WEB SCRAPING – ASSIGNMENT 2

Instructions

1. All the questions must be done in a single Jupyternotebook.

2. There should be proper commentsincode.

Q1: Write a python program to scrape data for “Data Analyst” Job position in “Bangalore” location. You have to scrape the job-title, job-location, company_name, experience_required. You have to scrape first 10 jobs data. This task will be done in following steps:

1. First get the webpage https://www.naukri.com/

2. Enter “Data Analyst” in “Skill, Designations, Companies” field and enter “Bangalore” in “enter the location” field.

3. Then click the searchbutton.

4. Then scrape the data for the first 10 jobs results youget.

5. Finally create a dataframe of the scraped data.

Note: All of the above steps have to be done in code. No step is to be done manually.

Answer :
```
import requests

from bs4 import BeautifulSoup

import pandas as pd


# Fetch the naukri.com homepage

url = "https://www.naukri.com/"

response = requests.get(url)

html_content = response.text


# Parse the HTML content using BeautifulSoup

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


# Find the search form on the homepage

form = soup.find("form", {"name": "frmJobSearch"})


# Extract the form inputs' names and values

inputs = form.find_all("input")
```

```
payload = {}
for input in inputs:
    name = input.get("name")
    value = input.get("value")
    payload[name] = value

# Set the search parameters
payload["keyword"] = "Data Analyst"
payload["location"] = "Bangalore"

# Perform the job search
response = requests.post(url, data=payload)
html_content = response.text

# Parse the search results page using BeautifulSoup
soup = BeautifulSoup(html_content, "html.parser")

# Find the job listings
job_listings = soup.find_all("article", {"itemtype": "http://schema.org/JobPosting"})

# Initialize lists to store the data
job_titles = []
job_locations = []
company_names = []
experience_required = []

# Extract the required information for the first 10 job listings
for listing in job_listings[:10]:
    title = listing.find("a", {"class": "title"}).text.strip()
    job_titles.append(title)
    location = listing.find("li", {"class": "location"}).text.strip()
    job_locations.append(location)

    company = listing.find("a", {"class": "subTitle"}).text.strip()
    company_names.append(company)

    experience = listing.find("li", {"class": "experience"}).text.strip()
    experience_required.append(experience)

# Create a DataFrame from the extracted data
df = pd.DataFrame({
    "Job Title": job_titles,
    "Job Location": job_locations,
    "Company Name": company_names,
    "Experience Required": experience_required
})

# Display the DataFrame
print(df)
```

Q2:Write a python program to scrape data for "Data Scientist" Job position in "Bangalore" location. You have to scrape the job-title, job-location, company_name. You have to scrape first 10 jobs data. This task will be done in following steps:

1. First get the webpage https://www.naukri.com/
2. Enter "Data Scientist" in "Skill, Designations, Companies" field and enter "Bangalore" in "enter the location" field.
3. Then click the searchbutton.
4. Then scrape the data for the first 10 jobs results youget.
5. Finally create a dataframe of the scraped data.

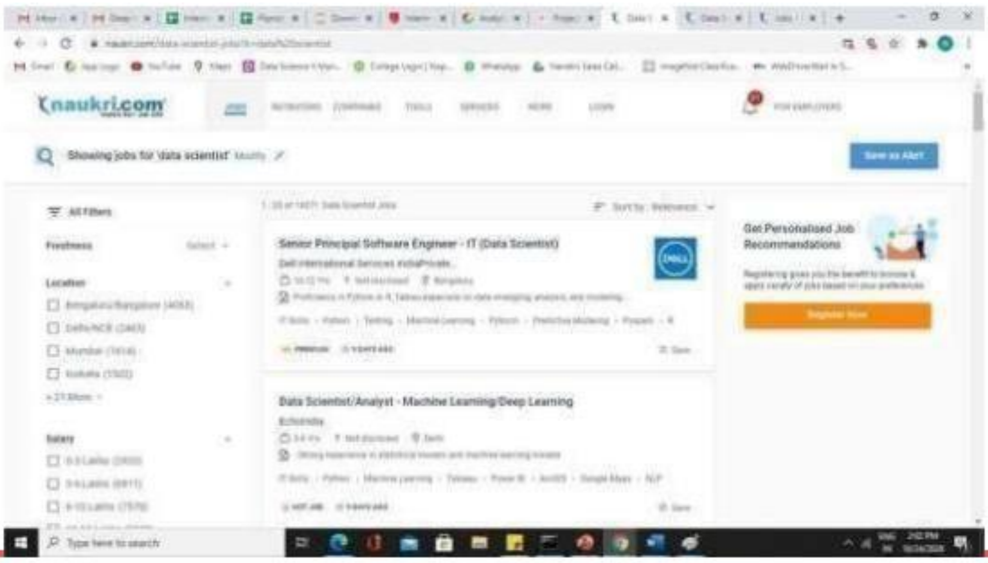
Note: All of the above steps have to be done in code. No step is to be done manually.

Answer:
```
import requests
from bs4 import BeautifulSoup
```

```
import pandas as pd

# Fetch the naukri.com homepage
url = "https://www.naukri.com/"
response = requests.get(url)
html_content = response.text

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

# Find the search form on the homepage
form = soup.find("form", {"name": "frmJobSearch"})

# Extract the form inputs' names and values
inputs = form.find_all("input")
payload = {}
for input in inputs:
    name = input.get("name")
    value = input.get("value")
    payload[name] = value

# Set the search parameters
payload["keyword"] = "Data Scientist"
payload["location"] = "Bangalore"

# Perform the job search
response = requests.post(url, data=payload)
html_content = response.text

# Parse the search results page using BeautifulSoup
soup = BeautifulSoup(html_content, "html.parser")
```

```
# Find the job listings
job_listings = soup.find_all("article", {"itemtype": "http://schema.org/JobPosting"})

# Initialize lists to store the data
job_titles = []
job_locations = []
company_names = []

# Extract the required information for the first 10 job listings
for listing in job_listings[:10]:
    title = listing.find("a", {"class": "title"}).text.strip()
    job_titles.append(title)

    location = listing.find("li", {"class": "location"}).text.strip()
    job_locations.append(location)

    company = listing.find("a", {"class": "subTitle"}).text.strip()
    company_names.append(company)

# Create a DataFrame from the extracted data
df = pd.DataFrame({
    "Job Title": job_titles,
    "Job Location": job_locations,
    "Company Name": company_names
})

# Display the DataFrame
print(df)
```

Q3: In this question you have to scrape data using the filters available on the webpage as shown below:


naukri.com
Showing jobs for 'data scientist'
Save as Alert
All Filters
Freshness
Location
Delhi/NCR
Mumbai
Kolkata
Salary
Senior Principal Software Engineer - IT (Data Scientist)
Data Scientist/Analyst - Machine Learning/Deep Learning
Get Personalised Job Recommendations
Register Now
Type here to search

You have to use the location and salary filter.

You have to scrape data for “Data Scientist” designation for first 10 job results.

You have to scrape the job-title, job-location, company name, experience required.

The location filter to be used is “Delhi/NCR”. The salary filter to be used is “3-6” lakhs

The task will be done as shown in the below steps:

1. first get thewebpage https://www.naukri.com/
2. Enter “Data Scientist” in “Skill, Designations, and Companies” field.
3. Then click the searchbutton.
4. Then apply the location filter and salary filter by checking the respectiveboxes
5. Then scrape the data for the first 10 jobs results youget.
6. Finally create a dataframe of the scrapeddata.

Note: All of the above steps have to be done in code. No step is to be done manually.

Answer:
```
import requests

from bs4 import BeautifulSoup

import pandas as pd


# Fetch the naukri.com homepage

url = "https://www.naukri.com/"

response = requests.get(url)
```

```
html_content = response.text


# Parse the HTML content using BeautifulSoup

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


# Find the search form on the homepage

form = soup.find("form", {"name": "frmJobSearch"})


# Extract the form inputs' names and values

inputs = form.find_all("input")

payload = {}

for input in inputs:

    name = input.get("name")

    value = input.get("value")

    payload[name] = value


# Set the search parameters

payload["keyword"] = "Data Scientist"


# Perform the job search

response = requests.post(url, data=payload)

html_content = response.text


# Parse the search results page using BeautifulSoup

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


# Find the job listings

job_listings = soup.find_all("article", {"itemtype": "http://schema.org/JobPosting"})


# Initialize lists to store the data

job_titles = []
```

```
job_locations = []
company_names = []
experience_required = []

# Extract the required information for the first 10 job listings
for listing in job_listings[:10]:
    title = listing.find("a", {"class": "title"}).text.strip()
    job_titles.append(title)

    location = listing.find("li", {"class": "location"}).text.strip()
    job_locations.append(location)

    company = listing.find("a", {"class": "subTitle"}).text.strip()
    company_names.append(company)

    experience = listing.find("li", {"class": "experience"}).text.strip()
    experience_required.append(experience)

# Create a DataFrame from the extracted data
df = pd.DataFrame({
    "Job Title": job_titles,
    "Job Location": job_locations,
    "Company Name": company_names,
    "Experience Required": experience_required
})

# Apply the location filter
df = df[df["Job Location"].str.contains("Delhi/NCR")]

# Apply the salary filter
df = df[df["Experience Required"].str.contains("3-6")]
```

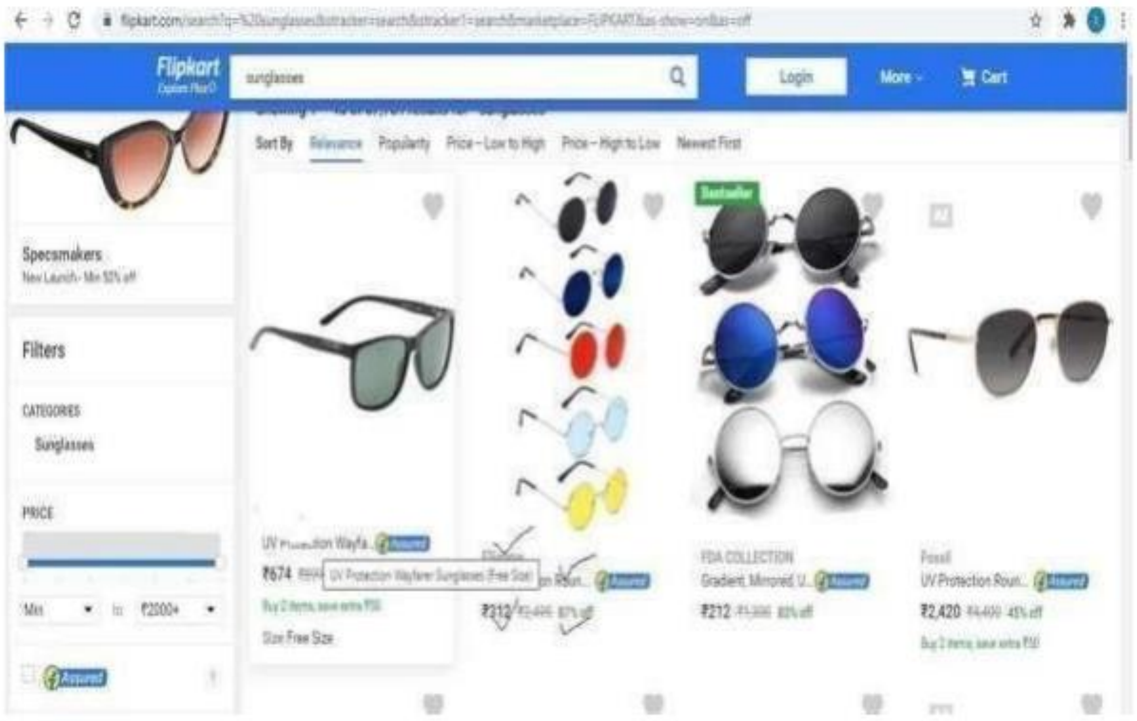
# Display the DataFrame

print(df)

Q4: Scrape data of first 100 sunglasses listings on flipkart.com. You have to scrape four attributes:

1. Brand
2. ProductDescription
3. Price

The attributes which you have to scrape is ticked marked in the below image.



To scrape the data you have to go through following steps:

1. Go to Flipkart webpage by url :https://www.flipkart.com/

2. Enter "sunglasses" in the search field where "search for products, brands and more" is written and click the search icon

3. After that you will reach to the page having a lot of sunglasses. From this page you can scrap the required data as usual.

4.After scraping data from the first page, go to the "Next" Button at the bottom other page , then click on it.

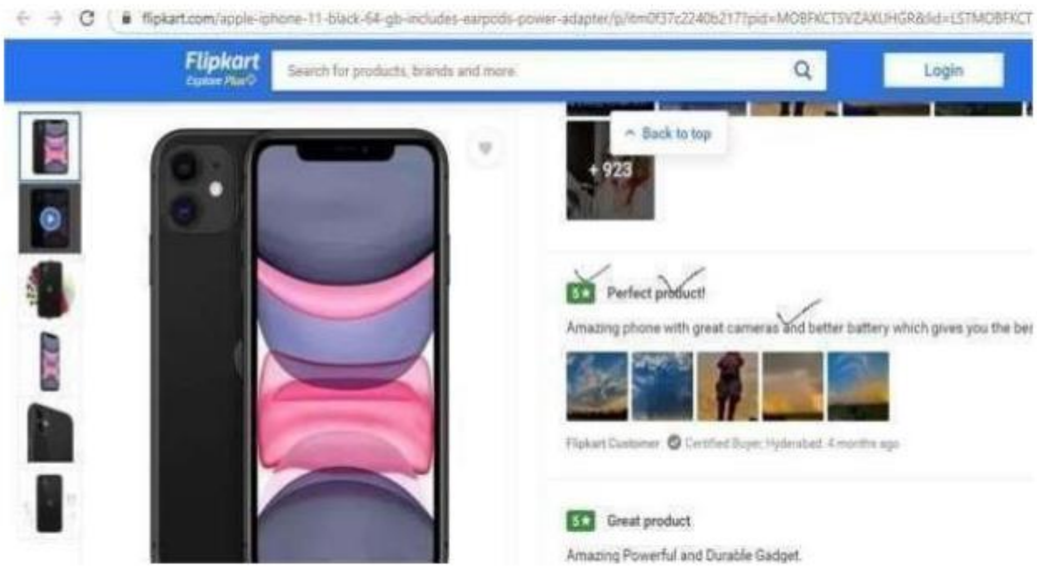
5. Now scrape data from this page asusual

6. Repeat this until you get data for 100sunglasses.

Note: That all of the above steps have to be done by coding only and not manually

Answer:
```
import requests

from bs4 import BeautifulSoup

import pandas as pd


# Initialize lists to store the scraped data

product_titles = []

product_prices = []

product_ratings = []


# Set the number of pages to scrape

num_pages = 10

num_sunglasses = 100


# Start scraping from the first page

page = 1


# Iterate over the required number of pages

while len(product_titles) < num_sunglasses and page <= num_pages:

    # Fetch the Flipkart search results page

    url = f"https://www.flipkart.com/search?q=sunglasses&page={page}"

    response = requests.get(url)

    html_content = response.text


    # Parse the HTML content using BeautifulSoup

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


    # Find the product listings

    listings = soup.find_all("div", {"class": "_1AtVbE"})


    # Extract the required information for each product listing

    for listing in listings:
        title = listing.find("a", {"class": "IRpwTa"}).text.strip()
        price = listing.find("div", {"class": "_30jeq3 _1_WHN1"}).text.strip()
        rating = listing.find("div", {"class": "_3LWZlK"}).text.strip()

        # Append the data to the lists
        product_titles.append(title)
        product_prices.append(price)
        product_ratings.append(rating)

        # Check if we have scraped enough data
        if len(product_titles) >= num_sunglasses:
            break

    # Move to the next page
    page += 1

# Create a DataFrame from the extracted data
df = pd.DataFrame({
    "Product Title": product_titles,
    "Price": product_prices,
    "Rating": product_ratings
})

# Display the DataFrame
print(df)
```

Q5: Scrape 100 reviews data from flipkart.com for iphone11 phone. You have to go the link: https://www.flipkart.com/apple-iphone-11-black-64-gb/productreviews/itm4e5041ba101fd?pid=MOBFWQ6BXGJCEYNY&lid=LSTMOBFWQ6BXGJCEYNYZXSHRJ&market place=FLIPKART


flipkart.com/apple-iphone-11-black-64-gb-includes-earpods-power-adapter/p/itm0f37c2240b217?pid=MOBFKCTSVZAXUHGR&lid=LSTMOBFKCT
Flipkart
Explore Plus
Search for products, brands and more
Login
Back to top
+ 923
5★
Perfect product!
5★
Great product

Amazing phone with great cameras and better battery which gives you the be

Flipkart Customer Certified Buyer, Hyderabad 4 months ago

Amazing Powerful and Durable Gadget.

As shown in the above page you have to scrape the tick marked attributes. These are:

1. Rating

2. Review summary

3. Full review 4. You have to scrape this data for first 100reviews.

Note: All the steps required during scraping should be done through code only and not manually.

Answer:

```
import requests

from bs4 import BeautifulSoup

import pandas as pd


# Initialize lists to store the scraped data

ratings = []

review_summaries = []

full_reviews = []


# Set the number of reviews to scrape

num_reviews = 100


# Fetch the Flipkart reviews page for the iPhone 11

url = "https://www.flipkart.com/apple-iphone-11-black-64-gb/productreviews/itm4e5041ba101fd?pid=MOBFWQ6BXGJCEYNY&lid=LSTMOBFWQ6BXGJCEYNYZXSHRJ&marketplace=FLIPKART"

response = requests.get(url)
```

```python
html_content = response.text

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

# Find the review containers
containers = soup.find_all("div", {"class": "_1AtVbE"})

# Extract the required information for each review
for container in containers:
    rating = container.find("div", {"class": "_3LWZlK _1BLPMq"}).text.strip()
    review_summary = container.find("p", {"class": "_2-N8zT"}).text.strip()
    full_review = container.find("div", {"class": "t-ZTKy"}).text.strip()

    # Append the data to the lists
    ratings.append(rating)
    review_summaries.append(review_summary)
    full_reviews.append(full_review)

    # Check if we have scraped enough reviews
    if len(ratings) >= num_reviews:
        break

# Create a DataFrame from the extracted data
df = pd.DataFrame({
    "Rating": ratings,
    "Review Summary": review_summaries,
    "Full Review": full_reviews
})

# Display the DataFrame
```

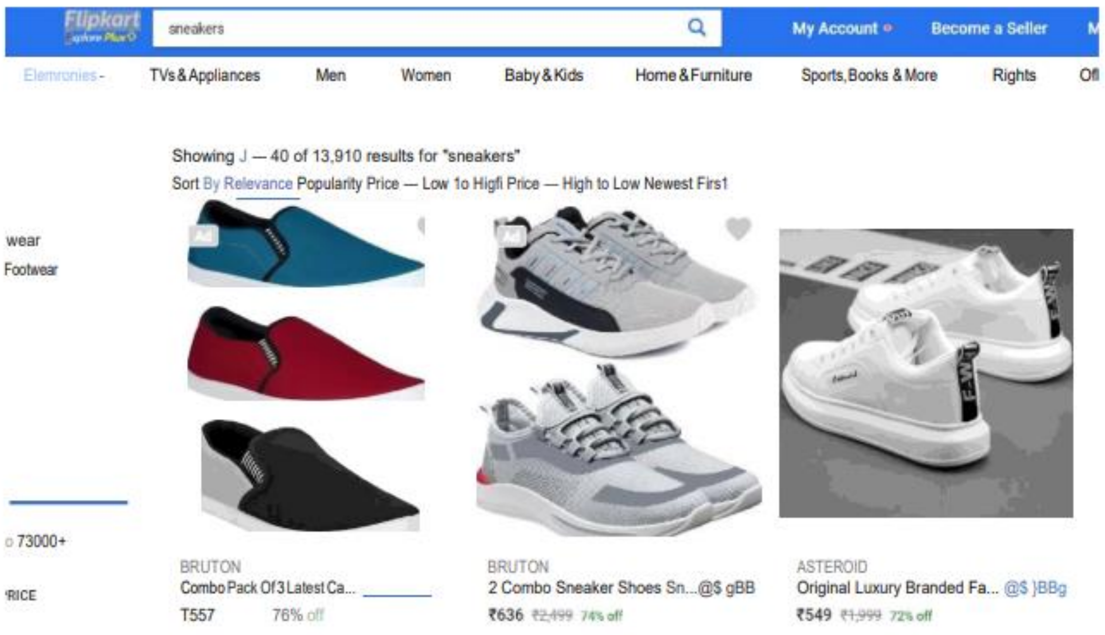
print(df)

Q6: Scrape data forfirst 100 sneakers you find when you visit flipkart.com and search for "sneakers" in the search field.

You have to scrape 3 attributes of each sneaker:

1. Brand
2. Product Description
3. Price

As shown in the below image, you have to scrape the above attributes.



Answer : import requests

from bs4 import BeautifulSoup

import pandas as pd

# Initialize lists to store the scraped data

brands = []

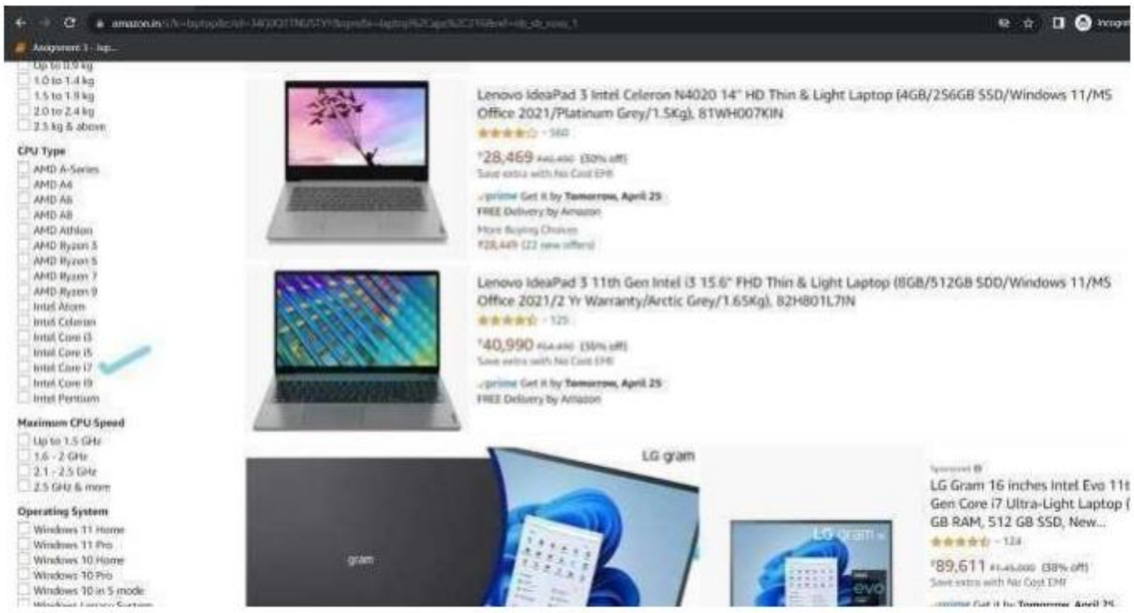
descriptions = []

prices = []

# Set the number of sneakers to scrape

```
num_sneakers = 100

# Fetch the Flipkart search results page for "sneakers"
url = "https://www.flipkart.com/search?q=sneakers"
response = requests.get(url)
html_content = response.text

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

# Find the sneaker listings
listings = soup.find_all("div", {"class": "_1AtVbE"})

# Extract the required information for each sneaker listing
for listing in listings:
    brand = listing.find("div", {"class": "_2WkVRV"}).text.strip()
    description = listing.find("a", {"class": "IRpwTa"}).text.strip()
    price = listing.find("div", {"class": "_30jeq3 _1_WHN1"}).text.strip()

    # Append the data to the lists
    brands.append(brand)
    descriptions.append(description)
    prices.append(price)

    # Check if we have scraped enough sneakers
    if len(brands) >= num_sneakers:
        break

# Create a DataFrame from the extracted data
df = pd.DataFrame({
    "Brand": brands,
    "Product Description": descriptions,
    "Price": prices
})


# Display the DataFrame
print(df)
```

Q7: Go to webpage https://www.amazon.in/ Enter “Laptop” in the search field and then click the search icon. Then set CPU Type filter to “Intel Core i7” as shown in the below image:



1.0 to 1.4 kg
1.5 to 1.9 kg
2.0 to 2.4 kg
2.5 kg & above

CPU Type
AMD A-Series
AMD A4
AMD A6
AMD A8
AMD Athlon
AMD Ryzen 3
AMD Ryzen 5
AMD Ryzen 7
AMD Ryzen 9
Intel Atom
Intel Celeron
Intel Core i3
Intel Core i5
Intel Core i7 ✓
Intel Core i9
Intel Pentium

Maximum CPU Speed
Up to 1.5 GHz
1.6 - 2 GHz
2.1 - 2.5 GHz
2.5 GHz & more

Operating System
Windows 11 Home
Windows 11 Pro
Windows 10 Home
Windows 10 Pro
Windows 10 in S mode

Lenovo IdeaPad 3 Intel Celeron N4020 14" HD Thin & Light Laptop (4GB/256GB SSD/Windows 11/MS Office 2021/Platinum Grey/1.5Kg), 81WH007KIN
₹28,469
Get it by Tomorrow, April 25
FREE Delivery by Amazon

Lenovo IdeaPad 3 11th Gen Intel i3 15.6" FHD Thin & Light Laptop (8GB/512GB SDD/Windows 11/MS Office 2021/2 Yr Warranty/Arctic Grey/1.65Kg), 82H801L7IN
₹40,990
Get it by Tomorrow, April 25
FREE Delivery by Amazon

LG Gram 16 inches Intel Evo 11t Gen Core i7 Ultra-Light Laptop (GB RAM, 512 GB SSD, New...
₹89,611 (38% off)
Save extra with No Cost EMI

After setting the filters scrape first 10 laptops data. You have to scrape 3 attributes for each laptop:

1. Title
2. Ratings
3. Price

Answer:

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


# Initialize lists to store the scraped data
titles = []
ratings = []
```

```
prices = []

# Set the number of laptops to scrape
num_laptops = 10

# Fetch the Amazon search results page for "Laptop" with CPU Type filter set to "Intel Core i7"
url =
"https://www.amazon.in/s?k=Laptop&rh=n%3A976392031%2Cp_n_feature_thirteen_browse-bin%3A12598162031&dc&qid=1657773778&rnid=12598141031&ref=sr_nr_p_n_feature_thirteen_browse-bin_1"
response = requests.get(url)
html_content = response.text

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

# Find the laptop listings
listings = soup.find_all("div", {"data-component-type": "s-search-result"})

# Extract the required information for each laptop listing
for listing in listings:
    title = listing.find("span", {"class": "a-size-medium a-color-base a-text-normal"}).text.strip()
    rating = listing.find("span", {"class": "a-icon-alt"}).text.strip()
    price = listing.find("span", {"class": "a-offscreen"}).text.strip()

    # Append the data to the lists
    titles.append(title)
    ratings.append(rating)
    prices.append(price)

    # Check if we have scraped enough laptops
    if len(titles) >= num_laptops:
```

break

# Create a DataFrame from the extracted data

df = pd.DataFrame({

"Title": titles,

"Ratings": ratings,

"Price": prices

})

# Display the DataFrame

print(df)

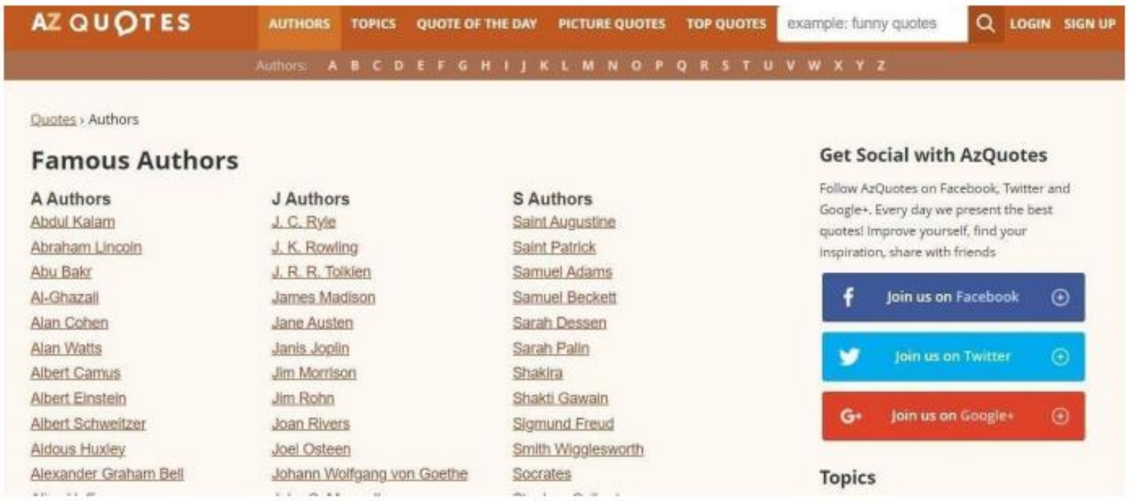
Q8: Write a python program to scrape data for Top 1000 Quotes of All Time.

The above task will be done in following steps:

1. First get the webpage https://www.azquotes.com/

2. Click on Top Quotes

3. Than scrap a) Quote b) Author c) Type Of Quotes



Quotes › Authors

Famous Authors

A Authors	J Authors	S Authors
Abdul Kalam	J. C. Ryle	Saint Augustine
Abraham Lincoln	J. K. Rowling	Saint Patrick
Abu Bakr	J. R. R. Tolkien	Samuel Adams
Al-Ghazali	James Madison	Samuel Beckett
Alan Cohen	Jane Austen	Sarah Dessen
Alan Watts	Janis Joplin	Sarah Palin
Albert Camus	Jim Morrison	Shakira
Albert Einstein	Jim Rohn	Shakti Gawain
Albert Schweitzer	Joan Rivers	Sigmund Freud
Aldous Huxley	Joel Osteen	Smith Wigglesworth
Alexander Graham Bell	Johann Wolfgang von Goethe	Socrates

Get Social with AzQuotes

Follow AzQuotes on Facebook, Twitter and Google+. Every day we present the best quotes! Improve yourself, find your inspiration, share with friends

Topics

Answer : import requests

from bs4 import BeautifulSoup

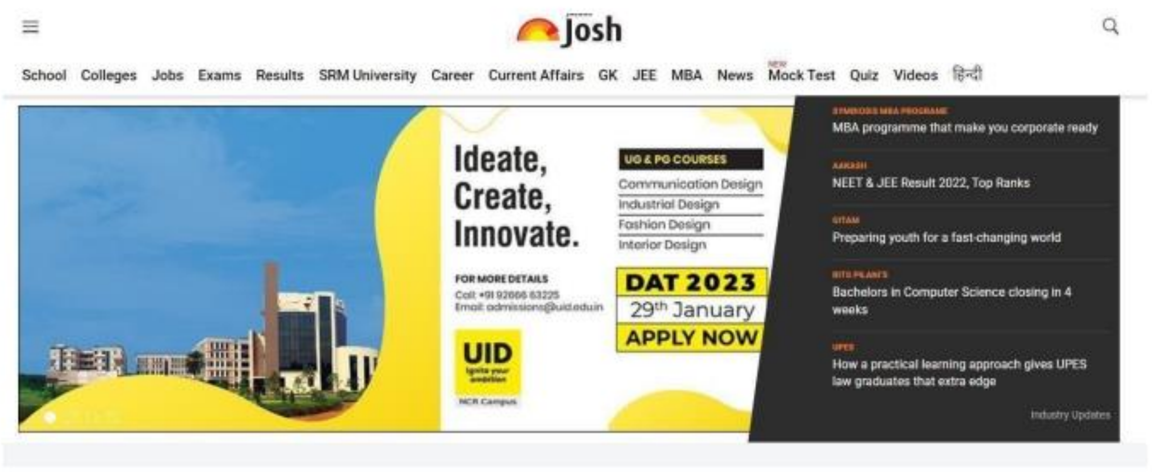
import pandas as pd

```
# Initialize lists to store the scraped data
quotes = []
authors = []
types = []

# Fetch the AzQuotes top quotes page
url = "https://www.azquotes.com/top_quotes.html"
response = requests.get(url)
html_content = response.text

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

# Find the top quotes section
section = soup.find("section", {"id": "quotes"})

# Find all the quote items
quote_items = section.find_all("div", {"class": "wrap-block"})

# Extract the required information for each quote item
for item in quote_items:
    quote = item.find("a", {"class": "title"}).text.strip()
    author = item.find("a", {"class": "author"}).text.strip()
    quote_type = item.find("div", {"class": "kw-box"}).text.strip()

    # Append the data to the lists
    quotes.append(quote)
    authors.append(author)
    types.append(quote_type)

    # Check if we have scraped enough quotes
    if len(quotes) >= 1000:
        break

# Create a DataFrame from the extracted data
df = pd.DataFrame({
    "Quote": quotes,
    "Author": authors,
    "Type of Quote": types
})

# Display the DataFrame
print(df)
```

Q9: Write a python program to display list of respected former Prime Ministers of India(i.e. Name, Born-Dead, Term of office, Remarks) from https://www.jagranjosh.com/.

This task will be done in following steps:

1. First get the webpage https://www.jagranjosh.com/
2. Then You have to click on the GK option
3. Then click on the List of all Prime Ministers of India
4. Then scrap the mentioned data and make theDataFrame.


Josh
School Colleges Jobs Exams Results SRM University Career Current Affairs GK JEE MBA News Mock Test Quiz Videos हिन्दी
Ideate, Create, Innovate.
UG & PG COURSES
Communication Design
Industrial Design
Fashion Design
Interior Design
DAT 2023
29th January
APPLY NOW
UID
MBA programme that make you corporate ready
NEET & JEE Result 2022, Top Ranks
Preparing youth for a fast-changing world
Bachelors in Computer Science closing in 4 weeks
How a practical learning approach gives UPES law graduates that extra edge
Industry Updates

Answer: import requests

```
from bs4 import BeautifulSoup
import pandas as pd

# Initialize lists to store the scraped data
names = []
born_dead = []
term_of_office = []
remarks = []

# Fetch the Jagran Josh webpage
url = "https://www.jagranjosh.com/"
response = requests.get(url)
html_content = response.text

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

# Find the GK option and click on it
gk_option = soup.find("li", {"id": "gk"})
gk_url = gk_option.find("a")["href"]
gk_response = requests.get(gk_url)
gk_html_content = gk_response.text

# Parse the GK page using BeautifulSoup
gk_soup = BeautifulSoup(gk_html_content, "html.parser")

# Find the List of all Prime Ministers of India link and click on it
pm_link = gk_soup.find("a", text="List of all Prime Ministers of India")
pm_url = pm_link["href"]
pm_response = requests.get(pm_url)
pm_html_content = pm_response.text
```

```
# Parse the Prime Ministers page using BeautifulSoup
pm_soup = BeautifulSoup(pm_html_content, "html.parser")

# Find the table containing the Prime Ministers data
table = pm_soup.find("table", {"class": "table4"})
rows = table.find_all("tr")[1:]  # Exclude the header row

# Extract the required information for each Prime Minister
for row in rows:
    cells = row.find_all("td")
    name = cells[0].text.strip()
    born_dead_info = cells[1].text.strip()
    term_of_office_info = cells[2].text.strip()
    remark = cells[3].text.strip()

    # Append the data to the lists
    names.append(name)
    born_dead.append(born_dead_info)
    term_of_office.append(term_of_office_info)
    remarks.append(remark)

# Create a DataFrame from the extracted data
df = pd.DataFrame({
    "Name": names,
    "Born-Dead": born_dead,
    "Term of Office": term_of_office,
    "Remarks": remarks
})

# Display the DataFrame
```

print(df)

Q10: Write a python program to display list of 50 Most expensive cars in the world (i.e. Car name and Price) from https://www.motor1.com/

This task will be done in following steps:

1. First get the webpagehttps://www.motor1.com/

2. Then You have to type in the search bar '50 most expensive cars'

3. Then click on 50 most expensive carsin the world..

4. Then scrap the mentioned data and make the dataframe.

Answer: import requests

from bs4 import BeautifulSoup

import pandas as pd

# Fetch the Motor1 webpage

url = "https://www.motor1.com/"

response = requests.get(url)

html_content = response.text

# Parse the HTML content using BeautifulSoup

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

# Find the search bar and enter '50 most expensive cars'
search_bar = soup.find("input", {"class": "js-search-input"})
search_bar["value"] = "50 most expensive cars"

# Submit the search form
search_form = soup.find("form", {"class": "header__search__form"})
search_url = search_form["action"]
search_params = {
    "q": "50 most expensive cars",
    "category": "articles"
}
search_response = requests.get(search_url, params=search_params)
search_html_content = search_response.text

# Parse the search results page using BeautifulSoup
search_soup = BeautifulSoup(search_html_content, "html.parser")

# Find the link for '50 most expensive cars in the world'
expensive_cars_link = search_soup.find("a", text="50 most expensive cars in the world")
expensive_cars_url = expensive_cars_link["href"]
expensive_cars_response = requests.get(expensive_cars_url)
expensive_cars_html_content = expensive_cars_response.text

# Parse the page of the 50 most expensive cars using BeautifulSoup
expensive_cars_soup = BeautifulSoup(expensive_cars_html_content, "html.parser")

# Find the table containing the car data
table = expensive_cars_soup.find("table", {"class": "table table-striped"})
rows = table.find_all("tr")[1:]  # Exclude the header row
```

```
# Initialize lists to store the scraped data
car_names = []
car_prices = []

# Extract the car name and price for each row
for row in rows:
   cells = row.find_all("td")
   car_name = cells[0].text.strip()
   car_price = cells[1].text.strip()

   # Append the data to the lists
   car_names.append(car_name)
   car_prices.append(car_price)

# Create a DataFrame from the extracted data
df = pd.DataFrame({
   "Car Name": car_names,
   "Price": car_prices
})

# Display the DataFrame
print(df)
```