Q 1)

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

url = "https://en.wikipedia.org/wiki/Main_Page"

response = requests.get(url)

if response.status_code == 200:

    soup = BeautifulSoup(response.content, 'html.parser')

    header_tags = soup.find_all(['h1', 'h2', 'h3', 'h4', 'h5', 'h6'])

    header_texts = [tag.text.strip() for tag in header_tags]


    df = pd.DataFrame({'Headers': header_texts})
      print(df)

else:
    print(f"Failed to retrieve the page. Status code: {response.status_code}")
```

Q 2)

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


url = "https://presidentofindia.nic.in/former-presidents.htm"


response = requests.get(url)
```

```
if response.status_code == 200:

    soup = BeautifulSoup(response.content, 'html.parser')


    table = soup.find('table', {'class': 'table-responsive'})


    presidents_data = []
    for row in table.find_all('tr')[1:]:
        columns = row.find_all('td')
        name = columns[0].text.strip()
        term_of_office = columns[1].text.strip()
        presidents_data.append({'Name': name, 'Term of Office': term_of_office})


    df = pd.DataFrame(presidents_data)


    print(df)

else:
    print(f"Failed to retrieve the page. Status code: {response.status_code}")
```

Q 5)

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

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

response = requests.get(url)
```

```
soup = BeautifulSoup(response.text, 'html.parser')

headlines = soup.find_all(class_='some-headline-class')
times = soup.find_all(class_='some-time-class')
links = soup.find_all(class_='some-link-class')

headline_list = [headline.text.strip() for headline in headlines]
time_list = [time.text.strip() for time in times]
link_list = [link['href'] for link in links]

data = {'Headline': headline_list, 'Time': time_list, 'News Link': link_list}
df = pd.DataFrame(data)

print(df)
```

Q6)

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

url = "https://www.journals.elsevier.com/artificial-intelligence/most-downloaded-articles"
response = requests.get(url)

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

  titles = []
  authors = []
  dates = []
  paper_urls = []

    for article in soup.find_all("li", class_="pod-listing"):
        title = article.find("a", class_="pod-listing-title").text.strip()
```

```
        author = article.find("div", class_="pod-authors").text.strip()
        date = article.find("div", class_="pod-listing-pubdate").text.strip()
        paper_url = article.find("a", class_="pod-listing-title")["href"].strip()

        titles.append(title)
        authors.append(author)
        dates.append(date)
        paper_urls.append(paper_url)

    data = {
        'Paper Title': titles,
        'Authors': authors,
        'Published Date': dates,
        'Paper URL': paper_urls
    }

    df = pd.DataFrame(data)

    print(df)

else:
    print(f"Failed to fetch the page. Status code: {response.status_code}")
```

Q7)

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

url = 'https://www.dineout.co.in/delhi-restaurants'

response = requests.get(url)

soup = BeautifulSoup(response.content, 'html.parser')
```

```
restaurant_names = []
cuisines = []
locations = []
ratings = []
image_urls = []

for restaurant in soup.find_all('div', class_='restnt-info'):
    restaurant_names.append(restaurant.find('div', class_='restnt-info-wrap').h3.text.strip())
    cuisines.append(restaurant.find('span', class_='double-line-ellipsis').text.strip())
    locations.append(restaurant.find('span', class_='double-line-ellipsis loc-ellipsis').text.strip())
    ratings.append(restaurant.find('span', class_='green-box').text.strip())
    image_urls.append(restaurant.find('img')['data-src'])



data = {'Restaurant Name': restaurant_names, 'Cuisine': cuisines, 'Location': locations, 'Ratings':
ratings, 'Image URL': image_urls}
df = pd.DataFrame(data)

print(df)
```

Q 3)

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

def scrape_odi_team_rankings():
    url = "https://www.icc-cricket.com/rankings/mens/team-rankings/odi"
    response = requests.get(url)
    soup = BeautifulSoup(response.text, 'html.parser')

    teams_data = {'Team': [], 'Matches': [], 'Points': [], 'Rating': []}

    for team in soup.find_all('td', class_='table-body__cell rankings-table__team'):
        teams_data['Team'].append(team.text.strip())
```

```
    for matches, points, rating in zip(
            soup.find_all('td', class_='table-body__cell u-center-text'),
            soup.find_all('td', class_='table-body__cell u-center-text'),
            soup.find_all('td', class_='table-body__cell u-text-right rating')
    ):
        teams_data['Matches'].append(matches.text.strip())
        teams_data['Points'].append(points.text.strip())
        teams_data['Rating'].append(rating.text.strip())



    df_teams = pd.DataFrame(teams_data)
    df_teams = df_teams.head(10)  # Selecting top 10 teams
    return df_teams

def scrape_odi_batsmen_rankings():



def scrape_odi_bowlers_rankings():



odi_teams_df = scrape_odi_team_rankings()
print("Top 10 ODI Teams:")
print(odi_teams_df)
```

We Can use Similar code for batsmen and bowlers records

Q 4)
This qiuestion can be done by following the same method as Q3.