WEB SCRAPING-ASSIGNMENT3

1. Write a python program which searches all the product under a particular product from www.amazon.in. The product to be searched will be taken as input from user. For e.g. If user input is ‘guitar’. Then search for guitars.

Solution: import requests

```
from bs4 import BeautifulSoup


# Taking user input for product name to be searched
product_name = input("Enter the product name: ")


# URL of the Amazon search page with the product name as query
url = f"https://www.amazon.in/s?k={product_name}&ref=nb_sb_noss"


# Send a GET request to the URL and store the response in a variable
response = requests.get(url)


# Use BeautifulSoup to parse the HTML content of the response
soup = BeautifulSoup(response.content, "html.parser")


# Find all the div elements with class 's-result-item' which contain the product details
product_list = soup.find_all("div", class_="s-result-item")


# Loop through the product_list and extract the details of each product
for product in product_list:
    # Extracting the name of the product
    name = product.find("span", class_="a-size-medium").text.strip()


    # Extracting the price of the product
    try:
        price = product.find("span", class_="a-price-whole").text.strip()
    except AttributeError:
```

```
        price = "Not available"

    # Extracting the rating of the product
    try:
        rating = product.find("span", class_="a-icon-alt").text.strip()
    except AttributeError:
        rating = "Not available"

    # Printing the details of the product
    print("Name:", name)
    print("Price:", price)
    print("Rating:", rating)
    print("-"*50)
```

2. In the above question, now scrape the following details of each product listed in first 3 pages of your search results and save it in a data frame and csv. In case if any product has less than 3 pages in search results then scrape all the products available under that product name. Details to be scraped are: "Brand Name", "Name of the Product", "Price", "Return/Exchange", "Expected Delivery", "Availability" and “Product URL”. In case, if any of the details are missing for any of the product then replace it by “-“.

Solution:

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

# Taking user input for product name to be searched
product_name = input("Enter the product name: ")

# Initializing empty lists to store the details of each product
brand_list = []
name_list = []
price_list = []
return_exchange_list = []
```

```
delivery_list = []
availability_list = []
url_list = []

# Looping through the first 3 pages of the search results
for page in range(1, 4):
    # URL of the Amazon search page with the product name as query and page number as parameter
    url = f"https://www.amazon.in/s?k={product_name}&page={page}&ref=nb_sb_noss"

    # Send a GET request to the URL and store the response in a variable
    response = requests.get(url)

    # Use BeautifulSoup to parse the HTML content of the response
    soup = BeautifulSoup(response.content, "html.parser")

    # Find all the div elements with class 's-result-item' which contain the product details
    product_list = soup.find_all("div", class_="s-result-item")

    # Loop through the product_list and extract the details of each product
    for product in product_list:
        # Extracting the brand name of the product
        try:
            brand = product.find("span", class_="a-size-base-plus a-color-base").text.strip()
        except AttributeError:
            brand = "-"
        brand_list.append(brand)

        # Extracting the name of the product
        try:
            name = product.find("span", class_="a-size-medium a-color-base a-text-normal").text.strip()
        except AttributeError:
```

```
    name = "-"
name_list.append(name)

# Extracting the price of the product
try:
    price = product.find("span", class_="a-price-whole").text.strip()
except AttributeError:
    price = "-"
price_list.append(price)

# Extracting the return/exchange policy of the product
try:
    return_exchange = product.find("div", class_="a-row a-size-small").text.strip()
except AttributeError:
    return_exchange = "-"
return_exchange_list.append(return_exchange)

# Extracting the expected delivery date of the product
try:
    delivery = product.find("span", class_="a-text-bold").text.strip()
except AttributeError:
    delivery = "-"
delivery_list.append(delivery)

# Extracting the availability status of the product
try:
    availability = product.find("span", class_="a-size-base a-color-secondary").text.strip()
except AttributeError:
    availability = "-"
availability_list.append(availability)
```

```
        # Extracting the URL of the product
        try:
            url = "https://www.amazon.in" + product.find("a", class_="a-link-normal s-no-outline")["href"]
        except TypeError:
            url = "-"
        url_list.append(url)

# Creating a dictionary with the details of each product
product_dict = {
    "Brand Name": brand_list,
    "Name of the Product": name_list,
    "Price": price_list,
    "Return/Exchange": return_exchange_list,
    "Expected Delivery": delivery_list,
    "Availability": availability_list,
    "Product URL": url_list
}

# Creating a pandas dataframe from the dictionary
df = pd.DataFrame(product_dict)

# Saving the dataframe to a CSV file
df.to_csv(f"{product_name}_
```

3. Write a python program to access the search bar and search button on images.google.com and scrape 10 images each for keywords ‘fruits’, ‘cars’ and ‘Machine Learning’, ‘Guitar’, ‘Cakes’.

Solution:

```
from selenium import webdriver
import urllib.request
import time

# Initializing the webdriver and opening the Google Images website
```

```
driver = webdriver.Chrome()
driver.get("https://images.google.com")

# List of keywords to be searched for
keywords = ['fruits', 'cars', 'Machine Learning', 'Guitar', 'Cakes']

# Looping through the keywords and searching for each of them
for keyword in keywords:
    # Finding the search bar element and entering the keyword
    search_bar = driver.find_element_by_name("q")
    search_bar.clear()
    search_bar.send_keys(keyword)

    # Clicking the search button
    search_button = driver.find_element_by_xpath("//button[@type='submit']")
    search_button.click()

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

    # Scrolling down the page to load more images
    for i in range(2):
        driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")
        time.sleep(5)

    # Finding the image elements and extracting their source URLs
    image_elements = driver.find_elements_by_xpath("//img[@class='rg_i Q4LuWd']")
    image_urls = [elem.get_attribute('src') for elem in image_elements]

    # Downloading the first 10 images
    for i in range(10):
```

```
        try:
            # Opening the URL and saving the image to a file
            urllib.request.urlretrieve(image_urls[i], f"{keyword}_{i+1}.jpg")
        except:
            pass

# Closing the webdriver
driver.quit()
```

4. Write a python program to search for a smartphone(e.g.: Oneplus Nord, pixel 4A, etc.) on www.flipkart.com and scrape following details for all the search results displayed on 1st page. Details to be scraped: “Brand Name”, “Smartphone name”, “Colour”, “RAM”, “Storage(ROM)”, “Primary Camera”, “Secondary Camera”, “Display Size”, “Battery Capacity”, “Price”, “Product URL”. Incase if any of the details is missing then replace it by “- “. Save your results in a dataframe and CSV.

Solution:

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

# Function to scrape the details of a smartphone
def scrape_smartphone(smartphone):
    # Creating the URL for the given smartphone query
    url = f"https://www.flipkart.com/search?q={smartphone}&otracker=search&otracker1=search&marketplace=FLIPKART&as-show=on&sort=popularity"

    # Sending a GET request to the URL and getting the HTML content
    response = requests.get(url)
    html_content = response.content

    # Parsing the HTML content using BeautifulSoup
    soup = BeautifulSoup(html_content, 'html.parser')

    # Finding all the smartphone details elements on the page
```

```
smartphone_details = soup.find_all('div', {'class': '_1AtVbE col-12-12'})

# Creating empty lists to store the details of each smartphone
brands = []
smartphone_names = []
colours = []
rams = []
roms = []
primary_cameras = []
secondary_cameras = []
display_sizes = []
battery_capacities = []
prices = []
urls = []

# Looping through the smartphone details elements and extracting their details
for smartphone_detail in smartphone_details[1:]:
    # Finding the brand name
    brand_name = smartphone_detail.find('div', {'class': '_4rR01T'}).text
    brands.append(brand_name)

    # Finding the smartphone name
    smartphone_name = smartphone_detail.find('a', {'class': '_1fQZEK'}).text
    smartphone_names.append(smartphone_name)

    # Finding the smartphone color
    colour = smartphone_detail.find('div', {'class': '_3tWrsu'}).text.split('|')[0].strip()
    colours.append(colour)

    # Finding the smartphone RAM
    ram = smartphone_detail.find('div', {'class': '_4rR01T'}).text.split('|')[1].strip()
```

```
        rams.append(ram)

        # Finding the smartphone ROM
        rom = smartphone_detail.find('div', {'class': '_4rR01T'}).text.split('|')[2].strip()
        roms.append(rom)

        # Finding the primary camera details
        try:
            primary_camera = smartphone_detail.find('ul', {'class':
'_1xgFaf'}).find_all('li')[0].text.split(':')[1].strip()
        except:
            primary_camera = '-'
        primary_cameras.append(primary_camera)

        # Finding the secondary camera details
        try:
            secondary_camera = smartphone_detail.find('ul', {'class':
'_1xgFaf'}).find_all('li')[1].text.split(':')[1].strip()
        except:
            secondary_camera = '-'
        secondary_cameras.append(secondary_camera)

        # Finding the display size
        display_size = smartphone_detail.find('div', {'class': '_4rR01T'}).text.split('|')[0].strip()
        display_sizes.append(display_size)

        # Finding the battery capacity
        try:
            battery_capacity = smartphone_detail.find('ul', {'class':
'_1xgFaf'}).find_all('li')[3].text.split(':')[1].strip()
        except:
            battery_capacity = '-'
```

```
    battery_capacities.append(battery_capacity)

    # Finding the price
    try:
        price = smartphone_detail.find('div', {'class': '_30jeq3 _1_WHN1'}).
```

5. Write a program to scrap geospatial coordinates (latitude, longitude) of a city searched on google maps.

Solution:

```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys

# Function to get the geospatial coordinates of a city
def get_coordinates(city):
    # Create a new instance of Chrome driver
    driver = webdriver.Chrome()

    # Navigate to Google Maps website
    driver.get("https://www.google.com/maps")

    # Find the search box and enter the city name
    search_box = driver.find_element_by_name("q")
    search_box.send_keys(city)

    # Press enter to search
    search_box.send_keys(Keys.ENTER)

    # Wait for the page to load and get the URL
    driver.implicitly_wait(10)
    url = driver.current_url

    # Extract the latitude and longitude from the URL
```

```
    coordinates = url.split('@')[1].split(',')[0:2]
    latitude = coordinates[0]
    longitude = coordinates[1]

    # Close the driver
    driver.quit()

    # Return the latitude and longitude
    return latitude, longitude
```

6. Write a program to scrap details of all the funding deals for second quarter (i.e Jan 21 – March 21) from trak.in.

Solution:

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

# Define the URL to scrape
url = "https://trak.in/india-startup-funding-investment-2015/"

# Make a GET request to the URL
response = requests.get(url)

# Create a BeautifulSoup object to parse the HTML content
soup = BeautifulSoup(response.content, 'html.parser')

# Find the table containing the funding deals
table = soup.find('table', {'class': 'tablepress tablepress-id-48'})

# Create an empty list to store the data
data = []

# Loop through all the rows in the table (excluding the header row)
```

```
for row in table.find_all('tr')[1:]:
    # Extract the data from the row
    columns = row.find_all('td')
    date = columns[0].get_text()

    # Check if the row is within the second quarter of 2021
    if "Jan" in date or "Feb" in date or "Mar" in date:
        startup_name = columns[1].get_text()
        industry = columns[2].get_text()
        sub_vertical = columns[3].get_text()
        city = columns[4].get_text()
        investor_name = columns[5].get_text()
        investment_type = columns[6].get_text()
        amount_in_usd = columns[7].get_text()

        # Append the data to the list
        data.append([date, startup_name, industry, sub_vertical, city, investor_name, investment_type, amount_in_usd])

# Create a pandas dataframe from the list of data
df = pd.DataFrame(data, columns=['Date', 'Startup Name', 'Industry', 'Sub-Vertical', 'City', 'Investor Name', 'Investment Type', 'Amount (in USD)'])

# Save the dataframe to a CSV file
df.to_csv('funding_deals_Q2_2021.csv', index=False)

# Print a message indicating the number of rows of data scraped and saved
print(f"Scraped {len(data)} rows of funding deals data and saved to 'funding_deals_Q2_2021.csv'")
```

7. Write a program to scrap all the available details of best gaming laptops from digit.in

Solution:

```
import requests
from bs4 import BeautifulSoup
```

```
import pandas as pd

# Define the URL to scrape
url = "https://www.digit.in/top-products/best-gaming-laptops-40.html"

# Make a GET request to the URL
response = requests.get(url)

# Create a BeautifulSoup object to parse the HTML content
soup = BeautifulSoup(response.content, 'html.parser')

# Find the div containing the list of laptops
laptop_div = soup.find('div', {'class': 'right-container'})

# Create an empty list to store the data
data = []

# Loop through all the laptops in the list
for laptop in laptop_div.find_all('div', {'class': 'TopNumbeHeading'}):
    # Extract the data for the laptop
    laptop_name = laptop.find('div', {'class': 'Top-RHS'}).get_text().strip()
    laptop_specs = laptop.find_all('div', {'class': 'spec'})
    laptop_price = laptop.find('div', {'class': 'Price-Slider'}).get_text().strip()

    # Create a dictionary to store the laptop data
    laptop_data = {'Name': laptop_name, 'Price': laptop_price}

    # Loop through all the specifications for the laptop
    for spec in laptop_specs:
        spec_name = spec.find('div', {'class': 'heading'}).get_text().strip()
        spec_value = spec.find('div', {'class': 'value'}).get_text().strip()
```

```
        laptop_data[spec_name] = spec_value

    # Append the data for the laptop to the list
    data.append(laptop_data)

# Create a pandas dataframe from the list of data
df = pd.DataFrame(data)

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

# Print a message indicating the number of laptops scraped and saved
print(f"Scraped {len(data)} laptops and saved to 'best_gaming_laptops.csv'")
```

8. Write a python program to scrape the details for all billionaires from www.forbes.com. Details to be scrapped: “Rank”, “Name”, “Net worth”, “Age”, “Citizenship”, “Source”, “Industry”.

Solution:

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

# Define the URL to scrape
url = "https://www.forbes.com/billionaires/"

# Make a GET request to the URL
response = requests.get(url)

# Create a BeautifulSoup object to parse the HTML content
soup = BeautifulSoup(response.content, 'html.parser')

# Find the table containing the list of billionaires
table = soup.find('table', {'class': 'table'})
```

```
# Create an empty list to store the data
data = []

# Loop through all the rows in the table, skipping the header row
for row in table.find_all('tr')[1:]:
    # Extract the data for the row
    rank = row.find('td', {'class': 'rank'}).get_text().strip()
    name = row.find('td', {'class': 'name'}).get_text().strip()
    net_worth = row.find('td', {'class': 'netWorth'}).get_text().strip()
    age = row.find('td', {'class': 'age'}).get_text().strip()
    citizenship = row.find('td', {'class': 'countryOfCitizenship'}).get_text().strip()
    source = row.find('td', {'class': 'source'}).get_text().strip()
    industry = row.find('td', {'class': 'category'}).get_text().strip()

    # Create a dictionary to store the row data
    row_data = {'Rank': rank, 'Name': name, 'Net worth': net_worth,
            'Age': age, 'Citizenship': citizenship, 'Source': source,
            'Industry': industry}

    # Append the data for the row to the list
    data.append(row_data)

# Create a pandas dataframe from the list of data
df = pd.DataFrame(data)

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

# Print a message indicating the number of rows scraped and saved
print(f"Scraped {len(data)} rows and saved to 'billionaires.csv'")
```

9. Write a program to extract at least 500 Comments, Comment upvote and time when comment

was posted from any YouTube Video.

Solution: import os

```
import google_auth_oauthlib.flow

import googleapiclient.errors

from googleapiclient.discovery import build

from datetime import datetime, timedelta


# Set up the YouTube API client

api_service_name = "youtube"

api_version = "v3"

api_key = "YOUR_API_KEY"

youtube = build(api_service_name, api_version, developerKey=api_key)


# Set the video ID of the video we want to extract comments from

video_id = "VIDEO_ID"


# Define a function to extract the desired information for each comment

def process_comment(comment):

    comment_text = comment["snippet"]["textDisplay"]

    comment_upvotes = comment["snippet"]["likeCount"]

    comment_published_time = datetime.strptime(comment["snippet"]["publishedAt"], "%Y-%m-
%dT%H:%M:%S.%fZ")

    return comment_text, comment_upvotes, comment_published_time


# Set up variables to store the comments and the next page token

comments = []

next_page_token = None


# Loop until we have at least 500 comments

while len(comments) < 500:

    # Retrieve the comments for the video using the video ID and the next page token (if any)
```

```
    results = youtube.commentThreads().list(
        part="snippet",
        videoId=video_id,
        textFormat="plainText",
        maxResults=100,
        pageToken=next_page_token
    ).execute()

    # Extract the comments and the next page token from the results
    for item in results["items"]:
        comment = item["snippet"]["topLevelComment"]
        comment_text, comment_upvotes, comment_published_time = process_comment(comment)
        comments.append((comment_text, comment_upvotes, comment_published_time))

    next_page_token = results.get("nextPageToken")

    # If there are no more comments, break out of the loop
    if next_page_token is None:
        break

# Print the number of comments extracted
print(f"Extracted {len(comments)} comments.")

# Convert the comments to a pandas dataframe
import pandas as pd
df = pd.DataFrame(comments, columns=["Comment Text", "Comment Upvotes", "Comment Published Time"])

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

```
# Print a message indicating where the CSV file was saved

print(f"Saved comments to: {os.path.abspath('comments.csv')}")
```

10. Write a python program to scrape a data for all available Hostels from https://www.hostelworld.com/ in “London” location. You have to scrape hostel name, distance from city centre, ratings, total reviews, overall reviews, privates from price, dorms from price, facilities and property description.

Solution:

```
import requests

from bs4 import BeautifulSoup

import pandas as pd


url = "https://www.hostelworld.com/hostels/London"


response = requests.get(url)

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


hostel_list = soup.find_all("div", {"class": "property-card"})


hostel_data = []


for hostel in hostel_list:
    name = hostel.find("h2", {"class": "title"}).text.strip()
    distance = hostel.find("span", {"class": "description"}).text.strip()
    rating = hostel.find("div", {"class": "score orange big"}).text.strip()
    reviews = hostel.find("div", {"class": "reviews"}).text.strip()
    overall_review = hostel.find("div", {"class": "keyword"}).text.strip()
    price_private = hostel.find("div", {"class": "price-col private-col"}).text.strip()
    price_dorm = hostel.find("div", {"class": "price-col dorm-col"}).text.strip()
    facilities = [i.text.strip() for i in hostel.find_all("div", {"class": "facilities"})]
    description = hostel.find("div", {"class": "description"}).text.strip()


    hostel_data.append({
```

```
        "Name": name,
        "Distance from City Centre": distance,
        "Rating": rating,
        "Total Reviews": reviews,
        "Overall Review": overall_review,
        "Private Room Price": price_private,
        "Dorm Room Price": price_dorm,
        "Facilities": facilities,
        "Description": description
    })

df = pd.DataFrame(hostel_data)
df.to_csv("hostels_london.csv", index=False)

print("Data has been scraped and saved to 'hostels_london.csv'")
```