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

In [1]:
```python
import requests
from bs4 import BeautifulSoup
import pandas as pd

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

    # Check if the request was successful
    if response.status_code == 200:
        # Parse the HTML content using BeautifulSoup
        soup = BeautifulSoup(response.text, 'html.parser')

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

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

        # Create a DataFrame from the header text
        header_df = pd.DataFrame({'Header': header_text})

        return header_df
    else:
        print("Failed to retrieve the web page.")
        return None
# Example usage:
if __name__ == "__main__":
    url = "https://en.wikipedia.org/wiki/Main_Page"  # Replace with the desired URL
    my_dataframe = get_header_tags_to_dataframe(url)  # Assign the result to a different variable name
    if my_dataframe is not None:
        print(my_dataframe)
```

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

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

In [2]:
```python
import requests
from bs4 import BeautifulSoup
import pandas as pd

# Send an HTTP GET request to the website
url = "https://presidentofindia.nic.in/former-presidents.htm"
response = requests.get(url)

# Check if the request was successful (status code 200)
if response.status_code == 200:
    soup = BeautifulSoup(response.text, 'html.parser')

    # Find the table containing the list of former Presidents
    table = soup.find('table')

    # Initialize lists to store President names and terms
    president_names = []
    president_terms = []

    # Extract data from the table
    for row in table.find_all('tr')[1:]:
        columns = row.find_all('td')
        if len(columns) == 2:
            name = columns[0].text.strip()
            term = columns[1].text.strip()
            president_names.append(name)
            president_terms.append(term)

    # Create a DataFrame
    data = {'Name': president_names, 'Term of Office': president_terms}
    df = pd.DataFrame(data)

    # Display the DataFrame
    print(df)

else:
    print("Failed to retrieve the website. Status code:", response.status_code)
```

```
Failed to retrieve the website. Status code: 404
```

3) Write a python program to scrape cricket rankings from icc-cricket.com. You have to scrape and make data frame a) Top 10 ODI teams in men’s cricket along with the records for matches, points and rating.

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

In [3]:
```python
import requests
from bs4 import BeautifulSoup
import pandas as pd

# Function to scrape and create a DataFrame for ranking data
def scrape_and_create_dataframe(url, columns):
    response = requests.get(url)

    if response.status_code == 200:
        soup = BeautifulSoup(response.text, 'html.parser')

        # Find the table containing the rankings
        table = soup.find('table')

        # Initialize lists to store data
        data = []

        # Extract data from the table
        for row in table.find_all('tr')[1:11]:  # Top 10 rankings
            columns_data = row.find_all('td')

            # Check if there are enough columns
            if len(columns_data) == len(columns):
                row_data = [col.text.strip() for col in columns_data]
                data.append(row_data)

        # Create a DataFrame
        df = pd.DataFrame(data, columns=columns)

        return df

    else:
        print("Failed to retrieve the website. Status code:", response.status_code)
        return None

# URLs for top 10 ODI teams, batsmen, and bowlers
teams_url = "https://www.icc-cricket.com/rankings/mens/team-rankings/odi"
batsmen_url = "https://www.icc-cricket.com/rankings/mens/player-rankings/odi/batting"
bowlers_url = "https://www.icc-cricket.com/rankings/mens/player-rankings/odi/bowling"

# Columns for the DataFrames
teams_columns = ["Position", "Team", "Matches", "Points", "Rating"]
batsmen_columns = ["Position", "Player", "Team", "Rating"]
bowlers_columns = ["Position", "Player", "Team", "Rating"]

# Scrape and create DataFrames
odi_teams_df = scrape_and_create_dataframe(teams_url, teams_columns)
odi_batsmen_df = scrape_and_create_dataframe(batsmen_url, batsmen_columns)
odi_bowlers_df = scrape_and_create_dataframe(bowlers_url, bowlers_columns)

# Print the DataFrames
if odi_teams_df is not None:
    print("Top 10 ODI Teams:")
    print(odi_teams_df)

if odi_batsmen_df is not None:
    print("\nTop 10 ODI Batsmen:")
    print(odi_batsmen_df)

if odi_bowlers_df is not None:
    print("\nTop 10 ODI Bowlers:")
    print(odi_bowlers_df)
```

```
Top 10 ODI Teams:
  Position              Team Matches Points Rating
0        1    Australia\nAUS      26  3,061    118
1        2     Pakistan\nPAK      26  3,061    118
2        3        India\nIND      39  4,516    116
3        4   New Zealand\nNZ      29  3,006    104
4        5      England\nENG      26  2,636    101
5        6  South Africa\nSA      22  2,218    101
6        7   Bangladesh\nBAN      32  2,941     92
7        8     Sri Lanka\nSL      36  3,280     91
8        9  Afghanistan\nAFG      21  1,687     80
9       10   West Indies\nWI      38  2,582     68

Top 10 ODI Batsmen:
Empty DataFrame
Columns: [Position, Player, Team, Rating]
Index: []

Top 10 ODI Bowlers:
Empty DataFrame
Columns: [Position, Player, Team, Rating]
Index: []
```

4) Write a python program to scrape cricket rankings from icc-cricket.com. You have to scrape and make data frame a) Top 10 ODI teams in women’s cricket along with the records for matches, points and rating.

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

In [7]:
```python
import requests
from bs4 import BeautifulSoup
import pandas as pd

# Function to scrape and create a DataFrame for ranking data
def scrape_and_create_dataframe(url, columns):
    response = requests.get(url)

    if response.status_code == 200:
        soup = BeautifulSoup(response.text, 'html.parser')

        # Find the table containing the rankings
        table = soup.find('table')

        # Initialize lists to store data
        data = []

        # Extract data from the table
        for row in table.find_all('tr')[1:11]:  # Top 10 rankings
            columns_data = row.find_all('td')

            # Check if there are enough columns
            if len(columns_data) == len(columns):
                row_data = [col.text.strip() for col in columns_data]
                data.append(row_data)

        # Create a DataFrame
        df = pd.DataFrame(data, columns=columns)

        return df

    else:
        print("Failed to retrieve the website. Status code:", response.status_code)
        return None

# URLs for top 10 women's ODI teams, batsmen, and all-rounders
teams_url = "https://www.icc-cricket.com/rankings/womens/team-rankings/odi"
batsmen_url = "https://www.icc-cricket.com/rankings/womens/player-rankings/odi/batting"
allrounders_url = "https://www.icc-cricket.com/rankings/womens/player-rankings/odi/all-rounder"

# Columns for the DataFrames
teams_columns = ["Position", "Team", "Matches", "Points", "Rating"]
batsmen_columns = ["Position", "Player", "Team", "Rating"]
allrounders_columns = ["Position", "Player", "Team", "Rating"]

# Scrape and create DataFrames
odi_womens_teams_df = scrape_and_create_dataframe(teams_url, teams_columns)
odi_womens_batsmen_df = scrape_and_create_dataframe(batsmen_url, batsmen_columns)
odi_womens_allrounders_df = scrape_and_create_dataframe(allrounders_url, allrounders_columns)

# Print the DataFrames
if odi_womens_teams_df is not None:
    print("Top 10 Women's ODI Teams:")
    print(odi_womens_teams_df)

if odi_womens_batsmen_df is not None:
    print("\nTop 10 Women's ODI Batsmen:")
    print(odi_womens_batsmen_df)

if odi_womens_allrounders_df is not None:
    print("\nTop 10 Women's ODI All-rounders:")
    print(odi_womens_allrounders_df)
```

```
Top 10 Women's ODI Teams:
  Position              Team Matches Points Rating
0        1    Australia\nAUS      26  4,290    165
1        2      England\nENG      31  3,875    125
2        3  South Africa\nSA      26  3,098    119
3        4        India\nIND      30  3,039    101
4        5   New Zealand\nNZ      28  2,688     96
5        6   West Indies\nWI      29  2,743     95
6        7   Bangladesh\nBAN      17  1,284     76
7        8     Sri Lanka\nSL      12    820     68
8        9     Thailand\nTHA      13    883     68
9       10     Pakistan\nPAK      27  1,678     62

Top 10 Women's ODI Batsmen:
Empty DataFrame
Columns: [Position, Player, Team, Rating]
Index: []

Top 10 Women's ODI All-rounders:
Empty DataFrame
Columns: [Position, Player, Team, Rating]
Index: []
```

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

make data frame i) Headline ii) Time iii) News Link

In [8]:
```python
import requests
from bs4 import BeautifulSoup
import pandas as pd

# Function to scrape news details
def scrape_cnbc_news(url):
    response = requests.get(url)
    if response.status_code == 200:
        soup = BeautifulSoup(response.text, 'html.parser')

        # Find the container with news articles
        articles_container = soup.find('div', class_='PageBuilder-content')

        if articles_container:
            # Find all news articles
            articles = articles_container.find_all('a', class_='Card-title-link')

            # Initialize lists to store data
            headlines = []
            times = []
            news_links = []

            for article in articles:
                # Extract headline
                headline = article.text.strip()
                headlines.append(headline)

                # Extract news link
                news_link = article['href'] if 'href' in article.attrs else ""
                news_links.append(news_link)

            # Find all timestamps
            timestamps = articles_container.find_all('time', class_='Card-time')
            times = [timestamp.text.strip() for timestamp in timestamps]

            # Create a DataFrame
            news_data = {
                'Headline': headlines,
                'Time': times,
                'News Link': news_links
            }

            df = pd.DataFrame(news_data)
            return df
        else:
            print("No news articles found on the web page.")
            return None
    else:
        print("Failed to retrieve the web page.")
        return None

if __name__ == "__main__":
    # URL of the web page to scrape
    url = "https://www.cnbc.com/world/?region=world"

    # Scrape news details
    news_df = scrape_cnbc_news(url)

    if news_df is not None:
        print("CNBC News Details:")
        print(news_df)
```

```
No news articles found on the web page.
```

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

In [3]:
```python
import requests
from bs4 import BeautifulSoup
import pandas as pd

# Function to scrape the most downloaded articles
def get_most_downloaded_articles(url, days):
    response = requests.get(url)
    if response.status_code == 200:
        soup = BeautifulSoup(response.text, 'html.parser')

        # Find the container with the list of articles
        article_container = soup.find('div', class_='downloaded-articles-container')

        if article_container is not None:
            # Extract data for each article
            articles_data = []
            for article in article_container.find_all('div', class_='article-info'):
                # Check if the article was published in the last 90 days
                publication_date = article.find('div', class_='text-xs').next_sibling.strip()
                if (publication_date > (datetime.datetime.now() - datetime.timedelta(days=days))):
                    title = article.find('a', class_='article-title').text.strip()
                    authors = article.find('div', class_='text-xs').text.strip()
                    downloads = article.find('div', class_='text-xs').find_next('div', class_='text-xs').text.strip()

                    articles_data.append([title, authors, publication_date, downloads])

            # Create a DataFrame from the extracted data
            articles_df = pd.DataFrame(articles_data, columns=['Title', 'Authors', 'Publication Date', 'Downloads'])
            return articles_df
        else:
            print("No articles found on the web page.")
            return None
    else:
        print("Failed to retrieve the web page.")
        return None

if __name__ == "__main__":
    # Number of days to consider
    days = 90
    url = "https://www.journals.elsevier.com/artificial-intelligence/most-downloaded-articles"
    most_downloaded_articles_df = get_most_downloaded_articles(url, days)
    if most_downloaded_articles_df is not None:
        print("Most Downloaded Articles in Artificial Intelligence (Last 90 Days):")
        print(most_downloaded_articles_df)
```

```
No articles found on the web page.
```

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

ii) Cuisine iii) Location iv) Ratings v) Image URL

In [6]:
```python
import requests
from bs4 import BeautifulSoup
import pandas as pd

# Function to scrape restaurant details
def scrape_restaurant_details(url):
    response = requests.get(url)
    if response.status_code == 200:
        soup = BeautifulSoup(response.text, 'html.parser')

        # Find the container with restaurant details
        restaurant_details = soup.find_all('div', class_='restnt-card restaurant')

        # Initialize lists to store data
        restaurant_names = []
        cuisines = []
        locations = []
        ratings = []
        image_urls = []

        for restaurant in restaurant_details:
            # Extract restaurant name
            name_elem = restaurant.find('a', class_='restnt-name')
            name = name_elem.text.strip() if name_elem else ""
            restaurant_names.append(name)

            # Extract cuisine
            cuisine_elem = restaurant.find('span', class_='cuisine')
            cuisine = cuisine_elem.text.strip() if cuisine_elem else ""
            cuisines.append(cuisine)

            # Extract location
            location_elem = restaurant.find('span', class_='locality')
            location = location_elem.text.strip() if location_elem else ""
            locations.append(location)

            # Extract ratings
            rating_elem = restaurant.find('div', class_='rating-stars')
            rating = rating_elem.text.strip() if rating_elem else ""
            ratings.append(rating)

            # Extract image URL
            img_elem = restaurant.find('img')
            img_url = img_elem['src'] if img_elem and 'src' in img_elem.attrs else ""
            image_urls.append(img_url)

        # Create a DataFrame
        restaurant_data = {
            'Restaurant Name': restaurant_names,
            'Cuisine': cuisines,
            'Location': locations,
            'Ratings': ratings,
            'Image URL': image_urls
        }

        df = pd.DataFrame(restaurant_data)
        return df
    else:
        print("Failed to retrieve the web page.")
        return None

if __name__ == "__main__":
    # URL of the web page to scrape
    url = "https://www.dineout.co.in/delhi-restaurants?search_str=biryani"

    # Scrape restaurant details
    restaurant_df = scrape_restaurant_details(url)

    if restaurant_df is not None:
        print("Restaurant Details:")
        print(restaurant_df)
```

```
Restaurant Details:
               Restaurant Name Cuisine Location Ratings Image URL
0                Biryani Blues
1              Biryani By Kilo
2                         Veda
3               Andhra Canteen
4                      Bagundi
5         Pakiza Shahi Biryani
6      Hyderabadi Biryani House
7              Biryani Kingdom
8      Hyderabadi Biryani House
9                      Oh! Pind
10               Biryani Blues
11             Biryani By Kilo
12             Behrouz Biryani
13          Bir's Food Factory
14             Behrouz Biryani
15               Biryani Blues
16             Behrouz Biryani
17               Biryani Tales
18               Biryani Blues
19  Andhra Spices Biryani Base
```

In [ ]: