=[]
for rating in scraped_ratings:
    rating=rating.get_text().replace("\n","")
    IMBD.append(rating)
IMBD
```

Out[58]:

```
['7.3',
 '7.6',
 '8.2',
 '7.5',
 '7.3',
 '7.8',
 '7.6',
 '8.4',
 '8',
 '8.2',
 '7.9',
 '7.3',
 '8',
 '8.1',
 '7.3',
 '7.4',
 '7.8',
 '8',
 '8',
 '7.5',
 '8.2',
 '7.4',
 '7.4',
 '8.1',
 '7.3',
 '7.3',
 '7.9',
 '7.7',
 '7.1',
 '7.3',
 '7.5',
 '7.7',
 '6.4',
 '7.4',
 '7',
 '7.6',
 '8.1',
 '8.2',
 '7.8',
 '7.5',
 '8.1',
 '7.3',
 '7.9',
 '7.1',
 '8.1',
 '8',
 '7.9',
 '8.1',
 '8.2',
 '8.1']
```

In [59]:

```
import pandas as pd
df=pd.DataFrame({'name of indian movie':name,'released years':years,'IMBD':IMBD})
df
```

Out[59]:

	name of indian movie	released years	IMBD
0	No Smoking	[(2007)]	7.3
1	Newton	[(2017)]	7.6
2	Shahid	[(2012)]	8.2
3	Trapped	[(XVII) (2016)]	7.5
4	City Lights	[(2014)]	7.3
5	Aligarh	[(2015)]	7.8
6	Paanch	[(2003)]	7.6
7	Black Friday	[(2004)]	8.4
8	Gulaal	[(2009)]	8
9	Gangs of Wasseypur	[(2012)]	8.2
10	Ugly	[(2013)]	7.9
11	Raman Raghav 2.0	[(2016)]	7.3
12	Mukkabaaz	[(2017)]	8
13	Masaan	[(2015)]	8.1
14	Firaaq	[(2008)]	7.3
15	Peepli [Live]	[(2010)]	7.4
16	The Lunchbox	[(2013)]	7.8
17	Manjhi: The Mountain Man	[(2015)]	8
18	Maqbool	[(2003)]	8
19	Titli	[(2014)]	7.5
20	Paan Singh Tomar	[(2012)]	8.2
21	Yeh Saali Zindagi	[(2011)]	7.4
22	Life in a Metro	[(2007)]	7.4
23	A Wednesday	[(2008)]	8.1
24	Shaurya: It Takes Courage to Make Right... Right	[(2008)]	7.3
25	Antardwand	[(2008)]	7.3
26	Ankhon Dekhi	[(2013)]	7.9
27	Sehar	[(2005)]	7.7
28	Manjunath	[(2014)]	7.1
29	Amu	[(2005)]	7.3
30	The Ghazi Attack	[(2017)]	7.5
31	Poorna	[(2017)]	7.7
32	Haraamkhor	[(2015)]	6.4
33	A Death in the Gunj	[(2016)]	7.4
34	Mukti Bhawan	[(2016)]	7
35	Aamir	[(2008)]	7.6
36	Talvar	[(2015)]	8.1

	name of indian movie	released years	IMBD
37	Drishyam	[(2015)]	8.2
38	Kaun?	[(1999)]	7.8
39	Manorama: Six Feet Under	[(2007)]	7.5
40	Udaan	[(2010)]	8.1
41	Shanghai	[(2012)]	7.3
42	Hazaaron Khwaishein Aisi	[(2003)]	7.9
43	Saare Jahaan Se Mehnga...	[(2013)]	7.1
44	Rang De Basanti	[(2006)]	8.1
45	Special Chabbis	[(2013)]	8
46	Baby	[(I) (2015)]	7.9
47	Omkara	[(2006)]	8.1
48	Nil Battey Sannata	[(2015)]	8.2
49	Pink	[(III) (2016)]	8.1

Question 4:- Write s python program to display list of respected former presidents of India(i.e. Name , Term ofoffice)

```
        from https://presidentofindia.nic.in/former-presidents.htm and make
    data frame.
```

In [60]:

```
president_page=requests.get('https://presidentofindia.nic.in/former-presidents.htm')
president_page
```

Out[60]:

```
<Response [200]>
```

In [61]:

```
soup4=BeautifulSoup(president_page.content)
soup4
```

```
    .facebookForumInn label{
        margin-right:10px;
    }
</style>
<link href="App_Themes/User/ie9.css" rel="stylesheet" type="text/css"/>
<link href="App_Themes/User/jquery.ui.autocomplete.css" rel="styleshee
t" type="text/css"/><link href="App_Themes/User/jquery.ui.autocomplete.
custom.css" rel="stylesheet" type="text/css"/><link href="App_Themes/Us
er/jQueryCalender.css" rel="stylesheet" type="text/css"/><link href="Ap
p_Themes/User/User.css" rel="stylesheet" type="text/css"/><link href="/
writereaddata/Portal/Design_CSS/2.css" id="css2" rel="stylesheet" type
="text/css"/><meta content="Former Presidents - The President of India"
name="keyword"/><meta content="Former Presidents - The President of Ind
ia" name="description"/><meta content="english" name="language"/><link
href="/writereaddata/Portal/Design_CSS/7.css" id="css7" media="print" r
el="stylesheet" type="text/css"/></head>
<body class="noJS" id="bdMainSite" style="font-size:87.5%">
<!--<script type="text/javascript">
        function AddBodyClass() {
                $("#" + "bdMainSite").addClass("home");
```

In [39]:

```
president=soup4.find_all('div',class_="presidentListing")
president
```

In [63]:

```
names=[]
for name in president:
    star=name.find('h3').get_text()
    star1=star.replace("\n","")
    names.append(star1)
```

In [64]:

```
names
```

Out[64]:

```
['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 [65]:

```
scrap_service=soup4.find_all('div',class_="presidentListing")
scrap_service
```

In [67]:

```
services=[]
for i in scrap_service:
    service=i.find('p').text
    services.append(service)
```

In [69]:

```
services
```

Out[69]:

```
['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 A
ugust, 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 [70]:

```
import pandas as pd
df1=pd.DataFrame({'president_name':names, 'service':services})
```

In [71]:

```
df1
```

Out[71]:

	president_name	service
0	Shri Ram Nath Kovind (birth - 1945)	Term of Office: 25 July, 2017 to 25 July, 2022
1	Shri Pranab Mukherjee (1935-2020)	Term of Office: 25 July, 2012 to 25 July, 2017
2	Smt Pratibha Devisingh Patil (birth - 1934)	Term of Office: 25 July, 2007 to 25 July, 2012
3	DR. A.P.J. Abdul Kalam (1931-2015)	Term of Office: 25 July, 2002 to 25 July, 2007
4	Shri K. R. Narayanan (1920 - 2005)	Term of Office: 25 July, 1997 to 25 July, 2002
5	Dr Shankar Dayal Sharma (1918-1999)	Term of Office: 25 July, 1992 to 25 July, 1997
6	Shri R Venkataraman (1910-2009)	Term of Office: 25 July, 1987 to 25 July, 1992
7	Giani Zail Singh (1916-1994)	Term of Office: 25 July, 1982 to 25 July, 1987
8	Shri Neelam Sanjiva Reddy (1913-1996)	Term of Office: 25 July, 1977 to 25 July, 1982
9	Dr. Fakhruddin Ali Ahmed (1905-1977)	Term of Office: 24 August, 1974 to 11 February...
10	Shri Varahagiri Venkata Giri (1894-1980)	Term of Office: 3 May, 1969 to 20 July, 1969 a...
11	Dr. Zakir Husain (1897-1969)	Term of Office: 13 May, 1967 to 3 May, 1969
12	Dr. Sarvepalli Radhakrishnan (1888-1975)	Term of Office: 13 May, 1962 to 13 May, 1967
13	Dr. Rajendra Prasad (1884-1963)	Term of Office: 26 January, 1950 to 13 May, 1962

5) Write a python program to scrape cricket rankings from icc-cricket.com. You have to scrape and make data frame

a) Top 10 ODI teams in men’s cricket along with the records for matches, points and rating.

In [34]:

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

```
Defaulting to user installation because normal site-packages is not writea
ble
Requirement already satisfied: bs4 in c:\users\shubh\appdata\roaming\pytho
n\python39\site-packages (0.0.1)
Requirement already satisfied: beautifulsoup4 in c:\programdata\anaconda3
\lib\site-packages (from bs4) (4.11.1)
Requirement already satisfied: soupsieve>1.2 in c:\programdata\anaconda3\l
ib\site-packages (from beautifulsoup4->bs4) (2.3.1)
Defaulting to user installation because normal site-packages is not writea
ble
Requirement already satisfied: requests in c:\programdata\anaconda3\lib\si
te-packages (2.27.1)
Requirement already satisfied: idna<4,>=2.5 in c:\programdata\anaconda3\li
b\site-packages (from requests) (3.3)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\programdata\ana
conda3\lib\site-packages (from requests) (1.26.9)
Requirement already satisfied: charset-normalizer~=2.0.0 in c:\programdata
\anaconda3\lib\site-packages (from requests) (2.0.4)
Requirement already satisfied: certifi>=2017.4.17 in c:\programdata\anacon
da3\lib\site-packages (from requests) (2021.10.8)
```

In [2]:

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

In [3]:

```
men_odi=requests.get('https://www.icc-cricket.com/rankings/mens/team-rankings/odi')
```

In [4]:

```
men_odi
```

Out[4]:

```
<Response [200]>
```

In [9]:

```
soup=BeautifulSoup(men_odi.content)
```

In [10]:

```
soup
```

Out[10]:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta content="ICC Men's ODI Team Rankings | ICC" name="twitter:title"/
>
<meta content="website" property="og:type"/>
<meta content="summary_large_image" property="twitter:card"/>
<meta content="Official International Cricket Council ranking for One D
ay International (ODI) cricket teams. Discover latest ICC rankings tabl
e, predict upcoming matches, see points and ratings for all teams." nam
e="description"/>
<meta content="@icc" property="twitter:site"/>
<meta content="Official International Cricket Council ranking for One D
ay International (ODI) cricket teams. Discover latest ICC rankings tabl
e, predict upcoming matches, see points and ratings for all teams." nam
e="twitter:description"/>
<meta content="https://www.icc-cricket.com/resources/ver/i/elements/def
ault-thumbnail.jpg" name="twitter:image"/>
```

In [17]:

```
teams=[]
for i in soup.find_all('span',class_="u-show-phablet"):
    teams.append(i.text.split())
```

In [18]:

```
teams
```

Out[18]:

```
[['AUS'],
 ['NZ'],
 ['IND'],
 ['ENG'],
 ['PAK'],
 ['SA'],
 ['BAN'],
 ['SL'],
 ['WI'],
 ['AFG'],
 ['IRE'],
 ['SCO'],
 ['ZIM'],
 ['NEP'],
 ['NED'],
 ['USA'],
 ['NAM'],
 ['OMA'],
 ['UAE'],
 ['PNG']]
```

In [24]:

```
match=[]
for i in soup.find_all('td',class_="table-body__cell u-center-text"):
    match.append(i.text)
```

In [25]:

```
match
```

Out[25]:

```
['31',
 '3,504',
 '47',
 '5,294',
 '36',
 '3,988',
 '25',
 '2,649',
 '31',
 '3,141',
 '38',
 '3,625',
 '36',
 '3,099',
 '43',
 '3,105',
 '20',
 '1,419',
 '27',
 '1,384',
 '35',
 '1,567',
 '31',
 '1,351',
 '41',
 '1,319',
 '26',
 '791',
 '36',
 '1,093',
 '37',
 '1,096',
 '31',
 '900',
 '40',
 '964',
 '43',
 '80']
```

In [28]:

```
points=match[1::2]
```

In [29]:

```
matches
```

Out[29]:

```
['3,504',
 '5,294',
 '3,988',
 '2,649',
 '3,141',
 '3,625',
 '3,099',
 '3,105',
 '1,419',
 '1,384',
 '1,567',
 '1,351',
 '1,319',
 '791',
 '1,093',
 '1,096',
 '900',
 '964',
 '80']
```

In [30]:

```
matches=match[0::2]
```

In [31]:

```
matches
```

Out[31]:

```
['31',
 '47',
 '36',
 '25',
 '31',
 '38',
 '36',
 '43',
 '20',
 '27',
 '35',
 '31',
 '41',
 '26',
 '36',
 '37',
 '31',
 '40',
 '43']
```

In [33]:

```
teams.remove(['AUS'])
```

In [34]:

```
teams
```

Out[34]:

```
[['NZ'],
 ['IND'],
 ['ENG'],
 ['PAK'],
 ['SA'],
 ['BAN'],
 ['SL'],
 ['WI'],
 ['AFG'],
 ['IRE'],
 ['SCO'],
 ['ZIM'],
 ['NEP'],
 ['NED'],
 ['USA'],
 ['NAM'],
 ['OMA'],
 ['UAE'],
 ['PNG']]
```

In [35]:

```
ratings=[]
for i in soup.find_all('td',class_="table-body__cell u-text-right rating"):
    ratings.append(i.text)
```

In [36]:

```
ratings
```

Out[36]:

```
['113',
 '113',
 '111',
 '106',
 '101',
 '95',
 '86',
 '72',
 '71',
 '51',
 '45',
 '44',
 '32',
 '30',
 '30',
 '30',
 '29',
 '24',
 '2']
```

In [37]:

```
import pandas as pd
df=pd.DataFrame({'teams':teams,'match':matches,'points':points,'rating':ratings})
```

In [39]:

```
df.head(10)
```

Out[39]:

	teams	match	points	rating
0	[NZ]	31	3,504	113
1	[IND]	47	5,294	113
2	[ENG]	36	3,988	111
3	[PAK]	25	2,649	106
4	[SA]	31	3,141	101
5	[BAN]	38	3,625	95
6	[SL]	36	3,099	86
7	[WI]	43	3,105	72
8	[AFG]	20	1,419	71
9	[IRE]	27	1,384	51

b) Top 10 ODI Batsmen along with the records of their team and rating.

In [41]:

```
men_odi_batsmen=requests.get('https://www.icc-cricket.com/rankings/mens/player-rankings/
```

In [42]:

```
men_odi_batsmen
```

Out[42]:

```
<Response [200]>
```

In [48]:

```
soup2=BeautifulSoup(men_odi_batsmen.content)
```

In [49]:

```
soup2
```

Out[49]:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta content="ICC Men's T20I Player Rankings | ICC" name="twitter:titl
e"/>
<meta content="website" property="og:type"/>
<meta content="summary_large_image" property="twitter:card"/>
<meta content="Official International Cricket Council rankings for T20I
match cricket players. Discover latest ICC rankings table, predict upco
ming matches, see points and ratings for all teams." name="descriptio
n"/>
<meta content="@icc" property="twitter:site"/>
<meta content="Official International Cricket Council rankings for T20I
match cricket players. Discover latest ICC rankings table, predict upco
ming matches, see points and ratings for all teams." name="twitter:desc
ription"/>
<meta content="https://www.icc-cricket.com/resources/ver/i/elements/def
ault-thumbnail.jpg" name="twitter:image"/>
```

In [51]:

```
player=[]
for i in soup2.find_all('td',class_="table-body__cell name"):
    players=i.find('a').text
    player.append(players)
```

In [52]:

```
player
```

Out[52]:

```
['Mohammad Rizwan',
 'Babar Azam',
 'Aiden Markram',
 'Rilee Rossouw',
 'Muhammad Waseem',
 'Devon Conway',
 'Dawid Malan',
 'Aaron Finch',
 'Jos Buttler',
 'Fazalhaq Farooqi',
 'Josh Hazlewood',
 'Wanindu Hasaranga',
 'Maheesh Theekshana',
 'Adil Rashid',
 'Adam Zampa',
 'Sam Curran',
 'Mujeeb Ur Rahman',
 'Anrich Nortje',
 'Hardik Pandya',
 'Mohammad Nabi',
 'Shadab Khan',
 'Wanindu Hasaranga',
 'J.J. Smit',
 'Sikandar Raza',
 'Aiden Markram',
 'David Wiese',
 'Moeen Ali',
 'Mohammad Rizwan',
 'Babar Azam',
 'Aiden Markram',
 'Rilee Rossouw',
 'Muhammad Waseem',
 'Devon Conway',
 'Dawid Malan',
 'Aaron Finch',
 'Jos Buttler',
 'Fazalhaq Farooqi',
 'Josh Hazlewood',
 'Wanindu Hasaranga',
 'Maheesh Theekshana',
 'Adil Rashid',
 'Adam Zampa',
 'Sam Curran',
 'Mujeeb Ur Rahman',
 'Anrich Nortje',
 'Hardik Pandya',
 'Mohammad Nabi',
 'Shadab Khan',
 'Wanindu Hasaranga',
 'J.J. Smit',
 'Sikandar Raza',
 'Aiden Markram',
 'David Wiese',
 'Moeen Ali']
```

In [57]:

```
team=[]
for i in soup2.find_all('td',class_="table-body__cell nationality-logo"):
    team.append(i.text.strip('\n'))
```

In [58]:

```
team
```

Out[58]:

```
['PAK',
 'PAK',
 'SA',
 'SA',
 'UAE',
 'NZ',
 'ENG',
 'AUS',
 'ENG',
 'AFG',
 'AUS',
 'SL',
 'SL',
 'ENG',
 'AUS',
 'ENG',
 'AFG',
 'SA',
 'IND',
 'AFG',
 'PAK',
 'SL',
 'NAM',
 'ZIM',
 'SA',
 'NAM',
 'ENG',
 'PAK',
 'PAK',
 'SA',
 'SA',
 'UAE',
 'NZ',
 'ENG',
 'AUS',
 'ENG',
 'AFG',
 'AUS',
 'SL',
 'SL',
 'ENG',
 'AUS',
 'ENG',
 'AFG',
 'SA',
 'IND',
 'AFG',
 'PAK',
 'SL',
 'NAM',
 'ZIM',
 'SA',
 'NAM',
 'ENG']
```

In [59]:

```
rating_of_player=[]
for i in soup2.find_all('td',class_="table-body__cell u-text-right rating"):
    rating_of_player.append(i.text)
```

In [60]:

```
rating_of_player
```

Out[60]:

```
['811',
 '756',
 '748',
 '724',
 '716',
 '709',
 '705',
 '680',
 '670',
 '692',
 '690',
 '686',
 '684',
 '684',
 '678',
 '673',
 '668',
 '667',
 '250',
 '230',
 '184',
 '182',
 '174',
 '173',
 '172',
 '170',
 '168',
 '811',
 '756',
 '748',
 '724',
 '716',
 '709',
 '705',
 '680',
 '670',
 '692',
 '690',
 '686',
 '684',
 '684',
 '678',
 '673',
 '668',
 '667',
 '250',
 '230',
 '184',
 '182',
 '174',
 '173',
 '172',
 '170',
 '168']
```

In [64]:

```
import pandas as pd
df1=pd.DataFrame({'player':player,'teams':team,'rating_of_player':rating_of_player})
```

In [66]:

```
df1.head(10)
```

Out[66]:

	player	teams	rating_of_player
0	Mohammad Rizwan	PAK	811
1	Babar Azam	PAK	756
2	Aiden Markram	SA	748
3	Rilee Rossouw	SA	724
4	Muhammad Waseem	UAE	716
5	Devon Conway	NZ	709
6	Dawid Malan	ENG	705
7	Aaron Finch	AUS	680
8	Jos Buttler	ENG	670
9	Fazalhaq Farooqi	AFG	692

c) Top 10 ODI bowlers along with the records of their team and rating.

In [68]:

```
men_odi_bowlers=requests.get('https://www.icc-cricket.com/rankings/mens/player-rankings/
```

In [70]:

```
men_odi_bowlers
```

Out[70]:

```
<Response [200]>
```

In [71]:

```
soup3=BeautifulSoup(men_odi_bowlers.content)
```

In [72]:

```
soup3
```

Out[72]:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta content="ICC Men's ODI Player Rankings | ICC" name="twitter:titl
e"/>
<meta content="website" property="og:type"/>
<meta content="summary_large_image" property="twitter:card"/>
<meta content="Official International Cricket Council rankings for ODI
match cricket players. Discover latest ICC rankings table, predict upco
ming matches, see points and ratings for all teams." name="descriptio
n"/>
<meta content="@icc" property="twitter:site"/>
<meta content="Official International Cricket Council rankings for ODI
match cricket players. Discover latest ICC rankings table, predict upco
ming matches, see points and ratings for all teams." name="twitter:desc
ription"/>
<meta content="https://www.icc-cricket.com/resources/ver/i/elements/def
ault-thumbnail.jpg" name="twitter:image"/>
```

In [73]:

```
player_bowler=[]
for i in soup3.find_all('td',class_="table-body__cell name"):
    bowler=i.find('a').text
    player_bowler.append(bowler)
```

In [74]:

```
player_bowler
```

Out[74]:

```
['Rassie van der Dussen',
 'Imam-ul-Haq',
 'Shubman Gill',
 'David Warner',
 'Virat Kohli',
 'Quinton de Kock',
 'Rohit Sharma',
 'Steve Smith',
 'Fakhar Zaman',
 'Trent Boult',
 'Mohammed Siraj',
 'Mitchell Starc',
 'Matt Henry',
 'Rashid Khan',
 'Adam Zampa',
 'Shaheen Afridi',
 'Mujeeb Ur Rahman',
 'Shakib Al Hasan',
 'Mohammad Nabi',
 'Rashid Khan',
 'Mitchell Santner',
 'Sikandar Raza',
 'Assad Vala',
 'Mehedi Hasan',
 'Zeeshan Maqsood',
 'Wanindu Hasaranga',
 'Chris Woakes',
 'Rassie van der Dussen',
 'Imam-ul-Haq',
 'Shubman Gill',
 'David Warner',
 'Virat Kohli',
 'Quinton de Kock',
 'Rohit Sharma',
 'Steve Smith',
 'Fakhar Zaman',
 'Trent Boult',
 'Mohammed Siraj',
 'Mitchell Starc',
 'Matt Henry',
 'Rashid Khan',
 'Adam Zampa',
 'Shaheen Afridi',
 'Mujeeb Ur Rahman',
 'Shakib Al Hasan',
 'Mohammad Nabi',
 'Rashid Khan',
 'Mitchell Santner',
 'Sikandar Raza',
 'Assad Vala',
 'Mehedi Hasan',
 'Zeeshan Maqsood',
 'Wanindu Hasaranga',
 'Chris Woakes']
```

In [77]:

```
bowler_team=[]
for i in soup3.find_all('td',class_="table-body__cell nationality-logo"):
    bowler_team.append(i.text.strip('\n'))
```

In [78]:

```
bowler_team
```

Out[78]:

```
['SA',
 'PAK',
 'IND',
 'AUS',
 'IND',
 'SA',
 'IND',
 'AUS',
 'PAK',
 'NZ',
 'IND',
 'AUS',
 'NZ',
 'AFG',
 'AUS',
 'PAK',
 'AFG',
 'BAN',
 'AFG',
 'AFG',
 'NZ',
 'ZIM',
 'PNG',
 'BAN',
 'OMA',
 'SL',
 'ENG',
 'SA',
 'PAK',
 'IND',
 'AUS',
 'IND',
 'SA',
 'IND',
 'AUS',
 'PAK',
 'NZ',
 'IND',
 'AUS',
 'NZ',
 'AFG',
 'AUS',
 'PAK',
 'AFG',
 'BAN',
 'AFG',
 'AFG',
 'NZ',
 'ZIM',
 'PNG',
 'BAN',
 'OMA',
 'SL',
 'ENG']
```

In [79]:

```
bowler_rating=[]
for i in soup3.find_all('td',class_="table-body__cell u-text-right rating"):
    bowler_rating.append(i.text)
```

In [81]:

```
bowler_rating
```

Out[81]:

```
['777',
 '740',
 '738',
 '726',
 '719',
 '718',
 '707',
 '702',
 '699',
 '694',
 '691',
 '686',
 '676',
 '659',
 '652',
 '641',
 '637',
 '636',
 '310',
 '280',
 '258',
 '253',
 '248',
 '248',
 '239',
 '216',
 '215',
 '777',
 '740',
 '738',
 '726',
 '719',
 '718',
 '707',
 '702',
 '699',
 '694',
 '691',
 '686',
 '676',
 '659',
 '652',
 '641',
 '637',
 '636',
 '310',
 '280',
 '258',
 '253',
 '248',
 '248',
 '239',
 '216',
 '215']
```

In [82]:

```
import pandas as pd
df2=pd.DataFrame({'name_bowler':player_bowler,'bowler_team':bowler_team,'bowler_rating':
```

In [84]:

```
df2.head(10)
```

Out[84]:

	name_bowler	bowler_team	bowler_rating
0	Rassie van der Dussen	SA	777
1	Imam-ul-Haq	PAK	740
2	Shubman Gill	IND	738
3	David Warner	AUS	726
4	Virat Kohli	IND	719
5	Quinton de Kock	SA	718
6	Rohit Sharma	IND	707
7	Steve Smith	AUS	702
8	Fakhar Zaman	PAK	699
9	Trent Boult	NZ	694

Question mo:-6 Write a python program to scrape cricket rankings from icc- cricket.com. You have to scrape and make data frame□

```
        a) Top 10 ODI teams in women’s cricket along with the records for matches,
    points and rating
```

In [85]:

```
women_odi=requests.get('https://www.icc-cricket.com/rankings/womens/team-rankings/odi')
```

