Технології програмування

Лабораторна робота №3

**ООП**

**Мета роботи:** Використовуючи теоретичне підґрунтя про ООП у мові Python переробити програму телефонного довідника студентів використовуючи принципи ООП для формування відомостей про студентів.

**Завдання до лабораторної роботи**

Переробити функціональність телефонного довідника студентів групи, що був розроблений у Лабораторній роботі №2 використовуючи принципи ООП:

1) розробити клас Студент групи з відповідними атрибутами;

2) розробити клас Список групи, має містити не словники, як виконано в лабораторній роботі №2, а об’єкти класу Студент групи; додавання нового запису, видаленні існуючого чи зміна даних має бути виконана через методи класу Список групи.

3) розробити клас для роботи з файлами для зчитування початкової інформації про список групи та збереження інформації по завершенню програми.

4) список студентів має містити не словники, як виконано в лабораторній роботі №2, а об’єкти класу Студент групи;

5) описання всіх класів мають міститися в окремих файлах, що мають відповідні імена(наприклад Student, StudentList, Utils)

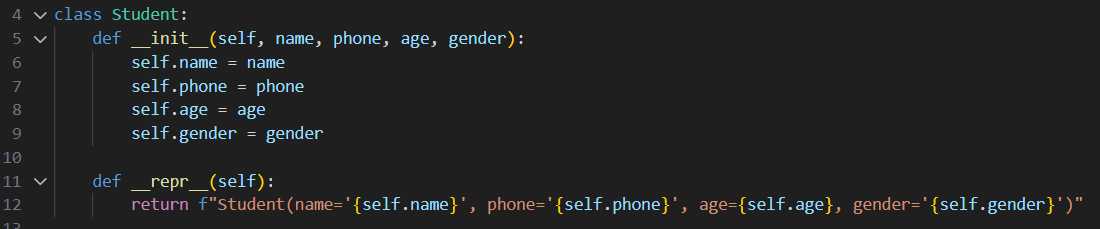
6) основний функціонал програми має бути покритий Юніт тестами.

**Хід роботи**

1. Розробка класу Student з атрибутами. Визначили чотири атрибути:

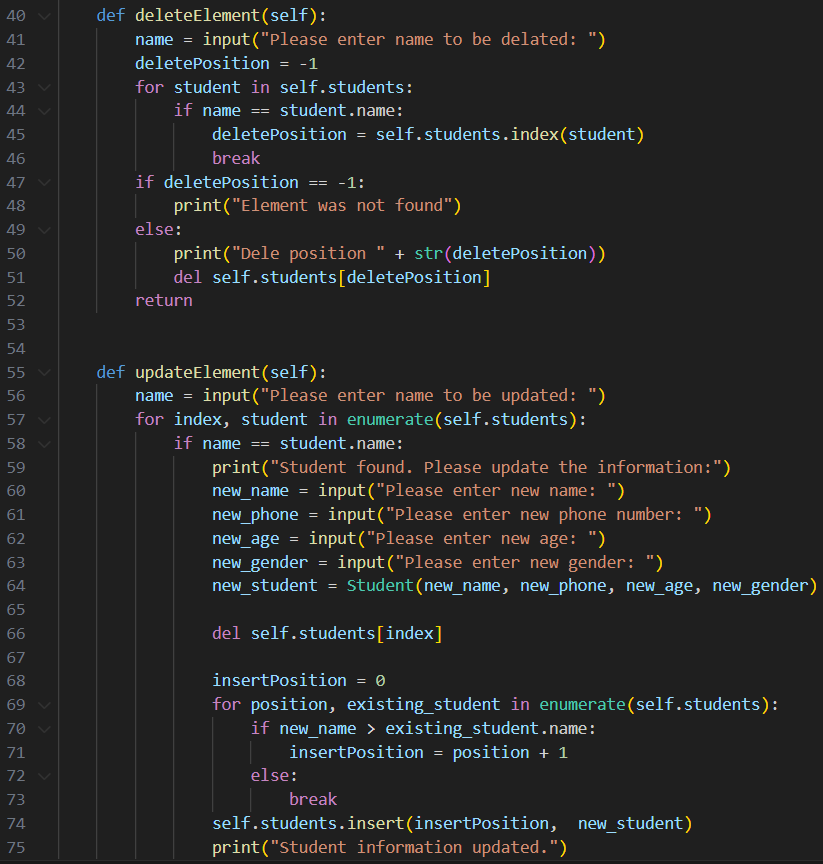
* name: Представляє ім'я студента.
* phone: Представляє номер телефону студента.
* age: Представляє вік студента.
* gender: Представляє стать студента.

Ці атрибути ініціалізуються при створенні об'єкта класу за допомогою конструктора \_\_init\_. А метод \_\_repr\_\_ визначає строкове представлення об'єкта для виведення при використанні print.

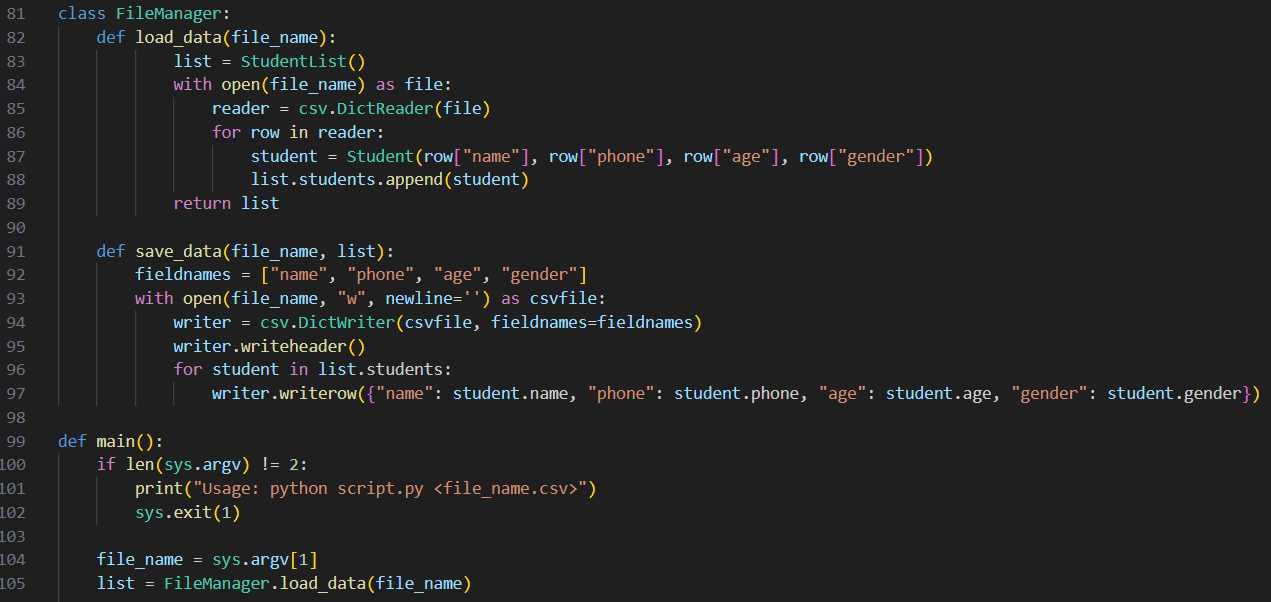


1. Далі розробила клас StudentList, що містить об’єкти класу Студент групи. В цьому класі створені методи додавання нового запису, видалення існуючого, зміна даних та вивід даних.

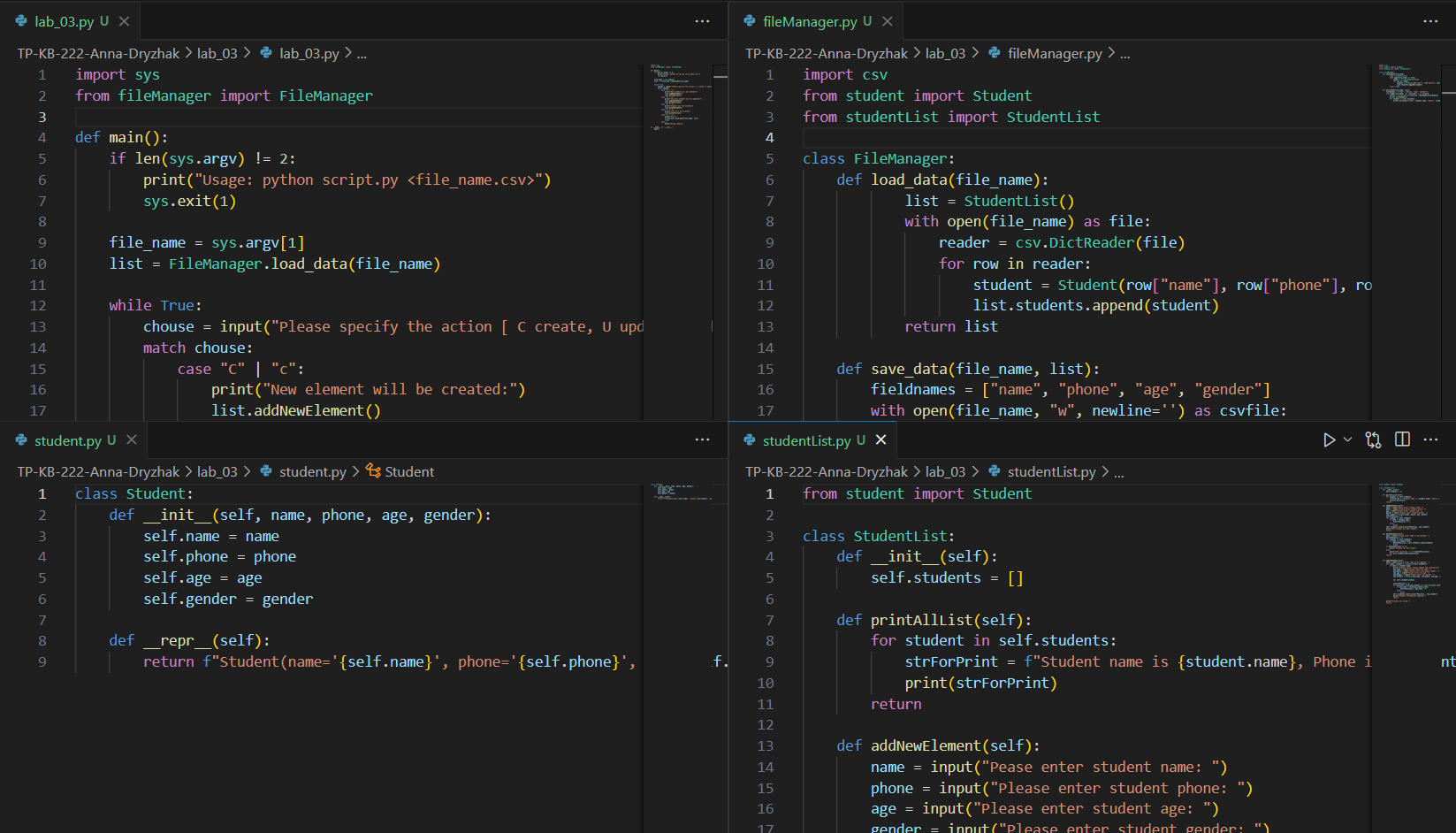




1. Розроблюємо клас FileManager для роботи з файлами для зчитування початкової інформації про список групи та збереження інформації по завершенню програми (методи load\_data та save\_data). Та використовуємо цей клас в основному коді.



1. Створюємо файли fileManager, student та studentList з вмістом відповідних класів.



1. Покриваємо основний функціонал програми Юніт тестами.

Test\_load\_data:

В першому рядку тесту ми створюємо тимчасовий CSV-файл test\_file.csv. Цей файл містить один рядок даних про студента з ім'ям "John", номером телефону "123-456-7890", віком "25" і статтю "male". Після цього ми викликаємо функцію FileManager.load\_data(test\_file) для завантаження даних з цього тимчасового файлу. За допомогою self.assertEqual(len(student\_list.students), 1) ми перевіряємо, чи завантажені дані містять одного студента.

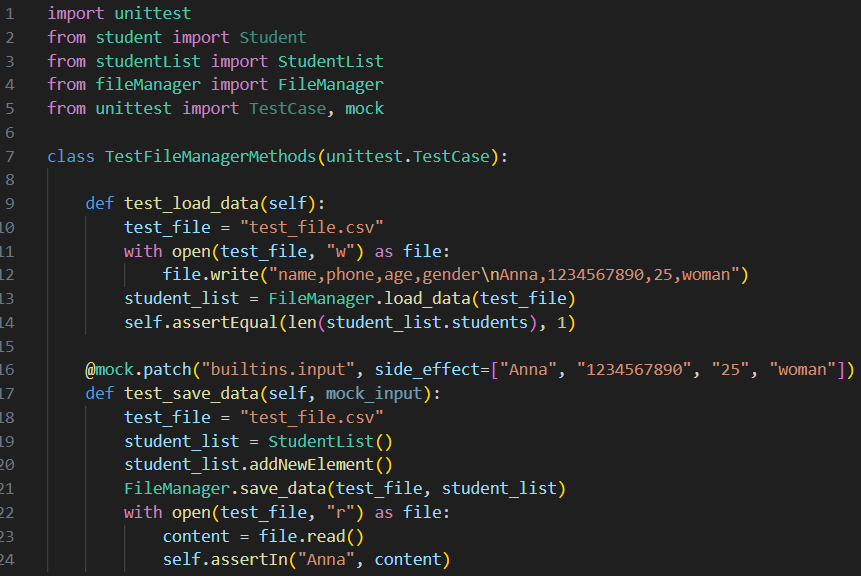
Test\_save\_data:

Декоратор @mock.patch імітує введення користувача. Кожне значення у списку side\_effect буде використано як введення при виклику input(). Створюється об'єкт StudentList і додається новий елемент до списку students з введеними значеннями "Anna", "1234567890", "25", "woman". Ці значення вводяться автоматично через механізм мокування введення. Викликається функція save\_data з об'єктом student\_list. Повине створитись файл "test\_file.csv" і зберегтись в ньому данні студента. Потім читаємо вміст файлу і перевіряємо, чи присутнє "Anna" у вмісті файлу.

Test\_addNewElement:

Знову декоратор @mock.patch для імітації введення користувача. Далі Створюється об'єкт StudentList, і функція addNewElement викликається. Ця функція очікує введення користувача для додавання нового елементу до списку студентів. Після виклику addNewElement очікується, що в списку students буде лише один елемент, і це ім'я елементу буде "Anna". Це перевіряється за допомогою методу assertEqual.

Аналогічно відбувається тестування функції оновлення елементу та видалення.





**Висновок.** Під час виконання даної лабораторної роботи я використовуючи теоретичне підґрунтя про ООП у мові Python переробила програму телефонного довідника студентів використовуючи принципи ООП для формування відомостей про студентів.