**Лабораторна робота №2**

**Тема:** Робота з файлами. Юніт тести.

**Мета:** 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.

**Завдання до лабораторної роботи**

Розширити функціональність телефонного довідника студентів групи, що був розроблений у Лабораторній роботі №1 наступними можливостями:

1) завантаження початкових данні для довідника з csv файлу. Ім’я файлу для завантаження даних повинно передаватись в якості параметрів командного рядку.

2)зберігання довідника студентів у форматі CSV перед виходом із програми.

3) покриття основного функціоналу програми Юніт тестами.

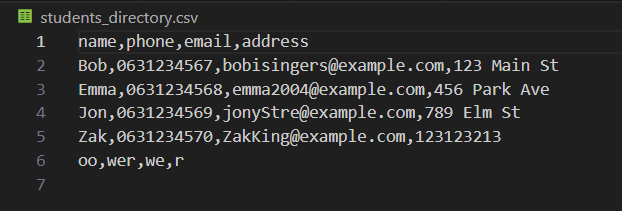
**Хід роботи**

Додав нові функції для опрацювання аргумента, завантаження даних із довідника та зберігання. Оптимізував існуючі функції.

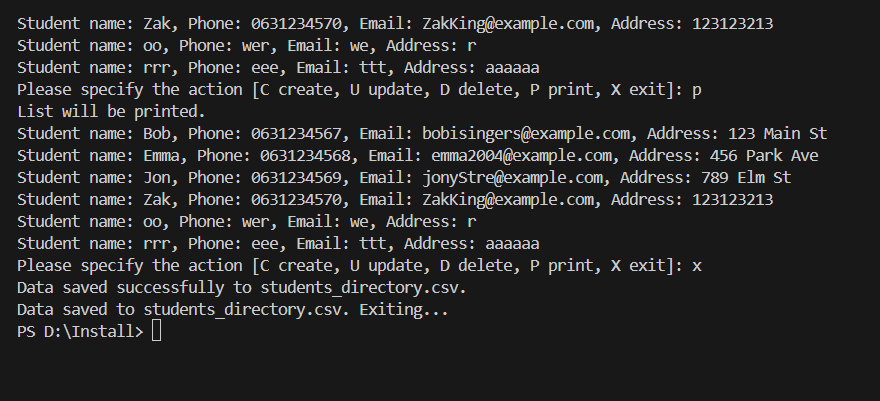
Код файла **Lab\_02.py**:

|  |
| --- |
| import csv  import sys  # Ім'я файлу за замовчуванням  DEFAULT\_FILENAME = "students\_directory.csv"  # Список студентів  list = []  # Друк списку  def printAllList():      if not list:          print("The student list is empty.")      else:          for elem in list:              print(f"Student name: {elem['name']}, Phone: {elem['phone']}, Email: {elem['email']}, Address: {elem['address']}")      return  # Додавання нового елемента  def addNewElement():      name = input("Please enter student name: ")      phone = input("Please enter student phone: ")      email = input("Please enter student email: ")      address = input("Please enter student address: ")        newItem = {"name": name, "phone": phone, "email": email, "address": address}        insertPosition = 0      for item in list:          if name > item["name"]:              insertPosition += 1          else:              break      list.insert(insertPosition, newItem)      print("New element has been added.")      return  # Видалення елемента  def deleteElement():      name = input("Please enter name to be deleted: ")      deletePosition = -1      for item in list:          if name == item["name"]:              deletePosition = list.index(item)              break      if deletePosition == -1:          print("Element was not found.")      else:          del list[deletePosition]          print("Element has been deleted.")      return  # Оновлення елемента  def updateElement():      name = input("Please enter the name of the student to be updated: ")      updatePosition = -1      for item in list:          if name == item["name"]:              updatePosition = list.index(item)              break        if updatePosition == -1:          print("Student not found.")      else:          new\_name = input(f"Please enter new name (current: {list[updatePosition]['name']}): ") or list[updatePosition]["name"]          new\_phone = input(f"Please enter new phone (current: {list[updatePosition]['phone']}): ") or list[updatePosition]["phone"]          new\_email = input(f"Please enter new email (current: {list[updatePosition]['email']}): ") or list[updatePosition]["email"]          new\_address = input(f"Please enter new address (current: {list[updatePosition]['address']}): ") or list[updatePosition]["address"]          if new\_name != list[updatePosition]["name"]:              old\_item = list[updatePosition]              del list[updatePosition]              newItem = {                  "name": new\_name,                  "phone": new\_phone,                  "email": new\_email,                  "address": new\_address              }              insertPosition = 0              for item in list:                  if new\_name > item["name"]:                      insertPosition += 1                  else:                      break              list.insert(insertPosition, newItem)              print(f"Student {old\_item['name']}'s details have been updated to {new\_name}.")          else:              list[updatePosition]["phone"] = new\_phone              list[updatePosition]["email"] = new\_email              list[updatePosition]["address"] = new\_address              print(f"Student {name}'s details have been updated.")      return  # Завантаження даних із CSV  def loadFromCSV(filename):      global list      try:          with open(filename, mode='r', newline='', encoding='utf-8') as file:              reader = csv.DictReader(file)              list = [row for row in reader]          print(f"Data loaded successfully from {filename}.")      except FileNotFoundError:          print(f"File {filename} not found. Starting with an empty list.")          list = []      except Exception as e:          print(f"Error loading file: {e}")  # Збереження даних у CSV  def saveToCSV(filename):      try:          with open(filename, mode='w', newline='', encoding='utf-8') as file:              fieldnames = ["name", "phone", "email", "address"]              writer = csv.DictWriter(file, fieldnames=fieldnames)              writer.writeheader()              writer.writerows(list)          print(f"Data saved successfully to {filename}.")      except Exception as e:          print(f"Error saving file: {e}")  # Меню  def main():      # Завантаження даних із файлу      filename = DEFAULT\_FILENAME      if len(sys.argv) > 1:          filename = sys.argv[1]      loadFromCSV(filename)      while True:          choice = input("Please specify the action [C create, U update, D delete, P print, X exit]: ")          match choice:              case "C" | "c":                  print("New element will be created:")                  addNewElement()                  printAllList()              case "U" | "u":                  print("Existing element will be updated.")                  updateElement()                  printAllList()              case "D" | "d":                  print("Element will be deleted.")                  deleteElement()                  printAllList()              case "P" | "p":                  print("List will be printed.")                  printAllList()              case "X" | "x":                  saveToCSV(filename)                  print(f"Data saved to {filename}. Exiting...")                  break              case \_:                  print("Wrong choice.")  # Запуск програми  if \_\_name\_\_ == "\_\_main\_\_":      main() |

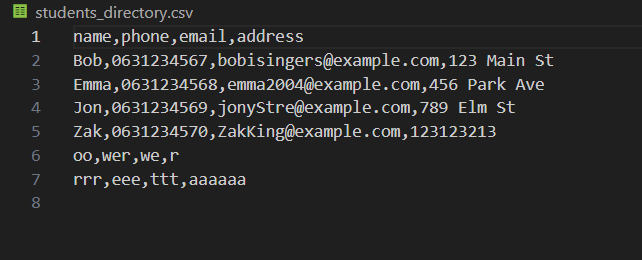
Файл students\_directory.csv:



Результат роботи програми:



Змінений файл studends.csv



Наступним завданням було покрити основний функціонал юніт тестами.

Створив новий файл Test.py та написав наступний код:

import unittest

from io import StringIO

import sys

import os

import tempfile

import csv

from lab\_02 import addNewElement, deleteElement, updateElement, loadFromCSV, saveToCSV, printAllList, list

class TestStudentDirectory(unittest.TestCase):

    def setUp(self):

        self.test\_file = tempfile.NamedTemporaryFile(delete=False, mode='w', newline='', encoding='utf-8')

        self.test\_filename = self.test\_file.name

        self.test\_file.close()

    def tearDown(self):

        if os.path.exists(self.test\_filename):

            os.remove(self.test\_filename)

    def test\_addNewElement(self):

        addNewElement()

        self.assertEqual(len(list), 1)

        self.assertEqual(list[0]['name'], "Test Student")

    def test\_deleteElement(self):

        list.append({"name": "Test Student", "phone": "123456", "email": "test@example.com", "address": "123 Test St"})

        deleteElement()

        self.assertEqual(len(list), 0)

    def test\_updateElement(self):

        list.append({"name": "Test Student", "phone": "123456", "email": "test@example.com", "address": "123 Test St"})

        updateElement()

        self.assertEqual(list[0]["phone"], "654321")

    def test\_loadFromCSV(self):

        data = [

            {"name": "Bob", "phone": "123456789", "email": "bob@example.com", "address": "1 Main St"},

            {"name": "Alice", "phone": "987654321", "email": "alice@example.com", "address": "2 Main St"}

        ]

        with open(self.test\_filename, mode='w', newline='', encoding='utf-8') as file:

            writer = csv.DictWriter(file, fieldnames=["name", "phone", "email", "address"])

            writer.writeheader()

            writer.writerows(data)

        loadFromCSV(self.test\_filename)

        self.assertEqual(len(list), 2)

        self.assertEqual(list[0]["name"], "Bob")

    def test\_saveToCSV(self):

        list.append({"name": "Charlie", "phone": "123123123", "email": "charlie@example.com", "address": "3 Main St"})

        saveToCSV(self.test\_filename)

        with open(self.test\_filename, mode='r', newline='', encoding='utf-8') as file:

            reader = csv.DictReader(file)

            rows = list(reader)

            self.assertEqual(len(rows), 1)

            self.assertEqual(rows[0]["name"], "Charlie")

    def test\_printAllList(self):

        list.append({"name": "Test Student", "phone": "123456", "email": "test@example.com", "address": "123 Test St"})

        captured\_output = StringIO()

        sys.stdout = captured\_output

        printAllList()

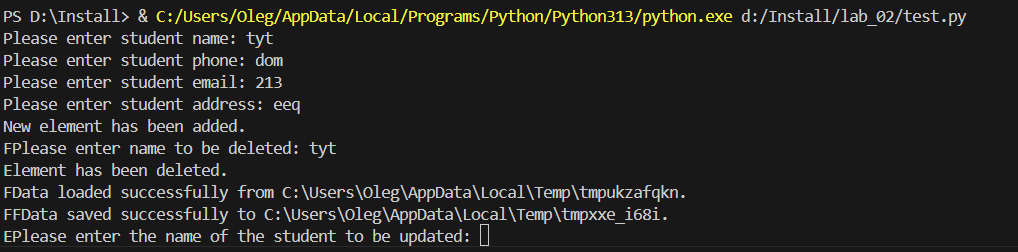
        sys.stdout = sys.\_\_stdout\_\_

        self.assertIn("Test Student", captured\_output.getvalue())

if \_\_name\_\_ == '\_\_main\_\_':

    unittest.main()

Результат:



**Висновок:** В рамках цієї лабораторної роботи я доопрацював існуючий код, зокрема додав нові функціональні можливості, а також вивчав та впроваджував обробку параметрів командного рядка. Окрім цього, я покрив основні частини функціоналу програми юніт тестами, що дозволило забезпечити її стабільність та надійність під час виконання різних операцій.