6. write a python program that reads a file containing a list of username and passwords. one pair per line (separated by comma). It checks each password to see if it has been leaked in a data breach . you can use the "Have i Been Pwned" api to check if a password has been leaked

```
import requests
import hashlib
import getpass

def check_password_leak(password):
    sha1_hash = hashlib.sha1(password.encode('utf-8')).hexdigest().upper()
    prefix, suffix = sha1_hash[:5], sha1_hash[5:]
    url = f'https://api.pwnedpasswords.com/range/{prefix}'
    response = requests.get(url)
    if response.status_code == 200:
        hashes = (line.split(':') for line in response.text.splitlines())
        return any(suffix in line for line in hashes)
    else:
        raise Exception(f'Error checking password: {response.status_code}')

def main():
    file_path = input("Enter the path to the file with username and password pairs: ")

    try:
        with open(file_path, 'r') as file:
            for line in file:
                username, password = map(str.strip, line.split(','))
                print(f'Checking password for {username}...')

                if check_password_leak(password):
                    print(f'Password for {username} has been leaked!')
                else:
                    print(f'Password for {username} is secure.')

    except FileNotFoundError:
        print(f"Error: File not found at {file_path}")

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

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

Input

Create a File Passwords.txt That must contain below data

user1,Password123
user2,SecurePwd456
user3,LeakedPwd789

Explanation of Code

```
import requests
import hashlib
```

- `import requests`: Imports the `requests` library, which will be used to make HTTP requests.
- `import hashlib`: Imports the `hashlib` library, which provides secure hash and message digest algorithms.

```
def check_password_leak(password, api_key):
    sha1_hash = hashlib.sha1(password.encode('utf-8')).hexdigest().upper()
    url = f'https://api.pwnedpasswords.com/pwnedpassword/{sha1_hash}'
    headers = {'Api-Key': api_key}

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

    if response.status_code == 200:
        return int(response.text) > 0
    elif response.status_code == 404:
        return False
    else:
        raise Exception(f'Error checking password: {response.status_code}')
```

- `check_password_leak` function: This function takes a password and an API key as parameters, computes the SHA-1 hash of the password, sends a request to the Have I Been Pwned API to check if the password has been leaked, and returns a boolean indicating whether the password has been leaked.
- `sha1_hash = hashlib.sha1(password.encode('utf-8')).hexdigest().upper()`: Computes the SHA-1 hash of the password and converts it to uppercase.
- `url = f'https://api.pwnedpasswords.com/pwnedpassword/{sha1_hash}'`: Constructs the URL for the HIBP API to check the password.
- `headers = {'Api-Key': api_key}`: Sets up the HTTP headers with the API key for authentication.
- `response = requests.get(url, headers=headers)`: Sends a GET request to the HIBP API to check if the password has been leaked.
- The subsequent `if-elif-else` block handles the API response. If the status code is 200, it checks if the count of occurrences is greater than 0 and returns `True` if so. If the status code is 404, it means the password is not found in the HIBP database, and it returns `False`. If any other status code is received, it raises an exception with an error message.

```
def main():
    file_path = 'passwords.txt'
    api_key = 'Your-API-Key'  # Replace with your actual API key from HIBP

    try:
        with open(file_path, 'r') as file:
            for line in file:
                username, password = map(str.strip, line.split(','))
                print(f'Checking password for {username}...')

                if check_password_leak(password, api_key):
                    print(f'Password for {username} has been leaked!')
                else:
                    print(f'Password for {username} is secure.')

    except FileNotFoundError:
        print(f"Error: File not found at {file_path}")

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

- `main` function: The main function of the script.
- `file_path = 'passwords.txt'`: Specifies the path to the file containing username and password pairs.
- `api_key = 'Your-API-Key'`: Your actual API key from Have I Been Pwned should be used here.
- The `try-except` block is used to handle potential errors, such as the file not being found or an unexpected exception occurring.
- `with open(file_path, 'r') as file:`: Opens the file specified by `file_path` in read mode.
- The `for line in file:` loop iterates through each line in the file.
- `username, password = map(str.strip, line.split(','))`: Splits each line into username and password, removing any leading or trailing whitespaces.
- The program then checks if the password has been leaked using the `check_password_leak` function and prints the result.

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

- `if __name__ == "__main__":`: Checks if the script is being run as the main program.- `main()`: Calls the `main` function if the script is the main program.