In [86]:

```
women_odi
```

Out[86]:

```
<Response [200]>
```

In [91]:

```
soup4=BeautifulSoup(women_odi.content)
```

In [92]:

```
soup4
```

Out[92]:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta content="ICC Women's ODI Team Rankings | ICC" name="twitter:titl
e"/>
<meta content="website" property="og:type"/>
<meta content="summary_large_image" property="twitter:card"/>
<meta content="Official International Cricket Council rankings for test
match cricket teams. Discover latest ICC rankings table, predict upcomi
ng matches, see points and ratings for all teams." name="description"/>
<meta content="@icc" property="twitter:site"/>
<meta content="Official International Cricket Council rankings for test
match cricket teams. Discover latest ICC rankings table, predict upcomi
ng matches, see points and ratings for all teams." name="twitter:descri
ption"/>
<meta content="https://www.icc-cricket.com/resources/ver/i/elements/def
ault-thumbnail.jpg" name="twitter:image"/>
<meta content="ICC Women's ODI Team Rankings | ICC" property="og:titl
```

In [93]:

```
women_team=[]
for i in soup4.find_all('span',class_="u-show-phablet"):
    women_team.append(i.text)
```

In [94]:

```
women_team
```

Out[94]:

```
['AUS',
 'ENG',
 'SA',
 'IND',
 'NZ',
 'WI',
 'BAN',
 'THA',
 'PAK',
 'SL',
 'IRE',
 'ZIM',
 'NED']
```

In [117]:

```
women_team.remove('AUS')
```

In [118]:

```
women_team
```

Out[118]:

```
['ENG',
 'SA',
 'IND',
 'NZ',
 'WI',
 'BAN',
 'THA',
 'PAK',
 'SL',
 'IRE',
 'ZIM',
 'NED']
```

In [95]:

```
women_match=[]
for i in soup4.find_all('td',class_="table-body__cell u-center-text"):
    women_match.append(i.text)
```

In [96]:

```
women_match
```

Out[96]:

```
['28',
 '3,342',
 '26',
 '3,098',
 '27',
 '2,820',
 '25',
 '2,553',
 '27',
 '2,535',
 '13',
 '983',
 '11',
 '821',
 '27',
 '1,678',
 '8',
 '353',
 '14',
 '548',
 '11',
 '0',
 '9',
 '0']
```

In [101]:

```
women_points=women_match[1::2]
```

In [102]:

```
women_points
```

Out[102]:

```
['3,342',
 '3,098',
 '2,820',
 '2,553',
 '2,535',
 '983',
 '821',
 '1,678',
 '353',
 '548',
 '0',
 '0']
```

In [114]:

```
women_matches=women_match[0::2]
```

In [115]:

```
women_matches
```

Out[115]:

```
['28', '26', '27', '25', '27', '13', '11', '27', '8', '14', '11', '9']
```

In [105]:

```
women_team_rating=[]
for i in soup4.find_all('td',class_="table-body__cell u-text-right rating"):
    women_team_rating.append(i.text)
```

In [106]:

```
women_team_rating
```

Out[106]:

```
['119', '119', '104', '102', '94', '76', '75', '62', '44', '39', '0', '0']
```

In [119]:

```
import pandas as pd
df4=pd.DataFrame({'name':women_team,'matches':women_matches,'points':women_points,'ratin
```

In [121]:

```
df4.head(10)
```

Out[121]:

	name	matches	points	ratings
0	ENG	28	3,342	119
1	SA	26	3,098	119
2	IND	27	2,820	104
3	NZ	25	2,553	102
4	WI	27	2,535	94
5	BAN	13	983	76
6	THA	11	821	75
7	PAK	27	1,678	62
8	SL	8	353	44
9	IRE	14	548	39

b) Top 10 women’s ODI Batting players along with the records of their team and rating

In [123]:

```
women_player_name=requests.get('https://www.icc-cricket.com/rankings/womens/player-ranki
```

In [124]:

```
women_player_name
```

Out[124]:

```
<Response [200]>
```

In [125]:

```
soup5=BeautifulSoup(women_player_name.content)
```

In [126]:

```
soup5
```

Out[126]:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta content="ICC Women's ODI Player Rankings | ICC" name="twitter:tit
le"/>
<meta content="website" property="og:type"/>
<meta content="summary_large_image" property="twitter:card"/>
<meta content="Official International Cricket Council rankings for ODI
match cricket players. Discover latest ICC rankings table, predict upco
ming matches, see points and ratings for all teams." name="descriptio
n"/>
<meta content="@icc" property="twitter:site"/>
<meta content="Official International Cricket Council rankings for ODI
match cricket players. Discover latest ICC rankings table, predict upco
ming matches, see points and ratings for all teams." name="twitter:desc
ription"/>
<meta content="https://www.icc-cricket.com/resources/ver/i/elements/def
ault-thumbnail.jpg" name="twitter:image"/>
```

In [127]:

```
women_player_names=[]
for i in soup5.find_all('td',class_="table-body__cell name"):
    star=i.find('a').text
    women_player_names.append(star)
```

In [128]:

```
women_player_names
```

Out[128]:

```
['Beth Mooney',
 'Laura Wolvaardt',
 'Natalie Sciver',
 'Meg Lanning',
 'Harmanpreet Kaur',
 'Smriti Mandhana',
 'Chamari Athapaththu',
 'Amy Satterthwaite',
 'Ellyse Perry',
 'Jess Jonassen',
 'Shabnim Ismail',
 'Megan Schutt',
 'Hayley Matthews',
 'Kate Cross',
 'Ayabonga Khaka',
 'Rajeshwari Gayakwad',
 'Marizanne Kapp',
 'Deepti Sharma',
 'Natalie Sciver',
 'Ellyse Perry',
 'Marizanne Kapp',
 'Amelia Kerr',
 'Deepti Sharma',
 'Ashleigh Gardner',
 'Jess Jonassen',
 'Nida Dar',
 'Sophie Ecclestone']
```

In [134]:

```
women_player_team=[]
for i in soup5.find_all('td',class_="table-body__cell nationality-logo"):
    women_player_team.append(i.text.strip('\n'))
```

In [135]:

```
women_player_team
```

Out[135]:

```
['AUS',
 'SA',
 'ENG',
 'AUS',
 'IND',
 'IND',
 'SL',
 'NZ',
 'AUS',
 'AUS',
 'SA',
 'AUS',
 'WI',
 'ENG',
 'SA',
 'IND',
 'SA',
 'IND',
 'ENG',
 'AUS',
 'SA',
 'NZ',
 'IND',
 'AUS',
 'AUS',
 'PAK',
 'ENG']
```

In [136]:

```
women_players_ratings=[]
for i in soup5.find_all('td',class_="table-body__cell u-text-right rating"):
    women_players_ratings.append(i.text)
```

In [137]:

```
women_players_ratings
```

Out[137]:

```
['754',
 '732',
 '731',
 '717',
 '716',
 '714',
 '655',
 '641',
 '626',
 '723',
 '722',
 '704',
 '660',
 '655',
 '634',
 '617',
 '598',
 '589',
 '371',
 '366',
 '349',
 '336',
 '322',
 '292',
 '250',
 '232',
 '205']
```

In [138]:

```
import pandas as pd
df5=pd.DataFrame({'rating':women_players_ratings,'names':women_player_names,'team':women
```

In [154]:

```
df5.head(10)
```

Out[154]:

	rating	names	team
0	754	Beth Mooney	AUS
1	732	Laura Wolvaardt	SA
2	731	Natalie Sciver	ENG
3	717	Meg Lanning	AUS
4	716	Harmanpreet Kaur	IND
5	714	Smriti Mandhana	IND
6	655	Chamari Athapaththu	SL
7	641	Amy Satterthwaite	NZ
8	626	Ellyse Perry	AUS
9	723	Jess Jonassen	AUS

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

In [140]:

```
women_allrounder=requests.get('https://www.icc-cricket.com/rankings/womens/player-rankin
```

In [141]:

```
women_allrounder
```

Out[141]:

```
<Response [200]>
```

In [142]:

```
soup6=BeautifulSoup(women_allrounder.content)
```

In [143]:

```
soup6
```

Out[143]:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta content="ICC Women's ODI Player Rankings | ICC" name="twitter:tit
le"/>
<meta content="website" property="og:type"/>
<meta content="summary_large_image" property="twitter:card"/>
<meta content="Official International Cricket Council rankings for ODI
match cricket players. Discover latest ICC rankings table, predict upco
ming matches, see points and ratings for all teams." name="descriptio
n"/>
<meta content="@icc" property="twitter:site"/>
<meta content="Official International Cricket Council rankings for ODI
match cricket players. Discover latest ICC rankings table, predict upco
ming matches, see points and ratings for all teams." name="twitter:desc
ription"/>
<meta content="https://www.icc-cricket.com/resources/ver/i/elements/def
ault-thumbnail.jpg" name="twitter:image"/>
```

In [146]:

```
women_allrounders=[]
for i in soup6.find_all('td',class_="table-body__cell name"):
    stars=i.find('a').text
    women_allrounders.append(stars)
```

In [147]:

```
women_allrounders
```

Out[147]:

```
['Beth Mooney',
 'Laura Wolvaardt',
 'Natalie Sciver',
 'Meg Lanning',
 'Harmanpreet Kaur',
 'Smriti Mandhana',
 'Chamari Athapaththu',
 'Amy Satterthwaite',
 'Ellyse Perry',
 'Jess Jonassen',
 'Shabnim Ismail',
 'Megan Schutt',
 'Hayley Matthews',
 'Kate Cross',
 'Ayabonga Khaka',
 'Rajeshwari Gayakwad',
 'Marizanne Kapp',
 'Deepti Sharma',
 'Natalie Sciver',
 'Ellyse Perry',
 'Marizanne Kapp',
 'Amelia Kerr',
 'Deepti Sharma',
 'Ashleigh Gardner',
 'Jess Jonassen',
 'Nida Dar',
 'Sophie Ecclestone']
```

In [149]:

```
woemn_allrounder_team=[]
for i in soup6.find_all('td',class_="table-body__cell nationality-logo"):
    woemn_allrounder_team.append(i.text.strip('\n'))
```

In [150]:

```
woemn_allrounder_team
```

Out[150]:

```
['AUS',
 'SA',
 'ENG',
 'AUS',
 'IND',
 'IND',
 'SL',
 'NZ',
 'AUS',
 'AUS',
 'SA',
 'AUS',
 'WI',
 'ENG',
 'SA',
 'IND',
 'SA',
 'IND',
 'ENG',
 'AUS',
 'SA',
 'NZ',
 'IND',
 'AUS',
 'AUS',
 'PAK',
 'ENG']
```

In [151]:

```
women_allrounder_ratings=[]
for i in soup6.find_all('td',class_="table-body__cell u-text-right rating"):
    women_allrounder_ratings.append(i.text)
```

In [152]:

```
women_allrounder_ratings
```

Out[152]:

```
['754',
 '732',
 '731',
 '717',
 '716',
 '714',
 '655',
 '641',
 '626',
 '723',
 '722',
 '704',
 '660',
 '655',
 '634',
 '617',
 '598',
 '589',
 '371',
 '366',
 '349',
 '336',
 '322',
 '292',
 '250',
 '232',
 '205']
```

In [156]:

```
import pandas as pd
df6=pd.DataFrame({'names':women_allrounders,'team':woemn_allrounder_team,'ratings':women
```

In [157]:

```
df6
```

Out[157]:

	names	team	ratings
0	Beth Mooney	AUS	754
1	Laura Wolvaardt	SA	732
2	Natalie Sciver	ENG	731
3	Meg Lanning	AUS	717
4	Harmanpreet Kaur	IND	716
5	Smriti Mandhana	IND	714
6	Chamari Athapaththu	SL	655
7	Amy Satterthwaite	NZ	641
8	Ellyse Perry	AUS	626
9	Jess Jonassen	AUS	723
10	Shabnim Ismail	SA	722

Question no:-7 Write a python program to scrape mentioned news details from https://www.cnbc.com/world/?region=world (https://www.cnbc.com/world/?region=world) and

```
make data frame
```

□i) Headline ii) Time iii) News Link

In [14]:

```
news_deatils=requests.get('https://www.cnbc.com/world/?region=world')
```

In [15]:

```
news_deatils
```

Out[15]:

```
<Response [200]>
```

In [16]:

```
soup7=BeautifulSoup(news_deatils.content)
```

In [17]:

```
soup7
```

Out[17]:

```
<!DOCTYPE html>
<html itemscope="" itemtype="https://schema.org/WebPage" lang="en" pref
ix="og=https://ogp.me/ns#"><head><meta content="telephone=no" name="for
mat-detection"/><style type="text/css">@charset "UTF-8";.RecaptchaAckno
wledgement-acknowledgement{color:#747474;flex:1;font-size:11px;font-wei
ght:600;line-height:15px;margin-bottom:7px;margin-top:24px;width:100%}.
RecaptchaAcknowledgement-acknowledgement a{color:#747474;font-weight:50
0;text-decoration:none}.RecaptchaAcknowledgement-acknowledgement a:hove
r{color:#747474;text-decoration:underline}.RecaptchaAcknowledgement-ack
nowledgement a:active{color:#747474}.RecaptchaAcknowledgement-reCaptcha
Padding{margin-top:15px}.RecaptchaAcknowledgement-centerAligned{text-al
ign:center}.RecaptchaAcknowledgement-leftAligned{text-align:left}.Recap
tchaAcknowledgement-rightAligned{text-align:right}.AuthForms-container
{margin:0 auto;padding:0 10px;width:458px}@media (max-width:759px){.Aut
hForms-container{max-width:458px;padding:20px 0 0;width:100%}}.AuthForm
s-container .AuthForms-signupContainer{margin:0 auto;padding:0 41px;tex
t-align:center;width:458px}@media (max-width:759px){.AuthForms-containe
r .AuthForms-signupContainer{padding:20px 0 0;width:100%}}.AuthForms-co
```

In [19]:

```
headline=[]
for i in soup7.find_all('a',class_="LatestNews-headline"):

    headline.append(i.text)
```

In [20]:

```
headline
```

Out[20]:

```
['The most overbought and oversold S&P 500 stocks include several tech nam
es',
 "Mark Cuban says paying Twitter for a blue check hasn't fixed his drop in
followers",
 'Goldman Sachs says these are its top picks coming out of earnings.',
 'What's next for SpaceX's Starship after a dramatic first launch',
 'Jim Cramer: Consumer goods stocks are set to keep running, so buy now',
 "Amazon drops on slowing cloud growth. Here's how the pros are playing i
t",
 'Why bitcoin keeps wavering between a store of value and a risk asset',
 'California bans the sale of new diesel trucks by 2036',
 'Markets next week have to contend with how aggressive the Fed drumbeat s
ounds ',
 'Here's how one investor is finding opportunities this earnings season',
 'These stocks reporting next week have a history of beating earnings expe
ctations',
 'How A.I. could change the future of work',
 'Carl Icahn calls Illumina Q1 results disappointing, slams cost-cutting p
lan',
 '31% of new crypto buyers influenced by friends. Be cautious, advisor say
s',
 'Series I bond rates fall to 4.3% amid cooling inflation',
 'House votes to restore solar panel tariffs, Biden vows veto if it passes
Senate',
 'These are the 7 best large U.S. cities for starting a new business',
 'Consumer-focused stocks outperform this week as earnings season rolls o
n',
 "Mark Cuban says he could get people to pay $100 for Twitter's blue check
marks",
 'Stocks making the biggest moves midday: First Republic, Snap, Amazon, In
tel and more',
 'These 7 Club holdings are using pricing power to boost profits this earn
ings season',
 'Tech earnings calls show mega-cap companies going big on A.I. ',
 "Spring cleaning can make you happier and more productive—here's how",
 'This Chinese social media platform is a buy that can surge 60%, UBS say
s',
 'Heading into retirement? Here are 4 tips to map out a plan',
 'Club meeting recap: Stocks gain, Eli Lilly, Amazon ',
 'Schumer demands Texas end judge cherry-picking after abortion pill rulin
g',
 'After being on pause for 3-plus years, student loan payments expected to
resume ',
 "United Airlines' plan to revamp narrow-body cabins faces supply chain de
lays",
 'Chinese hackers outnumber FBI cyber staff 50 to 1, bureau director say
s']
```

In [25]:

```
print(len(headline))
```

```
30
```

In [23]:

```
time=[]
for i in soup7.find_all('time',class_="LatestNews-timestamp"):
    time.append(i.text)
```

In [24]:

```
time
```

Out[24]:

```
['18 Min Ago',
 '22 Min Ago',
 '22 Min Ago',
 '52 Min Ago',
 '13 Hours Ago',
 '16 Hours Ago',
 '16 Hours Ago',
 '17 Hours Ago',
 '17 Hours Ago',
 '18 Hours Ago',
 '18 Hours Ago',
 '18 Hours Ago',
 '18 Hours Ago',
 '18 Hours Ago',
 '19 Hours Ago',
 '19 Hours Ago',
 '19 Hours Ago',
 '19 Hours Ago',
 '20 Hours Ago',
 '20 Hours Ago',
 '21 Hours Ago',
 '21 Hours Ago',
 '21 Hours Ago',
 '21 Hours Ago',
 '21 Hours Ago',
 '21 Hours Ago',
 '21 Hours Ago',
 '22 Hours Ago',
 '22 Hours Ago',
 '22 Hours Ago']
```

In [33]:

```
news_link=[]
for i in soup7.find_all('div',class_="LatestNews-container"):
    news=i.find('a')
    new=news.get('href')
    news_link.append(new)
```

In [34]:

```
news_link
```

Out[34]:

```
['/pro/',
 'https://www.cnbc.com/2023/04/29/mark-cuban-paying-for-twitter-blue-didnt
-stop-me-from-losing-followers.html',
 '/pro/',
 'https://www.cnbc.com/2023/04/29/spacex-starship-whats-next.html',
 'https://www.cnbc.com/2023/04/28/jim-cramer-consumer-goods-stocks-are-set
-to-keep-running-and-its-not-too-late-to-buy.html',
 '/pro/',
 '/pro/',
 'https://www.cnbc.com/2023/04/28/california-bans-the-sale-of-new-diesel-t
rucks-by-2036.html',
 '/pro/',
 '/pro/',
 '/pro/',
 'https://www.cnbc.com/2023/04/28/how-ai-could-change-the-future-of-work.h
tml',
 'https://www.cnbc.com/2023/04/28/carl-icahn-slams-illumina-q1-results-and
-cost-cutting-plan.html',
 'https://www.cnbc.com/2023/04/28/many-new-bitcoin-crypto-buyers-influence
d-by-friends-why-to-be-cautious.html',
 'https://www.cnbc.com/2023/04/28/series-i-bond-rates-fall-to-4point3perce
nt-amid-cooling-inflation.html',
 'https://www.cnbc.com/2023/04/28/house-votes-to-restore-solar-panel-tarif
fs-biden-vows-veto.html',
 'https://www.cnbc.com/2023/04/28/best-large-us-cities-for-new-startups-or
lando-durham-boise.html',
 '/pro/',
 'https://www.cnbc.com/2023/04/28/mark-cuban-elon-musk-made-huge-mistake-w
ith-twitter-blue-marketing.html',
 'https://www.cnbc.com/2023/04/28/stocks-making-the-biggest-moves-midday-f
rc-snap-amzn-intc-and-more.html',
 '/investingclub/',
 'https://www.cnbc.com/2023/04/28/tech-earnings-calls-show-mega-cap-compan
ies-going-big-on-ai-.html',
 'https://www.cnbc.com/2023/04/28/3-ways-spring-cleaning-can-boost-your-ha
ppiness-and-productivity.html',
 '/pro/',
 'https://www.cnbc.com/2023/04/28/heading-into-retirement-here-are-some-ke
y-tips-to-map-out-a-game-plan.html',
 '/investingclub/',
 'https://www.cnbc.com/2023/04/28/schumer-demands-judge-shopping-stop-afte
r-texas-abortion-lgbtq-fights.html',
 'https://www.cnbc.com/2023/04/28/after-3-year-pause-student-loan-payments
-expected-to-resume-soon.html',
 'https://www.cnbc.com/2023/04/28/united-airlines-revamped-cabins-behind-s
chedule.html',
 'https://www.cnbc.com/2023/04/28/chinese-hackers-outnumber-fbi-cyber-staf
f-50-to-1-director-wray-says.html']
```

In [37]:

```
import pandas as pd
df7=pd.DataFrame({'headline':headline,'time':time,'news_link':news_link})
df7
```

Out[37]:

	headline	time	news_link
0	The most overbought and oversold S&P 500 stock...	18 Min Ago	/pro/
1	Mark Cuban says paying Twitter for a blue chec...	22 Min Ago	https://www.cnbc.com/2023/04/29/mark-cuban-pay...
2	Goldman Sachs says these are its top picks com...	22 Min Ago	/pro/
3	What’s next for SpaceX’s Starship after a dram...	52 Min Ago	https://www.cnbc.com/2023/04/29/spacex-starshi...
4	Jim Cramer: Consumer goods stocks are set to k...	13 Hours Ago	https://www.cnbc.com/2023/04/28/jim-cramer-con...
5	Amazon drops on slowing cloud growth. Here's h...	16 Hours Ago	/pro/
6	Why bitcoin keeps wavering between a store of ...	16 Hours Ago	/pro/
7	California bans the sale of new diesel trucks ...	17 Hours Ago	https://www.cnbc.com/2023/04/28/california-ban...
8	Markets next week have to contend with how agg...	17 Hours Ago	/pro/
9	Here’s how one investor is finding opportuniti...	18 Hours Ago	/pro/
10	These stocks reporting next week have a histor...	18 Hours Ago	/pro/
11	How A.I. could change the future of work	18 Hours Ago	https://www.cnbc.com/2023/04/28/how-ai-could-c...
12	Carl Icahn calls Illumina Q1 results disappoin...	18 Hours Ago	https://www.cnbc.com/2023/04/28/carl-icahn-sla...
13	31% of new crypto buyers influenced by friends...	18 Hours Ago	https://www.cnbc.com/2023/04/28/many-new-bitco...
14	Series I bond rates fall to 4.3% amid cooling ...	19 Hours Ago	https://www.cnbc.com/2023/04/28/series-i-bond-...
15	House votes to restore solar panel tariffs, Bi...	19 Hours Ago	https://www.cnbc.com/2023/04/28/house-votes-to...
16	These are the 7 best large U.S. cities for sta...	19 Hours Ago	https://www.cnbc.com/2023/04/28/best-large-us-...
17	Consumer-focused stocks outperform this week a...	19 Hours Ago	/pro/
18	Mark Cuban says he could get people to pay $10...	20 Hours Ago	https://www.cnbc.com/2023/04/28/mark-cuban-elo...
19	Stocks making the biggest moves midday: First ...	20 Hours Ago	https://www.cnbc.com/2023/04/28/stocks-making-...
20	These 7 Club holdings are using pricing power ...	21 Hours Ago	/investingclub/
21	Tech earnings calls show mega-cap companies go...	21 Hours Ago	https://www.cnbc.com/2023/04/28/tech-earnings-...
22	Spring cleaning can make you happier and more ...	21 Hours Ago	https://www.cnbc.com/2023/04/28/3-ways-spring-...
23	This Chinese social media platform is a buy th...	21 Hours Ago	/pro/

	headline	time	news_link
24	Heading into retirement? Here are 4 tips to ma...	21 Hours Ago	https://www.cnbc.com/2023/04/28/heading-into-r...
25	Club meeting recap: Stocks gain, Eli Lilly, Am...	21 Hours Ago	/investingclub/
26	Schumer demands Texas end judge cherry-picking...	21 Hours Ago	https://www.cnbc.com/2023/04/28/schumer-demand...
27	After being on pause for 3-plus years, student...	22 Hours Ago	https://www.cnbc.com/2023/04/28/after-3-year-p...
28	United Airlines' plan to revamp narrow-body ca...	22 Hours Ago	https://www.cnbc.com/2023/04/28/united-airline...
29	Chinese hackers outnumber FBI cyber staff 50 t...	22 Hours Ago	https://www.cnbc.com/2023/04/28/chinese-hacker...

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 (https://www.journals.elsevier.com/artificial-intelligence/most-downloaded-articles)](https://www.journals.elsevier.com/artificial-intelligence/most-downloaded-articles) Scrape below mentioned details and make data frame□ i) Paper Title ii) Authors iii) Published Date iv) Paper URL

In [7]:

```
AI_article=requests.get('https://www.journals.elsevier.com/artificial-intelligence/most-
```

In [8]:

```
AI_article
```

Out[8]:

```
<Response [200]>
```

In [9]:

```
soup8=BeautifulSoup(AI_article.content)
```

In [10]:

```
soup8
```

Out[10]:

```
<!DOCTYPE html>
<html><head><meta charset="utf-8"/><meta content="width=device-width" n
ame="viewport"/><meta content="en_US" name="og:locale"/><meta content
="Most Downloaded Articles - Artificial Intelligence - Journal - Elsevi
er" property="og:title"/><meta content="The journal of Artificial Intel
ligence (AIJ)  welcomes papers on broad aspects of AI that constitute a
dvances in the overall field including, but not limited …" property="o
g:description"/><meta content="http://ars.els-cdn.com/content/image/X00
043702.jpg" name="og:image" property="og:image"/><meta content="http://
ars.els-cdn.com/content/image/X00043702.jpg" name="og:image:url" proper
ty="og:image:url"/><meta content="https://ars.els-cdn.com/content/imag
e/X00043702.jpg" name="og:image:secure_url" property="og:image:secure_u
rl"/><meta content="journals.elsevier.com/artificial-intelligence/most-
downloaded-articles" name="og:url"/><meta content="website" property="o
g:type"/><link href="/apple-touch-icon.png" rel="apple-touch-icon" size
s="180x180"/><link href="/favicon-32x32.png" rel="icon" sizes="32x32" t
ype="image/png"/><link href="/favicon-16x16.png" rel="icon" sizes="16x1
6" type="image/png"/><link color="#ff6c00" href="/safari-pinned-tab.sv
```

In [11]:

```
paper_title=[]
for i in soup8.find_all('li',class_="sc-9zxyh7-1 sc-9zxyh7-2 kOEIEO hvoVxs"):
    art=i.find('a').text
    paper_title.append(art)
```

In [12]:

```
paper_title
```

Out[12]:

```
['Reward is enough',
 'Making sense of raw input',
 'Law and logic: A review from an argumentation perspective',
 'Creativity and artificial intelligence',
 'Artificial cognition for social human–robot interaction: An implementati
on',
 'Explanation in artificial intelligence: Insights from the social science
s',
 'Making sense of sensory input',
 'Conflict-based search for optimal multi-agent pathfinding',
 'Between MDPs and semi-MDPs: A framework for temporal abstraction in rein
forcement learning',
 'The Hanabi challenge: A new frontier for AI research',
 'Evaluating XAI: A comparison of rule-based and example-based explanation
s',
 'Argumentation in artificial intelligence',
 'Algorithms for computing strategies in two-player simultaneous move game
s',
 'Multiple object tracking: A literature review',
 'Selection of relevant features and examples in machine learning',
 'A survey of inverse reinforcement learning: Challenges, methods and prog
ress',
 'Explaining individual predictions when features are dependent: More accu
rate approximations to Shapley values',
 'A review of possible effects of cognitive biases on interpretation of ru
le-based machine learning models',
 'Integrating social power into the decision-making of cognitive agents',
 "“That's (not) the output I expected!” On the role of end user expectatio
ns in creating explanations of AI systems",
 'Explaining black-box classifiers using post-hoc explanations-by-example:
The effect of explanations and error-rates in XAI user studies',
 'Algorithm runtime prediction: Methods & evaluation',
 'Wrappers for feature subset selection',
 'Commonsense visual sensemaking for autonomous driving – On generalised n
eurosymbolic online abduction integrating vision and semantics',
 'Quantum computation, quantum theory and AI']
```

In [15]:

```
authors=[]
for i in soup8.find_all('span',class_="sc-1w3fpd7-0 dnCnAO"):
    authors.append(i.text)
```

In [16]:

```
authors
```

Out[16]:

```
['Silver, David, Singh, Satinder, Precup, Doina, Sutton, Richard S. ',
 'Evans, Richard, Bošnjak, Matko and 5 more',
 'Prakken, Henry, Sartor, Giovanni ',
 'Boden, Margaret A. ',
 'Lemaignan, Séverin, Warnier, Mathieu and 3 more',
 'Miller, Tim ',
 'Evans, Richard, Hernández-Orallo, José and 3 more',
 'Sharon, Guni, Stern, Roni, Felner, Ariel, Sturtevant, Nathan R. ',
 'Sutton, Richard S., Precup, Doina, Singh, Satinder ',
 'Bard, Nolan, Foerster, Jakob N. and 13 more',
 'van der Waa, Jasper, Nieuwburg, Elisabeth, Cremers, Anita, Neerincx, Mar
k ',
 'Bench-Capon, T.J.M., Dunne, Paul E. ',
 'Bošanský, Branislav, Lisý, Viliam and 3 more',
 'Luo, Wenhan, Xing, Junliang and 4 more',
 'Blum, Avrim L., Langley, Pat ',
 'Arora, Saurabh, Doshi, Prashant ',
 'Aas, Kjersti, Jullum, Martin, Løland, Anders ',
 'Kliegr, Tomáš, Bahník, Štěpán, Fürnkranz, Johannes ',
 'Pereira, Gonçalo, Prada, Rui, Santos, Pedro A. ',
 'Riveiro, Maria, Thill, Serge ',
 'Kenny, Eoin M., Ford, Courtney, Quinn, Molly, Keane, Mark T. ',
 'Hutter, Frank, Xu, Lin, Hoos, Holger H., Leyton-Brown, Kevin ',
 'Kohavi, Ron, John, George H. ',
 'Suchan, Jakob, Bhatt, Mehul, Varadarajan, Srikrishna ',
 'Ying, Mingsheng ']
```

In [17]:

```
date=[]
for i in soup8.find_all('span',class_="sc-1thf9ly-2 dvggWt"):
    date.append(i.text)
```

In [18]:

```
date
```

Out[18]:

```
['October 2021',
 'October 2021',
 'October 2015',
 'August 1998',
 'June 2017',
 'February 2019',
 'April 2021',
 'February 2015',
 'August 1999',
 'March 2020',
 'February 2021',
 'October 2007',
 'August 2016',
 'April 2021',
 'December 1997',
 'August 2021',
 'September 2021',
 'June 2021',
 'December 2016',
 'September 2021',
 'May 2021',
 'January 2014',
 'December 1997',
 'October 2021',
 'February 2010']
```

In [28]:

```
paper_url=[]
for i in soup8.find_all('a'):
    paper_url.append(i.get('href'))
```

In [41]:

```
paper_urls=paper_url[0:25]
```

In [42]:

```
paper_urls
```

Out[42]:

```
['#skip-to-content-anchor',
 'http://www.elsevier.com',
 'https://account.elsevier.com/auth',
 'https://elsevier.com/about',
 'https://www.elsevier.com/connect',
 'https://www.elsevier.com/about/careers',
 'https://elsevier.com/about',
 'https://www.elsevier.com/connect',
 'https://www.elsevier.com/about/careers',
 'https://www.elsevier.com/rd-solutions',
 'https://www.elsevier.com/clinical-solutions',
 'https://www.elsevier.com/research-platforms',
 'https://www.elsevier.com/research-intelligence',
 'https://www.elsevier.com/education',
 'https://www.elsevier.com/solutions',
 'https://www.elsevier.com/rd-solutions',
 'https://www.elsevier.com/clinical-solutions',
 'https://www.elsevier.com/research-platforms',
 'https://www.elsevier.com/research-intelligence',
 'https://www.elsevier.com/education',
 'https://www.elsevier.com/solutions',
 'https://www.elsevier.com/authors',
 'https://www.elsevier.com/editors',
 'https://www.elsevier.com/reviewers',
 'https://www.elsevier.com/librarians']
```

In [43]:

```
import pandas as pd
df10=pd.DataFrame({'paper_title':paper_title,'authors':authors,'dates':date,'url':paper_
```

In [44]:

```
df10
```

Out[44]:

	paper_title	authors	dates	url
0	Reward is enough	Silver, David, Singh, Satinder, Precup, Doina,...	October 2021	#skip-to-content-anchor
1	Making sense of raw input	Evans, Richard, Bošnjak, Matko and 5 more	October 2021	http://www.elsevier.com
2	Law and logic: A review from an argumentation ...	Prakken, Henry, Sartor, Giovanni	October 2015	https://account.elsevier.com/auth
3	Creativity and artificial intelligence	Boden, Margaret A.	August 1998	https://elsevier.com/about
4	Artificial cognition for social human–robot in...	Lemaignan, Séverin, Warnier, Mathieu and 3 more	June 2017	https://www.elsevier.com/connect
5	Explanation in artificial intelligence: Insigh...	Miller, Tim	February 2019	https://www.elsevier.com/about/careers
6	Making sense of sensory input	Evans, Richard, Hernández-Orallo, José and 3 more	April 2021	https://elsevier.com/about
7	Conflict-based search for optimal multi-agent ...	Sharon, Guni, Stern, Roni, Felner, Ariel, Stur...	February 2015	https://www.elsevier.com/connect
8	Between MDPs and semi-MDPs: A framework for te...	Sutton, Richard S., Precup, Doina, Singh, Sati...	August 1999	https://www.elsevier.com/about/careers
9	The Hanabi challenge: A new frontier for AI re...	Bard, Nolan, Foerster, Jakob N. and 13 more	March 2020	https://www.elsevier.com/rd-solutions
10	Evaluating XAI: A comparison of rule-based and...	van der Waa, Jasper, Nieuwburg, Elisabeth, Cre...	February 2021	https://www.elsevier.com/clinical-solutions
11	Argumentation in artificial intelligence	Bench-Capon, T.J.M., Dunne, Paul E.	October 2007	https://www.elsevier.com/research-platforms
12	Algorithms for computing strategies in two-pla...	Bošanský, Branislav, Lisý, Viliam and 3 more	August 2016	https://www.elsevier.com/research-intelligence
13	Multiple object tracking: A literature review	Luo, Wenhan, Xing, Junliang and 4 more	April 2021	https://www.elsevier.com/education
14	Selection of relevant features and examples in...	Blum, Avrim L., Langley, Pat	December 1997	https://www.elsevier.com/solutions
15	A survey of inverse reinforcement learning: Ch...	Arora, Saurabh, Doshi, Prashant	August 2021	https://www.elsevier.com/rd-solutions
16	Explaining individual predictions when feature...	Aas, Kjersti, Jullum, Martin, Løland, Anders	September 2021	https://www.elsevier.com/clinical-solutions
17	A review of possible effects of cognitive bias...	Kliegr, Tomáš, Bahník, Štěpán, Fürnkranz, Joha...	June 2021	https://www.elsevier.com/research-platforms

	paper_title	authors	dates	url
18	Integrating social power into the decision-mak...	Pereira, Gonçalo, Prada, Rui, Santos, Pedro A.	December 2016	https://www.elsevier.com/research-intelligence
19	“That's (not) the output I expected!” On the r...	Riveiro, Maria, Thill, Serge	September 2021	https://www.elsevier.com/education
20	Explaining black-box classifiers using post-ho...	Kenny, Eoin M., Ford, Courtney, Quinn, Molly, ...	May 2021	https://www.elsevier.com/solutions
21	Algorithm runtime prediction: Methods & evalua...	Hutter, Frank, Xu, Lin, Hoos, Holger H., Leyto...	January 2014	https://www.elsevier.com/authors
22	Wrappers for feature subset selection	Kohavi, Ron, John, George H.	December 1997	https://www.elsevier.com/editors
23	Commonsense visual sensemaking for autonomous ...	Suchan, Jakob, Bhatt, Mehul, Varadarajan, Srik...	October 2021	https://www.elsevier.com/reviewers
24	Quantum computation, quantum theory and AI	Ying, Mingsheng	February 2010	https://www.elsevier.com/librarians

9) Write a python program to scrape mentioned details from dineout.co.in and make data frame□

i) Restaurant name ii) Cuisine iii) Location iv) Ratings v) Image UR

In [6]:

```
dineout=requests.get('https://www.dineout.co.in/delhi-restaurants/buffet-special')
```

In [7]:

```
dineout
```

Out[7]:

```
<Response [200]>
```

In [10]:

```
soup9=BeautifulSoup(dineout.content)
```

In [11]:

```
soup9
```

Out[11]:

```
<!DOCTYPE html>
<html lang="en"><head><meta charset="utf-8"/><meta content="IE=edge" ht
tp-equiv="X-UA-Compatible"/><meta content="width=device-width, initial-
scale=1.0, maximum-scale=1.0, user-scalable=no" name="viewport"/><link
href="/manifest.json" rel="manifest"/><style type="text/css">
            @font-face {
                font-family: 'dineicon';
                src:  url('/fonts/dineicon.eot');
                src:  url('/fonts/dineicon.eot#iefix') format('embedded
-opentype'),
                url('/fonts/dineicon.ttf') format('truetype'),
                url('/fonts/dineicon.woff') format('woff'),
                url('/fonts/dineicon.svg#dineicon') format('svg');
                font-weight: normal;
                                font-style: normal;
                                font-display: swap;
            }
            .hide {
```

In [17]:

```
resturant_name=[]
for i in soup9.find_all('div',class_="restnt-info cursor"):
    restur=i.find('a').text
    resturant_name.append(restur)
```

In [18]:

```
resturant_name
```

Out[18]:

```
['Castle Barbeque',
 'Jungle Jamboree',
 'Cafe Knosh',
 'Castle Barbeque',
 'The Barbeque Company',
 'India Grill',
 'Delhi Barbeque',
 'The Monarch - Bar Be Que Village',
 'Indian Grill Room']
```

In [19]:

```
cusine=[]
for i in soup9.find_all('span',class_="double-line-ellipsis"):
    cus=i.find('a').text
    cusine.append(cus)
```

In [20]:

```
cusine
```

Out[20]:

```
['Chinese',
 'North Indian',
 'Italian',
 'Chinese',
 'North Indian',
 'North Indian',
 'North Indian',
 'North Indian',
 'North Indian']
```

In [22]:

```
location=[]
for i in soup9.find_all('div',class_="restnt-loc ellipsis"):
    location.append(i.text)
```

In [24]:

```
location
```

Out[24]:

```
['Connaught Place, Central Delhi',
 '3CS Mall,Lajpat Nagar - 3, South Delhi',
 'The Leela Ambience Convention Hotel,Shahdara, East Delhi',
 'Pacific Mall,Tagore Garden, West Delhi',
 'Gardens Galleria,Sector 38A, Noida',
 'Hilton Garden Inn,Saket, South Delhi',
 'Taurus Sarovar Portico,Mahipalpur, South Delhi',
 'Indirapuram Habitat Centre,Indirapuram, Ghaziabad',
 'Suncity Business Tower,Golf Course Road, Gurgaon']
```

In [25]:

```
resturant_rating=[]
for i in soup9.find_all('div',class_="restnt-rating rating-4"):
    resturant_rating.append(i.text)
```

In [27]:

```
resturant_rating
```

Out[27]:

```
['4', '3.9', '4.3', '3.9', '3.9', '3.9', '3.7', '3.8', '4.3']
```

In [28]:

```
resturant_url=[]
for i in soup9.find_all('img',class_="no-img"):
    resturant_url.append(i.get('data-src'))
```

In [29]:

```
resturant_url
```

Out[29]:

```
['https://im1.dineout.co.in/images/uploads/restaurant/sharpen/8/k/b/p86792
-16062953735fbe1f4d3fb7e.jpg?tr=tr:n-medium',
 'https://im1.dineout.co.in/images/uploads/restaurant/sharpen/5/p/m/p59633
-166088382462ff137009010.jpg?tr=tr:n-medium',
 'https://im1.dineout.co.in/images/uploads/restaurant/sharpen/4/p/m/p406-1
5438184745c04ccea491bc.jpg?tr=tr:n-medium',
 'https://im1.dineout.co.in/images/uploads/restaurant/sharpen/3/j/o/p38113
-15959192065f1fcb666130c.jpg?tr=tr:n-medium',
 'https://im1.dineout.co.in/images/uploads/restaurant/sharpen/7/p/k/p79307
-16051787755fad1597f2bf9.jpg?tr=tr:n-medium',
 'https://im1.dineout.co.in/images/uploads/restaurant/sharpen/2/v/t/p2687-
1482477169585cce712b90f.jpg?tr=tr:n-medium',
 'https://im1.dineout.co.in/images/uploads/restaurant/sharpen/5/d/i/p52501
-1661855212630de5eceb6d2.jpg?tr=tr:n-medium',
 'https://im1.dineout.co.in/images/uploads/restaurant/sharpen/3/n/o/p34822
-15599107305cfa594a13c24.jpg?tr=tr:n-medium',
 'https://im1.dineout.co.in/images/uploads/restaurant/sharpen/5/y/f/p549-1
65000147262590640c0afc.jpg?tr=tr:n-medium']
```

In [32]:

```
import pandas as pd
df11=pd.DataFrame({'resturant_name':resturant_name,'cusine':cusine,'loaction':location,'
```

In [33]:

```
df11
```

Out[33]:

	resturant_name	cusine	loaction	resturant_rating	
0	Castle Barbeque	Chinese	Connaught Place, Central Delhi	4	https://im1.dineout.co.in/images/
1	Jungle Jamboree	North Indian	3CS Mall,Lajpat Nagar - 3, South Delhi	3.9	https://im1.dineout.co.in/images/
2	Cafe Knosh	Italian	The Leela Ambience Convention Hotel,Shahdara, ...	4.3	https://im1.dineout.co.in/images/
3	Castle Barbeque	Chinese	Pacific Mall,Tagore Garden, West Delhi	3.9	https://im1.dineout.co.in/images/
4	The Barbeque Company	North Indian	Gardens Galleria,Sector 38A, Noida	3.9	https://im1.dineout.co.in/images/
5	India Grill	North Indian	Hilton Garden Inn,Saket, South Delhi	3.9	https://im1.dineout.co.in/images/
6	Delhi Barbeque	North Indian	Taurus Sarovar Portico,Mahipalpur, South Delhi	3.7	https://im1.dineout.co.in/images/
7	The Monarch - Bar Be Que Village	North Indian	Indirapuram Habitat Centre,Indirapuram, Ghaziabad	3.8	https://im1.dineout.co.in/images/
8	Indian Grill Room	North Indian	Suncity Business Tower,Golf Course Road, Gurgaon	4.3	https://im1.dineout.co.in/images/

In [ ]:

```

```