Q.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):
    # Replace spaces with '+' for the search URL
    search_query = product.replace(' ', '+')
    url = f"https://www.amazon.in/s?k={search_query}"

    # Set headers to mimic a real browser visit
    headers = {
        "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/5
    }

    # Make a request to the Amazon search URL
    response = requests.get(url, headers=headers)

    if response.status_code == 200:
        # Parse the page content
        soup = BeautifulSoup(response.content, 'html.parser')

        # Find all product listings on the page
        products = soup.find_all('div', {'data-component-type': 's-search-resu

        for product in products:
            # Extract product title
            title = product.h2.text.strip()

            # Extract product URL
            link = "https://www.amazon.in" + product.h2.a['href']

            # Extract product price if available
            price = product.find('span', 'a-price-whole')
            if price:
                price = price.text.strip()
            else:
                price = "Price not available"

            print(f"Title: {title}\nLink: {link}\nPrice: {price}\n")

    else:
        print(f"Failed to retrieve search results. Status code: {response.statu

if __name__ == "__main__":
    product_to_search = input("Enter the product to search on Amazon.in: ")
    search_amazon(product_to_search)
```

```
Enter the product to search on Amazon.in:
```

Q.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

In [3]:
```
import requests
from bs4 import BeautifulSoup
import pandas as pd

def get_product_details(product):
    headers = {
        "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/5
    }

    # List to store product details
    products_list = []

    for page in range(1, 4):  # Iterate through the first 3 pages
        url = f"https://www.amazon.in/s?k={product.replace(' ', '+')}&page={pa
        response = requests.get(url, headers=headers)

        if response.status_code != 200:
            print(f"Failed to retrieve search results. Status code: {response.
            break

        soup = BeautifulSoup(response.content, 'html.parser')
        products = soup.find_all('div', {'data-component-type': 's-search-resu

        if not products:
            break  # Exit if no products are found

        for product in products:
            try:
                title = product.h2.text.strip()
                link = "https://www.amazon.in" + product.h2.a['href']
                brand = product.find('span', 'a-size-base-plus').text.strip() 
                price = product.find('span', 'a-price-whole').text.strip() if 
                availability = product.find('span', 'a-color-success').text.st
                delivery = product.find('span', 'a-text-bold').text.strip() if
                return_exchange = product.find('span', 'a-declarative').text.s

                products_list.append({
                    "Brand Name": brand,
                    "Name of the Product": title,
                    "Price": price,
                    "Return/Exchange": return_exchange,
                    "Expected Delivery": delivery,
                    "Availability": availability,
                    "Product URL": link
                })

            except Exception as e:
                print(f"An error occurred: {e}")
                continue

    # Create DataFrame
    df = pd.DataFrame(products_list)

    # Save to CSV
    df.to_csv(f"{product}_amazon_products.csv", index=False)
    print(f"Saved {len(products_list)} products to {product}_amazon_products.c
if __name__ == "__main__":
    product_to_search = input("Enter the product to search on Amazon.in: ")
    get_product_details(product_to_search)
```

```
Enter the product to search on Amazon.in: samsung
Failed to retrieve search results. Status code: 503
Saved 0 products to samsung_amazon_products.csv
```

Q.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’.

In [5]:
```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
import time
import pandas as pd

def scrape_images(keywords, num_images=10):
    # Set up the WebDriver
    driver = webdriver.Chrome(executable_path='/path/to/chromedriver')
    driver.get("https://images.google.com")

    results = []

    for keyword in keywords:
        # Find the search bar and search for 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 results to load
        time.sleep(2)

        # Scroll to load more images
        for _ in range(5):  # Adjust this range to load more images
            driver.execute_script("window.scrollBy(0, document.body.scrollHeigl
            time.sleep(2)

        # Find image elements
        image_elements = driver.find_elements_by_css_selector("img.rg_i.Q4LuWd

        count = 0
        for image_element in image_elements:
            if count >= num_images:
                break
            try:
                image_element.click()
                time.sleep(2)
                large_image = driver.find_element_by_css_selector("img.n3VNCb"
                image_url = large_image.get_attribute("src")
                if image_url and 'http' in image_url:
                    results.append({
                        "Keyword": keyword,
                        "Image URL": image_url
                    })
                    count += 1
            except Exception as e:
                print(f"Could not retrieve image for {keyword}: {e}")
                continue

    driver.quit()

    # Convert results to DataFrame and save to CSV
    df = pd.DataFrame(results)
    df.to_csv("google_images_scraped.csv", index=False)
    print(f"Scraped {len(results)} images. Saved to google_images_scraped.csv"

if __name__ == "__main__":
    keywords = ['fruits', 'cars', 'Machine Learning', 'Guitar', 'Cakes']
    scrape_images(keywords)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[5], line 59
     57 if __name__ == "__main__":
     58     keywords = ['fruits', 'cars', 'Machine Learning', 'Guitar', 'Cake
s']
---> 59     scrape_images(keywords)

Cell In[5], line 8, in scrape_images(keywords, num_images)
      6 def scrape_images(keywords, num_images=10):
      7     # Set up the WebDriver
----> 8     driver = webdriver.Chrome(executable_path='/path/to/chromedrive
r')
      9     driver.get("https://images.google.com")
     11     results = []

TypeError: WebDriver.__init__() got an unexpected keyword argument 'executabl
e_path'
```

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 CSV.

In [6]:
```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
import time
import pandas as pd
from bs4 import BeautifulSoup

def scrape_flipkart(smartphone):
    # Set up the WebDriver
    driver = webdriver.Chrome(executable_path='/path/to/chromedriver')
    driver.get("https://www.flipkart.com")

    # Close the login popup if it appears
    try:
        close_popup = driver.find_element_by_css_selector("button._2KpZ6l._2do
        close_popup.click()
    except:
        pass

    # Find the search bar and search for the smartphone
    search_bar = driver.find_element_by_name("q")
    search_bar.clear()
    search_bar.send_keys(smartphone)
    search_bar.send_keys(Keys.RETURN)

    # Wait for results to load
    time.sleep(3)

    # Parse the page content
    soup = BeautifulSoup(driver.page_source, 'html.parser')
    products = soup.find_all('div', {'class': '_1AtVbE'})

    results = []

    for product in products:
        try:
            # Product URL
            product_url = "https://www.flipkart.com" + product.find('a', {'cla

            # Brand and smartphone name
            name_div = product.find('div', {'class': '_4rR01T'})
            brand_name = name_div.text.split()[0]
            smartphone_name = name_div.text

            # Extracting other details
            details_div = product.find('ul', {'class': '_1xgFaf'})
            details_list = details_div.find_all('li')

            details = {
                "Brand Name": brand_name,
                "Smartphone name": smartphone_name,
                "Colour": "-",
                "RAM": "-",
                "Storage(ROM)": "-",
                "Primary Camera": "-",
                "Secondary Camera": "-",
                "Display Size": "-",
                "Battery Capacity": "-",
                "Price": "-",
                "Product URL": product_url
            }

            for detail in details_list:
                text = detail.text
                if 'RAM' in text and 'ROM' in text:
                    details["RAM"] = text.split('|')[0].strip()
                    details["Storage(ROM)"] = text.split('|')[1].strip()
                elif 'Display' in text:
                    details["Display Size"] = text.split(' ')[0].strip()
                elif 'Battery' in text:
                    details["Battery Capacity"] = text.strip()
                elif 'Primary Camera' in text or 'Rear Camera' in text:
                    details["Primary Camera"] = text.strip()
                elif 'Secondary Camera' in text or 'Front Camera' in text:
                    details["Secondary Camera"] = text.strip()
                elif 'Color' in text or 'Colour' in text:
                    details["Colour"] = text.split(':')[1].strip()

            # Price
            price_div = product.find('div', {'class': '_30jeq3 _1_WHN1'})
            if price_div:
                details["Price"] = price_div.text

            results.append(details)
        except Exception as e:
            print(f"An error occurred: {e}")
            continue

    driver.quit()

    # Create DataFrame
    df = pd.DataFrame(results)

    # Save to CSV
    df.to_csv(f"{smartphone}_flipkart_products.csv", index=False)
    print(f"Saved {len(results)} products to {smartphone}_flipkart_products.cs

if __name__ == "__main__":
    smartphone_to_search = input("Enter the smartphone to search on Flipkart:
    scrape_flipkart(smartphone_to_search)
```

```
Enter the smartphone to search on Flipkart: Oneplus Nord
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[6], line 99
     97 if __name__ == "__main__":
     98     smartphone_to_search = input("Enter the smartphone to search on F
lipkart: ")
---> 99     scrape_flipkart(smartphone_to_search)

Cell In[6], line 9, in scrape_flipkart(smartphone)
      7 def scrape_flipkart(smartphone):
      8     # Set up the WebDriver
----> 9     driver = webdriver.Chrome(executable_path='/path/to/chromedrive
r')
     10     driver.get("https://www.flipkart.com")
     12     # Close the login popup if it appears

TypeError: WebDriver.__init__() got an unexpected keyword argument 'executabl
e_path'
```

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

In [12]:
```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
import time

def get_coordinates(city_name):
    # Set up the WebDriver
    driver = webdriver.Chrome(executable_path='/path/to/chromedriver')
    driver.get("https://www.google.com/maps")

    # Find the search bar and search for the city
    search_bar = driver.find_element_by_name("q")
    search_bar.clear()
    search_bar.send_keys(city_name)
    search_bar.send_keys(Keys.RETURN)

    # Wait for results to load
    time.sleep(3)

    # Extract the current URL
    current_url = driver.current_url

    # Extract coordinates from the URL
    try:
        if '/@' in current_url:
            lat_long_part = current_url.split('/@')[1].split(',')[0:2]
            latitude = lat_long_part[0]
            longitude = lat_long_part[1]
            print(f"Coordinates of {city_name}: Latitude = {latitude}, Longitu
        else:
            print("Coordinates not found in the URL.")
    except Exception as e:
        print(f"An error occurred: {e}")

    driver.quit()

if __name__ == "__main__":
    city_to_search = input("Enter the city to search on Google Maps: ")
    get_coordinates(city_to_search)
```

```
Enter the city to search on Google Maps: nagpur
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[12], line 38
     36 if __name__ == "__main__":
     37     city_to_search = input("Enter the city to search on Google Maps:
")
---> 38     get_coordinates(city_to_search)

Cell In[12], line 7, in get_coordinates(city_name)
      5 def get_coordinates(city_name):
      6     # Set up the WebDriver
----> 7     driver = webdriver.Chrome(executable_path='/path/to/chromedrive
r')
      8     driver.get("https://www.google.com/maps")
     10     # Find the search bar and search for the city

TypeError: WebDriver.__init__() got an unexpected keyword argument 'executabl
e_path'
```

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

In [13]:
```
import requests
from bs4 import BeautifulSoup
import pandas as pd

def get_gaming_laptops():
    url = "https://www.digit.in/top-products/best-gaming-laptops-40.html"
    headers = {
        "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/5
    }

    response = requests.get(url, headers=headers)

    if response.status_code != 200:
        print(f"Failed to retrieve the webpage. Status code: {response.status_
        return

    soup = BeautifulSoup(response.content, 'html.parser')
    laptops = soup.find_all('div', class_='TopNumbeHeading sticky-footer')

    laptop_details = []

    for laptop in laptops:
        try:
            title = laptop.find('h2', class_='heading-wraper').text.strip()
            specs = laptop.find('div', class_='product-detail')
            details = specs.text.strip().split('\n')
            details = [detail.strip() for detail in details if detail.strip()]

            detail_dict = {"Title": title}

            for detail in details:
                if ':' in detail:
                    key, value = detail.split(':', 1)
                    detail_dict[key.strip()] = value.strip()
                else:
                    detail_dict["Description"] = detail

            laptop_details.append(detail_dict)

        except Exception as e:
            print(f"An error occurred: {e}")
            continue

    # Create DataFrame
    df = pd.DataFrame(laptop_details)

    # Save to CSV
    df.to_csv("best_gaming_laptops_digit_in.csv", index=False)
    print(f"Saved {len(laptop_details)} laptops to best_gaming_laptops_digit_i

if __name__ == "__main__":
    get_gaming_laptops()
```

```
Saved 0 laptops to best_gaming_laptops_digit_in.csv
```

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 [14]:
```
import requests
from bs4 import BeautifulSoup
import pandas as pd

def get_billionaires():
    url = "https://www.forbes.com/billionaires/"
    headers = {
        "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/5
    }

    response = requests.get(url, headers=headers)

    if response.status_code != 200:
        print(f"Failed to retrieve the webpage. Status code: {response.status_
        return

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

    # Container for all billionaire details
    billionaires = []

    # Loop through the table and extract data
    rows = soup.select('div.table-row')
    for row in rows:
        try:
            rank = row.find('div', {'class': 'rank'}).text.strip()
            name = row.find('div', {'class': 'personName'}).text.strip()
            net_worth = row.find('div', {'class': 'netWorth'}).text.strip()
            age = row.find('div', {'class': 'age'}).text.strip() if row.find('
            citizenship = row.find('div', {'class': 'countryOfCitizenship'}).t
            source = row.find('div', {'class': 'source'}).text.strip()
            industry = row.find('div', {'class': 'category'}).text.strip()

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

        except Exception as e:
            print(f"An error occurred: {e}")
            continue

    # Create DataFrame
    df = pd.DataFrame(billionaires)

    # Save to CSV
    df.to_csv("forbes_billionaires.csv", index=False)
    print(f"Saved {len(billionaires)} billionaires to forbes_billionaires.csv"

if __name__ == "__main__":
    get_billionaires()
```

```
Saved 0 billionaires to forbes_billionaires.csv
```

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

In [15]:
```python
import os
import googleapiclient.discovery
import pandas as pd

# Replace with your API key
API_KEY = "YOUR_API_KEY"
VIDEO_ID = "YOUR_VIDEO_ID"

def get_comments(video_id, api_key, max_results=500):
    comments = []

    youtube = googleapiclient.discovery.build("youtube", "v3", developerKey=ap

    request = youtube.commentThreads().list(
        part="snippet",
        videoId=video_id,
        maxResults=100,
        textFormat="plainText"
    )

    response = request.execute()

    while response:
        for item in response['items']:
            comment = item['snippet']['topLevelComment']['snippet']
            comments.append({
                'Comment': comment['textDisplay'],
                'Upvotes': comment['likeCount'],
                'Timestamp': comment['publishedAt']
            })

            # Check if we have reached the max results
            if len(comments) >= max_results:
                return comments

        if 'nextPageToken' in response:
            request = youtube.commentThreads().list(
                part="snippet",
                videoId=video_id,
                pageToken=response['nextPageToken'],
                maxResults=100,
                textFormat="plainText"
            )
            response = request.execute()
        else:
            break

    return comments

def main():
    comments = get_comments(VIDEO_ID, API_KEY)

    # Convert to DataFrame
    df = pd.DataFrame(comments)

    # Save to CSV
    df.to_csv("youtube_comments.csv", index=False)
    print(f"Saved {len(comments)} comments to youtube_comments.csv")

if __name__ == "__main__":
    main()
```

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
Cell In[15], line 2
      1 import os
----> 2 import googleapiclient.discovery
      3 import pandas as pd
      5 # Replace with your API key

ModuleNotFoundError: No module named 'googleapiclient'
```

Q.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 reviews, overall reviews, privates from price, dorms from price, facilities and property description.

In [20]:
```
import requests
from bs4 import BeautifulSoup
import pandas as pd

def scrape_hostels_in_london():
    url = "https://www.hostelworld.com/"
    headers = {
        "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/5
    }

    response = requests.get(url, headers=headers)

    if response.status_code != 200:
        print(f"Failed to retrieve the webpage. Status code: {response.status_
        return

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

    # Container for all hostel details
    hostels = []

    # Extracting the hostel data from the page
    hostel_cards = soup.find_all('div', class_='fabresult')

    for hostel in hostel_cards:
        try:
            # Hostel name
            name = hostel.find('h2', class_='title').text.strip()

            # Distance from city centre
            distance = hostel.find('span', class_='description').text.strip()

            # Ratings and reviews
            rating = hostel.find('div', class_='score orange big').text.strip(
            total_reviews = hostel.find('div', class_='reviews').text.strip().
            overall_reviews = hostel.find('div', class_='keyword').text.strip(

            # Prices
            privates_from_price = hostel.find('a', class_='price privates').fi
            dorms_from_price = hostel.find('a', class_='price dorms').find('sp

            # Facilities
            facilities = [facility.text.strip() for facility in hostel.find_al

            # Property description
            property_description = hostel.find('div', class_='rating-factors p

            hostel_details = {
                "Hostel Name": name,
                "Distance from City Centre": distance,
                "Rating": rating,
                "Total Reviews": total_reviews,
                "Overall Reviews": overall_reviews,
                "Privates from Price": privates_from_price,
                "Dorms from Price": dorms_from_price,
                "Facilities": ', '.join(facilities),
                "Property Description": property_description
            }

            hostels.append(hostel_details)

        except Exception as e:
            print(f"An error occurred: {e}")
            continue

    # Create DataFrame
    df = pd.DataFrame(hostels)

    # Save to CSV
    df.to_csv("london_hostels.csv", index=False)
    print(f"Saved {len(hostels)} hostels to london_hostels.csv")

if __name__ == "__main__":
    scrape_hostels_in_london()
```

```
Saved 0 hostels to london_hostels.csv
```

In [21]:
```python
import requests
from bs4 import BeautifulSoup

# Send a GET request to the website
url = "https://www.hostelworld.com/"
response = requests.get(url)

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

# Find all the hostel containers
hostels = soup.find_all("div", class_="fabresult")

# Iterate over each hostel container and extract the required information
for hostel in hostels:
  # Extract hostel name
  name = hostel.find("h2", class_="fabresult-title").text.strip()

  # Extract distance from city centre
  distance = hostel.find("span", class_="distance").text.strip()

  # Extract ratings
  ratings = hostel.find("div", class_="rating").text.strip()

  # Extract total reviews
  total_reviews = hostel.find("div", class_="reviews").text.strip()

  # Extract overall reviews
  overall_reviews = hostel.find("div", class_="overall").text.strip()

  # Extract privates from price
  privates_price = hostel.find("div", class_="price-col").find("div", class_="
  
  # Extract dorms from price
  dorms_price = hostel.find("div", class_="price-col").find("div", class_="pri

  # Extract facilities
  facilities = hostel.find("div", class_="facilities").text.strip()

  # Extract property description
  description = hostel.find("div", class_="description").text.strip()

  # Print the extracted information
  print("Hostel Name:", name)
  print("Distance from City Centre:", distance)
  print("Ratings:", ratings)
  print("Total Reviews:", total_reviews)
  print("Overall Reviews:", overall_reviews)
  print("Privates from Price:", privates_price)
  print("Dorms from Price:", dorms_price)
  print("Facilities:", facilities)
  print("Property Description:", description)
  print()
```

In [ ]: