In this question you have to scrape data using the filters available on the webpage 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 the web page https://www.naukri.com/
2. Enter “Data Scientist” in “Skill, Designations, and Companies” field.
3. Then click the search button.
4. Then apply the location filter and salary filter by checking the respective boxes
5. Then scrape the data for the first 10 jobs results you get.
6. Finally create a dataframe of the scraped data.

```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.chrome.service import Service
from webdriver_manager.chrome import ChromeDriverManager
import pandas as pd
import time

# Initialize the browser
driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()))

# Step 1: Open Naukri.com
driver.get("https://www.naukri.com/")

# Step 2: Enter "Data Scientist" in the search field
search_field = driver.find_element(By.CLASS_NAME, "suggestor-input")
search_field.send_keys("Data Scientist")

# Step 3: Click on the search button
search_button = driver.find_element(By.CLASS_NAME, "qsbSubmit")
search_button.click()

# Wait for the results page to load
WebDriverWait(driver, 10).until(
    EC.presence_of_element_located((By.CLASS_NAME, "jobTuple"))
)

# Step 4: Apply the location filter "Delhi/NCR"
location_filter = driver.find_element(By.XPATH, "//span[text()='Delhi / NCR']")
location_filter.click()
```

```
# Apply the salary filter "3-6 Lakhs"
salary_filter = driver.find_element(By.XPATH, "//span[text()='3-6 Lakhs']")
salary_filter.click()

# Wait for the filters to be applied
time.sleep(5)

# Step 5: Scrape data for the first 10 jobs
job_data = []

# Scrape job details
jobs = driver.find_elements(By.CLASS_NAME, 'jobTuple')[:10]

for job in jobs:
    job_title = job.find_element(By.CLASS_NAME, 'title').text
    job_location = job.find_element(By.CLASS_NAME, 'location').text
    company_name = job.find_element(By.CLASS_NAME, 'subTitle').text
    experience_required = job.find_element(By.CLASS_NAME, 'experience').text

    job_data.append({
        "Job Title": job_title,
        "Location": job_location,
        "Company Name": company_name,
        "Experience Required": experience_required
    })

# Step 6: Create a DataFrame from the scraped data
df = pd.DataFrame(job_data)

# Print the DataFrame
print(df)

# Close the browser
driver.quit()
```

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, 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.shine.com/
2. Enter “Data Analyst” in “Job title, Skills” 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 you get.

5. Finally create a dataframe of the scraped data.

```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.chrome.service import Service
from webdriver_manager.chrome import ChromeDriverManager
import pandas as pd
import time

# Initialize the browser
driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()))

# Step 1: Open the Shine.com webpage
driver.get("https://www.shine.com/")

# Step 2: Enter "Data Analyst" in "Job title, Skills" field
job_title_field = driver.find_element(By.ID, "id_q")
job_title_field.send_keys("Data Scientist")

# Enter "Bangalore" in the location field
location_field = driver.find_element(By.ID, "id_loc")
location_field.send_keys("Bangalore")

# Step 3: Click the search button
search_button = driver.find_element(By.CLASS_NAME, "search-btn")
search_button.click()

# Wait for the results to load
time.sleep(5)

# Step 4: Scrape data for the first 10 job results
job_data = []

# Find all job listings (limiting to the first 10)
jobs = driver.find_elements(By.CLASS_NAME, 'wht-shd-bx')[:10]

for job in jobs:
    try:
        job_title = job.find_element(By.CLASS_NAME, 'job_title').text
    except:
        job_title = None

    try:
        job_location = job.find_element(By.CLASS_NAME, 'loc').text
    except:
        job_location = None
```

```
    try:
        company_name = job.find_element(By.CLASS_NAME, 'jobs-comp-name').text
    except:
        company_name = None

    try:
        experience_required = job.find_element(By.CLASS_NAME, 'exp').text
    except:
        experience_required = None

    job_data.append({
        "Job Title": job_title,
        "Location": job_location,
        "Company Name": company_name,
        "Experience Required": experience_required
    })

# Step 5: Create a DataFrame
df = pd.DataFrame(job_data)

# Display the DataFrame
print(df)

# Close the browser
driver.quit()
```

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/product-reviews/itm4e5041ba101fd?pid=MOBFWQ6BXGJCEYNY&lid=LSTMOBFWQ6BXGJCEYNYZXSHRJ&marketplace=F

LIPKART

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.

```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.chrome.service import Service
from webdriver_manager.chrome import ChromeDriverManager
import pandas as pd
import time
```

```
# Initialize the browser
driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()))

# Open the Flipkart iPhone 11 reviews page
url = "https://www.flipkart.com/apple-iphone-11-black-64-gb/product-
reviews/itm4e5041ba101fd?pid=MOBFWQ6BXGJCEYNY&lid=LSTMOBFWQ6BXGJCEYNYZXSH
RJ&marketplace=FLIPKART"
driver.get(url)

# Wait for the page to load
time.sleep(5)

# Lists to store scraped data
ratings = []
review_summaries = []
full_reviews = []

# Scrape reviews until we get 100 reviews
while len(ratings) < 100:
    # Find all the review blocks on the page
    reviews = driver.find_elements(By.CLASS_NAME, '_1AtVbE')

    for review in reviews:
        # Rating
        try:
            rating = review.find_element(By.CLASS_NAME, '_3LWZlK').text
        except:
            rating = None

        # Review Summary
        try:
            review_summary = review.find_element(By.CLASS_NAME, '_2-N8zT').text
        except:
            review_summary = None

        # Full Review
        try:
            full_review = review.find_element(By.CLASS_NAME, 't-ZTKy').text
        except:
            full_review = None

        # Append data if the review has valid information
        if rating and review_summary and full_review:
            ratings.append(rating)
            review_summaries.append(review_summary)
            full_reviews.append(full_review)
```

```
        # Stop if we have collected 100 reviews
        if len(ratings) >= 100:
            break

    # Click the "Next" button to go to the next page of reviews
    try:
        next_button = driver.find_element(By.CLASS_NAME, '_1LKTO3')
        next_button.click()
        time.sleep(3)  # Wait for the next page to load
    except:
        print("No more pages or unable to click the Next button.")
        break

# Step 6: Create a DataFrame from the scraped data
df = pd.DataFrame({
    "Rating": ratings[:100],  # Take only the first 100 if we overshoot
    "Review Summary": review_summaries[:100],
    "Full Review": full_reviews[:100]
})

# Print the DataFrame
print(df)

# Save the DataFrame to a CSV file
df.to_csv("iphone11_flipkart_reviews.csv", index=False)

# Close the browser
driver.quit()
```

Q4: Scrape data forfirst 100 sneakers you find whenyouvisitflipkart.com and search for “sneakers” inthe search
field.
You have to scrape 3 attributes of each sneaker:
1. Brand
2. ProductDescription
3. Price
As shown in the below image, you have to scrape the above attributes.

```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.chrome.service import Service
from webdriver_manager.chrome import ChromeDriverManager
import pandas as pd
import time
```

```
# Initialize the browser
driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()))

# Step 1: Open Flipkart and dismiss the login popup
driver.get("https://www.flipkart.com/")
time.sleep(3)

# Close the login popup if it appears
try:
    close_popup = driver.find_element(By.XPATH, '//button[contains(text(),"✕")]')
    close_popup.click()
except:
    pass

# Step 2: Search for "sneakers"
search_box = driver.find_element(By.NAME, "q")
search_box.send_keys("sneakers")
search_box.submit()

# Wait for the results to load
time.sleep(5)

# Lists to store scraped data
brands = []
product_descriptions = []
prices = []

# Step 3: Scrape the first 100 results
while len(brands) < 100:
    # Find all sneaker items on the page
    products = driver.find_elements(By.XPATH, "//div[@class='_1AtVbE']")

    for product in products:
        try:
            # Scrape brand name
            brand = product.find_element(By.CLASS_NAME, "_2WkVRV").text
        except:
            brand = None

        try:
            # Scrape product description
            product_description = product.find_element(By.CLASS_NAME, "IRpwTa").text
        except:
            product_description = None
```

```
        try:
            # Scrape price
            price = product.find_element(By.CLASS_NAME, "_30jeq3").text
        except:
            price = None

        # Append data only if all fields are available
        if brand and product_description and price:
            brands.append(brand)
            product_descriptions.append(product_description)
            prices.append(price)

        # Stop if we have collected 100 products
        if len(brands) >= 100:
            break

    # Step 4: Click the "Next" button to load the next page if we don't have 100 products yet
    if len(brands) < 100:
        try:
            next_button = driver.find_element(By.XPATH, "//a[@class='_1LKTO3'][2]")  # Select
the second button for "Next"
            next_button.click()
            time.sleep(5)  # Wait for the next page to load
        except:
            print("No more pages or unable to click the Next button.")
            break

# Step 5: Create a DataFrame from the scraped data
df = pd.DataFrame({
    "Brand": brands[:100],  # Taking only the first 100 if more data was scraped
    "Product Description": product_descriptions[:100],
    "Price": prices[:100]
})

# Print the DataFrame
print(df)

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

# Close the browser
driver.quit()
```

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:

Aftersetting the filters scrape first 10 laptops data. You have to scrape 3 attributes for each laptop:
1. Title
2. Ratings
3. Price

```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.chrome.service import Service
from webdriver_manager.chrome import ChromeDriverManager
import pandas as pd
import time

# Initialize the WebDriver
driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()))

# Step 1: Open Amazon India
driver.get("https://www.amazon.in/")
time.sleep(3)

# Step 2: Search for "Laptop"
search_box = driver.find_element(By.ID, "twotabsearchtextbox")
search_box.send_keys("Laptop")
search_box.submit()
time.sleep(3)

# Step 3: Apply the filter for "Intel Core i7"
try:
    # Scroll to CPU Type filter section
    driver.execute_script("window.scrollBy(0, 3000);")
    time.sleep(2)

    # Find the filter checkbox for "Intel Core i7"
    intel_i7_filter = driver.find_element(By.XPATH, "//span[contains(text(), 'Intel Core i7')]")
    intel_i7_filter.click()
    time.sleep(5)  # Wait for the filter to apply and page to reload
except:
    print("Unable to apply 'Intel Core i7' filter.")
    driver.quit()

# Lists to store scraped data
titles = []
ratings = []
prices = []

# Step 4: Scrape the first 10 laptop results
```

```
laptop_count = 0
while laptop_count < 10:
    # Find all laptop elements on the page
    laptops = driver.find_elements(By.XPATH, "//div[@data-component-type='s-search-result']")

    for laptop in laptops:
        if laptop_count >= 10:
            break

        try:
            # Title of the laptop
            title = laptop.find_element(By.XPATH, ".//span[@class='a-size-medium a-color-base a-text-normal']").text
        except:
            title = None

        try:
            # Ratings of the laptop
            rating = laptop.find_element(By.XPATH, ".//span[@class='a-icon-alt']").text
        except:
            rating = None

        try:
            # Price of the laptop
            price = laptop.find_element(By.XPATH, ".//span[@class='a-price-whole']").text
        except:
            price = None

        # Append the data if all fields are present
        if title and price:
            titles.append(title)
            ratings.append(rating)
            prices.append(price)
            laptop_count += 1

    # Step 5: Click the "Next" button if we don't have 10 laptops yet
    if laptop_count < 10:
        try:
            next_button = driver.find_element(By.XPATH, "//a[@class='s-pagination-item s-pagination-next']")
            next_button.click()
            time.sleep(5)  # Wait for the next page to load
        except:
            print("No more pages available or unable to click the next button.")
            break
```

```
# Step 6: Create a DataFrame from the scraped data
df = pd.DataFrame({
    "Title": titles,
    "Ratings": ratings,
    "Price": prices
})

# Print the DataFrame
print(df)

# Optionally save the DataFrame to a CSV file
df.to_csv("amazon_laptops_intel_i7.csv", index=False)

# Close the browser
driver.quit()
```

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 webpagehttps://www.azquotes.com/
2. Click on TopQuote
3. Than scrap a)Quote b) Author c) Type Of Quotes

```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.chrome.service import Service
from webdriver_manager.chrome import ChromeDriverManager
import pandas as pd
import time

# Initialize the WebDriver
driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()))

# Step 1: Open the azquotes website
driver.get("https://www.azquotes.com/")
time.sleep(3)

# Step 2: Click on the "Top Quotes" link
try:
    top_quotes_link = driver.find_element(By.LINK_TEXT, "Top Quotes")
    top_quotes_link.click()
    time.sleep(3)
except:
    print("Unable to find 'Top Quotes' link.")
    driver.quit()

# Lists to store scraped data
quotes = []
```

```
authors = []
types = []

# Step 3: Scrape the quotes, authors, and types
while len(quotes) < 1000:
    # Find all quote containers on the current page
    quote_elements = driver.find_elements(By.XPATH, "//div[@class='wrap block-grid-item']")

    for quote_element in quote_elements:
        try:
            # Scrape the quote
            quote = quote_element.find_element(By.CLASS_NAME, "title").text
        except:
            quote = None

        try:
            # Scrape the author
            author = quote_element.find_element(By.CLASS_NAME, "author").text
        except:
            author = None

        try:
            # Scrape the type of quote
            quote_type = quote_element.find_element(By.CLASS_NAME, "tags").text
        except:
            quote_type = None

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

        # Stop if we've scraped 1000 quotes
        if len(quotes) >= 1000:
            break

    # Step 4: Click the "Next" button to go to the next page if we need more quotes
    if len(quotes) < 1000:
        try:
            next_button = driver.find_element(By.LINK_TEXT, "Next")
            next_button.click()
            time.sleep(3)  # Wait for the next page to load
        except:
            print("No more pages available or unable to click the 'Next' button.")
            break
```

```
# Step 5: Create a DataFrame from the scraped data
df = pd.DataFrame({
    "Quote": quotes[:1000],  # Ensure we only have the first 1000 quotes
    "Author": authors[:1000],
    "Type of Quote": types[:1000]
})

# Print the DataFrame
print(df)

# Optionally save the DataFrame to a CSV file
df.to_csv("top_1000_quotes.csv", index=False)

# Close the browser
driver.quit()
```

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/general-knowledge/list-of-
all-prime-ministers-of-india-1473165149-1

scrap the mentioned data and make the DataFrame

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

# URL of the webpage to scrape
url = "https://www.jagranjosh.com/general-knowledge/list-of-all-prime-ministers-of-india-1473165149-1"

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

# Check if the request was successful
if response.status_code == 200:
    print("Successfully accessed the page!")
else:
    print(f"Failed to retrieve page with status code: {response.status_code}")

# Parse the webpage content using BeautifulSoup
soup = BeautifulSoup(response.text, 'html.parser')

# Find the table containing the list of Prime Ministers
table = soup.find('table')
```

```
# Create empty lists to store the scraped data
names = []
born_dead = []
term_of_office = []
remarks = []

# Step through each row of the table (excluding the header row)
for row in table.find_all('tr')[1:]:
    cols = row.find_all('td')

    if len(cols) >= 4:  # Ensure there are enough columns in the row
        # Extract and clean the data from each column
        name = cols[0].text.strip()
        born = cols[1].text.strip()
        term = cols[2].text.strip()
        remark = cols[3].text.strip()

        # Append the data to the respective lists
        names.append(name)
        born_dead.append(born)
        term_of_office.append(term)
        remarks.append(remark)

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

# Display the DataFrame
print(df)

# Optionally, save the DataFrame to a CSV file
df.to_csv("former_prime_ministers_of_india.csv", index=False)
```

Q8. 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 webpage https://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 thementioned data and make the dataframe.

```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.chrome.service import Service
from webdriver_manager.chrome import ChromeDriverManager
import pandas as pd
import time

# Step 1: Initialize the WebDriver
driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()))

# Step 2: Open the motor1 website
driver.get("https://www.motor1.com/")
time.sleep(3)

# Step 3: Type '50 most expensive cars' in the search bar and hit Enter
search_bar = driver.find_element(By.NAME, "query")
search_bar.send_keys("50 most expensive cars")
search_bar.submit()
time.sleep(3)

# Step 4: Click on the article about '50 most expensive cars in the world'
article_link = driver.find_element(By.PARTIAL_LINK_TEXT, "50 Most Expensive Cars")
article_link.click()
time.sleep(5)

# Step 5: Scrape the car name and price data
cars = []
prices = []

# Find all car elements (assuming they are in list items or divs with a specific class)
# Note: Depending on the structure of the article, you might need to adjust the XPath or CSS selector here
car_elements = driver.find_elements(By.XPATH, "//h2")  # Example XPath for car name; change as per actual structure

for car_element in car_elements:
    try:
        # Scrape the car name and price
        car_name = car_element.text.strip()
        car_price_element = car_element.find_element(By.XPATH, "following-sibling::p")  # Assuming price is in a sibling element
        car_price = car_price_element.text.strip()

        # Append to the respective lists
        cars.append(car_name)
        prices.append(car_price)
    except:
```

```
        continue

# Close the driver
driver.quit()

# Step 6: Create a DataFrame from the scraped data
df = pd.DataFrame({
    'Car Name': cars[:50],  # Only keep the first 50 entries
    'Price': prices[:50]
})

# Step 7: Display the DataFrame
print(df)

# Optionally, save the DataFrame to a CSV file
df.to_csv("50_most_expensive_cars.csv", index=False)
```