**Звіт про виконання лабораторних завдань до лекцій з курсу Технології програмування на мові Python**

Лабораторна робота №2

**Робота з файлами. Юніт тести.**

**Мета роботи**: Використовуючи теоретичне підґрунтя про роботу з файлами та тестування коду у мові Python розширити програму телефонного довідника студентів додавши функціонал, що буде вказано в завданні до лабораторної роботи.

**Завантаження початкових данні для довідника з csv файлу**

**Завантаження** початкових данні для довідника з csv файлу. Приклад csv файлу (lab2.csv), та оригінальній файл формату xlsx розміщені поряд з вказівками. Ім’я файлу для завантаження даних повинно передаватись в якості параметрів командного рядку. **Зберігання** довідника студентів у форматі CSV перед виходом із програми

Перед виконанням роботи слід ознайомитись з існуючим функціоналом.

Хід виконання завдання:

1. Відкрила ВСкод
2. Прочитала завдання
3. Виконала завдання
4. Затестила
5. Закинула в звіт

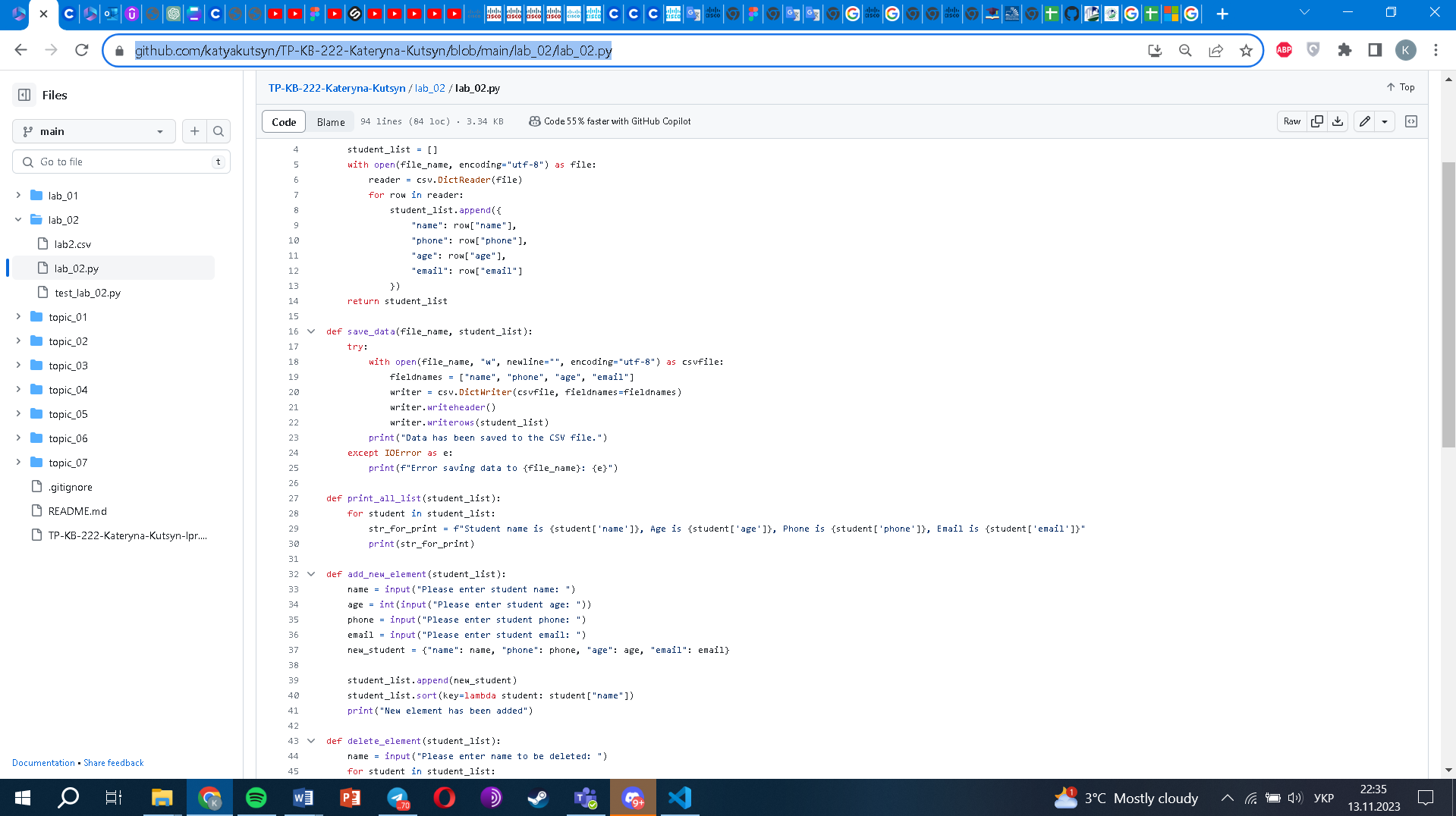
Текст програми:

|  |
| --- |
| import csv  def load\_data(file\_name):  student\_list = []  with open(file\_name, encoding="utf-8") as file:  reader = csv.DictReader(file)  for row in reader:  student\_list.append({  "name": row["name"],  "phone": row["phone"],  "age": row["age"],  "email": row["email"]  })  return student\_list  def save\_data(file\_name, student\_list):  try:  with open(file\_name, "w", newline="", encoding="utf-8") as csvfile:  fieldnames = ["name", "phone", "age", "email"]  writer = csv.DictWriter(csvfile, fieldnames=fieldnames)  writer.writeheader()  writer.writerows(student\_list)  print("Data has been saved to the CSV file.")  except IOError as e:  print(f"Error saving data to {file\_name}: {e}")  def print\_all\_list(student\_list):  for student in student\_list:  str\_for\_print = f"Student name is {student['name']}, Age is {student['age']}, Phone is {student['phone']}, Email is {student['email']}"  print(str\_for\_print)  def add\_new\_element(student\_list):  name = input("Please enter student name: ")  age = int(input("Please enter student age: "))  phone = input("Please enter student phone: ")  email = input("Please enter student email: ")  new\_student = {"name": name, "phone": phone, "age": age, "email": email}  student\_list.append(new\_student)  student\_list.sort(key=lambda student: student["name"])  print("New element has been added")  def delete\_element(student\_list):  name = input("Please enter name to be deleted: ")  for student in student\_list:  if name == student["name"]:  student\_list.remove(student)  print("Element has been deleted")  break  else:  print("Element was not found")  def update\_element(student\_list):  name = input("Please enter name to be updated: ")  for student in student\_list:  if name == student["name"]:  student["name"] = input("Enter new name: ")  student["age"] = input("Enter new age: ")  student["phone"] = input("Enter new phone: ")  student["email"] = input("Enter new email: ")  student\_list.sort(key=lambda student: student["name"])  print("Element has been updated")  break  else:  print("Student not found")  def main():  file\_name = input("Enter the CSV file name (e.g., lab2.csv): ")  student\_list = load\_data(file\_name)  while True:  choice = input("Please specify the action [C create, U update, D delete, S save, P print, Q exit]: ")  if choice.upper() == "C":  print("New element will be created:")  add\_new\_element(student\_list)  elif choice.upper() == "U":  print("Existing element will be updated")  update\_element(student\_list)  elif choice.upper() == "D":  print("Element will be deleted")  delete\_element(student\_list)  elif choice.upper() == "P":  print("List will be printed")  print\_all\_list(student\_list)  elif choice.upper() == "S":  save\_data(file\_name, student\_list)  elif choice.upper() == "Q":  print("Dosviduli")  break  else:  print("Dosviduli")  if \_\_name\_\_ == "\_\_main\_\_":  main() |

Посилання на github:

[https://github.com/katyakutsyn/TP-KB-222-Kateryna-Kutsyn/blob/main/lab\_02/lab\_02.py](https://github.com/katyakutsyn/TP-KB-222-Kateryna-Kutsyn/blob/main/lab_02/lab_02.py%20)

Знімок екрану з посилання на github:



**Покриття основного функціоналу програми Юніт тестами**

Процес виконання завдання може включати розбиття великої задачі на маленькі під задачі, при цьому результатом виконання маленької частина має бути певна кількість функцій, сумарний результат виконання яких задовольняє критеріям виділеної під задачі. Для того, щоб впевнитись, що написаний код виконує саме те, що вказано в завданні, розробник формує набір тестів яким піддається написаний в рамках виконання завдання код. **Тестування коду – написання коду для тестування коду.** При цьому поняття Юніт тест означає тестування окремого функціоналу, наприклад однієї функції (юніту).

Перед виконанням роботи слід ознайомитись з існуючим функціоналом.

Хід виконання завдання:

1. Відкрила ВСкод
2. Прочитала завдання
3. Виконала завдання
4. Затестила
5. Закинула в звіт

Текст програми:

|  |
| --- |
| import unittest  import io  import os  import csv  from unittest.mock import patch  from lab\_02 import load\_data, save\_data, add\_new\_element, delete\_element, update\_element, print\_all\_list  class TestLab2Script(unittest.TestCase):  def setUp(self):  self.test\_file = "test\_lab2.csv"  self.test\_data = [  {"name": "John", "phone": "9675643454", "age": "21", "email": "john@example.com"},  {"name": "Alice", "phone": "734586290", "age": "19", "email": "alice@example.com"},  ]  def tearDown(self):  # Clean up the test file after each test  try:  os.remove(self.test\_file)  except FileNotFoundError:  pass  def test\_load\_data(self):  with open(self.test\_file, "w", newline="", encoding="utf-8") as file:  fieldnames = ["name", "phone", "age", "email"]  writer = csv.DictWriter(file, fieldnames=fieldnames)  writer.writeheader()  writer.writerows(self.test\_data)  loaded\_data = load\_data(self.test\_file)  self.assertEqual(loaded\_data, self.test\_data)  def test\_save\_data(self):  save\_data(self.test\_file, self.test\_data)  loaded\_data = load\_data(self.test\_file)  self.assertEqual(loaded\_data, self.test\_data)  def test\_add\_new\_element(self):  with patch('builtins.input', side\_effect=["Bob", "5555555555", "30", "bob@example.com"]):  add\_new\_element(self.test\_data)  self.assertEqual(len(self.test\_data), 3)  def test\_delete\_element(self):  with patch('builtins.input', return\_value="John"):  delete\_element(self.test\_data)  self.assertEqual(len(self.test\_data), 1)  def test\_update\_element(self):  with patch('builtins.input', side\_effect=["Alice", "Bob", "5555555555", "30", "bob@example.com"]):  update\_element(self.test\_data)  self.assertEqual(self.test\_data[0]["name"], "Bob")  def test\_print\_all\_list(self):  with patch('sys.stdout', new\_callable=io.StringIO) as mock\_stdout:  print\_all\_list(self.test\_data)  output = mock\_stdout.getvalue()  self.assertIn("John", output)  self.assertIn("Alice", output)  if \_\_name\_\_ == '\_\_main\_\_':  unittest.main() |

Посилання на github:

<https://github.com/katyakutsyn/TP-KB-222-Kateryna-Kutsyn/blob/main/lab_02/test_lab_02.py>

Знімок екрану з посилання на github:

