1. Write a python program which searches all the product under a particular product from www.amazon.in (http://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.

In [ ]:
```
import requests
from bs4 import BeautifulSoup

def search_amazon_product(product_name):
    base_url = "https://www.amazon.in/s"
    params = {"k": product_name}

    response = requests.get(base_url, params=params)

    if response.status_code == 200:
        soup = BeautifulSoup(response.content, "html.parser")
        product_containers = soup.find_all("div", class_="s-result-item")

        for product_container in product_containers:
            product_title = product_container.find("h2").text.strip()
            product_price = product_container.find("span", class_="a-offscreen'
            if product_price:
                product_price = product_price.text
            else:
                product_price = "Price not available"

            print("Product:", product_title)
            print("Price:", product_price)
            print("=" * 50)
    else:
        print("Failed to fetch Amazon page")

if __name__ == "__main__":
    user_input = input("Enter the product to search for on Amazon.in: ")
    search_amazon_product(user_input)
```

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 “-“.

In [ ]:
```
import requests
from bs4 import BeautifulSoup
import pandas as pd
import time

def scrape_product_details(product_container):
    product_details = {}

    product_title = product_container.find("h2").text.strip()
    product_details["Product Name"] = product_title

    product_brand = product_container.find("span", class_="a-size-base-plus a-c
    if product_brand:
        product_details["Brand Name"] = product_brand.text.strip()
    else:
        product_details["Brand Name"] = "-"

    product_price = product_container.find("span", class_="a-offscreen")
    if product_price:
        product_details["Price"] = product_price.text
    else:
        product_details["Price"] = "-"

    product_return = product_container.find("div", class_="a-column a-span3 a-t
    product_details["Return/Exchange"] = product_return

    product_delivery = product_container.find("span", class_="a-text-bold").tex
    product_details["Expected Delivery"] = product_delivery

    product_availability = product_container.find("span", class_="a-declarative
    product_details["Availability"] = product_availability

    product_url = product_container.find("a", class_="a-link-normal")
    if product_url:
        product_details["Product URL"] = "https://www.amazon.in" + product_url[
    else:
        product_details["Product URL"] = "-"

    return product_details

def search_amazon_product(product_name):
    base_url = "https://www.amazon.in/s"
    params = {"k": product_name}

    product_data = []
    page_count = 0

    while len(product_data) < 3 * 16:  # Assuming 16 products per page
        response = requests.get(base_url, params=params)

        if response.status_code == 200:
            soup = BeautifulSoup(response.content, "html.parser")
            product_containers = soup.find_all("div", class_="s-result-item")

            for product_container in product_containers:
                product_details = scrape_product_details(product_container)
                product_data.append(product_details)
            next_page_link = soup.find("li", class_="a-last")
            if next_page_link:
                next_page_url = "https://www.amazon.in" + next_page_link.find('
                base_url = next_page_url
            else:
                break

            page_count += 1
            if page_count >= 3:
                break

            time.sleep(2)  # Adding a delay to avoid overloading the server
        else:
            print("Failed to fetch Amazon page")
            break

    return product_data

if __name__ == "__main__":
    user_input = input("Enter the product to search for on Amazon.in: ")
    product_data = search_amazon_product(user_input)

    # Creating a DataFrame from the scraped data
    df = pd.DataFrame(product_data)

    # Saving the DataFrame to a CSV file
    csv_filename = f"{user_input}_products.csv"
    df.to_csv(csv_filename, index=False)

    print(f"Scraped data saved to {csv_filename}")
```

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’.

In [ ]:
```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
import time

# Create a new instance of the Chrome browser (you need to have Chrome and chro
driver = webdriver.Chrome()

# Open Google Images
driver.get("https://www.google.com/imghp")

keywords = ['fruits', 'cars', 'Machine Learning', 'Guitar', 'Cakes']
num_images_to_scrape = 10

for keyword in keywords:
    # Find the search bar element and send the keyword
    search_bar = driver.find_element_by_name("q")
    search_bar.clear()
    search_bar.send_keys(keyword)
    search_bar.send_keys(Keys.RETURN)

    # Wait for search results to load
    time.sleep(2)

    # Scroll down to load more images (optional)
    # driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")

    # Collect image elements
    image_elements = driver.find_elements_by_css_selector(".rg_i")

    # Loop through image elements and download images
    for i, img_element in enumerate(image_elements[:num_images_to_scrape]):
        img_url = img_element.get_attribute("src")
        # You can use a library like requests to download the images
        # Here, we'll just print the URLs
        print(f"{keyword} Image {i+1}: {img_url}")

    # Wait before moving to the next keyword
    time.sleep(2)

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

4. Write a python program to search for a smartphone(e.g.: Oneplus Nord, pixel 4A, etc.) on www.flipkart.com (http://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

In [ ]:
```
from selenium import webdriver
from selenium.webdriver.common.by import By
import pandas as pd

# Create a new instance of the Chrome browser (you need to have Chrome and chro
driver = webdriver.Chrome()

def scrape_flipkart_smartphones(product_name):
    base_url = "https://www.flipkart.com"
    search_url = f"{base_url}/search?q={product_name}"

    driver.get(search_url)

    product_data = []

    product_containers = driver.find_elements(By.CSS_SELECTOR, "._1AtVbE")

    for product_container in product_containers:
        details = {
            "Brand Name": "-",
            "Smartphone Name": "-",
            "Colour": "-",
            "RAM": "-",
            "Storage(ROM)": "-",
            "Primary Camera": "-",
            "Secondary Camera": "-",
            "Display Size": "-",
            "Battery Capacity": "-",
            "Price": "-",
            "Product URL": "-"
        }

        product_title_element = product_container.find_element(By.CSS_SELECTOR,
        details["Smartphone Name"] = product_title_element.text

        product_link_element = product_container.find_element(By.CSS_SELECTOR,
        details["Product URL"] = base_url + product_link_element.get_attribute(

        try:
            product_price_element = product_container.find_element(By.CSS_SELE
            details["Price"] = product_price_element.text
        except:
            pass

        try:
            rating_element = product_container.find_element(By.CSS_SELECTOR, "c
            details["Rating"] = rating_element.text
        except:
            pass

        product_data.append(details)

    return product_data

if __name__ == "__main__":
    product_name = input("Enter the smartphone name to search for on Flipkart:
    product_data = scrape_flipkart_smartphones(product_name)
    # Creating a DataFrame from the scraped data
    df = pd.DataFrame(product_data)

    # Saving the DataFrame to a CSV file
    csv_filename = f"{product_name}_smartphones.csv"
    df.to_csv(csv_filename, index=False)

    print(f"Scraped data saved to {csv_filename}")

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

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

In [ ]:

```
import requests

def get_coordinates(city_name):
    api_key = "YOUR_GOOGLE_MAPS_API_KEY"
    base_url = "https://maps.googleapis.com/maps/api/geocode/json"

    params = {
        "address": city_name,
        "key": api_key
    }

    response = requests.get(base_url, params=params)
    data = response.json()

    if data["status"] == "OK":
        location = data["results"][0]["geometry"]["location"]
        latitude = location["lat"]
        longitude = location["lng"]
        return latitude, longitude
    else:
        print("Error:", data.get("status", "Unknown error"))

if __name__ == "__main__":
    city = input("Enter the name of the city: ")
    coordinates = get_coordinates(city)

    if coordinates:
        print(f"Coordinates of {city}: Latitude = {coordinates[0]}, Longitude =
```

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

In [ ]:
```
import requests
from bs4 import BeautifulSoup

def get_laptop_details(url):
    response = requests.get(url)
    if response.status_code == 200:
        soup = BeautifulSoup(response.content, 'html.parser')

        laptop_list = []

        laptops = soup.find_all('div', class_='TopNumbeHeading sticky-footer')
        for laptop in laptops:
            laptop_details = {}

            name = laptop.find('div', class_='TopNumbeHeading sticky-footer').t
            specs = laptop.find('div', class_='TopNumbeSecondHeading sticky-foo

            laptop_details['Name'] = name
            laptop_details['Specifications'] = specs

            laptop_list.append(laptop_details)

        return laptop_list
    else:
        print("Failed to retrieve the webpage.")
        return []

if __name__ == "__main__":
    url = 'https://www.digit.in/top-products/best-gaming-laptops-40.html'
    laptop_details = get_laptop_details(url)

    for index, laptop in enumerate(laptop_details, start=1):
        print(f"Laptop {index}:")
        print(f"Name: {laptop['Name']}")
        print(f"Specifications: {laptop['Specifications']}")
        print("=" * 50)
```

7. Write a python program to scrape the details for all billionaires from www.forbes.com (http://www.forbes.com). Details to be scrapped:

“Rank”, “Name”, “Net worth”, “Age”, “Citizenship”, “Source”, “Industry”.

In [ ]:
```
import requests
from bs4 import BeautifulSoup

def scrape_forbes_billionaires():
    url = "https://www.forbes.com/billionaires/"
    response = requests.get(url)

    if response.status_code == 200:
        soup = BeautifulSoup(response.content, "html.parser")

        billionaires = []
        rows = soup.select(".data")[0].find_all("tr")

        for row in rows[1:]:  # Skipping the header row
            columns = row.find_all("td")
            rank = columns[0].text.strip()
            name = columns[1].text.strip()
            net_worth = columns[2].text.strip()
            age = columns[3].text.strip()
            citizenship = columns[4].text.strip()
            source = columns[5].text.strip()
            industry = columns[6].text.strip()

            billionaire = {
                "Rank": rank,
                "Name": name,
                "Net worth": net_worth,
                "Age": age,
                "Citizenship": citizenship,
                "Source": source,
                "Industry": industry
            }

            billionaires.append(billionaire)

        return billionaires
    else:
        print("Failed to fetch data")
        return []

if __name__ == "__main__":
    billionaires_list = scrape_forbes_billionaires()

    for billionaire in billionaires_list:
        print("Billionaire Details:")
        for key, value in billionaire.items():
            print(f"{key}: {value}")
        print("=" * 30)
```

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

from any YouTube Video

In [ ]:
```
import os
import json
from googleapiclient.discovery import build

# Set your API key here
API_KEY = "YOUR_YOUTUBE_API_KEY"
VIDEO_ID = "YOUTUBE_VIDEO_ID"

# Create a YouTube Data API client
youtube = build('youtube', 'v3', developerKey=API_KEY)

def get_video_comments(youtube, **kwargs):
    comments = []
    results = youtube.commentThreads().list(**kwargs).execute()

    while results:
        for item in results['items']:
            comment = item['snippet']['topLevelComment']['snippet']
            comments.append({
                'text': comment['textDisplay'],
                'time': comment['publishedAt'],
                'upvotes': comment['likeCount']
            })

        # Check if there are more comments
        if 'nextPageToken' in results:
            kwargs['pageToken'] = results['nextPageToken']
            results = youtube.commentThreads().list(**kwargs).execute()
        else:
            break

    return comments

# Fetch comments from the video
comments = get_video_comments(youtube, part='snippet', videoId=VIDEO_ID, textFo

# Save the comments to a JSON file
output_file = 'youtube_comments.json'
with open(output_file, 'w', encoding='utf-8') as f:
    json.dump(comments, f, ensure_ascii=False, indent=4)

print(f'Successfully extracted {len(comments)} comments and saved to {output_fi
```

9. Write a python program to scrape a data for all available Hostels from https://www.hostelworld.com/ (https://www.hostelworld.com/) in

"London" location. You have to scrape hostel name, distance from city centre, ratings, total

In [ ]:
```
import requests
from bs4 import BeautifulSoup
import re

# URL of the page to scrape
url = "https://www.hostelworld.com/search?search_keywords=London,%20England&cou

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

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

# Find all hostel containers
hostel_containers = soup.find_all('div', class_='hostel-container')

# Initialize a list to store hostel data
hostel_data = []

# Iterate through each hostel container
for container in hostel_containers:
    name = container.find('h2', class_='title').text.strip()
    distance = container.find('span', class_='distance').text.strip()
    rating = container.find('div', class_='score orange').text.strip()
    total_reviews = container.find('div', class_='reviews').text.strip()
    overall_reviews = container.find('div', class_='rating').text.strip()
    privates_price = container.find('div', class_='price-col').find('a', class_
    dorms_price = container.find('div', class_='price-col').find('a', class_='p
    facilities = [item.text.strip() for item in container.find_all('span', clas
    description = container.find('div', class_='ratings').find_next('p').text.s

    hostel_data.append({
        'Name': name,
        'Distance from City Centre': distance,
        'Rating': rating,
        'Total Reviews': total_reviews,
        'Overall Reviews': overall_reviews,
        'Privates from Price': privates_price,
        'Dorms from Price': dorms_price,
        'Facilities': facilities,
        'Description': description
    })

# Print the scraped data
for hostel in hostel_data:
    print(hostel)
    print('-' * 50)

# You can also save the data to a file (e.g., CSV or JSON) if needed
```