In all the following questions, you have to use BeautifulSoup to scrape different websites and collect data as per the requirement of the question.

Every answer to the question should be in form of a python function which should take URL as the parameter. Use Jupyter Notebooks to program, upload it on your GitHub and send the link of the Jupyter notebook to your SME.

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

Answer:
```
import requests
from bs4 import BeautifulSoup
import pandas as pd

# Fetch the HTML content of the page
url = "https://en.wikipedia.org/wiki/Main_Page"
response = requests.get(url)
html_content = response.text

# Parse the HTML content using BeautifulSoup
soup = BeautifulSoup(html_content, "html.parser")

# Find all header tags (h1, h2, h3, h4, h5, h6)
header_tags = soup.find_all(["h1", "h2", "h3", "h4", "h5", "h6"])

# Extract the text from header tags
header_text = [tag.text.strip() for tag in header_tags]

# Create a DataFrame from the extracted data
df = pd.DataFrame({"Header Tags": header_text})

# Display the DataFrame
print(df)
```

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 and make data frame.

Answer:

```
import requests
from bs4 import BeautifulSoup
import pandas as pd

# Fetch the HTML content of the page
url = "https://presidentofindia.nic.in/former-presidents.htm"
response = requests.get(url)
html_content = response.text

# Parse the HTML content using BeautifulSoup
soup = BeautifulSoup(html_content, "html.parser")

# Find the table containing the former presidents' information
table = soup.find("table", {"class": "table-striped"})

# Initialize lists to store the data
names = []
terms = []

# Extract the names and terms of office from the table
rows = table.find_all("tr")
for row in rows[1:]:    # Skip the first row as it contains table headers
    columns = row.find_all("td")
    names.append(columns[0].text.strip())
    terms.append(columns[1].text.strip())

# Create a DataFrame from the extracted data
```

```
df = pd.DataFrame({"Name": names, "Term of Office": terms})


# Display the DataFrame

print(df)
```

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.

Answer:

```
import requests

from bs4 import BeautifulSoup

import pandas as pd


# Fetch the HTML content of the page

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

response = requests.get(url)

html_content = response.text


# Parse the HTML content using BeautifulSoup

soup = BeautifulSoup(html_content, "html.parser")


# Find the table containing the team rankings

table = soup.find("table", {"class": "table"})

table_body = table.find("tbody")


# Initialize lists to store the data

teams = []

matches = []

points = []

ratings = []
```

```
# Extract the data from the table
rows = table_body.find_all("tr")
for row in rows:
    columns = row.find_all("td")
    team = columns[1].text.strip()
    match = columns[2].text.strip()
    point = columns[3].text.strip()
    rating = columns[4].text.strip()

    teams.append(team)
    matches.append(match)
    points.append(point)
    ratings.append(rating)

# Create a DataFrame from the extracted data
df = pd.DataFrame({
    "Team": teams,
    "Matches": matches,
    "Points": points,
    "Rating": ratings
})

# Keep only the top 10 teams
df = df.head(10)

# Display the DataFrame
print(df)
```

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

Answer: import requests

```
from bs4 import BeautifulSoup
import pandas as pd

# Fetch the HTML content of the page
url = "https://www.icc-cricket.com/rankings/mens/player-rankings/odi/batting"
response = requests.get(url)
html_content = response.text

# Parse the HTML content using BeautifulSoup
soup = BeautifulSoup(html_content, "html.parser")

# Find the table containing the batsmen rankings
table = soup.find("table", {"class": "table"})
table_body = table.find("tbody")

# Initialize lists to store the data
batsmen = []
teams = []
ratings = []

# Extract the data from the table
rows = table_body.find_all("tr")
for row in rows[:10]:    # Limit to top 10 batsmen
    columns = row.find_all("td")
    batsman = columns[1].text.strip()
    team = columns[2].text.strip()
    rating = columns[3].text.strip()

    batsmen.append(batsman)
```

```
        teams.append(team)
        ratings.append(rating)

# Create a DataFrame from the extracted data
df = pd.DataFrame({
        "Batsman": batsmen,
        "Team": teams,
        "Rating": ratings
})

# Display the DataFrame
print(df)
```

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

Answer:
```
import requests
from bs4 import BeautifulSoup
import pandas as pd

# Fetch the HTML content of the page
url = "https://www.icc-cricket.com/rankings/mens/player-rankings/odi/bowling"
response = requests.get(url)
html_content = response.text

# Parse the HTML content using BeautifulSoup
soup = BeautifulSoup(html_content, "html.parser")

# Find the table containing the bowlers rankings
table = soup.find("table", {"class": "table"})
table_body = table.find("tbody")
```

```
# Initialize lists to store the data
bowlers = []
teams = []
ratings = []

# Extract the data from the table
rows = table_body.find_all("tr")
for row in rows[:10]:    # Limit to top 10 bowlers
    columns = row.find_all("td")
    bowler = columns[1].text.strip()
    team = columns[2].text.strip()
    rating = columns[3].text.strip()

    bowlers.append(bowler)
    teams.append(team)
    ratings.append(rating)

# Create a DataFrame from the extracted data
df = pd.DataFrame({
    "Bowler": bowlers,
    "Team": teams,
    "Rating": ratings
})

# Display the DataFrame
print(df)
```

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

i. Top 10 ODI teams in women's cricket along with the records for matches, points and rating.

Answer:

```
import requests
from bs4 import BeautifulSoup
import pandas as pd

# Fetch the HTML content of the page
url = "https://www.icc-cricket.com/rankings/womens/team-rankings/odi"
response = requests.get(url)
html_content = response.text

# Parse the HTML content using BeautifulSoup
soup = BeautifulSoup(html_content, "html.parser")

# Find the table containing the team rankings
table = soup.find("table", {"class": "table"})
table_body = table.find("tbody")

# Initialize lists to store the data
teams = []
matches = []
points = []
ratings = []

# Extract the data from the table
rows = table_body.find_all("tr")
for row in rows:
    columns = row.find_all("td")
    team = columns[1].text.strip()
```

```
        match = columns[2].text.strip()
        point = columns[3].text.strip()
        rating = columns[4].text.strip()

        teams.append(team)
        matches.append(match)
        points.append(point)
        ratings.append(rating)

# Create a DataFrame from the extracted data
df = pd.DataFrame({
        "Team": teams,
        "Matches": matches,
        "Points": points,
        "Rating": ratings
})

# Keep only the top 10 teams
df = df.head(10)

# Display the DataFrame
print(df)
```

ii. Top 10 women's ODI Batting players along with the records of their team and rating.

Answer:

```
import requests
from bs4 import BeautifulSoup
import pandas as pd

# Fetch the HTML content of the page
url = "https://www.icc-cricket.com/rankings/womens/player-rankings/odi/batting"
```

```
response = requests.get(url)
html_content = response.text

# Parse the HTML content using BeautifulSoup
soup = BeautifulSoup(html_content, "html.parser")

# Find the table containing the batting players rankings
table = soup.find("table", {"class": "table"})
table_body = table.find("tbody")

# Initialize lists to store the data
players = []
teams = []
ratings = []

# Extract the data from the table
rows = table_body.find_all("tr")
for row in rows[:10]:    # Limit to top 10 players
    columns = row.find_all("td")
    player = columns[1].text.strip()
    team = columns[2].text.strip()
    rating = columns[3].text.strip()

    players.append(player)
    teams.append(team)
    ratings.append(rating)

# Create a DataFrame from the extracted data
df = pd.DataFrame({
    "Player": players,
    "Team": teams,
    "Rating": ratings
})

# Display the DataFrame
print(df)
```

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

Answer:

```
import requests
from bs4 import BeautifulSoup
import pandas as pd

# Fetch the HTML content of the page
url = "https://www.icc-cricket.com/rankings/womens/player-rankings/odi/all-rounder"
response = requests.get(url)
html_content = response.text

# Parse the HTML content using BeautifulSoup
soup = BeautifulSoup(html_content, "html.parser")

# Find the table containing the all-rounders rankings
table = soup.find("table", {"class": "table"})
table_body = table.find("tbody")

# Initialize lists to store the data
players = []
teams = []
ratings = []
```

```
# Extract the data from the table
rows = table_body.find_all("tr")
for row in rows[:10]:    # Limit to top 10 all-rounders
    columns = row.find_all("td")
    player = columns[1].text.strip()
    team = columns[2].text.strip()
    rating = columns[3].text.strip()

    players.append(player)
    teams.append(team)
    ratings.append(rating)

# Create a DataFrame from the extracted data
df = pd.DataFrame({
    "Player": players,
    "Team": teams,
    "Rating": ratings
})

# Display the DataFrame
print(df)
```

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

Answer:

```
import requests
from bs4 import BeautifulSoup
import pandas as pd

# Fetch the HTML content of the page
```

```
url = "https://www.cnbc.com/world/?region=world"
response = requests.get(url)
html_content = response.text

# Parse the HTML content using BeautifulSoup
soup = BeautifulSoup(html_content, "html.parser")

# Find the news headline elements
headline_elements = soup.find_all("a", {"class": "Card-titleLink"})
headlines = [element.text.strip() for element in headline_elements]

# Create a DataFrame from the extracted data
df = pd.DataFrame({
    "Headline": headlines
})

# Display the DataFrame
print(df)
```

ii. Time

Answer:
```
import requests
from bs4 import BeautifulSoup
import pandas as pd

# Fetch the HTML content of the page
url = "https://www.cnbc.com/world/?region=world"
response = requests.get(url)
html_content = response.text

# Parse the HTML content using BeautifulSoup
```

```
soup = BeautifulSoup(html_content, "html.parser")

# Find the news elements
news_elements = soup.find_all("div", {"class": "Card-title"})
headlines = []
times = []

# Extract the headline and time from each news element
for element in news_elements:
    headline = element.find("a", {"class": "Card-titleLink"}).text.strip()
    time = element.find("time").text.strip()

    headlines.append(headline)
    times.append(time)

# Create a DataFrame from the extracted data
df = pd.DataFrame({
    "Headline": headlines,
    "Time": times
})

# Display the DataFrame
print(df)
```

iii. News Link

Answer:
```
import requests
from bs4 import BeautifulSoup
import pandas as pd

# Fetch the HTML content of the page
```

```
url = "https://www.cnbc.com/world/?region=world"

response = requests.get(url)

html_content = response.text


# Parse the HTML content using BeautifulSoup

soup = BeautifulSoup(html_content, "html.parser")


# Find the news elements

news_elements = soup.find_all("div", {"class": "Card-title"})

headlines = []

times = []

news_links = []


# Extract the headline, time, and news link from each news element

for element in news_elements:

    headline = element.find("a", {"class": "Card-titleLink"}).text.strip()

    time = element.find("time").text.strip()

    news_link = element.find("a", {"class": "Card-titleLink"})["href"]


    headlines.append(headline)

    times.append(time)

    news_links.append(news_link)


# Create a DataFrame from the extracted data

df = pd.DataFrame({

    "Headline": headlines,

    "Time": times,

    "News Link": news_links

})
```

# Display the DataFrame

print(df)

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

Answer: import requests

from bs4 import BeautifulSoup

import pandas as pd

# Fetch the HTML content of the page

url = "https://www.journals.elsevier.com/artificial-intelligence/most-downloaded-articles"

response = requests.get(url)

html_content = response.text

# Parse the HTML content using BeautifulSoup

soup = BeautifulSoup(html_content, "html.parser")

# Find the article elements

article_elements = soup.find_all("div", {"class": "pod-listing-header"})

# Initialize lists to store the data

paper_titles = []

# Extract the paper title from each article element

for element in article_elements:

```
    title = element.find("a").text.strip()

    paper_titles.append(title)
```

```
# Create a DataFrame from the extracted data
df = pd.DataFrame({
    "Paper Title": paper_titles
})

# Display the DataFrame
print(df)
```

ii. Authors

Answer:
```
import requests
from bs4 import BeautifulSoup
import pandas as pd

# Fetch the HTML content of the page
url = "https://www.journals.elsevier.com/artificial-intelligence/most-downloaded-articles"
response = requests.get(url)
html_content = response.text

# Parse the HTML content using BeautifulSoup
soup = BeautifulSoup(html_content, "html.parser")

# Find the article elements
article_elements = soup.find_all("div", {"class": "pod-listing-header"})

# Initialize lists to store the data
paper_titles = []
authors_list = []

# Extract the paper title and authors from each article element
```

```
for element in article_elements:
    title = element.find("a").text.strip()
    paper_titles.append(title)

    authors = []
    authors_elements = element.find_all("span", {"class": "author"})
    for author_element in authors_elements:
        author = author_element.text.strip()
        authors.append(author)

    authors_list.append(authors)

# Create a DataFrame from the extracted data
df = pd.DataFrame({
    "Paper Title": paper_titles,
    "Authors": authors_list
})

# Display the DataFrame
print(df)
```

iii. Published Date

Answer:

```
import requests
from bs4 import BeautifulSoup
import pandas as pd

# Fetch the HTML content of the page
url = "https://www.journals.elsevier.com/artificial-intelligence/most-downloaded-articles"
response = requests.get(url)
html_content = response.text
```

```
# Parse the HTML content using BeautifulSoup
soup = BeautifulSoup(html_content, "html.parser")

# Find the article elements
article_elements = soup.find_all("div", {"class": "pod-listing-header"})
date_elements = soup.find_all("div", {"class": "pod-listing-header__text"})

# Initialize lists to store the data
paper_titles = []
published_dates = []

# Extract the paper title and published date from each article element
for i in range(len(article_elements)):
    title = article_elements[i].find("a").text.strip()
    paper_titles.append(title)

    date = date_elements[i].find("span", {"class": "pod-listing-header__text_date"}).text.strip()
    published_dates.append(date)

# Create a DataFrame from the extracted data
df = pd.DataFrame({
    "Paper Title": paper_titles,
    "Published Date": published_dates
})

# Display the DataFrame
print(df)
```

iv. Paper URL

Answer: import requests

```
from bs4 import BeautifulSoup

import pandas as pd


# Fetch the HTML content of the page

url = "https://www.journals.elsevier.com/artificial-intelligence/most-downloaded-articles"

response = requests.get(url)

html_content = response.text


# Parse the HTML content using BeautifulSoup

soup = BeautifulSoup(html_content, "html.parser")


# Find the article elements

article_elements = soup.find_all("div", {"class": "pod-listing-header"})


# Initialize lists to store the data

paper_titles = []

paper_urls = []


# Extract the paper title and paper URL from each article element

for element in article_elements:

    title = element.find("a").text.strip()

    paper_titles.append(title)


    url = element.find("a")["href"]

    paper_urls.append(url)


# Create a DataFrame from the extracted data

df = pd.DataFrame({
    "Paper Title": paper_titles,
    "Paper URL": paper_urls
})

# Display the DataFrame
print(df)
```

7) Write a python program to scrape mentioned details from dineout.co.inand make data frame-
   i. Restaurant name

Answer:
```
import requests
from bs4 import BeautifulSoup
import pandas as pd

# Fetch the HTML content of the page
url = "https://www.dineout.co.in/delhi-restaurants"
response = requests.get(url)
html_content = response.text

# Parse the HTML content using BeautifulSoup
soup = BeautifulSoup(html_content, "html.parser")

# Find the restaurant elements
restaurant_elements = soup.find_all("div", {"class": "restnt-info"})
restaurant_names = []

# Extract the restaurant names from each restaurant element
for element in restaurant_elements:
    name = element.find("div", {"class": "restnt-info-top clearfix"}).find("a").text.strip()
    restaurant_names.append(name)
```

```
# Create a DataFrame from the extracted data
df = pd.DataFrame({
    "Restaurant Name": restaurant_names
})

# Display the DataFrame
print(df)
```

ii. Cuisine

```
Answer: import requests
from bs4 import BeautifulSoup
import pandas as pd

# Fetch the HTML content of the page
url = "https://www.dineout.co.in/delhi-restaurants"
response = requests.get(url)
html_content = response.text

# Parse the HTML content using BeautifulSoup
soup = BeautifulSoup(html_content, "html.parser")

# Find the restaurant elements
restaurant_elements = soup.find_all("div", {"class": "restnt-info"})
restaurant_cuisines = []

# Extract the cuisine details from each restaurant element
for element in restaurant_elements:
    cuisine = element.find("div", {"class": "restnt-info-cuisine"}).text.strip()
    restaurant_cuisines.append(cuisine)
```

```
# Create a DataFrame from the extracted data
df = pd.DataFrame({
    "Cuisine": restaurant_cuisines
})

# Display the DataFrame
print(df)
```

iii. Location

Answer:
```
import requests
from bs4 import BeautifulSoup
import pandas as pd

# Fetch the HTML content of the page
url = "https://www.dineout.co.in/delhi-restaurants"
response = requests.get(url)
html_content = response.text

# Parse the HTML content using BeautifulSoup
soup = BeautifulSoup(html_content, "html.parser")

# Find the restaurant elements
restaurant_elements = soup.find_all("div", {"class": "restnt-info"})
restaurant_locations = []

# Extract the location details from each restaurant element
for element in restaurant_elements:
    location = element.find("div", {"class": "restnt-info-location"}).text.strip()
    restaurant_locations.append(location)
```

```
# Create a DataFrame from the extracted data
df = pd.DataFrame({
    "Location": restaurant_locations
})

# Display the DataFrame
print(df)
```

iv. Ratings

Answer: import requests

```
from bs4 import BeautifulSoup
import pandas as pd

# Fetch the HTML content of the page
url = "https://www.dineout.co.in/delhi-restaurants"
response = requests.get(url)
html_content = response.text

# Parse the HTML content using BeautifulSoup
soup = BeautifulSoup(html_content, "html.parser")

# Find the restaurant elements
restaurant_elements = soup.find_all("div", {"class": "restnt-info"})
restaurant_ratings = []

# Extract the ratings from each restaurant element
for element in restaurant_elements:
    rating = element.find("span", {"class": "rating"}).text.strip()
    restaurant_ratings.append(rating)
```

```
# Create a DataFrame from the extracted data
df = pd.DataFrame({
    "Ratings": restaurant_ratings
})

# Display the DataFrame
print(df)
```

v. Image URL

Answer:
```
import requests
from bs4 import BeautifulSoup
import pandas as pd

# Fetch the HTML content of the page
url = "https://www.dineout.co.in/delhi-restaurants"
response = requests.get(url)
html_content = response.text

# Parse the HTML content using BeautifulSoup
soup = BeautifulSoup(html_content, "html.parser")

# Find the restaurant elements
restaurant_elements = soup.find_all("div", {"class": "restnt-info"})
restaurant_image_urls = []

# Extract the image URLs from each restaurant element
for element in restaurant_elements:
    image_url = element.find("img", {"class": "img-responsive"})["src"]
    restaurant_image_urls.append(image_url)
```

```
# Create a DataFrame from the extracted data
df = pd.DataFrame({
    "Image URL": restaurant_image_urls
})

# Display the DataFrame
print(df)
```