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.

**Хід роботи**

1.Завантаження початкових даних з CSV файлу: Додана функція завантаження початкових даних з CSV файлу. Файл передається як параметр командного рядку.

Зберігання довідника у форматі CSV перед виходом: Додана функція зберігання довідника студентів у форматі CSV перед завершенням програми. Це дозволяє зберегти внесені зміни та оновлені дані.

3. Розроблені тестові випадки для основних функцій програми, таких як завантаження даних, збереження даних, додавання нового елемента, видалення елемента та оновлення інформації про студента. Це дозволяє перевірити правильність роботи кожної окремої функції.

**Текс програми**

**1.**

import csv

import sys

def load\_data(file\_name):

student\_list = []

with open(file\_name) as file:

reader = csv.DictReader(file)

for row in reader:

student\_list.append({"name": row["name"], "phone": row["phone"], "age": row["age"], "grade": row["grade"]})

return student\_list

def save\_data(file\_name, student\_list):

fieldnames = ["name", "phone", "age", "grade"]

with open(file\_name, "w", newline='') as csvfile:

writer = csv.DictWriter(csvfile, fieldnames=fieldnames)

writer.writeheader()

writer.writerows(student\_list)

def printAllList(student\_list):

for student in student\_list:

strForPrint = f"Student name: {student['name']}, Phone: {student['phone']}, Age: {student['age']}, Grade: {student['grade']}"

print(strForPrint)

return

def addNewElement(student\_list):

name = input("Please enter student name: ")

phone = input("Please enter student phone: ")

age = input("Please enter student age:")

grade = input("Please enter student grade:")

new\_student = {"name": name, "phone": phone, "age": age, "grade": grade}

insert\_position = 0

for student in student\_list:

if name > student["name"]:

insert\_position += 1

else:

break

student\_list.insert(insert\_position, new\_student)

print("New element has been added")

return

def deleteElement(student\_list):

name = input("Please enter name to be deleted: ")

delete\_position = -1

for student in student\_list:

if name == student["name"]:

delete\_position = student\_list.index(student)

break

if delete\_position == -1:

print("Element was not found")

else:

print("Delete position " + str(delete\_position))

del student\_list[delete\_position]

return

def updateElement(student\_list):

name = input("Please enter name to be updated: ")

delete\_position = -1

for student in student\_list:

if name == student["name"]:

delete\_position = student\_list.index(student)

break

if delete\_position == -1:

print("Element was not found")

else:

print("Delete position " + str(delete\_position))

del student\_list[delete\_position]

name = input("Please enter student name: ")

phone = input("Please enter student phone: ")

age = input("Please enter student age:")

grade = input("Please enter student grade:")

new\_student = {"name": name, "phone": phone, "age": age, "grade": grade}

insert\_position = 0

for student in student\_list:

if name > student["name"]:

insert\_position += 1

else:

break

student\_list.insert(insert\_position, new\_student)

print("New element has been added")

return

def main():

if len(sys.argv) != 2:

print("Usage: python script.py <file\_name.csv>")

sys.exit(1)

file\_name = sys.argv[1]

student\_list = load\_data(file\_name)

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(student\_list)

printAllList(student\_list)

case "U" | "u":

print("Existing element will be updated:")

updateElement(student\_list)

printAllList(student\_list)

case "D" | "d":

print("Element will be deleted")

deleteElement(student\_list)

case "P" | "p":

print("List will be printed")

printAllList(student\_list)

case "X" | "x":

print("Exit")

save\_data(file\_name, student\_list)

break

case \_:

print("Wrong choice")

if \_\_name\_\_ == "\_\_main\_\_":

main()

**2.**

import unittest

import io

from unittest.mock import patch

from lab\_02 import load\_data, save\_data, addNewElement, deleteElement, updateElement, printAllList

class TestLab02(unittest.TestCase):

def setUp(self):

self.test\_file = "test\_lab2.csv"

self.test\_data = [

{"name": "Bob", "phone": "0631234567", "age": "22", "grade": "83"},

{"name": "Emma", "phone": "0631234567", "age": "16", "grade": "65"},

]

def test\_load\_data(self):

with open(self.test\_file, "w") as file:

file.write("name,phone,age,grade\nBob,0631234567,22,83\nEmma,0631234567,16,65")

data = load\_data(self.test\_file)

self.assertEqual(data, [

{"name": "Bob", "phone": "0631234567", "age": "22", "grade": "83"},

{"name": "Emma", "phone": "0631234567", "age": "16", "grade": "65"},

])

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\_addNewElement(self):

with patch('builtins.input', side\_effect=["Jon", "0631234567", "19", "87"]):

addNewElement(self.test\_data)

self.assertEqual(len(self.test\_data), 3)

def test\_deleteElement(self):

with patch('builtins.input', return\_value="Emma"):

deleteElement(self.test\_data)

self.assertEqual(len(self.test\_data), 1)

def test\_updateElement(self):

with patch('builtins.input', side\_effect=["Bob", "Jon", "5555555555", "30", "35"]):

updateElement(self.test\_data)

self.assertEqual(self.test\_data[1]["name"], "Jon")

def test\_printAllList(self):

with patch('sys.stdout', new\_callable=io.StringIO) as mock\_stdout:

printAllList(self.test\_data)

output = mock\_stdout.getvalue()

self.assertIn("Bob", output)

self.assertIn("Emma", output)

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

unittest.main()

**Висновок**

Розширення функціональності телефонного довідника студентів було виконано. Тепер програма може працювати з великим обсягом даних, зберігаючи та завантажуючи їх з CSV файлів. Додані тестові випадки підвищують надійність програми та полегшують виявлення та виправлення помилок.