WEB SCRAPING – ASSIGNMENT 4

1. Scrape the details of most viewed videos on YouTube from Wikipedia. Url = https://en.wikipedia.org/wiki/List_of_most-viewed_YouTube_videos You need to find following details: A) Rank
B) Name
C) Artist
D) Upload date
E) Views

Solution:

```
import requests
from bs4 import BeautifulSoup

url = "https://en.wikipedia.org/wiki/List_of_most-viewed_YouTube_videos"

# Send a GET request to the URL
response = requests.get(url)

# Create a BeautifulSoup object to parse the HTML content
soup = BeautifulSoup(response.text, "html.parser")

# Find the table containing the video details
table = soup.find("table", class_="wikitable sortable")

# Initialize lists to store the details
rank_list = []
name_list = []
artist_list = []
upload_date_list = []
views_list = []

# Iterate over each row in the table, skipping the header row
for row in table.find_all("tr")[1:]:
    # Find the cells in the row
```

```
    cells = row.find_all("td")

    # Extract the required details from the cells
    rank = cells[0].text.strip()
    name = cells[1].text.strip()
    artist = cells[2].text.strip()
    upload_date = cells[4].text.strip()
    views = cells[5].text.strip()

    # Append the details to the respective lists
    rank_list.append(rank)
    name_list.append(name)
    artist_list.append(artist)
    upload_date_list.append(upload_date)
    views_list.append(views)

# Print the details of the most viewed videos
for i in range(len(rank_list)):
    print(f"Rank: {rank_list[i]}")
    print(f"Name: {name_list[i]}")
    print(f"Artist: {artist_list[i]}")
    print(f"Upload Date: {upload_date_list[i]}")
    print(f"Views: {views_list[i]}")
    print()
```

2. Scrape the details teamIndia'sinternationalfixtures from bcci.tv. Url = https://www.bcci.tv/. You need to find following details:

A) Match title (I.e. 1stODI)
B) Series
C) Place
D) Date
E) Time

Note: - From bcci.tv home page you have reach to the international fixture page through code

Solution: import requests

from bs4 import BeautifulSoup

```
url = "https://www.bcci.tv/"

# Send a GET request to the URL
response = requests.get(url)

# Create a BeautifulSoup object to parse the HTML content
soup = BeautifulSoup(response.text, "html.parser")

# Find the "International" dropdown menu on the homepage
dropdown = soup.find("li", class_="navigation__item--en-navigation-item--3d1Ty")

# Find the URL of the international fixtures page
fixtures_url = dropdown.find("a", text="Fixtures")["href"]

# Construct the full URL of the fixtures page
fixtures_full_url = url.rstrip("/") + fixtures_url

# Send a GET request to the fixtures URL
fixtures_response = requests.get(fixtures_full_url)

# Create a BeautifulSoup object for the fixtures page
fixtures_soup = BeautifulSoup(fixtures_response.text, "html.parser")

# Find the container element that holds the fixture details
container = fixtures_soup.find("div", class_="js-list")

# Find all the fixture items
```

```
fixture_items = container.find_all("div", class_="fixture__format")


# Iterate over each fixture item and extract the required details
for item in fixture_items:
    match_title = item.find("div", class_="fixture__format-strip").text.strip()
    series = item.find("span", class_="u-unskewed-text fixture__tournament-label").text.strip()
    place = item.find("p", class_="fixture__additional-info").text.strip()
    date = item.find("span", class_="fixture__date").text.strip()
    time = item.find("span", class_="fixture__time").text.strip()


    print("Match Title:", match_title)
    print("Series:", series)
    print("Place:", place)
    print("Date:", date)
    print("Time:", time)
    print()
```

3. Scrape the details of State-wise GDP ofIndia fromstatisticstime.com. Url = http://statisticstimes.com/ You have to find following details:
A) Rank
B) State
C) GSDP(18-19)- at current prices
D) GSDP(19-20)- at current prices
E) Share(18-19)
F) GDP($ billion)
Note: - From statisticstimes home page you have to reach to economy page through code

Solution:

```
import requests
from bs4 import BeautifulSoup


url = "http://statisticstimes.com/"


# Send a GET request to the URL
response = requests.get(url)
```

```
# Create a BeautifulSoup object to parse the HTML content
soup = BeautifulSoup(response.text, "html.parser")

# Find the "Economy" dropdown menu on the homepage
dropdown = soup.find("li", class_="no-arrow mega-dropdown")
dropdown_link = dropdown.find("a", class_="dropdown-toggle")
economy_url = dropdown_link["href"]

# Construct the full URL of the economy page
economy_full_url = url.rstrip("/") + economy_url

# Send a GET request to the economy URL
economy_response = requests.get(economy_full_url)

# Create a BeautifulSoup object for the economy page
economy_soup = BeautifulSoup(economy_response.text, "html.parser")

# Find the "India" section on the economy page
india_section = economy_soup.find("div", class_="col-md-12")

# Find the URL of the state-wise GDP page
gdp_url = india_section.find("a", text="GDP of Indian states").get("href")

# Construct the full URL of the state-wise GDP page
gdp_full_url = url.rstrip("/") + gdp_url

# Send a GET request to the state-wise GDP URL
gdp_response = requests.get(gdp_full_url)

# Create a BeautifulSoup object for the state-wise GDP page
gdp_soup = BeautifulSoup(gdp_response.text, "html.parser")
```

```
# Find the table containing the GDP details
table = gdp_soup.find("table", class_="display dataTable")

# Initialize lists to store the details
rank_list = []
state_list = []
gsdp_18_19_list = []
gsdp_19_20_list = []
share_18_19_list = []
gdp_billion_list = []

# Iterate over each row in the table, skipping the header row
for row in table.find_all("tr")[1:]:
    # Find the cells in the row
    cells = row.find_all("td")

    # Extract the required details from the cells
    rank = cells[0].text.strip()
    state = cells[1].text.strip()
    gsdp_18_19 = cells[2].text.strip()
    gsdp_19_20 = cells[3].text.strip()
    share_18_19 = cells[4].text.strip()
    gdp_billion = cells[5].text.strip()

    # Append the details to the respective lists
    rank_list.append(rank)
    state_list.append(state)
    gsdp_18_19_list.append(gsdp_18_19)
    gsdp_19_20_list.append(gsdp_19_20)
    share_18_19_list.append(share_18_19)
```

```
    gdp_billion_list.append(gdp_billion)


# Print the details of state-wise GDP
for i in range(len(rank_list)):
    print(f"Rank: {rank_list[i]}")
    print(f"State: {state_list[i]}")
    print(f"GSDP (18-19) - at current prices: {gsdp_18_19_list[i]}")
    print(f"GSDP (19-20) - at current prices: {gsdp_19_20_list[i]}")
    print(f"Share (18-19): {share_18_19_list[i]}")
    print(f"GDP ($
```

4. Scrape the details of trending repositories on Github.com. Url = https://github.com/ You have to find the following details:
A) Repository title
B) Repository description
C) Contributors count
D) Language used

Solution:

```
from selenium import webdriver
from bs4 import BeautifulSoup

url = "https://github.com/"

# Set up the Selenium webdriver (make sure to have the appropriate driver installed)
driver = webdriver.Chrome("path_to_chromedriver")  # Replace "path_to_chromedriver" with the actual path

# Open the URL using Selenium
driver.get(url)

# Get the page source after the dynamic content has loaded
page_source = driver.page_source

# Create a BeautifulSoup object to parse the HTML content
```

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

# Find the container element that holds the trending repositories
container = soup.find("ol", class_="repo-list")

# Find all the repository items
repository_items = container.find_all("li")

# Iterate over each repository item and extract the required details
for item in repository_items:
    title = item.find("h3").text.strip()
    description = item.find("p", class_="mb-1").text.strip()
    contributors_count = item.find("a", href=lambda href: href and "/network/members" in href).text.strip()
    language = item.find("span", itemprop="programmingLanguage").text.strip()

    print("Repository Title:", title)
    print("Repository Description:", description)
    print("Contributors Count:", contributors_count)
    print("Language Used:", language)
    print()

# Close the Selenium webdriver
driver.quit()
```

(Note: Make sure to replace `"path_to_chromedriver"` with the actual path to the Chrome WebDriver executable file, which you can download from the official Selenium website (https://sites.google.com/a/chromium.org/chromedriver/downloads).

5. Scrape the details of top 100 songs on billiboard.com. Url = https:/www.billboard.com/ You have to find the following details:
A) Song name
B) Artist name
C) Last week rank

D) Peak rank
E) Weeks on board
Note: - From the home page you have to click on the charts option then hot 100-page link through code.

Solution: import requests

from bs4 import BeautifulSoup

```
url = "https://www.billboard.com/"

# Send a GET request to the URL
response = requests.get(url)

# Create a BeautifulSoup object to parse the HTML content
soup = BeautifulSoup(response.text, "html.parser")

# Find the "Charts" option on the homepage
charts_option = soup.find("a", class_="header__main-link", text="Charts")

# Find the URL of the Hot 100 page
hot_100_url = charts_option["href"]

# Construct the full URL of the Hot 100 page
hot_100_full_url = url.rstrip("/") + hot_100_url

# Send a GET request to the Hot 100 URL
hot_100_response = requests.get(hot_100_full_url)

# Create a BeautifulSoup object for the Hot 100 page
hot_100_soup = BeautifulSoup(hot_100_response.text, "html.parser")

# Find the container element that holds the song details
container = hot_100_soup.find("ol", class_="chart-list")
```

```
# Find all the song items
song_items = container.find_all("li", class_="chart-list__element")

# Iterate over each song item and extract the required details
for item in song_items:
    song_name = item.find("span", class_="chart-element__information__song").text.strip()
    artist_name = item.find("span", class_="chart-element__information__artist").text.strip()
    last_week_rank = item.find("span", class_="chart-element__meta text--last").text.strip()
    peak_rank = item.find("span", class_="chart-element__meta text--peak").text.strip()
    weeks_on_board = item.find("span", class_="chart-element__meta text--week").text.strip()

    print("Song Name:", song_name)
    print("Artist Name:", artist_name)
    print("Last Week Rank:", last_week_rank)
    print("Peak Rank:", peak_rank)
    print("Weeks on Board:", weeks_on_board)
    print()
```

6. Scrape the details of Highest sellingnovels. Url = https://www.theguardian.com/news/datablog/2012/aug/09/best-selling-books-all-time-fifty-shades-greycompare You have to find the following details:
A) Book name
B) Author name
C) Volumes sold
D) Publisher
E) Genre

Solution:

```
import requests
from bs4 import BeautifulSoup

url = "https://www.theguardian.com/news/datablog/2012/aug/09/best-selling-books-all-time-fifty-shades-grey-compare"

# Send a GET request to the URL
```

```
response = requests.get(url)

# Create a BeautifulSoup object to parse the HTML content
soup = BeautifulSoup(response.text, "html.parser")

# Find the table containing the book details
table = soup.find("table", class_="in-article sortable")

# Initialize lists to store the details
book_name_list = []
author_name_list = []
volumes_sold_list = []
publisher_list = []
genre_list = []

# Iterate over each row in the table, skipping the header row
for row in table.find_all("tr")[1:]:
    # Find the cells in the row
    cells = row.find_all("td")

    # Extract the required details from the cells
    book_name = cells[1].text.strip()
    author_name = cells[2].text.strip()
    volumes_sold = cells[3].text.strip()
    publisher = cells[4].text.strip()
    genre = cells[5].text.strip()

    # Append the details to the respective lists
    book_name_list.append(book_name)
    author_name_list.append(author_name)
    volumes_sold_list.append(volumes_sold)
```

```
    publisher_list.append(publisher)
    genre_list.append(genre)

# Print the details of the highest-selling novels
for i in range(len(book_name_list)):
    print(f"Book Name: {book_name_list[i]}")
    print(f"Author Name: {author_name_list[i]}")
    print(f"Volumes Sold: {volumes_sold_list[i]}")
    print(f"Publisher: {publisher_list[i]}")
    print(f"Genre: {genre_list[i]}")
    print()
```

7. Scrape the details most watched tv series of all time from imdb.com. Url = https://www.imdb.com/list/ls095964455/ You have to find the following details:
A) Name
B) Year span
C) Genre
D) Run time
E) Ratings
F) Votes

Solution:

```
import requests
from bs4 import BeautifulSoup

url = "https://www.imdb.com/list/ls095964455/"

# Send a GET request to the URL
response = requests.get(url)

# Create a BeautifulSoup object to parse the HTML content
soup = BeautifulSoup(response.text, "html.parser")

# Find the container that holds the TV series details
container = soup.find("div", class_="lister-list")
```

```
# Find all the TV series items
tv_series_items = container.find_all("div", class_="lister-item-content")

# Iterate over each TV series item and extract the required details
for item in tv_series_items:
    name = item.find("h3", class_="lister-item-header").a.text.strip()
    year_span = item.find("span", class_="lister-item-year").text.strip("()")

    genre = item.find("span", class_="genre").text.strip()
    runtime = item.find("span", class_="runtime").text.strip()
    ratings = item.find("div", class_="ipl-rating-star").span.text.strip()
    votes = item.find("span", attrs={"name": "nv"}).text.strip()

    print("Name:", name)
    print("Year Span:", year_span)
    print("Genre:", genre)
    print("Run Time:", runtime)
    print("Ratings:", ratings)
    print("Votes:", votes)
    print()
```

8. Details of Datasetsfrom UCI machine learning repositories. Url = https://archive.ics.uci.edu/ You have to find the following details:
A) Dataset name
B) Data type
C) Task
D) Attribute type
E) No of instances
F) No of attribute
G) Year
Note: - from the home page you have to go to the ShowAllDataset page through code.

Solution: import requests

from bs4 import BeautifulSoup

```
url = "https://archive.ics.uci.edu/"

# Send a GET request to the URL
response = requests.get(url)

# Create a BeautifulSoup object to parse the HTML content
soup = BeautifulSoup(response.text, "html.parser")

# Find the link to the "Show All Dataset" page
link = soup.find("a", href="ml/datasets.php")

# Construct the full URL of the "Show All Dataset" page
show_all_url = url.rstrip("/") + "/" + link["href"]

# Send a GET request to the "Show All Dataset" page
show_all_response = requests.get(show_all_url)

# Create a BeautifulSoup object for the "Show All Dataset" page
show_all_soup = BeautifulSoup(show_all_response.text, "html.parser")

# Find the table containing the dataset details
table = show_all_soup.find("table", cellpadding="3")

# Find all the rows in the table
rows = table.find_all("tr")

# Iterate over each row, skipping the header row
for row in rows[1:]:
    # Find the cells in the row
    cells = row.find_all("td")
```

```
    # Extract the required details from the cells
    dataset_name = cells[0].text.strip()
    data_type = cells[1].text.strip()
    task = cells[2].text.strip()
    attribute_type = cells[3].text.strip()
    no_of_instances = cells[4].text.strip()
    no_of_attributes = cells[5].text.strip()
    year = cells[6].text.strip()

    # Print the dataset details
    print("Dataset Name:", dataset_name)
    print("Data Type:", data_type)
    print("Task:", task)
    print("Attribute Type:", attribute_type)
    print("No. of Instances:", no_of_instances)
    print("No. of Attributes:", no_of_attributes)
    print("Year:", year)
    print()
```

9. Scrape the details of Data science recruiters
Url = https://www.naukri.com/hr-recruiters-consultants You have to find the following details:
A) Name
B) Designation
C)Company
D)Skills they hire for
E) Location
Note: - From naukri.com homepage click on the recruiters option and the on the search pane type Data science and click on search. All this should be done through code

Solution:

```
import requests
from bs4 import BeautifulSoup

url = "https://www.naukri.com/hr-recruiters-consultants"
```

```
# Set the search parameters for Data Science recruiters
search_params = {
    "keyword": "Data Science"
}

# Send a POST request with the search parameters
response = requests.post(url, data=search_params)

# Create a BeautifulSoup object to parse the HTML content
soup = BeautifulSoup(response.text, "html.parser")

# Find the container that holds the recruiter details
container = soup.find("div", class_="recSec")

# Find all the recruiter items
recruiter_items = container.find_all("div", class_="recDetails")

# Iterate over each recruiter item and extract the required details
for item in recruiter_items:
    name = item.find("span", class_="fl").text.strip()
    designation = item.find("span", class_="designation").text.strip()
    company = item.find("p", class_="highlightable").text.strip()
    skills = item.find("div", class_="hireSec highlightable").text.strip()
    location = item.find("small", class_="ellipsis").text.strip()

    print("Name:", name)
    print("Designation:", designation)
    print("Company:", company)
    print("Skills they hire for:", skills)
    print("Location:", location)
    print()
```