1) Write a python program to display **IMDB's Top** rated **100 Indian movies'** data https://www.imdb.com/list/ls056092300/ (i.e. **name, rating, year ofrelease**) and make **data frame**.

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

url = "https://www.imdb.com/list/ls056092300/"

response = requests.get(url)

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

movie_containers = soup.find_all('div', class_='lister-item mode-detail')

# Lists to store data
titles = []
years = []
ratings = []

for container in movie_containers:
    title = container.h3.a.text
    titles.append(title)

    year = container.h3.find('span', class_='lister-item-year').text.strip('()')
    years.append(year)


    rating = container.find('span', class_='ipl-rating-star__rating').text
    ratings.append(rating)

movies_df = pd.DataFrame({
    'Title': titles,
    'Year': years,
    'Rating': ratings
})

print(movies_df)
```

2Write a python program to scrape details of all the posts from https://www.patreon.com/coreyms .Scrape the **heading, date, content** and the **likes for the video from the link for the youtube video** from the post.

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

```
url = "https://www.patreon.com/coreyms"
response = requests.get(url)

# Parse the page content using BeautifulSoup
soup = BeautifulSoup(driver.page_source, 'html.parser')


# Extract post details
posts = soup.find_all('div', class_='sc-1eq90u-0')

data = []
for post in posts:
    try:
        heading = post.find('h2').text

        date = post.find('time').get('datetime')

        # Extract content
        content = post.find('div', class_='sc-1eq90u-0').text

        # Extract YouTube link
        youtube_link = post.find('a', href=True)
        if youtube_link and "youtube.com" in youtube_link['href']:
            youtube_link = youtube_link['href']
        else:
            youtube_link = None

        # Store the data
        data.append({
            'Heading': heading,
            'Date': date,
            'Content': content,
            'YouTube Link': youtube_link
        })
    except Exception as e:
        print(f"Error processing post: {e}")

# Create DataFrame
df = pd.DataFrame(data)

# Display the DataFrame
print(df)

df.to_csv('patreon_posts.csv', index=False)
```

3) Write a python program to scrape house details from mentioned URL. It should include house title, location, area, EMI and price from

https://www.nobroker.in/ .Enter three localities which are Indira Nagar, Jayanagar, Rajaji Nagar.

```
import requests
import pandas as pd

def get_property_data(locality):
    # URL for NoBroker API
    url = "https://www.nobroker.in/api/v1/property/filter/region"

    # Parameters for the API request
    params = {
        "city": "bangalore",
        "locality": locality.replace(" ", "%20"),
        "rentAmount": "0,500000",  # Adjust price range if needed
        "pageNo": 1,
        "pageSize": 50,
        "type": "SALE"
    }

    # Headers to mimic a browser visit
    headers = {
        "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"
    }

    # Send GET request to NoBroker API
    response = requests.get(url, params=params, headers=headers)

    # Check if request was successful
    if response.status_code == 200:
        data = response.json()
        return data.get('data', [])
    else:
        print(f"Failed to retrieve data for locality: {locality}")
        return []

def extract_property_details(properties):
    property_list = []
    for property in properties:
        # Extract relevant details
        title = property.get('title', 'No Title')
        location = property.get('locality', 'No Location')
        area = property.get('propertySize', 'No Area')
        price = property.get('price', 'No Price')
        emi = property.get('emi', 'No EMI')
```

```
        # Convert price and EMI to a readable format if necessary
        price = f"₹{price:,}"
        emi = f"₹{emi:,}" if emi else "No EMI"

        property_list.append({
            "Title": title,
            "Location": location,
            "Area (Sqft)": area,
            "Price": price,
            "EMI": emi
        })

    return property_list

# Localities to scrape
localities = ["Indira Nagar", "Jayanagar", "Rajaji Nagar"]

# Aggregate all property data
all_properties = []

for locality in localities:
    print(f"Scraping data for {locality}...")
    properties = get_property_data(locality)
    if properties:
        all_properties.extend(extract_property_details(properties))

# Create a DataFrame
df = pd.DataFrame(all_properties)

# Save the data to a CSV file
df.to_csv('nobroker_properties.csv', index=False)

# Display the DataFrame
print(df)
```

4) Write a python program to scrape first 10 product details which include product name , price , Image URL from https://www.bewakoof.com/bestseller?sort=popular .

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

# URL of the page to scrape
url = "https://www.bewakoof.com/bestseller?sort=popular"
```

```
# Send a GET request to the URL
headers = {
   "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/91.0.4472.124 Safari/537.36"
}
response = requests.get(url, headers=headers)

# Parse the page content using BeautifulSoup
soup = BeautifulSoup(response.content, "html.parser")

# Find all product containers
products = soup.find_all("div", class_="productCardBox")[:10]  # Get only the first 10
products

# Lists to store the data
product_names = []
product_prices = []
product_images = []

# Loop through each product and extract details
for product in products:
   # Extract product name
   name = product.find("h3").text
   product_names.append(name)

   # Extract product price
   price = product.find("span", class_="discountedPriceText").text
   product_prices.append(price)

   # Extract image URL
   image_url = product.find("img")["src"]
   product_images.append(image_url)

# Create a DataFrame to store the scraped data
df = pd.DataFrame({
   "Product Name": product_names,
   "Price": product_prices,
   "Image URL": product_images
})

# Display the DataFrame
print(df)

# Save the DataFrame to a CSV file
df.to_csv("bewakoof_products.csv", index=False)
```

5) Please visit https://www.cnbc.com/world/?region=world and scrap-
a) headings
b) date
c) News link

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

# URL of the CNBC World News page
url = "https://www.cnbc.com/world/?region=world"

# Send a GET request to the URL
headers = {
   "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/91.0.4472.124 Safari/537.36"
}
response = requests.get(url, headers=headers)

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

# Lists to store the data
headings = []
dates = []
news_links = []

# Find all news articles
articles = soup.find_all("div", class_="Card-titleContainer")

# Loop through each article and extract the required details
for article in articles:
   # Extract heading
   heading = article.find("a").get_text(strip=True)
   headings.append(heading)

   # Extract link
   news_link = article.find("a")["href"]
   news_links.append(news_link)

   # Extract date (if available)
   date_tag = article.find_next_sibling("div", class_="Card-time")
   if date_tag:
      date = date_tag.get_text(strip=True)
   else:
      date = "No date provided"
   dates.append(date)
```

```
# Create a DataFrame to store the scraped data
df = pd.DataFrame({
   "Heading": headings,
   "Date": dates,
   "News Link": news_links
})

# Display the DataFrame
print(df)

# Save the DataFrame to a CSV file
df.to_csv("cnbc_world_news.csv", index=False)
```

6) Please visit https://www.keaipublishing.com/en/journals/artificial-intelligence-in-agriculture/most-downloaded- articles/ andscrap-

a) Paper title
b) date
c) Author

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

# URL of the page to scrape
url = "https://www.keaipublishing.com/en/journals/artificial-intelligence-in-agriculture/most-downloaded-articles/"

# Send a GET request to the URL
headers = {
   "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"
}
response = requests.get(url, headers=headers)

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

# Lists to store the data
paper_titles = []
authors = []
dates = []

# Find all articles listed in the "Most Downloaded Articles" section
articles = soup.find_all("div", class_="pod-listing")
```

```
# Loop through each article and extract the required details
for article in articles:
    # Extract paper title
    title = article.find("a").get_text(strip=True)
    paper_titles.append(title)

    # Extract author(s)
    author = article.find("p", class_="pod-authors").get_text(strip=True)
    authors.append(author)

    # Extract publication date
    date = article.find("span", class_="pod-meta-item").get_text(strip=True)
    dates.append(date)

# Create a DataFrame to store the scraped data
df = pd.DataFrame({
    "Paper Title": paper_titles,
    "Authors": authors,
    "Date": dates
})

# Display the DataFrame
print(df)

# Save the DataFrame to a CSV file
df.to_csv("most_downloaded_articles.csv", index=False)
```