In [138]:

```
!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\nitu calla\appdata\roaming
\python\python310\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) (2.3.2.post1)
Defaulting to user installation because normal site-packages is not writea
ble
Requirement already satisfied: requests in c:\programdata\anaconda3\lib\si
te-packages (2.28.1)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\programdata\ana
conda3\lib\site-packages (from requests) (1.26.14)
Requirement already satisfied: certifi>=2017.4.17 in c:\programdata\anacon
da3\lib\site-packages (from requests) (2022.12.7)
Requirement already satisfied: charset-normalizer<3,>=2 in c:\programdata
\anaconda3\lib\site-packages (from requests) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\programdata\anaconda3\li
b\site-packages (from requests) (3.4)
```

In [139]:

```
#Now importing necessory  libraries
from bs4 import BeautifulSoup
import requests
```

Question1) Write a python program to display all the header tags from wikipedia.org and make data frame.

In [140]:

```
# First requesting website to get permission for scraping
page = requests.get ('https://www.wikipedia.org/')
```

In [141]:

```
page
```

Out[141]:

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

In [142]:

```
# Now when our request got approved as we got response [200]
# Now we will scrap all content from the said page and store in variable soup
soup = BeautifulSoup(page.content)
soup
```

Out[142]:

```
<!DOCTYPE html>
<html class="no-js" lang="en">
<head>
<meta charset="utf-8"/>
<title>Wikipedia</title>
<meta content="Wikipedia is a free online encyclopedia, created and edit
ed by volunteers around the world and hosted by the Wikimedia Foundatio
n." name="description"/>
<script>
document.documentElement.className = document.documentElement.className.
replace( /(^|\s)no-js(\s|$)/, "$1js-enabled$2" );
</script>
<meta content="initial-scale=1,user-scalable=yes" name="viewport"/>
<link href="/static/apple-touch/wikipedia.png" rel="apple-touch-icon"/>
<link href="/static/favicon/wikipedia.ico" rel="shortcut icon"/>
<link href="//creativecommons.org/licenses/by-sa/4.0/" rel="license"/>
<style>
.sprite{background-image:linear-gradient(transparent,transparent),url(po
```

In [143]:

```
# lets find all header tags (from h1 to h6)
# creating a list
header_tags= []
for i in soup.find_all(['h1','h2','h3','h4','h5','h6']):
    header_tags.append(i.text.strip())

header_tags
```

Out[143]:

```
['Wikipedia\n\nThe Free Encyclopedia',
 '1\xa0000\xa0000+\n\n\narticles',
 '100\xa0000+\n\n\narticles',
 '10\xa0000+\n\n\narticles',
 '1\xa0000+\n\n\narticles',
 '100+\n\n\narticles']
```

In [144]:

```python
# Now creating a dataframe
import pandas as pd
df = pd.DataFrame({'Header':header_tags})
df
```

Out[144]:

	Header
0	Wikipedia\n\nThe Free Encyclopedia
1	1 000 000+\n\n\narticles
2	100 000+\n\n\narticles
3	10 000+\n\n\narticles
4	1 000+\n\n\narticles
5	100+\n\n\narticles

In [ ]:

```python

```

Question 2) 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 (https://presidentofindia.nic.in/former-presidents.htm) and make data frame.

In [145]:

```python
# Now same first requesting website to get permission for scraping
# Question given url is giving wrong result, so taking below
page = requests.get ('https://presidentofindia.nic.in/former-presidents')
page
```

Out[145]:

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

In [146]:

```
# Now when our request got approved as we got response [200]
# Now we will scrap all content from the said page and store in variable soup
soup = BeautifulSoup(page.content)
soup
```

Out[146]:

```
<!DOCTYPE html>
<html dir="ltr" lang="en">
<head>
<meta charset="utf-8"/>
<meta content="Drupal 9 (https://www.drupal.org)" name="Generator"/>
<meta content="width" name="MobileOptimized"/>
<meta content="true" name="HandheldFriendly"/>
<meta content="width=device-width, initial-scale=1.0" name="viewport"/>
<link href="/sites/default/files/tiranga_1.png" rel="icon" type="image/p
ng"/>
<title>Former Presidents of India | President of India</title>
<link href="/libraries/superfish/css/superfish.css?rziu9l" media="all" r
el="stylesheet"/>
<link href="/core/modules/system/css/components/ajax-progress.module.cs
s?rziu9l" media="all" rel="stylesheet"/>
<link href="/core/modules/system/css/components/align.module.css?rziu9l"
media="all" rel="stylesheet"/>
<link href="/core/modules/system/css/components/autocomplete-loading.mod
```

In [147]:

```
# Now we have to display all name of all former presidents and term of office .
president_name=[]
for i in soup.find_all('div',class_="desc-sec"):
    name=i.text.strip()
    president_name.append(name)
president_name
```

Out[147]:

```
['Shri Ram Nath Kovind\n14th President of India',
 'Shri Pranab Mukherjee\n13th President of India',
 'Smt Pratibha Devisingh Patil\n12th President of India',
 'DR. A.P.J. Abdul Kalam\n11th President of India',
 'Shri K. R. Narayanan\n10th President of India',
 'Dr Shankar Dayal Sharma\n9th  President of India',
 'Shri R Venkataraman\n8th President of India',
 'Giani Zail Singh\n7th President of India',
 'Shri Neelam Sanjiva Reddy\n6th President of India',
 'Dr. Fakhruddin Ali Ahmed\n5th President of India',
 'Shri Varahagiri Venkata Giri\n4th President of India',
 'Dr. Zakir Husain\n3rd President of India',
 'Dr. Sarvepalli Radhakrishnan\n2nd President of India',
 'Dr. Rajendra Prasad\n1st President of India']
```

In [148]:

```
# Now we creating dataframe
import pandas as pd
df = pd.DataFrame({'Former President Name ':president_name })
df
```

Out[148]:

	Former President Name
0	Shri Ram Nath Kovind\n14th President of India
1	Shri Pranab Mukherjee\n13th President of India
2	Smt Pratibha Devisingh Patil\n12th President o...
3	DR. A.P.J. Abdul Kalam\n11th President of India
4	Shri K. R. Narayanan\n10th President of India
5	Dr Shankar Dayal Sharma\n9th President of India
6	Shri R Venkataraman\n8th President of India
7	Giani Zail Singh\n7th President of India
8	Shri Neelam Sanjiva Reddy\n6th President of India
9	Dr. Fakhruddin Ali Ahmed\n5th President of India
10	Shri Varahagiri Venkata Giri\n4th President of...
11	Dr. Zakir Husain\n3rd President of India
12	Dr. Sarvepalli Radhakrishnan\n2nd President of...
13	Dr. Rajendra Prasad\n1st President of India

In [ ]:

```

```

Question 3) 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.

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

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

In [149]:

```
#a) Top 10 ODI teams in men’s cricket along with the records for matches, points and
# Now same first requesting website to get permission for scraping
# Question given url is giving wrong result, so taking below
page = requests.get ('https://www.icc-cricket.com/rankings/mens/team-rankings/odi')
page
```

Out[149]:

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

In [150]:

```
# Now when our request got approved as we got response [200]
# Now we will scrap all content from the said page and store in variable soup
soup = BeautifulSoup(page.content)
soup
```

Out[150]:

```
<!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 Da
y International (ODI) cricket teams. Discover latest ICC rankings table,
predict upcoming matches, see points and ratings for all teams." name="d
escription"/>
<meta content="@icc" property="twitter:site"/>
<meta content="Official International Cricket Council ranking for One Da
y International (ODI) cricket teams. Discover latest ICC rankings table,
predict upcoming matches, see points and ratings for all teams." name="t
witter:description"/>
<meta content="https://www.icc-cricket.com/resources/ver/i/elements/defa
ult-thumbnail.jpg" name="twitter:image"/>
<meta content="ICC Men's ODI Team Rankings | ICC" property="og:title"/>
```

In [151]:

```
# we observed the table in website ( apart from name of team, rest all) designed suc
# extracting data seperatly from banner and rest of the table

```

In [152]:

```
# Now lets create a  list of name the banner team name .
banner_team_name=[]
for i in soup.find_all('span',class_="u-hide-phablet")[0:1]:
    banner_team_name.append(i.text)
banner_team_name
```

Out[152]:

```
['Australia']
```

In [153]:

```
# Now lets create a  list of name the banner team matches .
banner_team_matches=[]
for i in soup.find_all('td',class_="rankings-block__banner--matches"):
    banner_team_matches.append(i.text)
banner_team_matches
```

Out[153]:

```
['23']
```

In [154]:

```
# similarly lets create a  list of banner team points .
banner_team_points=[]
for i in soup.find_all('td',class_="rankings-block__banner--points"):
    banner_team_points.append(i.text)
banner_team_points
```

Out[154]:

```
['2,714']
```

In [155]:

```
# similarly lets create a  list of banner team rating .
banner_team_rating=[]
for i in soup.find_all('td',class_="rankings-block__banner--rating u-text-right"):
    rating=i.text.strip()
    banner_team_rating.append(rating)
banner_team_rating
```

Out[155]:

```
['118']
```

In [156]:

```
# Now lets create a  list of name the rest of top 9 teams .
rest_teams=[]
for i in soup.find_all('span',class_="u-hide-phablet")[1:10]:
    rest_teams.append(i.text)
rest_teams
```

Out[156]:

```
['Pakistan',
 'India',
 'New Zealand',
 'England',
 'South Africa',
 'Bangladesh',
 'Afghanistan',
 'Sri Lanka',
 'West Indies']
```

In [157]:

```
# Now we observed number of matches and points are mentioned in same tag
#lets create a  list  first and then seperate the list of the rest of top 9 teams .
matches_points=[]
for i in soup.find_all('td',class_="table-body__cell u-center-text")[0:18]:
    matches_points.append(i.text)
matches_points
```

Out[157]:

```
['20',
 '2,316',
 '36',
 '4,081',
 '27',
 '2,806',
 '24',
 '2,426',
 '19',
 '1,910',
 '28',
 '2,661',
 '16',
 '1,404',
 '32',
 '2,794',
 '38',
 '2,582']
```

In [158]:

```
# Now seperating list in 2 lists number of matches and points
matches = matches_points[::2]  # it will extract number of matches - even indexed ele
points = matches_points[1::2]  # it will extract points of team - odd indexed element
matches

```

Out[158]:

```
['20', '36', '27', '24', '19', '28', '16', '32', '38']
```

In [159]:

```
points
```

Out[159]:

```
['2,316',
 '4,081',
 '2,806',
 '2,426',
 '1,910',
 '2,661',
 '1,404',
 '2,794',
 '2,582']
```

In [160]:

```
# Now lets create a  list of rating of  rest of top 9 teams .
ratings=[]
for i in soup.find_all('td',class_="table-body__cell u-text-right rating")[0:9]:
    ratings.append(i.text)
ratings
```

Out[160]:

```
['116', '113', '104', '101', '101', '95', '88', '87', '68']
```

In [161]:

```
# Now let us make a dataframe
import pandas as pd
df1=pd.DataFrame({'Team':banner_team_name,'Matches':banner_team_matches,'Points':ban
df1
```

Out[161]:

	Team	Matches	Points	Ratings
0	Australia	23	2,714	118

In [162]:

```
# creating dataframe for rest of teams
df2=pd.DataFrame({'Team':rest_teams,'Matches':matches,'Points':points,'Ratings':rati
df2
```

Out[162]:

	Team	Matches	Points	Ratings
0	Pakistan	20	2,316	116
1	India	36	4,081	113
2	New Zealand	27	2,806	104
3	England	24	2,426	101
4	South Africa	19	1,910	101
5	Bangladesh	28	2,661	95
6	Afghanistan	16	1,404	88
7	Sri Lanka	32	2,794	87
8	West Indies	38	2,582	68

In [163]:

```
# Combining both DFs
df=pd.concat([df1,df2],ignore_index=True)
df.index=range(1,11)
print(df)
```

```
            Team Matches Points Ratings
1      Australia      23  2,714     118
2       Pakistan      20  2,316     116
3          India      36  4,081     113
4    New Zealand      27  2,806     104
5        England      24  2,426     101
6   South Africa      19  1,910     101
7     Bangladesh      28  2,661      95
8    Afghanistan      16  1,404      88
9      Sri Lanka      32  2,794      87
10   West Indies      38  2,582      68
```

In [ ]:

```

```

Question 3.b) Top 10 ODI Batsmen along with the records of their team andrating.

In [164]:

```
# requesting
page = requests.get ('https://www.icc-cricket.com/rankings/mens/player-rankings/odi'
page
```

Out[164]:

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

In [165]:

```
# Now when our request got approved as we got response [200]
# Now we will scrap all content from the said page and store in variable soup
soup = BeautifulSoup(page.content)
soup
```

Out[165]:

```
<!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 m
atch cricket players. 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 ODI m
atch cricket players. Discover latest ICC rankings table, predict upcomi
ng matches, see points and ratings for all teams." name="twitter:descrip
tion"/>
<meta content="https://www.icc-cricket.com/resources/ver/i/elements/defa
ult-thumbnail.jpg" name="twitter:image"/>
<meta content="ICC Men's ODI Player Rankings | ICC" property="og:title"/
```

In [166]:

```
# Here also we observed the website is designed in format that banner and rest of the
# we will create 2 DFs one for banner and other for rest of the content
# creating banner df
banner_player_name=[]
for i in soup.find_all('div',class_="rankings-block__banner--name")[0:1]:
    banner_player_name.append(i.text)
banner_player_name
```

Out[166]:

```
['Babar Azam']
```

In [167]:

```
banner_team_rating=[]
for i in soup.find_all('div',class_="rankings-block__banner--nationality")[0:1]:
    banner_team_rating.append(i.text.split())
banner_team_rating
```

Out[167]:

```
[['PAK', '886']]
```

In [168]:

```
banner_team = [item[0]  for item in  banner_team_rating]
banner_rating = [item[1] for item in banner_team_rating]
banner_team
```

Out[168]:

```
['PAK']
```

In [169]:

```
banner_rating
```

Out[169]:

```
['886']
```

In [170]:

```
# now lets create list for players rest of the table
player=[]
for i in soup.find_all('td',class_="table-body__cell name")[0:9]:
    player.append(i.text.strip())
player
```

Out[170]:

```
['Rassie van der Dussen',
 'Fakhar Zaman',
 'Imam-ul-Haq',
 'Shubman Gill',
 'Harry Tector',
 'David Warner',
 'Quinton de Kock',
 'Virat Kohli',
 'Steve Smith']
```

In [171]:

```
# now lets create list for teams rest of the table
team=[]
for i in soup.find_all('span',class_="table-body__logo-text")[0:9]:
    team.append(i.text)
team
```

Out[171]:

```
['SA', 'PAK', 'PAK', 'IND', 'IRE', 'AUS', 'SA', 'IND', 'AUS']
```

In [172]:

```
# now lets create list for ratings rest of the table
rating=[]
for i in soup.find_all('td',class_="table-body__cell u-text-right rating")[0:9]:
    rating.append(i.text)
rating
```

Out[172]:

```
['777', '755', '745', '743', '726', '726', '718', '705', '702']
```

In [173]:

```python
# Now lets create Dataframe for banner and rest of table seperatly and then combine
import pandas as pd
df1=pd.DataFrame({'Player':banner_player_name,'Nationality':banner_team,'Ratings':ba
df1
```

Out[173]:

	Player	Nationality	Ratings
0	Babar Azam	PAK	886

In [174]:

```python
# Similarly creating dataframe for rest of teams
df2=pd.DataFrame({'Player':player,'Nationality':team,'Ratings':rating})
df2
```

Out[174]:

	Player	Nationality	Ratings
0	Rassie van der Dussen	SA	777
1	Fakhar Zaman	PAK	755
2	Imam-ul-Haq	PAK	745
3	Shubman Gill	IND	743
4	Harry Tector	IRE	726
5	David Warner	AUS	726
6	Quinton de Kock	SA	718
7	Virat Kohli	IND	705
8	Steve Smith	AUS	702

In [175]:

```python
# Now Combining both DFs
df=pd.concat([df1,df2],ignore_index=True)
df.index=range(1,11)
print(df)
```

```
                   Player Nationality Ratings
1              Babar Azam         PAK     886
2   Rassie van der Dussen          SA     777
3            Fakhar Zaman         PAK     755
4             Imam-ul-Haq         PAK     745
5            Shubman Gill         IND     743
6            Harry Tector         IRE     726
7            David Warner         AUS     726
8         Quinton de Kock          SA     718
9             Virat Kohli         IND     705
10            Steve Smith         AUS     702
```

In [ ]:

```

```

Question 3. c) Top 10 ODI bowlers along with the records of their team andrating.

In [176]:

```
# requesting
page = requests.get ('https://www.icc-cricket.com/rankings/mens/player-rankings/odi'
page
```

Out[176]:

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

In [177]:

```
# Now when our request got approved as we got response [200]
# Now we will scrap all content from the said page and store in variable soup
soup = BeautifulSoup(page.content)
soup
```

Out[177]:

```
<!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 m
atch cricket players. 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 ODI m
atch cricket players. Discover latest ICC rankings table, predict upcomi
ng matches, see points and ratings for all teams." name="twitter:descrip
tion"/>
<meta content="https://www.icc-cricket.com/resources/ver/i/elements/defa
ult-thumbnail.jpg" name="twitter:image"/>
<meta content="ICC Men's ODI Player Rankings | ICC" property="og:title"/
```

In [178]:

```
# Here also we observed the website is designed in format that banner and rest of the
# we will create 2 DFs one for banner and other for rest of the content
# creating banner df
banner_player_name=[]
for i in soup.find_all('div',class_="rankings-block__banner--name")[1:2]:
    banner_player_name.append(i.text)
banner_player_name
```

Out[178]:

```
['Josh Hazlewood']
```

In [179]:

```
banner_team_rating=[]
for i in soup.find_all('div',class_="rankings-block__banner--nationality")[1:2]:
    banner_team_rating.append(i.text.split())
banner_team_rating
```

Out[179]:

```
[['AUS', '705']]
```

In [180]:

```
banner_team = [item[0]  for item in  banner_team_rating]
banner_rating = [item[1] for item in banner_team_rating]
banner_team
```

Out[180]:

```
['AUS']
```

In [181]:

```
banner_rating
```

Out[181]:

```
['705']
```

In [182]:

```
# now lets create list for bowler from rest of the table
bowler=[]
for i in soup.find_all('td',class_="table-body__cell name")[9:18]:
    bowler.append(i.text.strip())
bowler
```

Out[182]:

```
['Mitchell Starc',
 'Rashid Khan',
 'Mohammed Siraj',
 'Matt Henry',
 'Mujeeb Ur Rahman',
 'Trent Boult',
 'Adam Zampa',
 'Shaheen Afridi',
 'Kuldeep Yadav']
```

In [183]:

```
# now lets create list for teams rest of the table
team=[]
for i in soup.find_all('span',class_="table-body__logo-text")[9:18]:
    team.append(i.text)
team
```

Out[183]:

```
['AUS', 'AFG', 'IND', 'NZ', 'AFG', 'NZ', 'AUS', 'PAK', 'IND']
```

In [184]:

```
# now lets create list for ratings rest of the table
rating=[]
for i in soup.find_all('td',class_="table-body__cell u-text-right rating")[9:18]:
    rating.append(i.text)
rating
```

Out[184]:

```
['686', '682', '670', '667', '661', '660', '652', '630', '622']
```

In [185]:

```
# Now lets create Dataframe for banner and rest of table seperatly and then combine 
import pandas as pd
df1=pd.DataFrame({'Bowler':banner_player_name,'Nationality':banner_team,'Ratings':ba
df1
```

Out[185]:

	Bowler	Nationality	Ratings
0	Josh Hazlewood	AUS	705

In [186]:

```
# Similarly creating dataframe for rest of teams
df2=pd.DataFrame({'Bowler':bowler,'Nationality':team,'Ratings':rating})
df2
```

Out[186]:

	Bowler	Nationality	Ratings
0	Mitchell Starc	AUS	686
1	Rashid Khan	AFG	682
2	Mohammed Siraj	IND	670
3	Matt Henry	NZ	667
4	Mujeeb Ur Rahman	AFG	661
5	Trent Boult	NZ	660
6	Adam Zampa	AUS	652
7	Shaheen Afridi	PAK	630
8	Kuldeep Yadav	IND	622

In [187]:

```python
# Now Combining both DFs
df=pd.concat([df1,df2],ignore_index=True)
df.index=range(1,11)
print(df)
```

```
              Bowler Nationality Ratings
1     Josh Hazlewood         AUS     705
2     Mitchell Starc         AUS     686
3        Rashid Khan         AFG     682
4     Mohammed Siraj         IND     670
5         Matt Henry          NZ     667
6   Mujeeb Ur Rahman         AFG     661
7        Trent Boult          NZ     660
8         Adam Zampa         AUS     652
9     Shaheen Afridi         PAK     630
10     Kuldeep Yadav         IND     622
```

In [ ]:

```python

```

Question 4) 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.

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

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

In [188]:

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

# Now same first requesting website to get permission for scraping
# Question given url is giving wrong result, so taking below
page = requests.get ('https://www.icc-cricket.com/rankings/womens/team-rankings/odi'
page
```

Out[188]:

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

In [189]:

```python
# Now when our request got approved as we got response [200]
# Now we will scrap all content from the said page and store in variable soup
soup = BeautifulSoup(page.content)
soup
```

Out[189]:

```
<!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 upcomin
g 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 upcomin
g matches, see points and ratings for all teams." name="twitter:descript
ion"/>
<meta content="https://www.icc-cricket.com/resources/ver/i/elements/defa
ult-thumbnail.jpg" name="twitter:image"/>
<meta content="ICC Women's ODI Team Rankings | ICC" property="og:title"/
```

In [190]:

```python
# we observed the table in website ( apart from name of team, rest all) designed suc
# extracting data seperatly from banner and rest of the table

```

In [191]:

```python
# Now lets create a  list of name the banner team name .
banner_team_name=[]
for i in soup.find_all('span',class_="u-hide-phablet")[0:1]:
    banner_team_name.append(i.text)
banner_team_name
```

Out[191]:

```
['Australia']
```

In [192]:

```python
# Now lets create a  list of name the banner team matches .
banner_team_matches=[]
for i in soup.find_all('td',class_="rankings-block__banner--matches"):
    banner_team_matches.append(i.text)
banner_team_matches
```

Out[192]:

```
['26']
```

In [193]:

```python
# similarly lets create a  list of banner team points .
banner_team_points=[]
for i in soup.find_all('td',class_="rankings-block__banner--points"):
    banner_team_points.append(i.text)
banner_team_points
```

Out[193]:

```
['4,290']
```

In [194]:

```python
# similarly lets create a  list of banner team rating .
banner_team_rating=[]
for i in soup.find_all('td',class_="rankings-block__banner--rating u-text-right"):
    rating=i.text.strip()
    banner_team_rating.append(rating)
banner_team_rating
```

Out[194]:

```
['165']
```

In [195]:

```python
# Now lets create a  list of name the rest of top 9 teams .
rest_teams=[]
for i in soup.find_all('span',class_="u-hide-phablet")[1:10]:
    rest_teams.append(i.text)
rest_teams
```

Out[195]:

```
['England',
 'South Africa',
 'India',
 'New Zealand',
 'West Indies',
 'Bangladesh',
 'Sri Lanka',
 'Thailand',
 'Pakistan']
```

In [196]:

```
# Now we observed number of matches and points are mentioned in same tag
#lets create a  list  first and then seperate the list of the rest of top 9 teams .
matches_points=[]
for i in soup.find_all('td',class_="table-body__cell u-center-text")[0:18]:
    matches_points.append(i.text)
matches_points
```

Out[196]:

```
['31',
 '3,875',
 '26',
 '3,098',
 '30',
 '3,039',
 '28',
 '2,688',
 '29',
 '2,743',
 '17',
 '1,284',
 '12',
 '820',
 '13',
 '883',
 '27',
 '1,678']
```

In [197]:

```
# Now seperating list in 2 lists number of matches and points
matches = matches_points[::2]  # it will extract number of matches - even indexed el
points = matches_points[1::2]  # it will extract points of team - odd indexed elemen
matches

```

Out[197]:

```
['31', '26', '30', '28', '29', '17', '12', '13', '27']
```

In [198]:

```
points
```

Out[198]:

```
['3,875', '3,098', '3,039', '2,688', '2,743', '1,284', '820', '883', '1,67
8']
```

In [199]:

```
# Now lets create a  list of rating of  rest of top 9 teams .
ratings=[]
for i in soup.find_all('td',class_="table-body__cell u-text-right rating")[0:9]:
    ratings.append(i.text)
ratings
```

Out[199]:

```
['125', '119', '101', '96', '95', '76', '68', '68', '62']
```

In [200]:

```
# Now let us make a dataframe
import pandas as pd
df1=pd.DataFrame({'Team':banner_team_name,'Matches':banner_team_matches,'Points':bann
df1
```

Out[200]:

	Team	Matches	Points	Ratings
0	Australia	26	4,290	165

In [201]:

```
# creating dataframe for rest of teams
df2=pd.DataFrame({'Team':rest_teams,'Matches':matches,'Points':points,'Ratings':rati
df2
```

Out[201]:

	Team	Matches	Points	Ratings
0	England	31	3,875	125
1	South Africa	26	3,098	119
2	India	30	3,039	101
3	New Zealand	28	2,688	96
4	West Indies	29	2,743	95
5	Bangladesh	17	1,284	76
6	Sri Lanka	12	820	68
7	Thailand	13	883	68
8	Pakistan	27	1,678	62

In [202]:

```
# Combining both DFs
df=pd.concat([df1,df2],ignore_index=True)
df.index=range(1,11)
print(df)
```

```
            Team Matches Points Ratings
1      Australia      26  4,290     165
2        England      31  3,875     125
3   South Africa      26  3,098     119
4          India      30  3,039     101
5    New Zealand      28  2,688      96
6    West Indies      29  2,743      95
7     Bangladesh      17  1,284      76
8      Sri Lanka      12    820      68
9       Thailand      13    883      68
10      Pakistan      27  1,678      62
```

In [ ]:

```

```

Question 4.b) Top 10 women's ODI Bating player along with the records of their team and rating.

In [203]:

```python
# Now same first requesting website to get permission for scraping
# Question given url is giving wrong result, so taking below
page = requests.get ('https://www.icc-cricket.com/rankings/womens/player-rankings/od
page
```

Out[203]:

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

In [204]:

```python
# Now when our request got approved as we got response [200]
# Now we will scrap all content from the said page and store in variable soup
soup = BeautifulSoup(page.content)
soup
```

Out[204]:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta content="ICC Women'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 m
atch cricket players. 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 ODI m
atch cricket players. Discover latest ICC rankings table, predict upcomi
ng matches, see points and ratings for all teams." name="twitter:descrip
tion"/>
<meta content="https://www.icc-cricket.com/resources/ver/i/elements/defa
ult-thumbnail.jpg" name="twitter:image"/>
<meta content="ICC Women's ODI Player Rankings | ICC" property="og:titl
```

In [205]:

```python
# Here also we observed the website is designed in format that banner and rest of th
# we will create 2 DFs one for banner and other for rest of the content
# creating banner df
banner_player_name=[]
for i in soup.find_all('div',class_="rankings-block__banner--name")[0:1]:
    banner_player_name.append(i.text)
banner_player_name
```

Out[205]:

```
['Natalie Sciver-Brunt']
```

In [206]:

```
banner_team_rating=[]
for i in soup.find_all('div',class_="rankings-block__banner--nationality")[0:1]:
    banner_team_rating.append(i.text.split())
banner_team_rating
```

Out[206]:

```
[['ENG', '803']]
```

In [207]:

```
banner_team = [item[0]  for item in  banner_team_rating]
banner_rating = [item[1] for item in banner_team_rating]
banner_team
```

Out[207]:

```
['ENG']
```

In [208]:

```
banner_rating
```

Out[208]:

```
['803']
```

In [209]:

```
# now lets create list for players rest of the table
player=[]
for i in soup.find_all('td',class_="table-body__cell name")[0:9]:
    player.append(i.text.strip())
player
```

Out[209]:

```
['Chamari Athapaththu',
 'Beth Mooney',
 'Laura Wolvaardt',
 'Smriti Mandhana',
 'Alyssa Healy',
 'Harmanpreet Kaur',
 'Ellyse Perry',
 'Meg Lanning',
 'Stafanie Taylor']
```

In [210]:

```
# now lets create list for teams rest of the table
team=[]
for i in soup.find_all('span',class_="table-body__logo-text")[0:9]:
    team.append(i.text)
team
```

Out[210]:

```
['SL', 'AUS', 'SA', 'IND', 'AUS', 'IND', 'AUS', 'AUS', 'WI']
```

In [211]:

```
# now lets create list for ratings rest of the table
rating=[]
for i in soup.find_all('td',class_="table-body__cell u-text-right rating")[0:9]:
    rating.append(i.text)
rating
```

Out[211]:

```
['758', '751', '732', '708', '702', '694', '686', '682', '618']
```

In [212]:

```
# Now lets create Dataframe for banner and rest of table seperatly and then combine 
import pandas as pd
df1=pd.DataFrame({'Player':banner_player_name,'Nationality':banner_team,'Ratings':ba
df1
```

Out[212]:

	Player	Nationality	Ratings
0	Natalie Sciver-Brunt	ENG	803

In [213]:

```
# Similarly creating dataframe for rest of teams
df2=pd.DataFrame({'Player':player,'Nationality':team,'Ratings':rating})
df2
```

Out[213]:

	Player	Nationality	Ratings
0	Chamari Athapaththu	SL	758
1	Beth Mooney	AUS	751
2	Laura Wolvaardt	SA	732
3	Smriti Mandhana	IND	708
4	Alyssa Healy	AUS	702
5	Harmanpreet Kaur	IND	694
6	Ellyse Perry	AUS	686
7	Meg Lanning	AUS	682
8	Stafanie Taylor	WI	618

In [214]:

```
# Now Combining both DFs
df=pd.concat([df1,df2],ignore_index=True)
df.index=range(1,11)
print(df)
```

```
                   Player Nationality Ratings
1     Natalie Sciver-Brunt         ENG     803
2      Chamari Athapaththu          SL     758
3              Beth Mooney         AUS     751
4          Laura Wolvaardt          SA     732
5          Smriti Mandhana         IND     708
6             Alyssa Healy         AUS     702
7         Harmanpreet Kaur         IND     694
8             Ellyse Perry         AUS     686
9              Meg Lanning         AUS     682
10         Stafanie Taylor          WI     618
```

In [ ]:

```

```

Question 4 c) Top 10 women’s ODI all-rounder along with the records of their team and rating.

In [215]:

```
# Now same first requesting website to get permission for scraping
# Question given url is giving wrong result, so taking below
page = requests.get ('https://www.icc-cricket.com/rankings/womens/player-rankings/od
page
```

Out[215]:

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

In [216]:

```
# Now when our request got approved as we got response [200]
# Now we will scrap all content from the said page and store in variable soup
soup = BeautifulSoup(page.content)
soup
```

Out[216]:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta content="ICC Women'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 m
atch cricket players. 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 ODI m
atch cricket players. Discover latest ICC rankings table, predict upcomi
ng matches, see points and ratings for all teams." name="twitter:descrip
tion"/>
<meta content="https://www.icc-cricket.com/resources/ver/i/elements/defa
ult-thumbnail.jpg" name="twitter:image"/>
<meta content="ICC Women's ODI Player Rankings | ICC" property="og:titl
```

In [217]:

```
# Here also we observed the website is designed in format that banner and rest of the
# we will create 2 DFs one for banner and other for rest of the content
# creating banner  df
banner_player_name=[]
for i in soup.find_all('div',class_="rankings-block__banner--name")[2:3]:
    banner_player_name.append(i.text)
banner_player_name
```

Out[217]:

```
['Natalie Sciver-Brunt']
```

In [218]:

```
banner_team_rating=[]
for i in soup.find_all('div',class_="rankings-block__banner--nationality")[2:3]:
    banner_team_rating.append(i.text.split())
banner_team_rating
```

Out[218]:

```
[['ENG', '421']]
```

In [219]:

```
banner_team = [item[0]  for item in  banner_team_rating]
banner_rating = [item[1] for item in banner_team_rating]
banner_team
```

Out[219]:

```
['ENG']
```

In [220]:

```python

banner_rating
```

Out[220]:

```
['421']
```

In [221]:

```python

# now lets create list for players rest of the table
player=[]
for i in soup.find_all('td',class_="table-body__cell name")[18:27]:
    player.append(i.text.strip())
player
```

Out[221]:

```
['Ashleigh Gardner',
 'Hayley Matthews',
 'Marizanne Kapp',
 'Ellyse Perry',
 'Amelia Kerr',
 'Deepti Sharma',
 'Jess Jonassen',
 'Sophie Devine',
 'Nida Dar']
```

In [222]:

```python
# now lets create list for teams rest of the table
team=[]
for i in soup.find_all('span',class_="table-body__logo-text")[18:27]:
    team.append(i.text)
team
```

Out[222]:

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

In [223]:

```python
# now lets create list for ratings rest of the table
rating=[]
for i in soup.find_all('td',class_="table-body__cell u-text-right rating")[18:27]:
    rating.append(i.text)
rating
```

Out[223]:

```
['389', '382', '349', '329', '328', '312', '241', '233', '232']
```

In [224]:

```python
# Now lets create Dataframe for banner and rest of table seperatly and then combine 
import pandas as pd
df1=pd.DataFrame({'Player':banner_player_name,'Nationality':banner_team,'Ratings':ba
df1
```

Out[224]:

	Player	Nationality	Ratings
0	Natalie Sciver-Brunt	ENG	421

In [225]:

```python
# Similarly creating dataframe for rest of teams
df2=pd.DataFrame({'Player':player,'Nationality':team,'Ratings':rating})
df2
```

Out[225]:

	Player	Nationality	Ratings
0	Ashleigh Gardner	AUS	389
1	Hayley Matthews	WI	382
2	Marizanne Kapp	SA	349
3	Ellyse Perry	AUS	329
4	Amelia Kerr	NZ	328
5	Deepti Sharma	IND	312
6	Jess Jonassen	AUS	241
7	Sophie Devine	NZ	233
8	Nida Dar	PAK	232

In [226]:

```python
# Now Combining both DFs
df=pd.concat([df1,df2],ignore_index=True)
df.index=range(1,11)
print(df)
```

```
                  Player Nationality Ratings
1   Natalie Sciver-Brunt         ENG     421
2       Ashleigh Gardner         AUS     389
3        Hayley Matthews          WI     382
4         Marizanne Kapp          SA     349
5           Ellyse Perry         AUS     329
6            Amelia Kerr          NZ     328
7          Deepti Sharma         IND     312
8          Jess Jonassen         AUS     241
9          Sophie Devine          NZ     233
10              Nida Dar         PAK     232
```

In [ ]:

```

```

5. `Write a python program to scrape mentioned news details from https://www.cnbc.com/world/?region=world and make data frame-`

i) Headline

ii) Time

iii) News Link

In [227]:

```python
# Now same first requesting website to get permission for scraping

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

Out[227]:

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

In [228]:

```python
# Now when our request got approved as we got response [200]
# Now we will scrap all content from the said page and store in variable soup
soup = BeautifulSoup(page.content)
soup
```

Out[228]:

```
<!DOCTYPE html>
<html itemscope="" itemtype="https://schema.org/WebPage" lang="en" prefi
x="og=https://ogp.me/ns#"><head><meta content="telephone=no" name="forma
t-detection"/><style type="text/css">@charset "UTF-8";.Modal-modalBackgr
ound{background:#000000b3;height:100%;left:0;overflow-y:auto;position:fi
xed;top:0;transition:background-color .4s;width:100%;z-index:100001}.Mod
al-bottomModal.Modal-modal{background:#f8f8f8;border-radius:3px;bottom:
0;box-shadow:5px 5px 20px #1717171a;display:inline-block;height:528px;le
ft:0;margin-top:0!important;max-width:100%;position:fixed;top:auto;trans
form:none;width:100%}@media (max-width:1019px){.Modal-bottomModal.Modal-
modal{height:642px}}@media (max-width:759px){.Modal-bottomModal.Modal-mo
dal{height:100%;position:relative;top:0}}.Modal-modal{background-color:#
fff;border-radius:3px;box-shadow:5px 5px 20px #1717171a;display:inline-b
lock;left:50%;margin-top:10vh;max-width:100%;overflow:auto;position:rela
tive;transform:translateX(-50%)}@media (max-width:759px){.Modal-modal{he
ight:100%;left:auto;margin:0;transform:none;width:100%}}.Modal-modalCont
ents{overflow:auto}@media (max-width:759px){.Modal-modalContents{height:
100%}}.Modal-closeButton{color:#a9a9a9;cursor:pointer;position:absolute;
```

In [229]:

```
headlines =[]

for i in soup.find_all('div',class_="RiverHeadline-headline RiverHeadline-hasThumbna
    headlines.append(i.text)
headlines
```

Out[229]:

```
["China's central bank steps up intervention after yuan hits 16-year low a
gainst greenback ",
 'These stocks pulled back in August but analysts expect them to bounce ba
ck — giving one 103% upside',
 'European markets set to slide as caution lingers around global stocks',
 "What China's big earnings say about the consumer",
 'S&P 500 futures are little changed after major indexes post third straig
ht losing day',
 'CNBC Daily Open: Treasury yields are putting pressure on stocks ',
 "S&P 500, Treasurys or Berkshire Hathaway? Here's where value investor Gu
y Spier would put his money",
 "He paid for the first date. When she didn't want a second, he asked for
his money back",
 "63% of divorcees say this is the No. 1 thing that would have saved their
marriage—it's not more money\xa0",
 'Novartis plans Sandoz spin-off around Oct. 4, proposes share',
 'Zero-day-options, weak technicals and China: Breaking down the market's
August headwinds',
 "China's property troubles aren't getting better, intensifying calls for
bolder policy help",
 "China's economic model is 'washed up on the beach,' says veteran investo
r David Roche",
 "China's growing aggression will be in focus as U.S. closes ranks with Ja
pan and South Korea",
 'Maui emergency chief resigns after defending decision to not activate si
rens during wildfire',
 'Running for the bus or hurrying up the stairs for just 3 minutes a day m
ay lower your risk of cancer',
 'Biden looks to solidify key ties with Japan and South Korea at Camp Davi
d meeting',
 'Gen Z, millennial couples say it's too expensive to get married in this
economy',
 "Top China official urges more secrecy in the country's energy sector",
 "China is considering countermeasures to Biden's executive order"]
```

In [230]:

```
time =[]

for i in soup.find_all('span',class_="RiverByline-datePublished"):
    time.append(i.text)
time
```

Out[230]:

```
['an hour ago',
 'an hour ago',
 '18 min ago',
 '18 min ago',
 'an hour ago',
 'an hour ago',
 '34 min ago',
 '34 min ago',
 '3 hours ago',
 '3 hours ago',
 'an hour ago',
 'an hour ago',
 '5 hours ago',
 '5 hours ago',
 '4 hours ago',
 '4 hours ago']
```

In [231]:

```
URL =[]

for i in soup.find_all('a',href="https://www.cnbc.com/2023/08/18/us-seeks-to-bring-ja
    URL.append(i.get('href'))
URL
```

Out[231]:

```
['https://www.cnbc.com/2023/08/18/us-seeks-to-bring-japan-and-south-korea-
closer-with-eye-on-china.html',
 'https://www.cnbc.com/2023/08/18/us-seeks-to-bring-japan-and-south-korea-
closer-with-eye-on-china.html',
 'https://www.cnbc.com/2023/08/18/us-seeks-to-bring-japan-and-south-korea-
closer-with-eye-on-china.html']
```

In [232]:

```
class="RiverByline-datePublished"
```

```
  Cell In[232], line 1
    class="RiverByline-datePublished"
         ^
SyntaxError: invalid syntax
```

In [ ]:

```

```

In [ ]:

```

```

In [233]:

```
href="https://www.cnbc.com/2023/08/18/us-seeks-to-bring-japan-and-south-korea-closer
```

In [ ]:

```

```

6. 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 Scrape below mentioned details and make data frame-

i) Paper Title ii) Authors iii) Published Date iv) Paper URL

In [234]:

```
# Now same first requesting website to get permission for scraping
page = requests.get ('https://www.journals.elsevier.com/artificial-intelligence/most
page
```

Out[234]:

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

In [235]:

```
# Now when our request got approved as we got response [200]
# Now we will scrap all content from the said page and store in variable soup
soup = BeautifulSoup(page.content)
soup
```

Out[235]:

```
<!DOCTYPE html>
<html><head><meta charset="utf-8"/><meta content="width=device-width" na
me="viewport"/><meta content="en_US" name="og:locale"/><meta content="Mo
st Downloaded Articles - Artificial Intelligence - Journal - Elsevier" p
roperty="og:title"/><meta content="The journal of Artificial Intelligenc
e (AIJ)  welcomes papers on broad aspects of AI that constitute advances
in the overall field including, but not limited …" property="og:descript
ion"/><meta content="http://ars.els-cdn.com/content/image/X00043702.jpg"
name="og:image" property="og:image"/><meta content="http://ars.els-cdn.c
om/content/image/X00043702.jpg" name="og:image:url" property="og:image:u
rl"/><meta content="https://ars.els-cdn.com/content/image/X00043702.jpg"
name="og:image:secure_url" property="og:image:secure_url"/><meta content
="journals.elsevier.com/artificial-intelligence/most-downloaded-article
s" name="og:url"/><meta content="website" property="og:type"/><link href
="/apple-touch-icon.png" rel="apple-touch-icon" sizes="180x180"/><link h
ref="/favicon-32x32.png" rel="icon" sizes="32x32" type="image/png"/><lin
k href="/favicon-16x16.png" rel="icon" sizes="16x16" type="image/png"/><
link color="#ff6c00" href="/safari-pinned-tab.svg" rel="mask-icon"/><tit
```

In [236]:

```
# Now finding all papertitles

paper_titles= []
for i in soup.find_all ('h2',class_="sc-1qrq3sd-1 gRGSUS sc-1nmom32-0 sc-1nmom32-1 bt
    paper_titles.append(i.text)
paper_titles
```

Out[236]:

```
['Reward is enough',
 'Explanation in artificial intelligence: Insights from the social scien
ces',
 'Creativity and artificial intelligence',
 'Conflict-based search for optimal multi-agent pathfinding',
 'Knowledge graphs as tools for explainable machine learning: A survey',
 'Law and logic: A review from an argumentation perspective',
 'Between MDPs and semi-MDPs: A framework for temporal abstraction in re
inforcement learning',
 'Explaining individual predictions when features are dependent: More ac
curate approximations to Shapley values',
 'Multiple object tracking: A literature review',
 'A survey of inverse reinforcement learning: Challenges, methods and pr
ogress',
 'Evaluating XAI: A comparison of rule-based and example-based explanati
ons',
 'Explainable AI tools for legal reasoning about cases: A study on the E
uropean Court of Human Rights',
```

In [237]:

```
# Now finding all authers

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

Out[237]:

```
['David Silver, Satinder Singh, Doina Precup, Richard S. Sutton ',
 'Tim Miller ',
 'Margaret A. Boden ',
 'Guni Sharon, Roni Stern, Ariel Felner, Nathan R. Sturtevant ',
 'Ilaria Tiddi, Stefan Schlobach ',
 'Henry Prakken, Giovanni Sartor ',
 'Richard S. Sutton, Doina Precup, Satinder Singh ',
 'Kjersti Aas, Martin Jullum, Anders Løland ',
 'Wenhan Luo, Junliang Xing and 4 more',
 'Saurabh Arora, Prashant Doshi ',
 'Jasper van der Waa, Elisabeth Nieuwburg, Anita Cremers, Mark Neerincx
',
 'Joe Collenette, Katie Atkinson, Trevor Bench-Capon ',
 'Roel Dobbe, Thomas Krendl Gilbert, Yonatan Mintz ',
 'Oskar Wysocki, Jessica Katharine Davies and 5 more',
 'Eoin M. Kenny, Courtney Ford, Molly Quinn, Mark T. Keane ',
 'Nolan Bard, Jakob N. Foerster and 13 more',
 'Ron Kohavi, George H. John ',
```

In [238]:

```
# Now finding all published_date

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

Out[238]:

```
['October 2021',
 'February 2019',
 'August 1998',
 'February 2015',
 'January 2022',
 'October 2015',
 'August 1999',
 'September 2021',
 'April 2021',
 'August 2021',
 'February 2021',
 'April 2023',
 'November 2021',
 'March 2023',
 'May 2021',
 'March 2020',
 'December 1997',
 'June 2017',
```

In [239]:

```
# Now lets creat dataframe
df=pd.DataFrame({'PaperTitle':paper_titles,'Auther':auther,'PublishedDate':date})
df
```

Out[239]:

	PaperTitle	Auther	PublishedDate
0	Reward is enough	David Silver, Satinder Singh, Doina Precup, Ri...	October 2021
1	Explanation in artificial intelligence: Insigh...	Tim Miller	February 2019
2	Creativity and artificial intelligence	Margaret A. Boden	August 1998
3	Conflict-based search for optimal multi-agent ...	Guni Sharon, Roni Stern, Ariel Felner, Nathan ...	February 2015
4	Knowledge graphs as tools for explainable mach...	Ilaria Tiddi, Stefan Schlobach	January 2022
5	Law and logic: A review from an argumentation ...	Henry Prakken, Giovanni Sartor	October 2015
6	Between MDPs and semi-MDPs: A framework for te...	Richard S. Sutton, Doina Precup, Satinder Singh	August 1999
7	Explaining individual predictions when feature...	Kjersti Aas, Martin Jullum, Anders Løland	September 2021
8	Multiple object tracking: A literature review	Wenhan Luo, Junliang Xing and 4 more	April 2021
9	A survey of inverse reinforcement learning: Ch...	Saurabh Arora, Prashant Doshi	August 2021
10	Evaluating XAI: A comparison of rule-based and...	Jasper van der Waa, Elisabeth Nieuwburg, Anita...	February 2021
11	Explainable AI tools for legal reasoning about...	Joe Collenette, Katie Atkinson, Trevor Bench-C...	April 2023
12	Hard choices in artificial intelligence	Roel Dobbe, Thomas Krendl Gilbert, Yonatan Mintz	November 2021
13	Assessing the communication gap between AI mod...	Oskar Wysocki, Jessica Katharine Davies and 5 ...	March 2023
14	Explaining black-box classifiers using post-ho...	Eoin M. Kenny, Courtney Ford, Molly Quinn, Mar...	May 2021
15	The Hanabi challenge: A new frontier for AI re...	Nolan Bard, Jakob N. Foerster and 13 more	March 2020
16	Wrappers for feature subset selection	Ron Kohavi, George H. John	December 1997
17	Artificial cognition for social human–robot in...	Séverin Lemaignan, Mathieu Warnier and 3 more	June 2017
18	A review of possible effects of cognitive bias...	Tomáš Kliegr, Štěpán Bahník, Johannes Fürnkranz	June 2021
19	The multifaceted impact of Ada Lovelace in the...	Luigia Carlucci Aiello	June 2016
20	Robot ethics: Mapping the issues for a mechani...	Patrick Lin, Keith Abney, George Bekey	April 2011
21	Reward (Mis)design for autonomous driving	W. Bradley Knox, Alessandro Allievi and 3 more	March 2023
22	Planning and acting in partially observable st...	Leslie Pack Kaelbling, Michael L. Littman, Ant...	May 1998
23	What do we want from Explainable Artificial In...	Markus Langer, Daniel Oster and 6 more	July 2021

In [ ]:

```

```

```
7)  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 URL

```

In [240]:

```python
# Now same first requesting website to get permission for scraping
page = requests.get ('https://www.dineout.co.in/mumbai-restaurants/welcome-back')
page
```

Out[240]:

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

In [241]:

```python
# Now when our request got approved as we got response [200]
# Now we will scrap all content from the said page and store in variable soup
soup = BeautifulSoup(page.content)
soup
```

Out[241]:

```
<!DOCTYPE html>
<html lang="en"><head><meta charset="utf-8"/><meta content="IE=edge" htt
p-equiv="X-UA-Compatible"/><meta content="width=device-width, initial-sc
ale=1.0, maximum-scale=1.0, user-scalable=no" name="viewport"/><link hre
f="/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 [242]:

```
# list we have to make are
name=[]

cuisine=[]

Location=[]

Ratings=[]
```

In [243]:

```
# Now finding name of all restra

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

Out[243]:

```
['Ventura Building,Powai, Powai',
 'Hiranandani, Powai',
 'Delphi Building,Powai, Powai',
 'Powai, Powai',
 'R City Mall,Ghatkopar West, Central Suburbs',
 'Kailash Business Park,Vikhroli West, Central Suburbs',
 'Kailash Business Park,Vikhroli West, Central Suburbs',
 'R City Mall,Ghatkopar West, Central Suburbs',
 'Powai, Powai',
 'Vikhroli West, Central Suburbs',
 'Heera Panna Shopping Centre,Powai, Powai',
 'R City Mall,Ghatkopar West, Central Suburbs',
 'R City Mall,Ghatkopar West, Central Suburbs',
 'R City Mall,Ghatkopar West, Central Suburbs',
 'Ventura Building,Powai, Powai',
 'R City Mall,Ghatkopar West, Central Suburbs',
 'City Park Building,Powai, Powai',
 'R City Mall,Ghatkopar West, Central Suburbs',
 'R City Mall,Ghatkopar West, Central Suburbs',
 'R City Mall,Ghatkopar West, Central Suburbs',
 'Delphi Building,Powai, Powai']
```

In [244]:

```python
# Now finding Location

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

Out[244]:

```
['Ventura Building,Powai, Powai',
 'Hiranandani, Powai',
 'Delphi Building,Powai, Powai',
 'Powai, Powai',
 'R City Mall,Ghatkopar West, Central Suburbs',
 'Kailash Business Park,Vikhroli West, Central Suburbs',
 'Kailash Business Park,Vikhroli West, Central Suburbs',
 'R City Mall,Ghatkopar West, Central Suburbs',
 'Powai, Powai',
 'Vikhroli West, Central Suburbs',
 'Heera Panna Shopping Centre,Powai, Powai',
 'R City Mall,Ghatkopar West, Central Suburbs',
 'R City Mall,Ghatkopar West, Central Suburbs',
 'R City Mall,Ghatkopar West, Central Suburbs',
 'Ventura Building,Powai, Powai',
 'R City Mall,Ghatkopar West, Central Suburbs',
 'City Park Building,Powai, Powai',
 'R City Mall,Ghatkopar West, Central Suburbs',
 'R City Mall,Ghatkopar West, Central Suburbs',
 'R City Mall,Ghatkopar West, Central Suburbs',
 'Delphi Building,Powai, Powai']
```

In [245]:

```python
# Now finding ratings

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

Out[245]:

```
['4.3',
 '4.4',
 '4.4',
 '4.3',
 '4.4',
 '4.2',
 '4.3',
 '4.4',
 '4.2',
 '4.3',
 '4.4',
 '4.2',
 '4.1',
 '4.1',
 '4',
 '4.3']
```

In [246]:

```
# cui

cui= []
for i in soup.find_all ('span',class_="double-line-ellipsis"):
    cui.append(i.text)
cui
```

Out[246]:

```
['₹ 1,500 for 2 (approx) | North Indian, Biryani',
 '₹ 1,200 for 2 (approx) | Chinese, Asian, Thai',
 '₹ 2,100 for 2 (approx) | Asian, Chinese, Japanese',
 '₹ 800 for 2 (approx) | Chinese, North Indian, Biryani',
 '₹ 1,500 for 2 (approx) | North Indian, Mughlai',
 '₹ 800 for 2 (approx) | South Indian, North Indian',
 '₹ 700 for 2 (approx) | Pizza, Fast Food, American, Italian',
 '₹ 600 for 2 (approx) | South Indian',
 '₹ 2,000 for 2 (approx) | Chinese, European, Asian, North Indian',
 '₹ 300 for 2 (approx) | Street Food, Tea',
 '₹ 700 for 2 (approx) | Biryani, North Indian, Fast Food',
 '₹ 400 for 2 (approx) | Desserts, Beverages',
 '₹ 1,200 for 2 (approx) | Gujarati, Rajasthani, North Indian',
 '₹ 1,200 for 2 (approx) | Continental, Italian, Beverages',
 '₹ 1,600 for 2 (approx) | Italian, Pizza, Beverages',
 '₹ 500 for 2 (approx) | Desserts, Beverages',
 '₹ 500 for 2 (approx) | Desserts, Beverages',
 '₹ 700 for 2 (approx) | Pizza, Fast Food, American, Italian',
 '₹ 700 for 2 (approx) | Continental, Health Food',
 '₹ 500 for 2 (approx) | Tex Mex',
 '₹ 2,000 for 2 (approx) | Continental, European']
```

In [248]:

```python
# Now lets creat dataframe
df=pd.DataFrame({'RestaurantName':name,'Cuisine':cui,'Location':loc,'Rating':rat })
df
```

```
---------------------------------------------------------------------------
-
ValueError                                Traceback (most recent call las
t)
Cell In[248], line 2
      1 # Now lets creat dataframe
----> 2 df=pd.DataFrame({'RestaurantName':name,'Cuisine':cui,'Location':lo
c,'Rating':rat })
      3 df

File C:\ProgramData\anaconda3\lib\site-packages\pandas\core\frame.py:664,
in DataFrame.__init__(self, data, index, columns, dtype, copy)
    658     mgr = self._init_mgr(
    659         data, axes={"index": index, "columns": columns}, dtype=dty
pe, copy=copy
    660     )
    662 elif isinstance(data, dict):
    663     # GH#38939 de facto copy defaults to False only in non-dict ca
ses
--> 664     mgr = dict_to_mgr(data, index, columns, dtype=dtype, copy=cop
y, typ=manager)
    665 elif isinstance(data, ma.MaskedArray):
    666     import numpy.ma.mrecords as mrecords

File C:\ProgramData\anaconda3\lib\site-packages\pandas\core\internals\cons
truction.py:493, in dict_to_mgr(data, index, columns, dtype, typ, copy)
    489     else:
    490         # dtype check to exclude e.g. range objects, scalars
    491         arrays = [x.copy() if hasattr(x, "dtype") else x for x in
arrays]
--> 493 return arrays_to_mgr(arrays, columns, index, dtype=dtype, typ=typ,
consolidate=copy)

File C:\ProgramData\anaconda3\lib\site-packages\pandas\core\internals\cons
truction.py:118, in arrays_to_mgr(arrays, columns, index, dtype, verify_in
tegrity, typ, consolidate)
    115 if verify_integrity:
    116     # figure out the index, if necessary
    117     if index is None:
--> 118         index = _extract_index(arrays)
    119     else:
    120         index = ensure_index(index)

File C:\ProgramData\anaconda3\lib\site-packages\pandas\core\internals\cons
truction.py:666, in _extract_index(data)
    664 lengths = list(set(raw_lengths))
    665 if len(lengths) > 1:
--> 666     raise ValueError("All arrays must be of the same length")
    668 if have_dicts:
    669     raise ValueError(
    670         "Mixing dicts with non-Series may lead to ambiguous orderi
ng."
    671     )

ValueError: All arrays must be of the same length
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: