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

**Лабораторна робота №3**

**Тема:** ООП

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

**Суть завдання:**

**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.

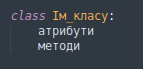
**4)** Список студентів має містити не словники, як виконано в лабораторній роботі №2, а об’єкти класу Студент групи;

**5)** Описання всіх класів мають міститися в окремих файлах, що мають відповідні імена(наприклад Studen, StudentList, Utils)

**6)** Основний функціонал програми має бути покритий Юніт тестами.

**Теоретичні відомості:**

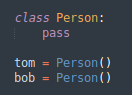
Об'єктно-орієнтоване програмування (ООП) є парадигмою програмування, що базується на концепції "об'єктів" - елементів, які об'єднують дані та функції, що з ними пов'язані, в один вузол. Мова програмування Python, окрім вбудованих типів даних, таких як int та str, також надає можливість визначати власні типи за допомогою класів. Клас визначає загальні характеристики та дії, а об'єкт - конкретний екземпляр цього класу. Можна уявити клас як шаблон, що визначає набір атрибутів (характеристик) та методів (функцій), які пов'язані з цим класом. Кожен об'єкт, який створений з цього класу, представляє собою конкретний екземпляр цього шаблону. Синтаксис визначення класу виглядає наступним чином:



Усередині класу визначаються атрибути, що зберігають різні характеристики класу, та методи - функції класу, які визначають його поведінку. Наприклад, визначимо клас "Person":



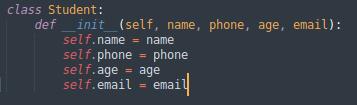
У цьому випадку клас "Person" мається на увазі як загальний шаблон для представлення осіб. Хоча у цьому прикладі не визначено жодних методів чи атрибутів, визначення класу дозволяє створити об'єкти цього класу. Після визначення класу можна створити його об'єкти. Конструктор, який є спеціальною функцією, що створює об'єкт класу, викликається з ім'ям класу.



У цьому випадку "tom" та "bob" - це два об'єкти класу "Person". Зауважте, що кожен об'єкт має свою власну копію атрибутів класу. Класи і об'єкти у Python надають зручний спосіб організації коду та вираження концепцій ООП.

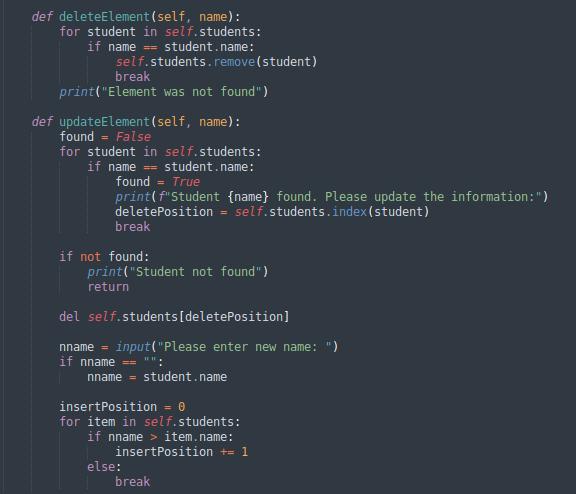
**Хід роботи:**

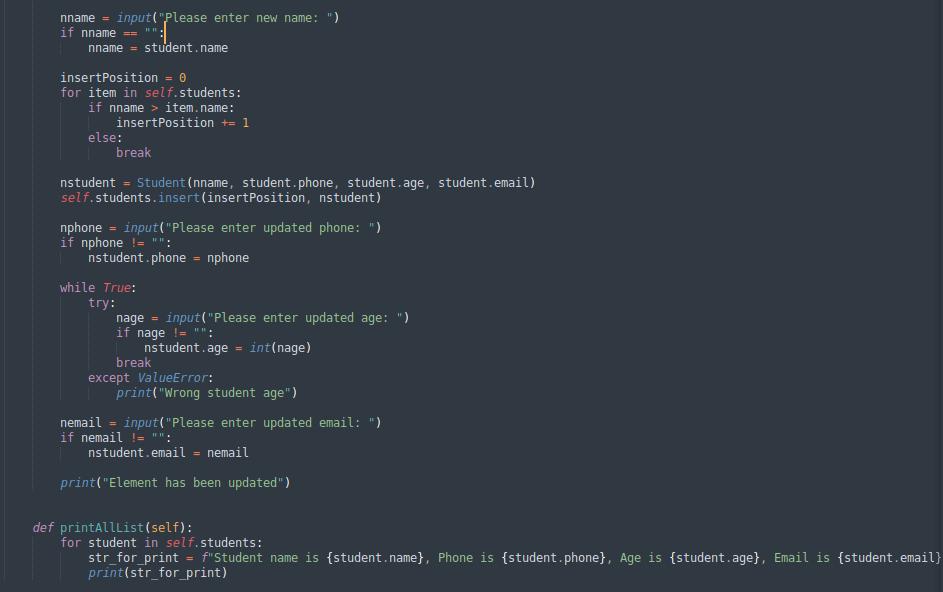
**1.** Я створив клас Student з необхідними атрибутами: ім'ям, телефоном, віком та електронною адресою. Для цього я використовував конструктор класу (\_\_init\_\_), який приймає чотири аргументи - ім'я, телефон, вік, електронна адреса і присвоює їх відповідним атрибутам об'єкта.



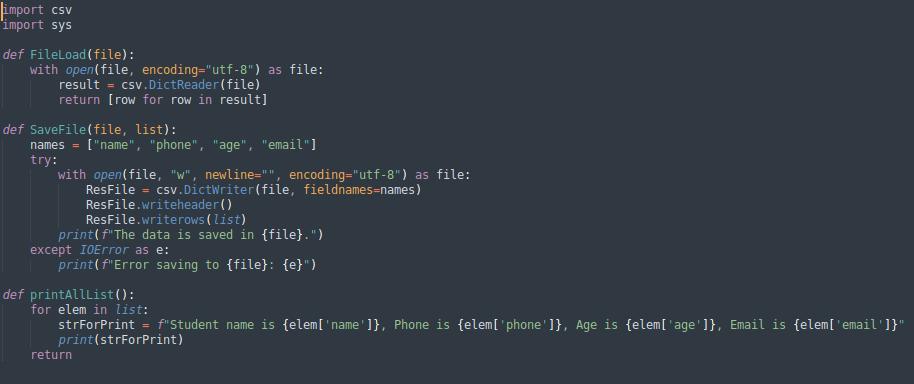
2. Далі я оновив використовуючи клас Student який міститься файлі Student.py. Тепер кожен студент в списку є об'єктом класу Student, якиий має атрибути, такі як ім'я, телефон, вік і електронна пошта. Я відредагував метод addNewElement, який додає нового студента до списку, зчитуючи від користувача ім'я, телефон, вік і електронну пошту. Він тепер створює об'єкт класу Student з введеними даними і додає його до списку студентів. Список потім сортується за ім'ям та віком студентів. Також я відредагував метод deleteElement, який видаляє студента із списку за ім'ям, яке вказується як аргумент методу. Далі я відредагував метод updateElement, він оновлює інформацію про студента за ім'ям, яке вказується як аргумент методу. Якщо студент існує, він видаляється зі списку, а користувачу пропонується ввести нові дані для оновлення інформації про студента. Оновлений об'єкт Student потім додається до списку на відповідну позицію. Також було відредаговано метод printAllList, який виводить на екран усю інформацію про студентів.

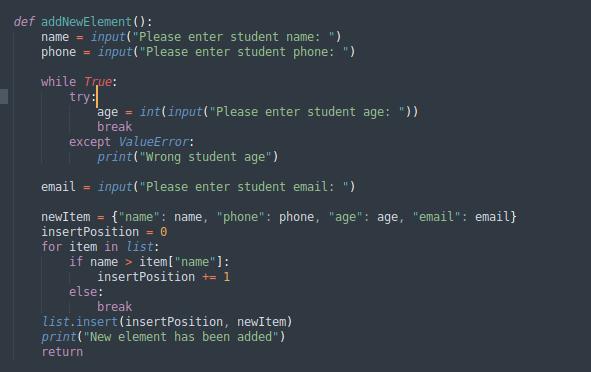




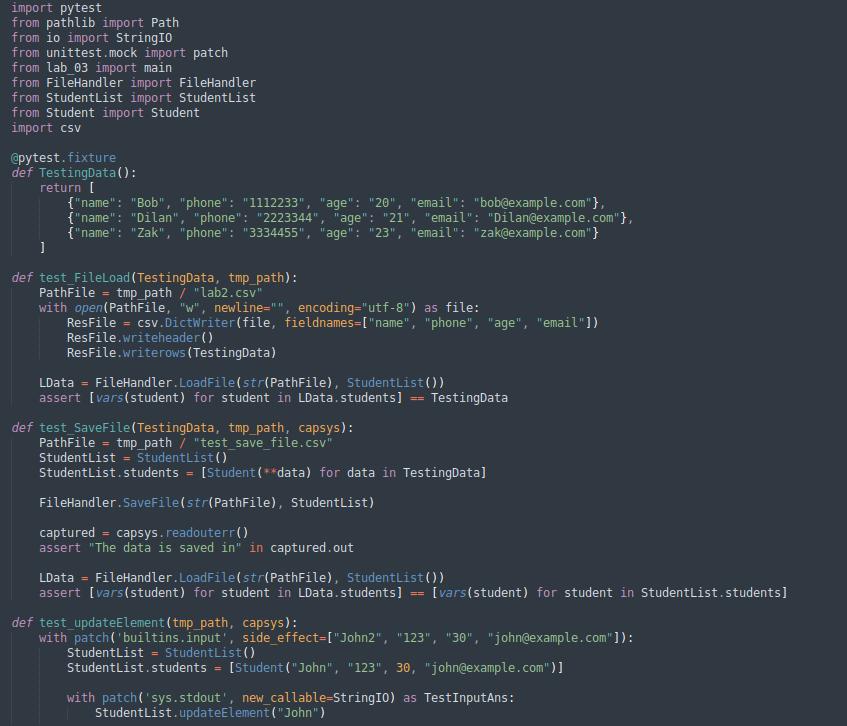


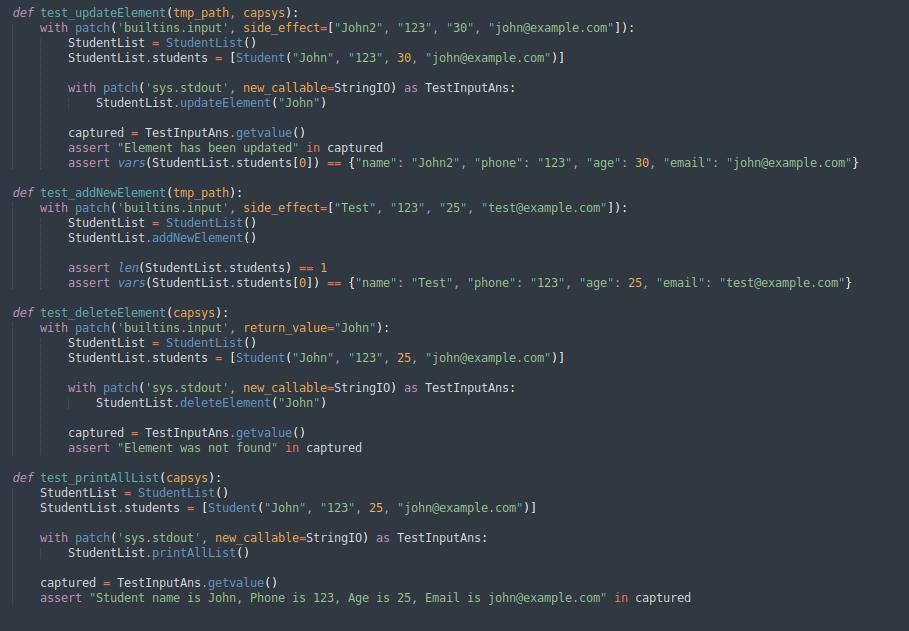
3. Спочатку я розробив клас FileHandler для роботи з файлами, який дозволяє зчитувати початкову інформацію про список групи та зберігати інформацію після завершення програми, в цей класс я помітив методи LoadFile і SaveFile. Метод LoadFile відповідає за зчитування початкової інформації про список групи з файлу. Я передавав ім'я файлу та клас StudentListClass. Метод очищує список студентів у класі StudentListClass та відкриває файл у кодуванні "utf-8", після цього, використовуючи модуль csv, я створив словниковий читач для обробки рядків у файлі. У циклі я створював екземпляри класу Student, передаючи дані з рядка файлу. Після цього я додаю ці студенти до списку студентів у класі StudentListClass. На кінець, повертаю оновлений клас StudentListClass. Потім я створив метод SaveFile, який відповідає за збереження інформації у файл. Я також передавав ім'я файлу та клас StudentListClass. Метод відкриває файл для запису у кодуванні "utf-8". Я визначив імена полів у файлі ("name", "phone", "age", "email") та використовував модуль csv для створення об'єкта DictWriter. Записавши заголовок за допомогою writeheader(), я використовував метод writerows для запису даних у файл. Для отримання даних про студентів у вигляді словників, я використовував vars(student) для кожного об'єкта студента в списку. Я також обробив виключення IOError у разі невдачі запису. На кінець, вивів повідомлення про успішне збереження даних, або повідомлення про помилку, якщо щось пішло не так.

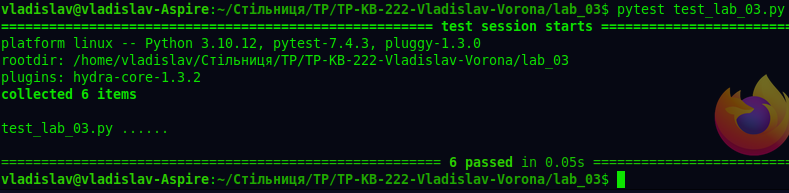




4. Далі я змінив код для юніт-тестів pytest, я використав фікстури pytest для передачі тестових даних і тимчасового шляху, замість lab\_02.FileLoad та lab\_02.SaveFile, я тепер викликаю відповідні методи з класу FileHandler і використовую клас StudentList для операцій над студентами в списку. Також я використав StringIO для перехоплення виводу та його подальшого порівняння в тестах і для зручності я використав unittest.mock.patch для заміни введення користувача під час тестування. Також я пристосував всі увесь код до роботи із об’єктами і класами.







**Висновок:** під час виконання цієї лабораторної роботи я створив програму-довідник студентів, використовуючи принципи об'єктно-орієнтованого програмування (ООП). Основні завдання включали розробку класів для представлення студента, списку групи та роботи з файлами, а також покриття основного функціоналу програми юніт-тестами. Почавши з створення класу Student з відповідними атрибутами, я використовував конструктор класу для ініціалізації об'єктів. Далі, розробивши клас StudentList, я перейшов до реалізації методів для додавання, видалення та оновлення інформації про студентів у списку. Також перейшов до використання об'єктів класу Student замість словників для представлення інформації про студентів у списку. У кроці з розробкою класу FileHandler для роботи з файлами, я реалізував методи для зчитування та збереження даних, використовуючи модуль csv, я забезпечив ефективну обробку інформації у файлах. Під час виконання лабораторної роботи я отримав навички в розробці об'єктно-орієнтованих програм, роботі з файлами та написанні тестів для перевірки правильності функціоналу. Ці навички будуть корисні у подальших проектах, де потрібно використовувати принципи ООП для організації коду та забезпечення його ефективності та надійності.