Task 1. Python Script for Web Scraping and Automation

Aim

The aim of the Python script is to automate web scraping of specific data from a website and perform data processing, cleaning, analysis and performing automation to update. Here we are choosing Amazon website and analyzing data for the product of wireless keyboard and mouse.

Defining Libraries

```
import requests
import pandas as pd
import numpy as np
import re
import sqlite3
```

Defining Scrape_amazon method to scrape

```
# Define a function to scrape data from Amazon
def scrape_amazon():
    headers = {
        'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64)AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36',
        'Accept-Language': 'en-US,en;q=0.5',
        'Referer': 'https://www.google.com/',
        'DNT': '1',
        'Upgrade-Insecure-Requests': '1'
    }

    # Initialize DataFrame to store all results
    all_products = pd.DataFrame(columns=['product_name', 'price', 'rating', 'reviews', 'availability', 'product_id'])

    for page in range(1, 11):
        print(f"Scraping page {page}")
        url = f"https://www.amazon.com/s?k=wireless+keyboard+and+mouse&page={page}&ref=sr_pg_{page}"
        try:
            response = requests.get(url, headers=headers)
            if response.status_code != 200:
            print(f"Failed to fetch page {page}, status code: {response.status_code}")
            continue

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

            if len(products) == 0:
            print("No products found on this page")
            break

            for product in products:
                try:
                    name = product.find('span', {'class': 'a-size-medium a-color-base a-text-normal'})
                    if name:
                        name = name.text.strip()

                    price = product.find('span', {'class': 'a-price-whole'})
                    if price:
                        price = price.text.strip()

                    rating = product.find('span', {'class': 'a-icon-alt'})
                    if rating:
                        rating = rating.text.strip()

                    reviews = product.find('span', {'class': 'a-size-base'})
                    if reviews:
                        reviews = reviews.text.strip()

                    availability = product.find('span', {'class': 'a-color-success'})
                    if availability:
                        availability = availability.text.strip()

                    product_id = product.get('data-asin')

                    if name:
                        new_row = {
                            'product_name': name,
                            'price': price if price else None,
                            'rating': rating if rating else None,
                            'reviews': reviews if reviews else None,
                            'availability': availability if availability else None,
                            'product_id': product_id if product_id else None
                        }
                        all_products = pd.concat([all_products, pd.DataFrame([new_row])], ignore_index=True)
                    time.sleep(random.uniform(1, 2))
                except Exception:
                    continue

            time.sleep(random.uniform(3, 5))
        except Exception:
            time.sleep(10)

    return all_products
```

Function call to scrape

```
# Execute the scraping
print("Starting scraping...")
df = scrape_amazon()

# Display basic information
print("\nScraping finished.")
print("Total products scraped: {len(df)}")

# Save to CSV
df.to_csv('amazon_products.csv', index=False)
print("Data saved to amazon_products.csv")

# Display first few rows
print("\nFirst 5 rows of the data:")
print(df.head())
```

```
Starting scraping...
Scraping page 1
Scraping page 2
Scraping page 3
Scraping page 4
Scraping page 5
Scraping page 6
Scraping page 7
Scraping page 8
Scraping page 9
Scraping page 10

Scraping finished.
Total products scraped: {len(df)}
Data saved to amazon_products.csv

First 5 rows of the data:
                                        product_name  price  rating  reviews  \
0  Logitech MK270 Wireless Combo, Wireless Keyboard  None  4.7 out  (14,195)
1  Logitech MK540 Advanced Wireless Keyboard and ...  None  4.6 out  (20,000)
2  Logitech MK850 Performance Wireless Keyboard a...  None  4.5 out  (2,000)
3  Logitech MK345 Wireless Combo, Wireless Keyboa...  None  4.7 out  (14,195)
4  Logitech MK235 Wireless Keyboard and Mouse Com...  None  4.7 out  (14,195)

  availability  product_id
0         NaN  B0CHX1W1XY
1         NaN  B0CHWRXH8B
2         NaN  B0CNKNZ8FT
3         NaN  B0CHX3QBCH
4         NaN  B0BP3W3WZ6
```

Defining update_daily() method to automate the updation schedule

```
# Function to update data daily
def update_daily():
    print("Starting daily update...")
    df = scrape_amazon()
    df.to_csv('amazon_products.csv', index=False)
    print("Daily update completed.")
```

Scheduling the update

```
# Schedule the scraping to run every day at 10:00 AM
schedule.every().day.at("10:00").do(update_daily)

print("Scraping scheduler started.")
print("Next scraping: {schedule.next_run()}")
```

```
Scraping scheduler started.
Next scraping: 2025-04-28 10:00:00
```

Main Loop for Scheduling

```
# Keep the script running
while True:
    schedule.run_pending()
    time.sleep(1)
```