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

In [1]:
```
import selenium
import pandas as pd
from selenium import webdriver
import warnings
warnings.filterwarnings('ignore')
from selenium.webdriver.common.by import By
import time
```

In [ ]:
```
S=input("Search =")

driver=webdriver.Chrome(r"chromedriver.exe")
time.sleep(2)


driver.get("https://www.amazon.in/")
time.sleep(2)

search=driver.find_element(By.ID,"twotabsearchtextbox")
search.send_keys(S)
look=driver.find_element(By.CLASS_NAME,"nav-right")
look.click()
```

1. 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 [44]:
```
name_pr=[]
rating=[]
no_rating=[]
price=[]
re_ex=[]
exp_del=[]
avail=[]
other_detail=[]
brand=[]


for page in range(0,3):

    brands=driver.find_elements(By.XPATH,"//span[@class='a-size-base.a-link-normal']")
    for i in brands:
        brand.append(i.text)

    name_product=driver.find_elements(By.XPATH,'//span[@class="a-size-base-plus a-color-
    for j in name_product:
        name_pr.append(j.text)

    prices=driver.find_elements(By.XPATH,'//span[@class="a-price-whole"]')
    for k in prices:
        price.append(k.text)
```

In [45]:
```
len(brand),len(price),len(name_pr)
```

Out[45]:
```
(0, 213, 204)
```

In [40]:
```
page_urls=[]
for page in range(0,3):
    url1=driver.find_elements(By.XPATH,"//a[@class='a-link-normal a-text-normal']")
    for t in url1:
        page_urls.append(t.get_attribute('href'))
    page_urls

    time.sleep(3)

    no_ratings=driver.find_elements(By.XPATH,'//a[@id="acrCustomerReviewLink"]')
    for l in no_ratings:
        if l.text is None:
            no_rating.append("--")
        else:
            no_rating.append(l.text)

    time.sleep(3)

    ratings=driver.find_elements(By.XPATH,'//span[@data-hook="acr-average-stars-rating-t
    for m in ratings:
        rating.append(m.text)

    time.sleep(2)

    return_ex= driver.find_elements(By.XPATH,'//a[@class="a-size-small a-link-normal a-t
    for n in return_ex:
        re_ex.append(n.text)

    time.sleep(2)

    expec_del=driver.find_elements(By.XPATH,'//div[@id="ddmDeliveryMessage"]')
    for o in expec_del:
        exp_del.append(o.text)

    time.sleep(2)

    pr_detail=driver.find_elements(By.XPATH,'//hr[@aria-hidden="true"]')
    for p in pr_detail:
        other_detail.append(p.text)

    time.sleep(2)

    pr_avail=driver.find_elements(By.XPATH,'//div[@id="availability"]')
    for q in pr_avail:
        avail.append(q.text)

    time.sleep(2)
```

In [43]:
```
df_guitar=pd.DataFrame({'Brand_name':brand,'Product_name':name_pr,'Ratings':rating,'No_r
df_guitar
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[43], line 1
----> 1 df_guitar=pd.DataFrame({'Brand name':brand,'Product name':name pr,'Ratings':rati
ng,'No ratings':no rating,'Price':price,'Return/Exchange':re ex,'Expected_del':exp_del,'
Availability':avail,'Other_detail':other_detail,'Product_URL':page_urls})
      2 df_guitar

File ~\anaconda4\lib\site-packages\pandas\core\frame.py:664, in DataFrame.__init__(self,
 data, index, columns, dtype, copy)
```

```
    658     mgr = self._init_mgr(
    659         data, axes={"index": index, "columns": columns}, dtype=dtype, copy=copy
    660     )
    662 elif isinstance(data, dict):
    663     # GH#38939 de facto copy defaults to False only in non-dict cases
--> 664     mgr = dict_to_mgr(data, index, columns, dtype=dtype, copy=copy, typ=manager)
    665 elif isinstance(data, ma.MaskedArray):
    666     import numpy.ma.mrecords as mrecords

File ~\anaconda4\lib\site-packages\pandas\core\internals\construction.py:493, in dict_to
_mgr(data, index, columns, dtype, typ, copy)
    489     else:
    490         # dtype check to exclude e.g. range objects, scalars
    491         arrays = [x.copy() if hasattr(x, "dtype") else x for x in arrays]
--> 493 return arrays_to_mgr(arrays, columns, index, dtype=dtype, typ=typ, consolidate=c
opy)

File ~\anaconda4\lib\site-packages\pandas\core\internals\construction.py:118, in arrays_
to_mgr(arrays, columns, index, dtype, verify_integrity, typ, consolidate)
    115 if verify_integrity:
    116     # figure out the index, if necessary
    117     if index is None:
--> 118         index = _extract_index(arrays)
    119     else:
    120         index = ensure_index(index)

File ~\anaconda4\lib\site-packages\pandas\core\internals\construction.py:666, in _extrac
t_index(data)
    664 lengths = list(set(raw_lengths))
    665 if len(lengths) > 1:
--> 666     raise ValueError("All arrays must be of the same length")
    668 if have_dicts:
    669     raise ValueError(
    670         "Mixing dicts with non-Series may lead to ambiguous ordering."
    671     )

ValueError: All arrays must be of the same length
```

1. 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 [18]:

```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.common.by import By
import time


screenshot_dir = './screenshots/'
webdriver_service = Service('path_to_chromedriver')
driver = webdriver.Chrome(service=webdriver_service)


driver.implicitly_wait(10)


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


for keyword in keywords:
    driver.get('https://images.google.com')
    search_box = driver.find_element(By.CLASS_NAME,"gLFyf")
```

```
    search_box.clear()
    search_box.send_keys(keyword)
    search_box.send_keys(Keys.RETURN)


    time.sleep(3)


    for _ in range(3):
        driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")
        time.sleep(2)


    images = driver.find_elements(By.XPATH, '//img[@class="rg_i"]')


    count = 0
    for image in images:

        image_url = image.get_attribute('src')


        if not image_url or image_url.startswith('data:'):
            continue


        count += 1
        image_path = f'{screenshot_dir}{keyword}_{count}.png'
        image.screenshot(image_path)
        print(f'Saved image: {image_path}')


        if count == 10:
            break


driver.quit()
```

1. 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

In [20]:
```
import pandas as pd
from selenium import webdriver


driver = webdriver.Chrome()


driver.get("https://www.flipkart.com/")


try:
    driver.find_element_by_xpath("//button[text()='✕']").click()
except:
    pass
time.sleep(10)

search=driver.find_element(By.CLASS_NAME,"_3704LK")
search.send_keys("Oneplus Nord")
```

```
look=driver.find_element(By.CLASS_NAME,"L0Z3Pu")
look.click()


driver.implicitly_wait(10)


results = driver.find_elements(By.XPATH,"//div[@class='_1AtVbE']/a[@class='_1fQZEK']")
data = []

for result in results:
    brand = driver.find_elements(By.XPATH,".//div[@class='_4rR01T']")
    name = driver.find_elements(By.XPATH,".//a[@class='IRpwTa']")
    url = driver.find_elements(By.XPATH,".//a[@href]")

    try:
        color = driver.find_elements(By.XPATH,".//a[@class='_1KHd47']/div/div[2]")
    except:
        color = "-"

    try:
        specs = driver.find_elements(By.XPATH,".//ul[@class='_1xgFaf']/li[1]")
        ram = specs.split("|")[0].strip()
        rom = specs.split("|")[1].strip()
    except:
        ram = "-"
        rom = "-"

    try:
        camera = driver.find_elements(By.XPATH,".//ul[@class='_1xgFaf']/li[2]")
        primary_camera = camera.split("|")[0].strip()
        secondary_camera = camera.split("|")[1].strip()
    except:
        primary_camera = "-"
        secondary_camera = "-"

    try:
        display_size = driver.find_elements(By.XPATH,".//ul[@class='_1xgFaf']/li[3]")
    except:
        display_size = "-"

    try:
        battery = driver.find_elements(By.XPATH,".//ul[@class='_1xgFaf']/li[4]")
    except:
        battery = "-"

    try:
        price = driver.find_elements(By.XPATH,".//div[@class='_30jeq3 _1_WHN1']")
    except:
        price = "-"

    data.append({
        "Brand Name": brand,
        "Smartphone name": name,
        "Colour": color,
        "RAM": ram,
        "Storage(ROM)": rom,
        "Primary Camera": primary_camera,
        "Secondary Camera": secondary_camera,
        "Display Size": display_size,
        "Battery Capacity": battery,
        "Price": price,
        "Product URL": url
    })
```

```
df = pd.DataFrame(data)


df.to_csv("smartphones.csv", index=False)

driver.quit()
```

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

In [2]:
```
from selenium import webdriver


driver = webdriver.Chrome()


driver.get("https://www.google.com/maps")


search_query = "New York"
search_box = driver.find_element(By.ID,"searchboxinput")
search_box.send_keys(search_query)
look=driver.find_element(By.CLASS_NAME,"pzfvzf")
look.click()



driver.implicitly_wait(10)

try:
    coordinates_element = driver.find_elements(By.XPATH,"//button[@data-value='Direction
    coordinates = coordinates_element.get_attribute("data-coordinates")
    latitude, longitude = coordinates.split(",")
    print("Latitude:", latitude)
    print("Longitude:", longitude)
except:
    print("Coordinates not found.")
```

```
Coordinates not found.
```

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

In [4]:
```
import requests
from bs4 import BeautifulSoup

# Send a GET request to the URL
url = "https://www.digit.in/top-products/best-gaming-laptops-40.html"
response = requests.get(url)

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

# Find the container that holds the laptop details
laptop_container = soup.find("div", class_="TopNumbeHeading")

# Find all the laptop details within the container
laptop_details = laptop_container.find_all("div", class_="right-container")

# Iterate over the laptop details and extract the required information
for laptop in laptop_details:
    # Extract the laptop name
    laptop_name = laptop.find("div", class_="heading-wraper").text.strip()
```

```
    # Extract the laptop specifications
    specs = laptop.find_all("div", class_="Specs-Wrap")
    laptop_specifications = []
    for spec in specs:
        title = spec.find("div", class_="title").text.strip()
        value = spec.find("div", class_="value").text.strip()
        laptop_specifications.append(f"{title}: {value}")

    # Print the laptop details
    print("Laptop Name:", laptop_name)
    print("Specifications:")
    for specification in laptop_specifications:
        print("-", specification)
    print()
```

In [5]:

```
from selenium import webdriver


driver = webdriver.Chrome('path_to_chromedriver')


url = "https://www.digit.in/top-products/best-gaming-laptops-40.html"
driver.get(url)


laptop_names = driver.find_elements(By.XPATH,"//div[@class='right-container']/div[@class
laptop_specs = driver.find_elements(By.XPATH,"//div[@class='right-container']/div[@class


for i in range(len(laptop_names)):

    laptop_name = laptop_names[i].text.strip()


    laptop_specifications = []
    specs = laptop_specs[i].find_elements(By.XPATH,".//div[@class='title']")
    values = laptop_specs[i].find_elements(By.XPATH,".//div[@class='value']")
    for j in range(len(specs)):
        title = specs[j].text.strip()
        value = values[j].text.strip()
        laptop_specifications.append(f"{title}: {value}")


    print("Laptop Name:", laptop_name)
    print("Specifications:")
    for specification in laptop_specifications:
        print("-", specification)
    print()
driver.quit()
```

1. 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”.

In [7]:

```
from selenium import webdriver


driver = webdriver.Chrome('path_to_chromedriver')


url = 'https://www.forbes.com/billionaires/'
driver.get(url)
```

```
driver.implicitly_wait(10)


container = driver.find_elements(By.XPATH,'//div[@class="rank"]')


billionaires = driver.find_elements(By.XPATH,'//div[contains(@class, "person")]')


for billionaire in billionaires:

    rank = billionaire.find_elements(By.XPATH,'.//div[@class="rank"]').text
    name = billionaire.find_elements(By.XPATH,'.//div[@class="name"]').text
    net_worth = billionaire.find_elements(By.XPATH,'.//div[@class="netWorth"]').text
    age = billionaire.find_elements(By.XPATH,'.//div[@class="age"]').text


    citizenship = billionaire.find_elements(By.XPATH,'.//div[@class="countryOfCitizenshi
    citizenship_list = [c.text for c in citizenship]

    source = billionaire.find_elements(By.XPATH,'.//div[@class="source"]/div')
    source_list = [s.text for s in source]


    industry = billionaire.find_elements(By.XPATH,'.//div[@class="category"]').text


    print("Rank:", rank)
    print("Name:", name)
    print("Net Worth:", net_worth)
    print("Age:", age)
    print("Citizenship:", ", ".join(citizenship_list))
    print("Source:", ", ".join(source_list))
    print("Industry:", industry)
    print()


driver.quit()
```

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

In [2]:
```
import time
from selenium import webdriver


driver = webdriver.Chrome('path_to_chromedriver')


url = 'https://www.youtube.com/watch?v=X3paOmcrTjQ'
driver.get(url)


while True:

    driver.execute_script('window.scrollTo(0, document.documentElement.scrollHeight);')
    time.sleep(2)

    comment_elements = driver.find_element(By.XPATH,'//div[@id="content"]//yt-formatted-
    if len(comment_elements) >= 500:
        break
```

```
comments = driver.find_elements(By.XPATH,'//div[@id="content"]//yt-formatted-string[@id=
upvotes = driver.find_elements(By.XPATH,'//div[@id="content"]//span[@id="vote-count-midd
times = driver.find_elements(By.XPATH,'//div[@id="content"]//a[contains(@class, "ytd-com


for comment, upvote, time in zip(comments, upvotes, times):
    print("Comment:", comment.text)
    print("Upvotes:", upvote.text)
    print("Time:", time.get_attribute('textContent'))
    print()


driver.quit()
```

```
---------------------------------------------------------------------------
NoSuchElementException                    Traceback (most recent call last)
Cell In[2], line 17
     14 driver.execute_script('window.scrollTo(0, document.documentElement.scrollHeigh
t);')
     15 time.sleep(2)
---> 17 comment_elements = driver.find_element(By.XPATH,'//div[@id="content"]//yt-format
ted-string[@id="content-text"]')
     18 if len(comment_elements) >= 500:
     19     break

File ~\anaconda4\lib\site-packages\selenium\webdriver\remote\webdriver.py:831, in WebDri
ver.find_element(self, by, value)
    828     by = By.CSS_SELECTOR
    829     value = f'[name="{value}"]'
--> 831 return self.execute(Command.FIND_ELEMENT, {"using": by, "value": value})["value"
]

File ~\anaconda4\lib\site-packages\selenium\webdriver\remote\webdriver.py:440, in WebDri
ver.execute(self, driver_command, params)
    438 response = self.command_executor.execute(driver_command, params)
    439 if response:
--> 440     self.error_handler.check_response(response)
    441     response["value"] = self._unwrap_value(response.get("value", None))
    442     return response

File ~\anaconda4\lib\site-packages\selenium\webdriver\remote\errorhandler.py:245, in Err
orHandler.check_response(self, response)
    243         alert_text = value["alert"].get("text")
    244     raise exception_class(message, screen, stacktrace, alert_text)  # type: igno
re[call-arg]  # mypy is not smart enough here
--> 245 raise exception_class(message, screen, stacktrace)

NoSuchElementException: Message: no such element: Unable to locate element: {"method":"x
path","selector":"//div[@id="content"]//yt-formatted-string[@id="content-text"]"}
  (Session info: chrome=114.0.5735.134)
Stacktrace:
Backtrace:
        GetHandleVerifier [0x0056A813+48355]
        (No symbol) [0x004FC4B1]
        (No symbol) [0x00405358]
        (No symbol) [0x004309A5]
        (No symbol) [0x00430B3B]
        (No symbol) [0x0045E232]
        (No symbol) [0x0044A784]
        (No symbol) [0x0045C922]
        (No symbol) [0x0044A536]
        (No symbol) [0x004282DC]
        (No symbol) [0x004293DD]
        GetHandleVerifier [0x007CAABD+2539405]
```

```
        GetHandleVerifier [0x0080A78F+2800735]
        GetHandleVerifier [0x0080456C+2775612]
        GetHandleVerifier [0x005F51E0+616112]
        (No symbol) [0x00505F8C]
        (No symbol) [0x00502328]
        (No symbol) [0x0050240B]
        (No symbol) [0x004F4FF7]
        BaseThreadInitThunk [0x76417D59+25]
        RtlInitializeExceptionChain [0x77E9B74B+107]
        RtlClearBits [0x77E9B6CF+191]
```

1. 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.

In [5]:
```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC


driver = webdriver.Chrome("path_to_chromedriver")


driver.get("https://www.hostelworld.com/")


search_input = driver.find_element(By.XPATH, "//input[@id='search-input-field']")
search_input.clear()
search_input.send_keys("London")


search_button = driver.find_element(By.XPATH, "//button[@id='search-button']")
search_button.click()


wait = WebDriverWait(driver, 10)
wait.until(EC.visibility_of_element_located((By.XPATH, "//div[@id='fabResultsContainer']


hostel_cards = driver.find_elements(By.XPATH, "//li[contains(@class, 'fabresult')]")


for card in hostel_cards:
    hostel_name = card.find_element(By.XPATH, ".//h2/a").text
    distance = card.find_element(By.XPATH, ".//span[contains(text(), 'km from city centr
    ratings = card.find_element(By.XPATH, ".//div[contains(@class, 'score')]").text
    total_reviews = card.find_element(By.XPATH, ".//div[@class='reviews']//span[contains
    overall_reviews = card.find_element(By.XPATH, ".//div[contains(@class, 'rating')]//s
    privates_price = card.find_element(By.XPATH, ".//div[contains(@class, 'price-col')]/
    dorms_price = card.find_element(By.XPATH, ".//div[contains(@class, 'price-col')]//di
    facilities = card.find_elements(By.XPATH, ".//ul[contains(@class, 'facilities')]//li
    facilities_list = [facility.text for facility in facilities]
    description = card.find_element(By.XPATH, ".//div[contains(@class, 'desc')]//p").tex


    print("Hostel Name:", hostel_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_list)
    print("Description:", description)
    print("=" * 50)


driver.quit()
```

```
---------------------------------------------------------------------------
NoSuchElementException                    Traceback (most recent call last)
Cell In[5], line 13
     10 driver.get("https://www.hostelworld.com/")
     12 # Find the search input field and enter "London"
---> 13 search_input = driver.find_element(By.XPATH, "//input[@id='search-input-field']"
)
     14 search_input.clear()
     15 search_input.send_keys("London")

File ~\anaconda4\lib\site-packages\selenium\webdriver\remote\webdriver.py:831, in WebDri
ver.find_element(self, by, value)
    828     by = By.CSS_SELECTOR
    829     value = f'[name="{value}"]'
--> 831 return self.execute(Command.FIND_ELEMENT, {"using": by, "value": value})["value"
]

File ~\anaconda4\lib\site-packages\selenium\webdriver\remote\webdriver.py:440, in WebDri
ver.execute(self, driver_command, params)
    438 response = self.command_executor.execute(driver_command, params)
    439 if response:
--> 440     self.error_handler.check_response(response)
    441     response["value"] = self._unwrap_value(response.get("value", None))
    442     return response

File ~\anaconda4\lib\site-packages\selenium\webdriver\remote\errorhandler.py:245, in Err
orHandler.check_response(self, response)
    243         alert_text = value["alert"].get("text")
    244     raise exception_class(message, screen, stacktrace, alert_text)  # type: igno
re[call-arg]  # mypy is not smart enough here
--> 245 raise exception_class(message, screen, stacktrace)

NoSuchElementException: Message: no such element: Unable to locate element: {"method":"x
path","selector":"//input[@id='search-input-field']"}
  (Session info: chrome=114.0.5735.134)
Stacktrace:
Backtrace:
        GetHandleVerifier [0x009AA813+48355]
        (No symbol) [0x0093C4B1]
        (No symbol) [0x00845358]
        (No symbol) [0x008709A5]
        (No symbol) [0x00870B3B]
        (No symbol) [0x0089E232]
        (No symbol) [0x0088A784]
        (No symbol) [0x0089C922]
        (No symbol) [0x0088A536]
        (No symbol) [0x008682DC]
        (No symbol) [0x008693DD]
        GetHandleVerifier [0x00C0AABD+2539405]
        GetHandleVerifier [0x00C4A78F+2800735]
        GetHandleVerifier [0x00C4456C+2775612]
        GetHandleVerifier [0x00A351E0+616112]
        (No symbol) [0x00945F8C]
        (No symbol) [0x00942328]
        (No symbol) [0x0094240B]
        (No symbol) [0x00934FF7]
        BaseThreadInitThunk [0x76417D59+25]
```

```
        RtlInitializeExceptionChain [0x77E9B74B+107]
        RtlClearBits [0x77E9B6CF+191]
```

In [ ]: