In [47]:

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

In [48]:

```python
page=requests.get('https://coreyms.com/')
page
```

Out[48]: <Response [200]>

In [49]:

```python
soup=BeautifulSoup(page.content)
#soup
```

In [50]:

```python
heading=[]

for i in soup.find_all('h2',class_='entry-title'):
    heading.append(i.text)

heading=heading[0:9]
heading
```

Out[50]:

```
['Python Tutorial: Zip Files – Creating and Extracting Zip Archives',
 'Python Data Science Tutorial: Analyzing the 2019 Stack Overflow Developer Survey',
 'Python Multiprocessing Tutorial: Run Code in Parallel Using the Multiprocessing Module',
 'Python Threading Tutorial: Run Code Concurrently Using the Threading Module',
 'Update (2019-09-03)',
 'Python Quick Tip: The Difference Between “==” and “is” (Equality vs Identity)',
 'Python Tutorial: Calling External Commands Using the Subprocess Module',
 'Visual Studio Code (Windows) – Setting up a Python Development Environment and Complete
Overview',
 'Visual Studio Code (Mac) – Setting up a Python Development Environment and Complete Over
view']
```

In [51]:

```python
date=[]

for i in soup.find_all('time',class_='entry-time'):
    date.append(i.text)

date= date[0:9]
date
```

Out[51]:

```
['November 19, 2019',
 'October 17, 2019',
 'September 21, 2019',
 'September 12, 2019',
 'September 3, 2019',
 'August 6, 2019',
 'July 24, 2019',
 'May 1, 2019',
 'May 1, 2019']
```

In [52]:

```python
content=[]

for i in soup.find_all('div',class_='entry-content'):
    content.append(i.text)

content= content[0:9]
content
```

Loading [MathJax]/extensions/Safe.js deo, we will be learning how to create and extract zip archives. We will sta

rt by using the zipfile module, and then we will see how to do this using the shutil module. We will learn how to do this with single files and directories, as well as learning how to use gzip as well. Let’s get started…\n\n',
 '\nIn this Python Programming video, we will be learning how to download and analyze real-world data from the 2019 Stack Overflow Developer Survey. This is terrific practice for anyone getting into the data science field. We will learn different ways to analyze this data and also some best practices. Let’s get started…\n\n\n\n\n',
 '\nIn this Python Programming video, we will be learning how to run code in parallel using the multiprocessing module. We will also look at how to process multiple high-resolution images at the same time using a ProcessPoolExecutor from the concurrent.futures module. Let’s get started…\n\n\n\n\n',
 '\nIn this Python Programming video, we will be learning how to run threads concurrently using the threading module. We will also look at how to download multiple high-resolution images online using a ThreadPoolExecutor from the concurrent.futures module. Let’s get started…\n\n\n\n\n',
 '\nHey everyone. I wanted to give you an update on my videos. I will be releasing videos on threading and multiprocessing within the next week. Thanks so much for your patience. I currently have a temporary recording studio setup at my Airbnb that will allow me to record and edit the threading/multiprocessing videos. I am going to be moving into my new house in 10 days and once I have my recording studio setup then you can expect much faster video releases. I really appreciate how patient everyone has been while I go through this move, especially those of you who are contributing monthly through YouTube \n',
 '\nIn this Python Programming Tutorial, we will be learning the difference between using “==” and the “is” keyword when doing comparisons. The difference between these is that “==” checks to see if values are equal, and the “is” keyword checks their identity, which means it’s going to check if the values are identical in terms of being the same object in memory. We’ll learn more in the video. Let’s get started…\n\n\n\n\n',
 '\nIn this Python Programming Tutorial, we will be learning how to run external commands using the subprocess module from the standard library. We will learn how to run commands, capture the output, handle errors, and also how to pipe output into other commands. Let’s get started…\n\n\n\n\n',
 '\nIn this Python Programming Tutorial, we will be learning how to set up a Python development environment in VSCode on Windows. VSCode is a very nice free editor for writing Python applications and many developers are now switching over to this editor. In this video, we will learn how to install VSCode, get the Python extension installed, how to change Python interpreters, create virtual environments, format/lint our code, how to use Git within VSCode, how to debug our programs, how unit testing works, and more. We have a lot to cover, so let’s go ahead and get started…\nVSCode on MacOS – https://youtu.be/06I63_p-2A4\nTimestamps for topics in this tutorial: Installation – 1:13 Python Extension – 5:48 Switching Interpreters – 10:04 Changing Color Themes – 12:35 VSCode Settings – 16:16 Set Default Python – 21:33 Using Virtual Environments – 25:10 IntelliSense – 29:45 Code Formatting – 32:13 Code Linting – 37:06 Code Runner Extension – 39:42 Git Integration – 47:44 Use Different Terminal – 51:07 Debugging – 58:45 Unit Testing – 1:03:25 Zen Mode – 1:09:55\n\n\n\n',
 '\nIn this Python Programming Tutorial, we will be learning how to set up a Python development environment in VSCode on MacOS. VSCode is a very nice free editor for writing Python applications and many developers are now switching over to this editor. In this video, we will learn how to install VSCode, get the Python extension installed, how to change Python interpreters, create virtual environments, format/lint our code, how to use Git within VSCode, how to debug our programs, how unit testing works, and more. We have a lot to cover, so let’s go ahead and get started…\nVSCode on Windows – https://youtu.be/-nh9rCzPJ20\nTimestamps for topics in this tutorial: Installation – 1:11 Python Extension – 6:21 Switching Interpreters – 10:16 Changing Color Themes – 13:08 VSCode Settings – 17:12 Set Default Python – 22:24 Using Virtual Environments – 25:52 IntelliSense – 30:28 Code Formatting – 33:08 Code Linting – 38:01 Code Runner Extension – 40:45 Git Integration – 49:05 Debugging – 58:15 Unit Testing – 1:02:38 Zen Mode – 1:10:42 \n\n\n\n']

In [53]:
```
videolink=[]

for i in soup.find_all('iframe',class_='youtube-player'):
    videolink.append(i['src'])

videolink
```

Out[53]:
```
['https://www.youtube.com/embed/z0gguhEmWiY?version=3&rel=1&showsearch=0&showinfo=1&iv_load_policy=1&fs=1&hl=en-US&autohide=2&wmode=transparent',
 'https://www.youtube.com/embed/_P7X8tMplsw?version=3&rel=1&showsearch=0&showinfo=1&iv_load_policy=1&fs=1&hl=en-US&autohide=2&wmode=transparent',
 'https://www.youtube.com/embed/fKl2JW_qrso?version=3&rel=1&showsearch=0&showinfo=1&iv_loa
```

```
d_policy=1&fs=1&hl=en-US&autohide=2&wmode=transparent',
 'https://www.youtube.com/embed/IEEhzQoKtQU?version=3&rel=1&showsearch=0&showinfo=1&iv_loa
d_policy=1&fs=1&hl=en-US&autohide=2&wmode=transparent',
 'https://www.youtube.com/embed/mO_dS3rXDIs?version=3&rel=1&showsearch=0&showinfo=1&iv_loa
d_policy=1&fs=1&hl=en-US&autohide=2&wmode=transparent',
 'https://www.youtube.com/embed/2Fp1N6dof0Y?version=3&rel=1&showsearch=0&showinfo=1&iv_loa
d_policy=1&fs=1&hl=en-US&autohide=2&wmode=transparent',
 'https://www.youtube.com/embed/-nh9rCzPJ20?version=3&rel=1&showsearch=0&showinfo=1&iv_loa
d_policy=1&fs=1&hl=en-US&autohide=2&wmode=transparent',
 'https://www.youtube.com/embed/06I63_p-2A4?version=3&rel=1&showsearch=0&showinfo=1&iv_loa
d_policy=1&fs=1&hl=en-US&autohide=2&wmode=transparent',
 'https://www.youtube.com/embed/_JGmemuINww?version=3&rel=1&showsearch=0&showinfo=1&iv_loa
d_policy=1&fs=1&hl=en-US&autohide=2&wmode=transparent']
```

In [63]:
```
#
```

In [62]:
```
#
```

In [ ]:
```
print(len(heading), len(date), len(content),len( videolink))
```

In [21]:
```
import pandas as pd
```

In [22]:
```
df=pd.DataFrame({'Heading': heading, 'Date': date, 'Content': content, 'VideoLink': videol
df
```

Out[22]:

	Heading	Date	Content	VideoLink
0	Python Tutorial: Zip Files – Creating and Extr...	November 19, 2019	\nIn this video, we will be learning how to cr...	https://www.youtube.com/embed/z0gguhEmWiY?vers...
1	Python Data Science Tutorial: Analyzing the 20...	October 17, 2019	\nIn this Python Programming video, we will be...	https://www.youtube.com/embed/_P7X8tMplsw?vers...
2	Python Multiprocessing Tutorial: Run Code in P...	September 21, 2019	\nIn this Python Programming video, we will be...	https://www.youtube.com/embed/fKl2JW_qrso?vers...
3	Python Threading Tutorial: Run Code Concurrent...	September 12, 2019	\nIn this Python Programming video, we will be...	https://www.youtube.com/embed/IEEhzQoKtQU?vers...
4	Update (2019-09-03)	September 3, 2019	\nHey everyone. I wanted to give you an update...	https://www.youtube.com/embed/mO_dS3rXDIs?vers...
5	Python Quick Tip: The Difference Between “==” ...	August 6, 2019	\nIn this Python Programming Tutorial, we will...	https://www.youtube.com/embed/2Fp1N6dof0Y?vers...
6	Python Tutorial: Calling External Commands Usi...	July 24, 2019	\nIn this Python Programming Tutorial, we will...	https://www.youtube.com/embed/-nh9rCzPJ20?vers...
7	Visual Studio Code (Windows) – Setting up a Py...	May 1, 2019	\nIn this Python Programming Tutorial, we will...	https://www.youtube.com/embed/06I63_p-2A4?vers...
8	Visual Studio Code (Mac) – Setting up a Python...	May 1, 2019	\nIn this Python Programming Tutorial, we will...	https://www.youtube.com/embed/_JGmemuINww?vers...

In [ ]:
```

```

Program to display products from Bewakoof

Import Libraries

In [1]:
```
from bs4 import BeautifulSoup
import requests
```

Send Get request from Web page

In [2]:
```
url= requests.get('https://www.bewakoof.com/women-t-shirts')
url
```

Out[2]:
```
<Response [200]>
```

Page Content

In [3]:
```
request = url.text
```

Scrapping header

In [ ]:
```
soup_data= BeautifulSoup(request, 'html.parser')
soup_data
```

In [5]:
```
soup_data.title.text
```

Out[5]:
```
'Buy Funky T-Shirts for Women Online India at Bewakoof'
```

In [6]:
```
#
```

In [7]:
```
#first_tshirt=tshirts
product=[]

for i in soup_data.find_all('div',class_="productCardDetail"):
    product.append(i)#.text.split('₹'))

#product
```

In [8]:
```
#Name of the product
prd1= product[0]
prd1.div.h3.text
```

Out[8]:
```
"Women's Red Busy Doing Nothing  Plus Size Boyfriend T-shirt"
```

In [9]:

```
#prd1.find('span',{'class':"discountedPriceText"}).text
prd1.span.text.replace('₹','')
```

Out[9]:
```
' 499'
```

Image

In [10]:
```
#URL
img=[]

for i in soup_data.find_all('img',class_="productImgTag"):
    img.append(i['src'])

img
```

Out[10]:
```
['https://images.bewakoof.com/t320/busy-doing-nothing-2-0-480110-1648529612-1.jpg',
 'https://images.bewakoof.com/t320/jeelo-merlot-oversize-fit-t-shirt-485544-1648738864-1.jpg',
 'https://images.bewakoof.com/t320/women-s-yellow-get-over-it-relaxed-fit-short-top-483259-1648732978-1.jpg',
 'https://images.bewakoof.com/t320/women-s-yellow-garfield-oddie-boyfriend-t-shirt-483257-1648732931-1.jpg',
 'https://images.bewakoof.com/t320/difference-of-opinion-women-s-green-colourblock-oversized-fit-t-shirt-457881-1639754729-1.jpg',
 'https://images.bewakoof.com/t320/dillinger-black-typography-t-shirt-425674-1633605968-1.jpg',
 'https://images.bewakoof.com/t320/dillinger-blue-typography-t-shirt-425763-1633603449-1.jpg',
 'https://images.bewakoof.com/t320/dillinger-black-typography-t-shirt-425306-1633603421-1.jpg',
 'https://images.bewakoof.com/t320/dillinger-women-yellow-floral-print-oversized-t-shirt-450220-1637331836-1.jpg',
 'https://images.bewakoof.com/t320/dillinger-women-s-pink-typographic-oversized-fit-t-shirt-456583-1639550340-1.jpg',
 'https://images.bewakoof.com/t320/orchid-petal-tank-top-454616-1640787382-1.jpg']
```

Looping

In [11]:
```
Prdname=[]
Price=[]

for i in product:
    Prdname.append(i.div.h3.text)
    Price.append(i.span.text.replace('₹',''))
```

In [12]:
```
data= list(zip(Prdname,Price,img))
```

In [13]:
```
import pandas as pd
```

In [14]:
```
df=pd.DataFrame(data, columns=['Prdname','Price','img'])
df.head(10)
```

Out[14]:

	Prdname	Price	img
0	Women's Red Busy Doing Nothing Plus Size Boyf...	499	https://images.bewakoof.com/t320/busy-doing-no...

	Prdname	Price	img
1	Women's Thoda Jeelo Thoda Merlot Printed Super...	499	https://images.bewakoof.com/t320/jeelo-merlot-...
2	Women's Yellow Get Over It Relaxed Fit Short Top	399	https://images.bewakoof.com/t320/women-s-yello...
3	Women's Yellow Garfield Oddie Boyfriend T-shirt	399	https://images.bewakoof.com/t320/women-s-yello...
4	Women's Green Color Block Oversized Fit T-shirt	558	https://images.bewakoof.com/t320/difference-of...
5	Women's Black Typography T-shirt	575	https://images.bewakoof.com/t320/dillinger-bla...
6	Women's Blue Typography T-shirt	527	https://images.bewakoof.com/t320/dillinger-blu...
7	Women's Green Striped T-shirt	527	https://images.bewakoof.com/t320/dillinger-bla...
8	Women's Yellow Floral Print Oversized T-shirt	527	https://images.bewakoof.com/t320/dillinger-wom...
9	Women's Pink Typographic Oversized Fit T Shirt	531	https://images.bewakoof.com/t320/dillinger-wom...

In [ ]:

First Install all Libraries

First we will import all libraries required for web scraping. Two libraries are required: Request BeautifulSoup

In [1]:
```
#!pip install bs4
#!pip install requests
```

Import Libraries

In [2]:
```
from bs4 import BeautifulSoup
import requests
```

Send get request to the webpage server to get the source code of the page

In [3]:
```
page= requests.get('https://www.dineout.co.in/delhi-restaurants/buffet-special')
page
```

Out[3]:
```
<Response [200]>
```

Page Content

In [ ]:
```
soup= BeautifulSoup(page.content)
soup
```

Scraping First name

In [5]:
```
#First we will use html tag where we have the first title of the restuarants

first_title= soup.find('div', class_="restnt-info cursor")
first_title
```

Out[5]:
```
<div class="restnt-info cursor" data-gatype="RestaurantNameClick"><a analytics-action="Res
taurantCardClick" analytics-label="86792_Castle Barbeque" class="restnt-name ellipsis" dat
a-w-onclick="sendAnalyticsCommon|w1-restarant" href="/delhi/castle-barbeque-connaught-plac
e-central-delhi-86792">Castle Barbeque</a><div class="restnt-loc ellipsis" data-w-onclick
="stopClickPropagation|w1-restarant"><a data-name="Connaught Place" data-type="LocalityCli
ck" href="/delhi-restaurants/central-delhi/connaught-place">Connaught Place</a>, <a data-n
ame="Central Delhi" data-type="AreaClick" href="/delhi-restaurants/central-delhi">Central
Delhi</a></div></div>
```

In [6]:
```
first_title.text
```

Out[6]:
```
'Castle BarbequeConnaught Place, Central Delhi'
```

Scraping the First Location)

In [7]:
```
loc= soup.find('div', class_="restnt-loc ellipsis")
loc.text
```

Out[7]:
```
'Connaught Place, Central Delhi'
```

Scraping First Price

In [8]:
```
price= soup.find('span', class_="double-line-ellipsis")
price.text.split()[1]
```

Out[8]:
```
'2,000'
```

Scraping Multiple Titles

In [9]:
```
# Now we have all the tags in which there are job titles

# Now we will extract the text from these tags one by one by looping over these tags

titles=[]# empty list to store the data

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

titles
```

Out[9]:
```
['Castle BarbequeConnaught Place, Central Delhi',
 'Jungle Jamboree3CS Mall,Lajpat Nagar - 3, South Delhi',
 'Castle BarbequePacific Mall,Tagore Garden, West Delhi',
 'Cafe KnoshThe Leela Ambience Convention Hotel,Shahdara, East Delhi',
 'The Barbeque CompanyGardens Galleria,Sector 38A, Noida',
 'India GrillHilton Garden Inn,Saket, South Delhi',
 'Delhi BarbequeTaurus Sarovar Portico,Mahipalpur, South Delhi',
 'The Monarch - Bar Be Que VillageIndirapuram Habitat Centre,Indirapuram, Ghaziabad',
 'World CafeVibe by The Lalit Traveller,Sector 35, Faridabad',
 'Indian Grill RoomSuncity Business Tower,Golf Course Road, Gurgaon',
 'Mad 4 Bar B QueSector 29, Faridabad',
 'Barbeque 29NIT, Faridabad',
 'GlasshouseDoubleTree By Hilton Gurugram Baani Square,Sector 50, Gurgaon']
```

Scraping Multiple Location

In [10]:
```
loc=[]

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

loc
```

Out[10]:
```
['Connaught Place, Central Delhi',
 '3CS Mall,Lajpat Nagar - 3, South Delhi',
 'Pacific Mall,Tagore Garden, West Delhi',
 'The Leela Ambience Convention Hotel,Shahdara, East Delhi',
 'Gardens Galleria,Sector 38A, Noida',
 'Hilton Garden Inn,Saket, South Delhi',
 'Taurus Sarovar Portico,Mahipalpur, South Delhi',
 Habitat Centre,Indirapuram, Ghaziabad',
```

```
 'Vibe by The Lalit Traveller,Sector 35, Faridabad',
 'Suncity Business Tower,Golf Course Road, Gurgaon',
 'Sector 29, Faridabad',
 'NIT, Faridabad',
 'DoubleTree By Hilton Gurugram Baani Square,Sector 50, Gurgaon']
```

Scraping Multiple Prices

In [11]:

```
price=[]

for i in soup.find_all('span', class_="double-line-ellipsis" ):
    price.append(i.text.replace('₹', ''))

price
```

Out[11]:

```
[' 2,000 for 2 (approx) | North Indian, Chinese',
 ' 1,400 for 2 (approx) | North Indian, Asian, Italian',
 ' 2,000 for 2 (approx) | Chinese, North Indian',
 ' 3,000 for 2 (approx) | Italian, Continental',
 ' 1,700 for 2 (approx) | North Indian, Chinese',
 ' 2,400 for 2 (approx) | North Indian, Italian',
 ' 1,800 for 2 (approx) | North Indian',
 ' 1,900 for 2 (approx) | North Indian, Chinese',
 ' 1,800 for 2 (approx) | North Indian, Chinese, Continental',
 ' 2,000 for 2 (approx) | North Indian, Mughlai',
 ' 800 for 2 (approx) | North Indian',
 ' 1,500 for 2 (approx) | North Indian, Mughlai, Desserts, Beverages',
 ' 3,400 for 2 (approx) | European, Italian, Asian, Continental']
```

Scraping the images

In [12]:

```
images=[]

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


images
```

Out[12]:

```
['https://im1.dineout.co.in/images/uploads/restaurant/sharpen/8/k/b/p86792-16062953735fbe1
f4d3fb7e.jpg?tr=tr:n-medium',
 'https://im1.dineout.co.in/images/uploads/restaurant/sharpen/3/h/c/p3643-144497865356209f
dd65746.jpg?tr=tr:n-medium',
 'https://im1.dineout.co.in/images/uploads/restaurant/sharpen/3/j/o/p38113-15959192065f1fc
b666130c.jpg?tr=tr:n-medium',
 'https://im1.dineout.co.in/images/uploads/restaurant/sharpen/4/p/m/p406-15438184745c04cce
a491bc.jpg?tr=tr:n-medium',
 'https://im1.dineout.co.in/images/uploads/restaurant/sharpen/7/p/k/p79307-16051787755fad1
597f2bf9.jpg?tr=tr:n-medium',
 'https://im1.dineout.co.in/images/uploads/restaurant/sharpen/2/v/t/p2687-1482477169585cce
712b90f.jpg?tr=tr:n-medium',
 'https://im1.dineout.co.in/images/uploads/restaurant/sharpen/5/v/f/p52501-16006856545f688
65616659.jpg?tr=tr:n-medium',
 'https://im1.dineout.co.in/images/uploads/restaurant/sharpen/3/n/o/p34822-15599107305cfa5
94a13c24.jpg?tr=tr:n-medium',
 'https://im1.dineout.co.in/images/uploads/restaurant/sharpen/1/p/y/p12366-1466935020576fa
6ecdc359.jpg?tr=tr:n-medium',
 'https://im1.dineout.co.in/images/uploads/restaurant/sharpen/5/y/d/p549-15291237715b2493b
b2a415.jpg?tr=tr:n-medium',
 'https://im1.dineout.co.in/images/uploads/restaurant/sharpen/4/n/t/p43488-164732311162302
7e763947.jpg?tr=tr:n-medium',
 'https://im1.dineout.co.in/images/uploads/restaurant/sharpen/5/w/r/p58842-15624171585d209
r=tr:n-medium',
```

```
 'https://im1.dineout.co.in/images/uploads/restaurant/sharpen/9/m/a/p9875-1645177960620f6c
68ecfef.jpg?tr=tr:n-medium']
```

Ratings

In [13]:
```
ratings=[]

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

ratings
```

Out[13]:
```
['3.5',
 '3.9',
 '3.9',
 '4.3',
 '4',
 '3.9',
 '3.7',
 '3.9',
 '4.2',
 '4.3',
 '3.6',
 '4.2',
 '4']
```

Print Length

In [14]:
```
print(len(titles), len(loc), len(price), len(images), len(ratings))
```

```
13 13 13 13 13
```

Make Dataframe

In [15]:
```
import pandas as pd
```

In [16]:
```
df= pd.DataFrame({'Titles': titles, 'Location': loc, 'Price': price, 'Images_URL': images,
df
```

Out[16]:

	Titles	Location	Price	Images_URL	Ratings
0	Castle BarbequeConnaught Place, Central Delhi	Connaught Place, Central Delhi	2,000 for 2 (approx) | North Indian, Chinese	https://im1.dineout.co.in/images/uploads/resta...	3.5
1	Jungle Jamboree3CS Mall,Lajpat Nagar - 3, Sout...	3CS Mall,Lajpat Nagar - 3, South Delhi	1,400 for 2 (approx) | North Indian, Asian, I...	https://im1.dineout.co.in/images/uploads/resta...	3.9
2	Castle BarbequePacific Mall,Tagore Garden, Wes...	Pacific Mall,Tagore Garden, West Delhi	2,000 for 2 (approx) | Chinese, North Indian	https://im1.dineout.co.in/images/uploads/resta...	3.9

	Titles	Location	Price	Images_URL	Ratings
3	Cafe KnoshThe Leela Ambience Convention Hotel,...	The Leela Ambience Convention Hotel,Shahdara, ...	3,000 for 2 (approx) | Italian, Continental	https://im1.dineout.co.in/images/uploads/resta...	4.3
4	The Barbeque CompanyGardens Galleria,Sector 38...	Gardens Galleria,Sector 38A, Noida	1,700 for 2 (approx) | North Indian, Chinese	https://im1.dineout.co.in/images/uploads/resta...	4
5	India GrillHilton Garden Inn,Saket, South Delhi	Hilton Garden Inn,Saket, South Delhi	2,400 for 2 (approx) | North Indian, Italian	https://im1.dineout.co.in/images/uploads/resta...	3.9
6	Delhi BarbequeTaurus Sarovar Portico,Mahipalpu...	Taurus Sarovar Portico,Mahipalpur, South Delhi	1,800 for 2 (approx) | North Indian	https://im1.dineout.co.in/images/uploads/resta...	3.7
7	The Monarch - Bar Be Que VillageIndirapuram Ha...	Indirapuram Habitat Centre,Indirapuram, Ghaziabad	1,900 for 2 (approx) | North Indian, Chinese	https://im1.dineout.co.in/images/uploads/resta...	3.9
8	World CafeVibe by The Lalit Traveller,Sector 3...	Vibe by The Lalit Traveller,Sector 35, Faridabad	1,800 for 2 (approx) | North Indian, Chinese,...	https://im1.dineout.co.in/images/uploads/resta...	4.2
9	Indian Grill RoomSuncity Business Tower,Golf C...	Suncity Business Tower,Golf Course Road, Gurgaon	2,000 for 2 (approx) | North Indian, Mughlai	https://im1.dineout.co.in/images/uploads/resta...	4.3
10	Mad 4 Bar B QueSector 29, Faridabad	Sector 29, Faridabad	800 for 2 (approx) | North Indian	https://im1.dineout.co.in/images/uploads/resta...	3.6
11	Barbeque 29NIT, Faridabad	NIT, Faridabad	1,500 for 2 (approx) | North Indian, Mughlai,...	https://im1.dineout.co.in/images/uploads/resta...	4.2
12	GlasshouseDoubleTree By Hilton Gurugram Baani ...	DoubleTree By Hilton Gurugram Baani Square,Sec...	3,400 for 2 (approx) | European, Italian, Asi...	https://im1.dineout.co.in/images/uploads/resta...	4

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [27]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

Program to display Houses from NoBroker.com

IMPORT LIBRARIES

In [1]:
```
from bs4 import BeautifulSoup
import requests
```

In [2]:
```
url= requests.get('https://www.nobroker.in/property/sale/chennai/Indira%20Nagar?searchPar
url
```

Out[2]:
```
<Response [200]>
```

In [3]:
```
request=url.text
```

Scrapping

In [ ]:
```
soup_data= BeautifulSoup(request, 'html.parser')
soup_data.prettify()
```

In [5]:
```
#title= soup_data.findAll('div', class_="nb__7nqQI")
#title
title=[]

for i in soup_data.find_all('h2',class_="heading-6 flex items-center font-semi-bold m-0")
    title.append(i.text)

title
```

Out[5]:
```
['2 BHK Flat  For Sale  In Dhakshni Dhirshti In 12th Cross Street',
 '2 BHK Apartment  For Sale  In Tnhb In Adyar',
 '2 BHK Apartment  For Sale  In The Marvel Tower In 155 Indira Nagar 4th Avenue',
 '3 BHK Flat  For Sale  In Adyar',
 '2 BHK Flat  For Sale  In Tnhb Mig Aptssmarveltower Indiranagaradayar In Indira Nagar',
 '3 BHK In Independent House  For Sale  In Indira Nagar',
 '2 BHK Flat  For Sale  In Tnhb Flats, Adyar In Adyar',
 '2 BHK Flat  For Sale  In Tnhb The Marvel Towers  In Adyar',
 '1 BHK Flat  For Sale  In Indira Palace Legacy Towers, In  Adyar',
 '3 BHK In Independent House  For Sale  In Indira Nagar',
 '1 BHK In Independent House  For Sale  In Indirangar',
 '1 BHK Flat  For Sale  In Aishwarya Colony In Adyar']
```

In [ ]:
```

```

In [6]:
```
address=[]

for i in soup_data.findAll('div', class_="mt-0.5p overflow-hidden overflow-ellipsis white
    address.append(i.text)

address
```



```
ilway station',
```

```
 'TNHBÂ\xa0 Dr Muthu Lakshmi Rd, Indira Nagar, near periya amman temple ',
 'The Marvel TowerÂ\xa0 4th Ave, Indira Nagar ,Near Navadurgai Amman Temple',
 'Standalone Building, indra nagar,Near Shri Periya Palayathamman Temple, Sri Anjaneyar Te
mple',
 '4th Aveneue Indira Nagar behind TTKhospi',
 'Independent House, indira nagar, adyar',
 '4th Ave, Indira Nagar Near TTK Hospital (T.T. Ranganathan Clinical Research Foundatio
n)',
 '4th Ave, Indira Nagar Near TTK Hospital',
 '4th avenue, Indra Nagar,near TTK Hospital (T.T. Ranganathan Clinical Research Foundatio
n',
 'Independent House,  OMR Service Rd, Tharamani, CIT Campus  near  Lakshmi maternity hospi
tal',
 'Independent House, 2nd Ave, Indira Nagar, Adyar, Chennai, Tamil Nadu 600020',
 ' 3rd Ave, Indira Nagar, Adyar,  Near - Indira Nagar library']
```

In [7]:

```
area=[]

for i in soup_data.find_all('div',class_="flex flex-col w-33pe items-center border-r bord
    area.append(i.text.replace('â\x82¹',''))

area
```

Out[7]:

```
['1,132 sqftBuiltup',
 '57,314/MonthEstimated EMI',
 '1,017 sqftBuiltup',
 '74,508/MonthEstimated EMI',
 '1,618 sqftBuiltup',
 '1.2 Lacs/MonthEstimated EMI',
 '1,025 sqftBuiltup',
 '68,777/MonthEstimated EMI',
 '1,592 sqftBuiltup',
 '1.15 Lacs/MonthEstimated EMI',
 '2,000 sqftBuiltup',
 '1.72 Lacs/MonthEstimated EMI',
 '1,100 sqftBuiltup',
 '91,703/MonthEstimated EMI',
 '1,584 sqftBuiltup',
 '1.18 Lacs/MonthEstimated EMI',
 '650 sqftBuiltup',
 '41,266/MonthEstimated EMI',
 '1,600 sqftBuiltup',
 '25,791/MonthEstimated EMI',
 '657 sqftBuiltup',
 '40,120/MonthEstimated EMI',
 '600 sqftBuiltup',
 '22,925/MonthEstimated EMI']
```

In [8]:

```
emi= area[1::2]
emi
```

Out[8]:

```
['57,314/MonthEstimated EMI',
 '74,508/MonthEstimated EMI',
 '1.2 Lacs/MonthEstimated EMI',
 '68,777/MonthEstimated EMI',
 '1.15 Lacs/MonthEstimated EMI',
 '1.72 Lacs/MonthEstimated EMI',
 '91,703/MonthEstimated EMI',
 '1.18 Lacs/MonthEstimated EMI',
 '41,266/MonthEstimated EMI',
 '25,791/MonthEstimated EMI',
 '40,120/MonthEstimated EMI',
 '22,925/MonthEstimated EMI']
```

In [9]:

```
build_area=area[0::2]
```

Out[9]:
```
['1,132 sqftBuiltup',
 '1,017 sqftBuiltup',
 '1,618 sqftBuiltup',
 '1,025 sqftBuiltup',
 '1,592 sqftBuiltup',
 '2,000 sqftBuiltup',
 '1,100 sqftBuiltup',
 '1,584 sqftBuiltup',
 '650 sqftBuiltup',
 '1,600 sqftBuiltup',
 '657 sqftBuiltup',
 '600 sqftBuiltup']
```

In [10]:
```
price=[]

for i in soup_data.find_all('div',class_="flex flex-col w-33pe items-center bo tp:w-half
    price.append(i.text.replace('â\x82¹1 Croreâ\x82¹',''))

price
```

Out[10]:
```
['8,834 per sq.ft.',
 'â\x82¹1.3 Croresâ\x82¹12,783 per sq.ft.',
 'â\x82¹2.1 Croresâ\x82¹12,979 per sq.ft.',
 'â\x82¹1.2 Croresâ\x82¹11,707 per sq.ft.',
 'â\x82¹2 Croresâ\x82¹12,563 per sq.ft.',
 'â\x82¹3 Croresâ\x82¹15,000 per sq.ft.',
 'â\x82¹1.6 Croresâ\x82¹14,545 per sq.ft.',
 'â\x82¹2.06 Croresâ\x82¹13,000 per sq.ft.',
 'â\x82¹72 Lacsâ\x82¹11,077 per sq.ft.',
 'â\x82¹45 Lacsâ\x82¹2,813 per sq.ft.',
 'â\x82¹70 Lacsâ\x82¹10,654 per sq.ft.',
 'â\x82¹40 Lacsâ\x82¹6,667 per sq.ft.']
```

In [11]:
```
import pandas as pd
```

In [12]:
```
df=pd.DataFrame({'Title': title, 'Address': address, 'Build_area': build_area, 'EMI': emi
df
```

Out[12]:

	Title	Address	Build_area	EMI	Price
0	2 BHK Flat For Sale In Dhakshni Dhirshti In ...	indra nagar railway station	1,132 sqftBuiltup	57,314/MonthEstimated EMI	8,834 per sq.ft.
1	2 BHK Apartment For Sale In Tnhb In Adyar	TNHBÂ Dr Muthu Lakshmi Rd, Indira Nagar, near...	1,017 sqftBuiltup	74,508/MonthEstimated EMI	â ¹1.3 Croresâ ¹12,783 per sq.ft.
2	2 BHK Apartment For Sale In The Marvel Tower...	The Marvel TowerÂ 4th Ave, Indira Nagar ,Near...	1,618 sqftBuiltup	1.2 Lacs/MonthEstimated EMI	â ¹2.1 Croresâ ¹12,979 per sq.ft.
3	3 BHK Flat For Sale In Adyar	Standalone Building, indra nagar,Near Shri Per...	1,025 sqftBuiltup	68,777/MonthEstimated EMI	â ¹1.2 Croresâ ¹11,707 per sq.ft.
4	2 BHK Flat For Sale In Tnhb Mig Aptssmarvelt...	4th Aveneue Indira Nagar behind TTKhospi	1,592 sqftBuiltup	1.15 Lacs/MonthEstimated EMI	â ¹2 Croresâ ¹12,563 per sq.ft.
5	3 BHK In Independent House For Sale In Indir...	Independent House, indira nagar, adyar	2,000 sqftBuiltup	1.72 Lacs/MonthEstimated EMI	â ¹3 Croresâ ¹15,000 per sq.ft.
6	2 BHK Flat For Sale In Tnhb Flats, Adyar In ...	4th Ave, Indira Nagar Near TTK Hospital (T.T. ...	1,100 sqftBuiltup	91,703/MonthEstimated EMI	â ¹1.6 Croresâ ¹14,545 per sq.ft.

	Title	Address	Build_area	EMI	Price
7	2 BHK Flat For Sale In Tnhb The Marvel Tower...	4th Ave, Indira Nagar Near TTK Hospital	1,584 sqftBuiltup	1.18 Lacs/MonthEstimated EMI	â ¹2.06 Croresâ ¹13,000 per sq.ft.
8	1 BHK Flat For Sale In Indira Palace Legacy ...	4th avenue, Indra Nagar,near TTK Hospital (T.T...	650 sqftBuiltup	41,266/MonthEstimated EMI	â ¹72 Lacsâ ¹11,077 per sq.ft.
9	3 BHK In Independent House For Sale In Indir...	Independent House, OMR Service Rd, Tharamani,...	1,600 sqftBuiltup	25,791/MonthEstimated EMI	â ¹45 Lacsâ ¹2,813 per sq.ft.
10	1 BHK In Independent House For Sale In Indir...	Independent House, 2nd Ave, Indira Nagar, Adya...	657 sqftBuiltup	40,120/MonthEstimated EMI	â ¹70 Lacsâ ¹10,654 per sq.ft.
11	1 BHK Flat For Sale In Aishwarya Colony In A...	3rd Ave, Indira Nagar, Adyar, Near - Indira ...	600 sqftBuiltup	22,925/MonthEstimated EMI	â ¹40 Lacsâ ¹6,667 per sq.ft.

In [ ]:

Program to scrap Top 10 ODI Bowlers in Womens Cricket

Import Libraries

In [1]:
```
from bs4 import BeautifulSoup
import requests
```

Send Get request from Web page

In [2]:
```
page= requests.get('https://www.icc-cricket.com/rankings/womens/player-rankings/odi')
page
```

Out[2]:
```
<Response [200]>
```

Page Content

In [ ]:
```
soup= BeautifulSoup(page.text)
soup
```

Scrapping

In [4]:
```
player=[]

for i in soup.find_all('td', class_="table-body__cell name"):
    player.append(i.text.replace('\n',''))



player.insert(9,'Sophie Ecclestone')
player[9:19]
```

Out[4]:
```
['Sophie Ecclestone',
 'Jess Jonassen',
 'Shabnim Ismail',
 'Megan Schutt',
 'Jhulan Goswami',
 'Marizanne Kapp',
 'Ayabonga Khaka',
 'Kate Cross',
 'Ellyse Perry',
 'Hayley Matthews']
```

In [5]:
```
team=[]

for i in soup.find_all('span', class_="table-body__logo-text"):
    team.append(i.text)

team.insert(9,'ENG')
```

Out[5]:
```
['ENG', 'AUS', 'SA', 'AUS', 'IND', 'SA', 'SA', 'ENG', 'AUS', 'WI']
```

In [13]:
```
#
```

In [7]:
```
ratings=[]

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


ratings.insert(9,'787')
ratings[9:19]
```

Out[7]:
```
['787', '727', '724', '706', '663', '660', '650', '626', '623', '622']
```

In [8]:
```
pos=[]

for i in soup.find_all('span', class_="rankings-table__pos-number"):
    pos.append(i.text.replace('\n',''))

pos.insert(9,'1')
pos[9:19]
```

Out[8]:
```
['1',
 '                                    2                                    ',
 '                                    3                                    ',
 '                                    4                                    ',
 '                                    5                                    ',
 '                                    6                                    ',
 '                                    7                                    ',
 '                                    8                                    ',
 '                                    9                                    ',
 '                                    10                                   ']
```

In [12]:
```
#
```

In [10]:
```
import pandas as pd
```

In [11]:
```
df_bowlers=pd.DataFrame({'position': pos, 'Player': player, 'Team': team, 'Ratings': ratin
df_bowlers
df= df_bowlers.set_index('position')
df[9:19]
```

Out[11]:

position	Player	Team	Ratings
1	Sophie Ecclestone	ENG	787
2	Jess Jonassen	AUS	727
3	Shabnim Ismail	SA	724
4	Megan Schutt	AUS	706
5	Jhulan Goswami	IND	663
6	Marizanne Kapp	SA	660

position	Player	Team	Ratings
7	Ayabonga Khaka	SA	650
8	Kate Cross	ENG	626
9	Ellyse Perry	AUS	623
10	Hayley Matthews	WI	622

In [ ]:

Program to scrap Top 10 ODI Bowlers in Mens Cricket

Import Libraries

In [1]:
```
from bs4 import BeautifulSoup
import requests
```

Send Get request from Web page

In [2]:
```
page= requests.get('https://www.icc-cricket.com/rankings/mens/player-rankings/odi')
page
```

Out[2]:
```
<Response [200]>
```

Page Content

In [ ]:
```
soup= BeautifulSoup(page.text)
soup
```

Scrapping

In [4]:
```
player=[]

for i in soup.find_all('td', class_="table-body__cell name"):
    player.append(i.text.replace('\n',''))



player.insert(9,'Trent Boult')
player[9:19]
```

Out[4]:
```
['Trent Boult',
 'Josh Hazlewood',
 'Chris Woakes',
 'Matt Henry',
 'Mujeeb Ur Rahman',
 'Jasprit Bumrah',
 'Mehedi Hasan',
 'Shakib Al Hasan',
 'Adam Zampa',
 'Rashid Khan']
```

In [5]:
```
team=[]

for i in soup.find_all('span', class_="table-body__logo-text"):
    team.append(i.text)

team.insert(9,'NZ')
```

Out[5]:
```
['NZ', 'AUS', 'ENG', 'NZ', 'AFG', 'IND', 'BAN', 'BAN', 'AUS', 'AFG']
```

In [13]:
```
#
```

In [7]:
```
ratings=[]

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


ratings.insert(9,'733')
ratings[9:19]
```

Out[7]:
```
['733', '705', '700', '687', '681', '679', '661', '657', '650', '650']
```

In [8]:
```
pos=[]

for i in soup.find_all('span', class_="rankings-table__pos-number"):
    pos.append(i.text.replace('\n',''))

pos.insert(9,'1')
pos[9:19]
```

Out[8]:
```
['1',
 '                                    2                                    ',
 '                                    3                                    ',
 '                                    4                                    ',
 '                                    5                                    ',
 '                                    6                                    ',
 '                                    7                                    ',
 '                                    8                                    ',
 '                                    9                                    ',
 '                                    10                                   ']
```

In [12]:
```
#
```

In [10]:
```
import pandas as pd
```

In [11]:
```
df_bowlers=pd.DataFrame({'position': pos, 'Player': player, 'Team': team, 'Ratings': ratin
df_bowlers
df= df_bowlers.set_index('position')
df[9:19]
```

Out[11]:

position	Player	Team	Ratings
1	Trent Boult	NZ	733
2	Josh Hazlewood	AUS	705
3	Chris Woakes	ENG	700
4	Matt Henry	NZ	687
5	Mujeeb Ur Rahman	AFG	681
6	Jasprit Bumrah	IND	679

position	Player	Team	Ratings
7	Mehedi Hasan	BAN	661
8	Shakib Al Hasan	BAN	657
9	Adam Zampa	AUS	650
10	Rashid Khan	AFG	650

In [ ]:

Program to scrap Top 10 ODI Batsmen in Womens Cricket

Import Libraries

In [1]:
```
from bs4 import BeautifulSoup
import requests
```

Send Get request from Web page

In [2]:
```
page= requests.get('https://www.icc-cricket.com/rankings/womens/player-rankings/odi')
page
```

Out[2]:
```
<Response [200]>
```

Page Content

In [ ]:
```
soup= BeautifulSoup(page.text)
soup
```

Scrapping

In [4]:
```
player=[]

for i in soup.find_all('td', class_="table-body__cell name"):
    player.append(i.text.replace('\n',''))



player.insert(0,'Laura Wolvaardt')
player[0:10]
```

Out[4]:
```
['Laura Wolvaardt',
 'Beth Mooney',
 'Meg Lanning',
 'Natalie Sciver',
 'Alyssa Healy',
 'Mithali Raj',
 'Rachael Haynes',
 'Tammy Beaumont',
 'Amy Satterthwaite',
 'Smriti Mandhana']
```

In [5]:
```
team=[]

for i in soup.find_all('span', class_="table-body__logo-text"):
    team.append(i.text)

team.insert(0,'SA')
```

Out[5]:
```
['SA', 'AUS', 'AUS', 'ENG', 'AUS', 'IND', 'AUS', 'ENG', 'NZ', 'IND']
```

In [6]:
```
#
```

In [7]:
```
ratings=[]

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


ratings.insert(0,'740')
ratings[0:10]
```

Out[7]:
```
['740', '726', '718', '705', '703', '686', '684', '682', '681', '669']
```

In [8]:
```
pos=[]

for i in soup.find_all('span', class_="rankings-table__pos-number"):
    pos.append(i.text.replace('\n',''))

pos.insert(0,'1')
pos[0:10]
```

Out[8]:
```
['1',
 '                                  2                                  ',
 '                                  3                                  ',
 '                                  4                                  ',
 '                                  5                                  ',
 '                                  6                                  ',
 '                                  7                                  ',
 '                                  8                                  ',
 '                                  9                                  ',
 '                                  10                                 ']
```

In [9]:
```
#
```

In [10]:
```
import pandas as pd
```

In [11]:
```
df_batting=pd.DataFrame({'position': pos, 'Player': player, 'Team': team, 'Ratings': ratir
df_batting
df= df_batting.set_index('position')
df[0:10]
```

Out[11]:

position	Player	Team	Ratings
1	Laura Wolvaardt	SA	740
2	Beth Mooney	AUS	726
3	Meg Lanning	AUS	718
4	Natalie Sciver	ENG	705
5	Alyssa Healy	AUS	703
6	Mithali Raj	IND	686

position	Player	Team	Ratings
7	Rachael Haynes	AUS	684
8	Tammy Beaumont	ENG	682
9	Amy Satterthwaite	NZ	681
10	Smriti Mandhana	IND	669

In [ ]:

Program to scrap Top 10 ODI Batsmen in Mens Cricket

Import Libraries

In [1]:
```
from bs4 import BeautifulSoup
import requests
```

Send Get request from Web page

In [2]:
```
page= requests.get('https://www.icc-cricket.com/rankings/mens/player-rankings/odi')
page
```

Out[2]:
```
<Response [200]>
```

Page Content

In [ ]:
```
soup= BeautifulSoup(page.text)
soup
```

Scrapping

In [4]:
```
player=[]

for i in soup.find_all('td', class_="table-body__cell name"):
    player.append(i.text.replace('\n',''))



player.insert(0,'Babar Azam')
player[0:10]
```

Out[4]:
```
['Babar Azam',
 'Virat Kohli',
 'Ross Taylor',
 'Rohit Sharma',
 'Quinton de Kock',
 'Jonny Bairstow',
 'Aaron Finch',
 'Rassie van der Dussen',
 'David Warner',
 'Imam-ul-Haq']
```

In [5]:
```
team=[]

for i in soup.find_all('span', class_="table-body__logo-text"):
    team.append(i.text)

team.insert(0,'PAK')
```

Out[5]:
```
['PAK', 'IND', 'NZ', 'IND', 'SA', 'ENG', 'AUS', 'SA', 'AUS', 'PAK']
```

In [6]:
```
#
```

In [7]:
```
ratings=[]

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


ratings.insert(0,'872')
ratings[0:10]
```

Out[7]:
```
['872', '811', '794', '791', '789', '775', '771', '769', '758', '746']
```

In [8]:
```
pos=[]

for i in soup.find_all('span', class_="rankings-table__pos-number"):
    pos.append(i.text.replace('\n',''))

pos.insert(0,'1')
pos[0:10]
```

Out[8]:
```
['1',
 '                                    2                                    ',
 '                                    3                                    ',
 '                                    4                                    ',
 '                                    5                                    ',
 '                                    6                                    ',
 '                                    7                                    ',
 '                                    8                                    ',
 '                                    9                                    ',
 '                                    10                                   ']
```

In [12]:
```
#
```

In [10]:
```
import pandas as pd
```

In [11]:
```
df_batting=pd.DataFrame({'position': pos, 'Player': player, 'Team': team, 'Ratings': ratin
df_batting
df= df_batting.set_index('position')
df[0:10]
```

Out[11]:

position	Player	Team	Ratings
1	Babar Azam	PAK	872
2	Virat Kohli	IND	811
3	Ross Taylor	NZ	794
4	Rohit Sharma	IND	791
5	Quinton de Kock	SA	789
6	Jonny Bairstow	ENG	775

position	Player	Team	Ratings
7	Aaron Finch	AUS	771
8	Rassie van der Dussen	SA	769
9	David Warner	AUS	758
10	Imam-ul-Haq	PAK	746

In [ ]:

Program to scrap Top 10 ODI Teams in Womens Cricket

Import Libraries

In [1]:
```
from bs4 import BeautifulSoup
import requests
```

Send Get request from Web page

In [2]:
```
page= requests.get('https://www.icc-cricket.com/rankings/womens/team-rankings/odi')
page
```

Out[2]:
```
<Response [200]>
```

Page Content

In [ ]:
```
soup= BeautifulSoup(page.text)
soup
```

Scrapping

In [4]:
```
team=[]

for i in soup.find_all('span', class_="u-hide-phablet"):
    team.append(i.text)


team=team[0:10]
team
```

Out[4]:
```
['Australia',
 'South Africa',
 'England',
 'India',
 'New Zealand',
 'West Indies',
 'Bangladesh',
 'Pakistan',
 'Ireland',
 'Sri Lanka']
```

In [12]:
```
matchepoints=[]

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

matchepoints
```

Out[12]:
```
['28',
 '3,504',
 '29',
 '3,425',
 '29',
 '2,890',
 '31',
 '3,018',
 '28',
 '2,478',
 '12',
 '935',
 '26',
 '1,753',
 '5',
 '240',
 '5',
 '233',
 '8',
 '0']
```

In [18]:
```
matches= matchepoints[0::2]
matches.insert(0,'28')
matches =matches[0:10]
matches
```

Out[18]:
```
['28', '28', '29', '29', '31', '28', '12', '26', '5', '5']
```

In [19]:
```
points= matchepoints[1::2]
points.insert(0,'4,663')
points= points[0:10]
points
```

Out[19]:
```
['4,663',
 '3,504',
 '3,425',
 '2,890',
 '3,018',
 '2,478',
 '935',
 '1,753',
 '240',
 '233']
```

In [7]:
```
ratings=[]

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


ratings.insert(0,'166')
ratings= ratings[0:10]
ratings
```

Out[7]:
```
['166', '125', '118', '100', '97', '89', '78', '67', '48', '47']
```

In [8]:
```
pos=[]

for i in soup.find_all('td', class_="table-body__cell table-body__cell--position u-text-r
    pos.append(i.text)

pos.insert(0,'1')
```

```
pos= pos[0:10]
pos
```

Out[8]:
```
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10']
```

In [20]:
```
print(len(team), len(ratings), len(pos),len(matches),len(points))
```

```
10 10 10 10 10
```

In [21]:
```
import pandas as pd
```

In [23]:
```
df=pd.DataFrame({'Position': pos, 'Team': team, 'Matches': matches, 'Points': points, 'Ra
df
```

Out[23]:

	Position	Team	Matches	Points	Ratings
0	1	Australia	28	4,663	166
1	2	South Africa	28	3,504	125
2	3	England	29	3,425	118
3	4	India	29	2,890	100
4	5	New Zealand	31	3,018	97
5	6	West Indies	28	2,478	89
6	7	Bangladesh	12	935	78
7	8	Pakistan	26	1,753	67
8	9	Ireland	5	240	48
9	10	Sri Lanka	5	233	47

In [ ]:
```

```

Program to scrap Top 10 ODI Teams in Mens Cricket

Import Libraries

In [2]:
```
from bs4 import BeautifulSoup
import requests
```

Send Get request from Web page

In [3]:
```
page= requests.get('https://www.icc-cricket.com/rankings/mens/team-rankings/odi')
page
```

Out[3]:
```
<Response [200]>
```

Page Content

In [ ]:
```
soup= BeautifulSoup(page.text)
soup
```

Scrapping

In [5]:
```
team=[]

for i in soup.find_all('span', class_="u-hide-phablet"):
    team.append(i.text)


team=team[0:20]
team
```

Out[5]:
```
['New Zealand',
 'England',
 'Australia',
 'India',
 'South Africa',
 'Bangladesh',
 'Pakistan',
 'Sri Lanka',
 'West Indies',
 'Afghanistan',
 'Ireland',
 'Scotland',
 'Zimbabwe',
 'Netherlands',
 'UAE',
 'Oman',
 'Namibia',
 'Nepal',
 'United States',
```

In [28]:
```
matchepoints=[]

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

matchepoints
```

Out[28]:
```
['32',
 '3,793',
 '29',
 '3,387',
 '38',
 '4,162',
 '31',
 '3,167',
 '36',
 '3,350',
 '28',
 '2,590',
 '35',
 '2,835',
 '36',
 '2,788',
 '23',
 '1,562',
 '28',
 '1,445',
 '10',
 '452',
 '23',
 '951',
 '11',
 '406',
 '20',
 '651',
 '24',
 '720',
 '13',
 '268',
 '17',
 '308',
 '14',
 '232',
 '19',
 '207']
```

In [31]:
```
matches= matchepoints[0::2]
matches.insert(0,'18')
matches
```

Out[31]:
```
['18',
 '32',
 '29',
 '38',
 '31',
 '36',
 '28',
 '35',
 '36',
 '23',
 '28',
 '10',
 '23',
 '11',
 '20',
```

```
'13',
'17',
'14',
'19']
```

In [32]:

```
points= matchepoints[1::2]
points.insert(0,'2,185')
points
```

Out[32]:

```
['2,185',
 '3,793',
 '3,387',
 '4,162',
 '3,167',
 '3,350',
 '2,590',
 '2,835',
 '2,788',
 '1,562',
 '1,445',
 '452',
 '951',
 '406',
 '651',
 '720',
 '268',
 '308',
 '232',
 '207']
```

In [40]:

```
ratings=[]

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


ratings.insert(0,'121')
ratings
```

Out[40]:

```
['121',
 '119',
 '117',
 '110',
 '102',
 '93',
 '93',
 '81',
 '77',
 '68',
 '52',
 '45',
 '41',
 '37',
 '33',
 '30',
 '21',
 '18',
 '17',
 '11']
```

In [38]:

```
pos=[]

for i in soup.find_all('td', class_="table-body__cell table-body__cell--position u-text-r
    pos.append(i.text)
```

```
pos.insert(0,'1')
pos
```

Out[38]:
```
['1',
 '2',
 '3',
 '4',
 '5',
 '6',
 '7',
 '8',
 '9',
 '10',
 '11',
 '12',
 '13',
 '14',
 '15',
 '16',
 '17',
 '18',
 '19',
 '20']
```

In [34]:
```
print(len(points), len(matches))
```

```
20 20
```

In [35]:
```
print(len(team),len(ratings), len(pos))
```

```
20 20 20
```

In [36]:
```
import pandas as pd
```

In [41]:
```
df=pd.DataFrame({'Position': pos, 'Team': team, 'Matches': matches, 'Points': points, 'Ra
df[0:10]
```

Out[41]:

	Position	Team	Matches	Points	Ratings
0	1	New Zealand	18	2,185	121
1	2	England	32	3,793	119
2	3	Australia	29	3,387	117
3	4	India	38	4,162	110
4	5	South Africa	31	3,167	102
5	6	Bangladesh	36	3,350	93
6	7	Pakistan	28	2,590	93
7	8	Sri Lanka	35	2,835	81
8	9	West Indies	36	2,788	77
9	10	Afghanistan	23	1,562	68

In [ ]:
```

```

Python Program to scrap Product name, price and discount from Meesho

Import Libraries

In [1]:
```
from bs4 import BeautifulSoup
import requests
```

Send Get request from Web page

In [2]:
```
page= requests.get('https://meesho.com/bags-ladies/pl/p7vbp')
page
```

Out[2]:
```
<Response [200]>
```

Page Content

In [ ]:
```
soup= BeautifulSoup(page.text)
soup
```

Scrapping

In [4]:
```
products=[]

for i in soup.find_all('p',class_="Text__StyledText-sc-oo0kvp-0 cPgaBh NewProductCard__Pr
    products.append(i.text)

products
```

Out[4]:
```
['Graceful Stylish Women Handbags',
 'Voguish Fashionable Women Handbags',
 'Elegant Attractive Women Handbags',
 'Classic Alluring Women Handbags',
 'Elite Stylish Women Handbags',
 'Elegant Stylish Women Handbags',
 'Classic Stylish Women Handbags',
 'Classic Fashionable Women Handbags',
 'Trendy Fashionable Women Handbags',
 'Ravishing Alluring Women Handbags',
 'Voguish Classy Women Handbags',
 'Classic Stylish Women Handbags',
 'Elite Stylish Women Handbags',
 'Trendy Versatile Women Handbags',
 'Classic Classy Women Handbags',
 'Elegant Fashionable Women Handbags',
 'Elegant Attractive Women Handbags',
 'Elegant Versatile Women Handbags',
 'Ameyson Attractive Women Jute Printed Lunch Time Handbags',
 'Graceful Attractive Women Handbags']
```

```
for i in soup.find_all('h5',class_="Text__StyledText-sc-oo0kvp-0 dLSsNI"):
    price.append(i.text.replace('₹',''))
price
```

Out[5]:
```
['434',
 '346',
 '386',
 '242',
 '429',
 '434',
 '334',
 '91',
 '434',
 '765',
 '354',
 '383',
 '449',
 '73',
 '434',
 '224',
 '424',
 '634',
 '176',
 '476']
```

In [6]:
```
dis=[]


for i in soup.find_all('span', class_="Text__StyledText-sc-oo0kvp-0 cZvGTZ"):
    dis.append(i.text.replace('off',''))

dis
```

Out[6]:
```
['10% ',
 '13% ',
 '11% ',
 '15% ',
 '10% ',
 '10% ',
 '13% ',
 '15% ',
 '10% ',
 '6% ',
 '12% ',
 '12% ',
 '10% ',
 '14% ',
 '10% ',
 '15% ',
 '11% ',
 '7% ',
 '15% ',
 '10% ']
```

In [13]:
```
#
```

Length

In [14]:
```
print(len(products), len(price), len(dis))
```

```
20 20 20
```

Make DataFrame

In [9]:
```
import pandas as pd
```

In [10]:
```
df= pd.DataFrame({'Product Name': products, 'Price': price, 'Dicount': dis})
df
```

Out[10]:

	Product Name	Price	Dicount
0	Graceful Stylish Women Handbags	434	10%
1	Voguish Fashionable Women Handbags	346	13%
2	Elegant Attractive Women Handbags	386	11%
3	Classic Alluring Women Handbags	242	15%
4	Elite Stylish Women Handbags	429	10%
5	Elegant Stylish Women Handbags	434	10%
6	Classic Stylish Women Handbags	334	13%
7	Classic Fashionable Women Handbags	91	15%
8	Trendy Fashionable Women Handbags	434	10%
9	Ravishing Alluring Women Handbags	765	6%
10	Voguish Classy Women Handbags	354	12%
11	Classic Stylish Women Handbags	383	12%
12	Elite Stylish Women Handbags	449	10%
13	Trendy Versatile Women Handbags	73	14%
14	Classic Classy Women Handbags	434	10%
15	Elegant Fashionable Women Handbags	224	15%
16	Elegant Attractive Women Handbags	424	11%
17	Elegant Versatile Women Handbags	634	7%
18	Ameyson Attractive Women Jute Printed Lunch Ti...	176	15%
19	Graceful Attractive Women Handbags	476	10%

In [ ]:
```

```

Program to display IMDBS Top Rated 100 INDIAN Movies

Import Libraries

In [1]:
```
from bs4 import BeautifulSoup
import requests
```

Send Get request from Web page

In [2]:
```
url= requests.get('https://www.imdb.com/india/top-rated-indian-movies/')
url
```

Out[2]:
```
<Response [200]>
```

Page Content

In [3]:
```
request = url.text
```

Scrapping header

In [ ]:
```
soup_data= BeautifulSoup(request, 'html.parser')
soup_data
```

In [5]:
```
soup_data.title.text
```

Out[5]:
```
'Top Rated Indian Movies - IMDb'
```

In [ ]:
```
movies = soup_data.findAll('tbody', {'class':"lister-list"})
movies
```

In [7]:
```
#
```

In [8]:
```
#
```

In [17]:
```
#Year Of Release
Year=[]

for i in soup_data.find_all('span', class_='secondaryInfo'):
    Year.append(i.text)

Year=Year[0:100]
Year
```

Out[17]:
```
['(2021)',
 '(2003)',
 '(2018)',
 '(1979)',
 '(1987)',
 '(2009)',
 '(1959)',
 '(2020)',
 '(2018)',
 '(2019)',
 '(2004)',
 '(2019)',
 '(2021)',
 '(2007)',
 '(2019)',
 '(1993)',
 '(2016)',
 '(1989)',
 '(2019)',
 '(2021)',
 '(2021)',
 '(1992)',
 '(2018)',
 '(1991)',
 '(1955)',
 '(2016)',
 '(2015)',
 '(2021)',
 '(2015)',
 '(2018)',
 '(2018)',
 '(1956)',
 '(1983)',
 '(2019)',
 '(2006)',
 '(2018)',
 '(2018)',
 '(2005)',
 '(2019)',
 '(1975)',
 '(2014)',
 '(2016)',
 '(2015)',
 '(1998)',
 '(1993)',
 '(2013)',
 '(2019)',
 '(2022)',
 '(1988)',
 '(2017)',
 '(2013)',
 '(2002)',
 '(1997)',
 '(2018)',
 '(2018)',
 '(2012)',
 '(1965)',
 '(2015)',
 '(1995)',
 '(2016)',
 '(2012)',
 '(1999)',
 '(2012)',
 '(2006)',
 '(2004)',
 '(2011)',
 '(2007)',
 '(2016)',
```

```
'(2015)',
'(2021)',
'(2013)',
'(1957)',
'(2005)',
'(1992)',
'(2021)',
'(2012)',
'(2019)',
'(2014)',
'(2013)',
'(1989)',
'(2014)',
'(2017)',
'(2015)',
'(2003)',
'(2014)',
'(2003)',
'(1999)',
'(2001)',
'(2012)',
'(2000)',
'(1995)',
'(2012)',
'(2010)',
'(2002)',
'(2018)',
'(1975)',
'(1982)',
'(2017)',
'(2016)']
```

In [18]:

```
#Ratings
#first_movie.find('td',{'class':"ratingColumn imdbRating"}).text.replace('\n','')

Ratings=[]

for i in soup_data.find_all('td', class_= 'ratingColumn imdbRating'):
    Ratings.append(i.text.replace('\n', ''))

Ratings= Ratings[0:100]
Ratings
```

Out[18]:

```
['8.4',
 '8.4',
 '8.4',
 '8.4',
 '8.4',
 '8.4',
 '8.4',
 '8.4',
 '8.3',
 '8.3',
 '8.3',
 '8.3',
 '8.3',
 '8.3',
 '8.3',
 '8.3',
 '8.3',
 '8.3',
 '8.3',
 '8.3',
 '8.3',
 '8.3',
 '8.3',
 '8.3',
 '8.3',
 '8.3',
 '8.3',
 '8.3',
```

```
'8.3',
'8.3',
'8.3',
'8.2',
'8.2',
'8.2',
'8.2',
'8.2',
'8.2',
'8.2',
'8.2',
'8.2',
'8.2',
'8.2',
'8.2',
'8.2',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
'8.1',
```

```
'8.0',
'8.0',
'8.0',
'8.0',
'8.0']
```

In [19]:

```
movie_name=[]

for i in soup_data.find_all('td', class_="titleColumn"):
    movie_name.append(i.a.text)

movie_name= movie_name[0:100]
movie_name
```

Out[19]:

```
['Jai Bhim',
 'Anbe Sivam',
 'Pariyerum Perumal',
 'Golmaal',
 'Nayakan',
 '3 Idiots',
 'Apur Sansar',
 'Soorarai Pottru',
 'C/o Kancharapalem',
 'Kumbalangi Nights',
 'Black Friday',
 'Jersey',
 '#Home',
 'Taare Zameen Par',
 'Kaithi',
 'Manichitrathazhu',
 'Dangal',
 'Kireedam',
 'Asuran',
 'Sardar Udham',
 'Sarpatta Parambarai',
 'Thevar Magan',
 '96',
 'Thalapathi',
 'Pather Panchali',
 'Natsamrat',
 'Visaaranai',
 'Drishyam 2',
 'Thani Oruvan',
 'Vada Chennai',
 'Peranbu',
 'Aparajito',
 'Jaane Bhi Do Yaaro',
 'Agent Sai Srinivasa Athreya',
 'Khosla Ka Ghosla!',
 'Mahanati',
 'Ratsasan',
 'Anniyan',
 'Super Deluxe',
 'Chupke Chupke',
 'Bangalore Days',
 'Aruvi',
 'Premam',
 'Satya',
 'Devasuram',
 'Drishyam',
 'Chhichhore',
 'RRR (Rise Roar Revolt)',
 'Chithram',
 'Vikram Vedha',
 'Bhaag Milkha Bhaag',
 'Kannathil Muthamittal',
 'Iruvar',
```

```
'Tumbbad',
'Gangs of Wasseypur',
'Guide',
'Drishyam',
'Spadikam',
'Sairat',
'Paan Singh Tomar',
'Mudhalvan',
'Shahid',
'Pudhu Pettai',
'Swades: We, the People',
'Zindagi Na Milegi Dobara',
'Chak De! India',
'Dhuruvangal Pathinaaru',
'Uri: The Surgical Strike',
'Papanasam',
'Mandela',
'Soodhu Kavvum',
'Pyaasa',
'Black',
'Jo Jeeta Wohi Sikandar',
'Shershaah',
'OMG: Oh My God!',
'Article 15',
'Jigarthanda',
'Queen',
'Oru Vadakkan Veeragatha',
'Kaakkaa Muttai',
'Theeran Adhigaaram Ondru',
'Talvar',
'Munna Bhai M.B.B.S.',
'PK',
'Pithamagan',
'Sarfarosh',
'Lagaan: Once Upon a Time in India',
'Ustad Hotel',
'Hera Pheri',
'Baasha',
'Barfi!',
'Udaan',
'The Legend of Bhagat Singh',
'K.G.F: Chapter 1',
'Sholay',
'Angoor',
'Baahubali 2: The Conclusion',
'Maheshinte Prathikaaram']
```

In [12]:
```
#
```

Make DataFrame

Import Pandas

In [15]:
```
import pandas as pd
```

In [20]:
```
df=pd.DataFrame({'Name of Movie': movie_name, 'Year': Year, 'Ratings': Ratings})
df
```

Out[20]:

	Name of Movie	Year	Ratings
0	Jai Bhim	(2021)	8.4
1	Anbe Sivam	(2003)	8.4

	Name of Movie	Year	Ratings
2	Pariyerum Perumal	(2018)	8.4
3	Golmaal	(1979)	8.4
4	Nayakan	(1987)	8.4
...	...	...	...
95	K.G.F: Chapter 1	(2018)	8.0
96	Sholay	(1975)	8.0
97	Angoor	(1982)	8.0
98	Baahubali 2: The Conclusion	(2017)	8.0
99	Maheshinte Prathikaaram	(2016)	8.0

100 rows × 3 columns

In [ ]:

Program to display IMDBS Top Rated 100 Movies

Import Libraries

In [18]:
```
from bs4 import BeautifulSoup
import requests
```

Send Get request from Web page

In [19]:
```
url= requests.get('https://www.imdb.com/search/title/?groups=top_100&sort=user_rating,des
url
```

Out[19]:
```
<Response [200]>
```

Page Content

In [20]:
```
request = url.text
```

Scrapping header

In [ ]:
```
soup_data= BeautifulSoup(request, 'html.parser')
soup_data
```

In [22]:
```
soup_data.title.text
```

Out[22]:
```
'IMDb "Top 100"\n(Sorted by IMDb Rating Descending) - IMDb'
```

In [23]:
```
movies = soup_data.findAll('div', {'class': 'lister-item mode-advanced'})
```

In [24]:
```
first_movie = movies[0]
```

In [25]:
```
#Name of the movie
first_movie.h3.a.text
```

Out[25]:
```
'The Shawshank Redemption'
```

In [26]:
```
#Year Of Release
first_movie.find('span',{'class':"lister-item-year text-muted unbold"}).text
```

Out[26]:
```
'(1994)'
```

In [27]:
```
#Ratings
first_movie.find('div',{'class':"inline-block ratings-imdb-rating"})['data-value']
```

Out[27]:
```
'9.3'
```

In [28]:
```
#MetaScore
first_movie.find('div',{'class':"inline-block ratings-metascore"}).span.text.strip()
```

Out[28]:
```
'81'
```

In [29]:
```
#No of votes
first_movie.find('span',{'name':'nv'})['data-value']
```

Out[29]:
```
'2567666'
```

In [30]:
```
Name=[]
Year=[]
Ratings=[]
Vote=[]

for i in movies:
    Name.append(i.h3.a.text)
    Year.append(i.find('span',{'class':"lister-item-year text-muted unbold"}).text)
    Ratings.append(i.find('div',{'class':"inline-block ratings-imdb-rating"})['data-value
    Vote.append(i.find('span',{'name':'nv'})['data-value'])
```

In [31]:
```
data= list(zip(Name,Year,Ratings,Vote))
```

In [32]:
```
import pandas as pd
```

In [33]:
```
df=pd.DataFrame(data, columns=['Name','Year','Ratings','Vote'])
df
```

Out[33]:

	Name	Year	Ratings	Vote
0	The Shawshank Redemption	(1994)	9.3	2567666
1	The Godfather	(1972)	9.2	1767603
2	The Dark Knight	(2008)	9.1	2531971
3	The Lord of the Rings: The Return of the King	(2003)	9	1766578
4	Schindler's List	(1993)	9	1308384
5	The Godfather: Part II	(1974)	9	1223014
6	12 Angry Men	(1957)	9	758329
7	The Lord of the Rings: The Fellowship of the Ring	(2001)	8.9	1788319
8	Pulp Fiction	(1994)	8.9	1971142
9	Inception	(2010)	8.8	2253500
10	The Lord of the Rings: The Two Towers	(2002)	8.8	1595994
11	Fight Club	(1999)	8.8	2020430
12	Forrest Gump	(1994)	8.8	1980529

	Name	Year	Ratings	Vote
13	Il buono, il brutto, il cattivo	(1966)	8.8	738587
14	Interstellar	(2014)	8.7	1711562
15	The Matrix	(1999)	8.7	1848216
16	Goodfellas	(1990)	8.7	1108035
17	The Empire Strikes Back	(1980)	8.7	1242389
18	One Flew Over the Cuckoo's Nest	(1975)	8.7	978272
19	Shichinin no samurai	(1954)	8.7	337342
20	It's a Wonderful Life	(1946)	8.7	442453
21	Cidade de Deus	(2002)	8.6	737962
22	The Pianist	(2002)	8.6	799926
23	Sen to Chihiro no kamikakushi	(2001)	8.6	725435
24	Saving Private Ryan	(1998)	8.6	1337633
25	The Green Mile	(1999)	8.6	1248504
26	La vita è bella	(1997)	8.6	672672
27	Se7en	(1995)	8.6	1574369
28	Léon	(1994)	8.6	1119434
29	Terminator 2: Judgment Day	(1991)	8.6	1062365
30	The Silence of the Lambs	(1991)	8.6	1376793
31	Back to the Future	(1985)	8.6	1152572
32	Star Wars	(1977)	8.6	1313902
33	Seppuku	(1962)	8.6	53129
34	Gisaengchung	(2019)	8.5	730565
35	Avengers: Infinity War	(2018)	8.5	997932
36	Whiplash	(2014)	8.5	800457
37	Django Unchained	(2012)	8.5	1483705
38	The Intouchables	(2011)	8.5	825937
39	3 Idiots	(2009)	8.5	382774
40	Spider-Man: No Way Home	(2021)	8.5	562905
41	The Prestige	(2006)	8.5	1286134
42	The Departed	(2006)	8.5	1278221
43	Memento	(2000)	8.5	1201082
44	Gladiator	(2000)	8.5	1444925
45	American History X	(1998)	8.5	1091500
46	The Usual Suspects	(1995)	8.5	1055733
47	The Lion King	(1994)	8.5	1017173
48	Nuovo Cinema Paradiso	(1988)	8.5	252991
49	Hotaru no haka	(1988)	8.5	264324

In [ ]:

Program to display all header tags from WIKIPEDIA.ORG

Import Libraries

In [1]:
```
from bs4 import BeautifulSoup
import requests
```

Send Get request from Web page

In [2]:
```
page= requests.get('https://en.wikipedia.org/wiki/Main_Page')
page
```

Out[2]:
```
<Response [200]>
```

Page Content

In [ ]:
```
Soup=BeautifulSoup(page.content)
Soup
```

Scrapping header

In [4]:
```
titles=[]

for i in Soup.find_all('span', class_="mw-headline"):
    titles.append(i.text)


titles
```

Out[4]:
```
['Welcome to Wikipedia',
 "From today's featured article",
 'Did you know\xa0...',
 'In the news',
 'On this day',
 "From today's featured list",
 "Today's featured picture",
 'Other areas of Wikipedia',
 "Wikipedia's sister projects",
 'Wikipedia languages']
```

Make DataFrame

In [5]:
```
import pandas as pd
```

In [6]:
```
df= pd.DataFrame({'Titles': titles})
df
```

Out[6]:

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

In [ ]: