Технології програмування

Лабораторна робота №1

**Списки. Словники. Кортежі**

**Мета роботи**: Використовуючи теоретичне підґрунтя про складні структури даних Списки, Словники Кортежі, та використовуючи існуючий код, доробити програму додавши функціонал, що буде вказано в завданні до лабораторної роботи.

**Завдання до лабораторної роботи:**

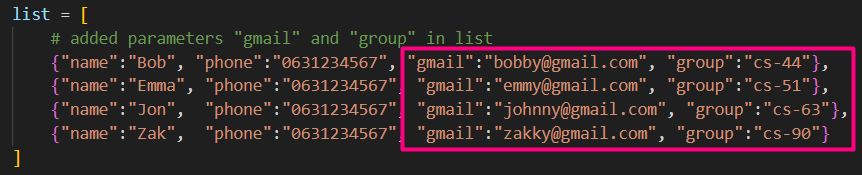
Реалізувати відсортований телефонний довідник студентів групи.

Для виконання задання надано частину готового функціоналу, яка реалізує безкінечний цикл запитів до користувача. Типи запитів: додати нового студента, змінити данні про існуючого студента, видалити запис, роздрукувати всю таблицю та вихід із програми. Реалізований функціонал додавання нового запису та видалення існуючого. Всі дії відбуваються з відсортованим списком студентів.

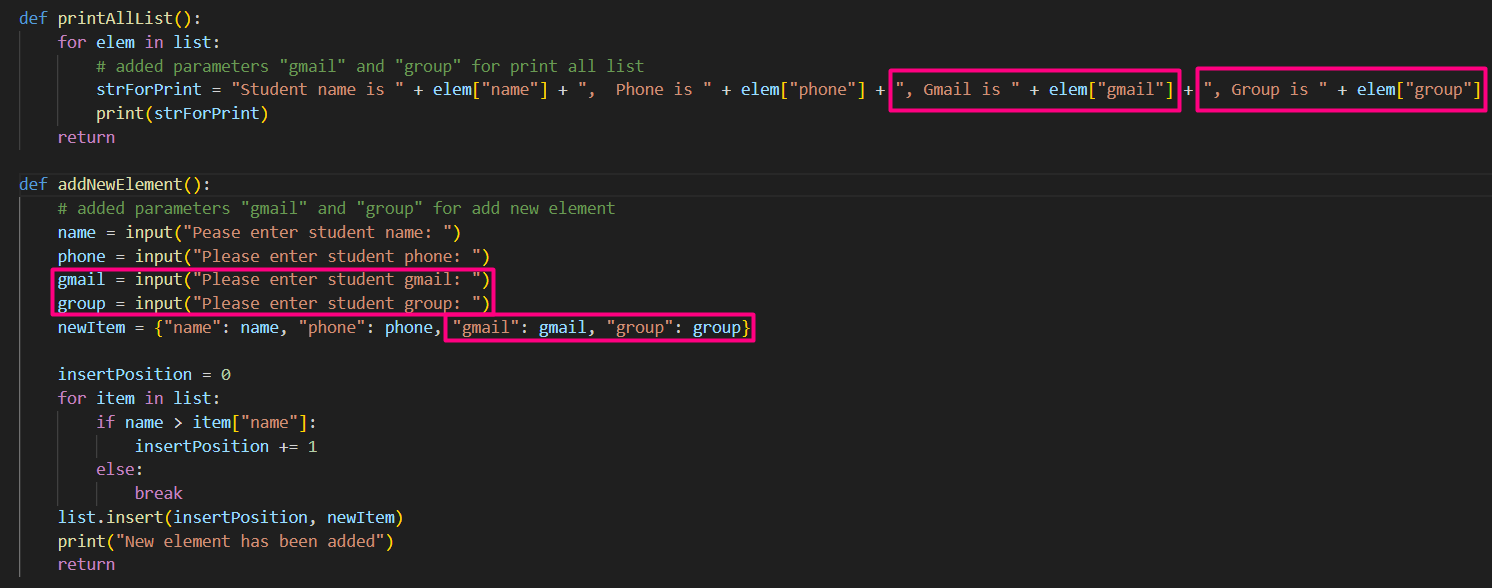
1. Необхідно розширити відомості про студента до 4х полів.
2. Необхідно переробити існуючий функціонал враховуючи розширення відомості про студента до 4х полів.
3. Необхідно реалізувати з нуля функціонал зміни інформації про студента враховуючи той факт, що вже існує реалізація додавання нового запису та видалення існуючого. При зміні інформації про студента список має залишатись відсортованим.

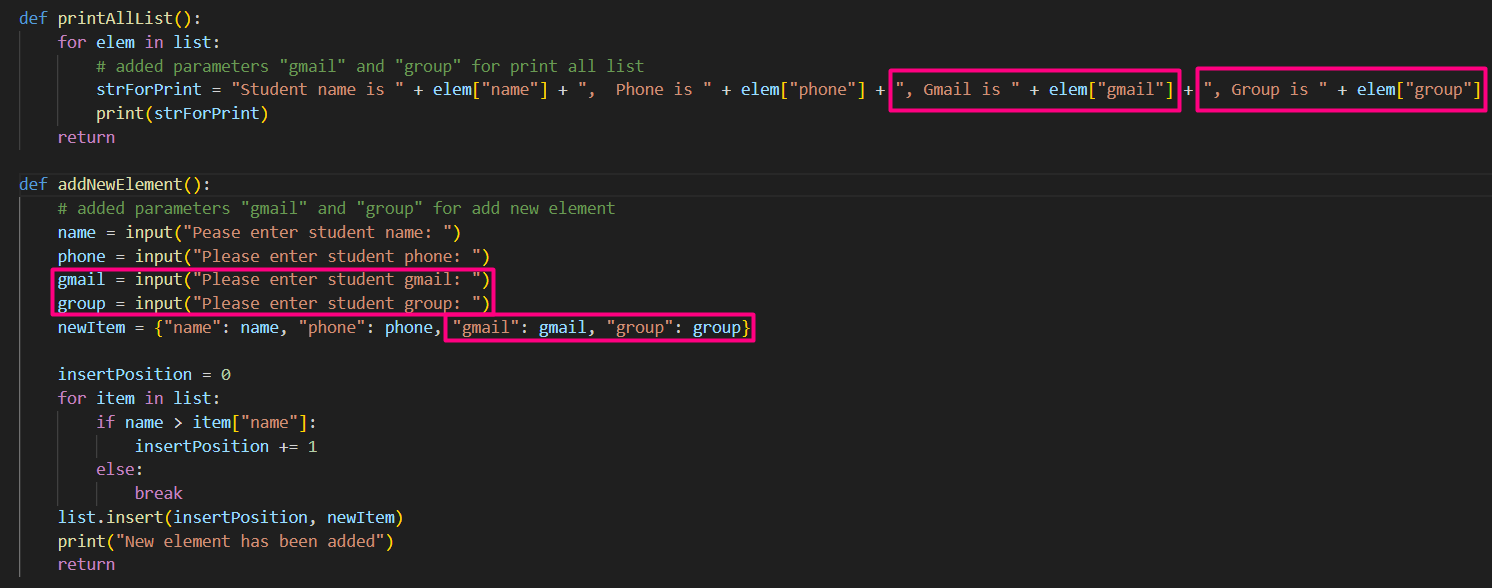
**Хід роботи:**

1. Розширили відомості про студентів додавши поля gmail та group. Заповнили їх даними. Тепер список словників list складається із 4 полів: name, phone, gmail, group.



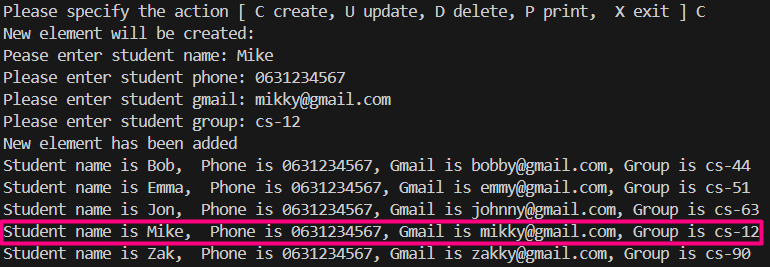
1. У функції виводу інформації printAllList() та додавання нових студентів addNewElement() додали поля gmail та group. Функція printAllList() перебирає список студентів і формує рядок, що містить інформацію про кожного студента: його ім'я, номер телефону, електронну пошту та групу. Функція addNewElement() запитує у користувача інформацію про нового студента та зберігає її у словнику newItem. Потім вона визначає позицію вставки в списку list, порівнюючи ім'я нового студента з існуючими. Якщо ім'я нового студента більше за ім'я поточного, позиція переміщається далі. Після визначення позиції новий студент вставляється в список, і виводиться повідомлення про успішне додавання. Тепер значення в функціях printAllList() та addNewElement() розширили до виводу та запису елементів gmail та group.





Результат роботи коду:

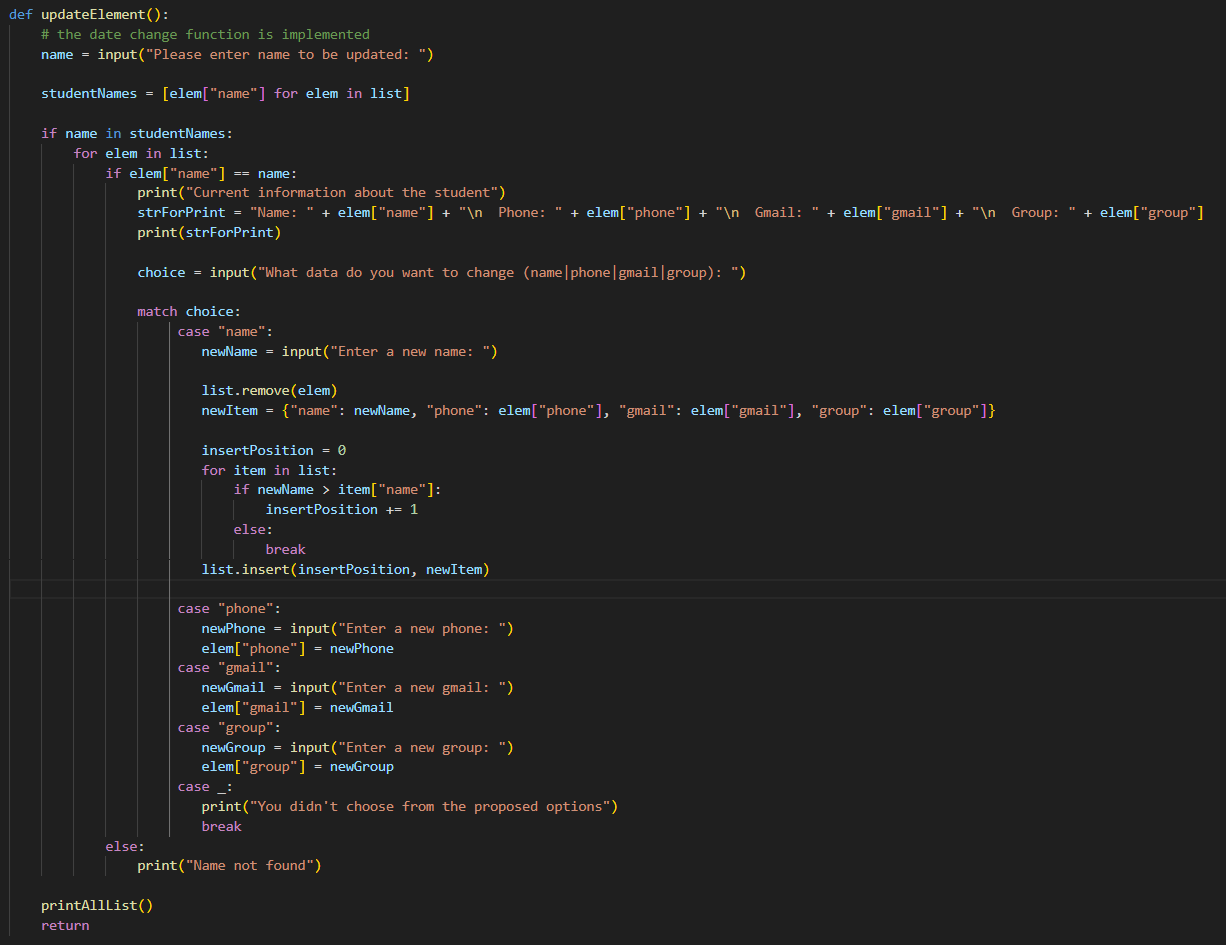
Додамо нового студента вибравши в меню “C” – create.



1. Реалізували функціонал зміни даних студента, таких як: імені, номеру телефону, електронної пошти, групи. Функція updateElement() дозволяє користувачеві оновлювати інформацію про студента в списку, запитуючи ім'я студента та перевіряючи його наявність. Якщо студент знайдений, функція виводить поточні дані, пропонує змінити конкретні поля (ім'я, телефон, пошту або групу), оновлює відповідну інформацію, сортує список і відображає нову інформацію. Якщо студент не знайдений, виводиться повідомлення про помилку.

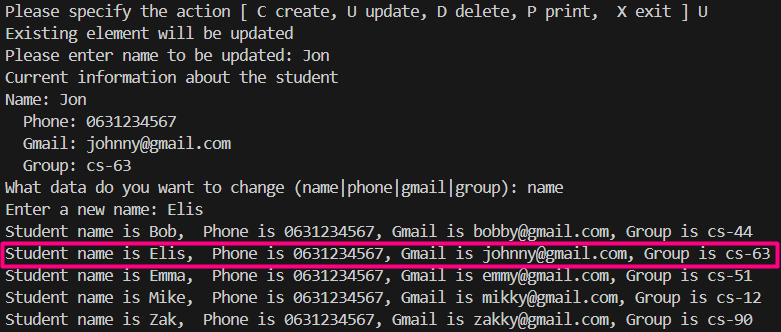
Пояснення коду:

* name = input(“Please enter name to be updated: “) – Запитує у користувача ім'я студента, інформацію про якого потрібно оновити.
* studentNames = [elem[“name”] for elem in list] – Створює список studentNames, що містить всі імена студентів з наявного списку.
* if name in studentNames: – Перевіряє, чи наявне ім'я є в списку імен студентів.
* Цикл for elem in list: – Проходить через всі елементи списку для пошуку студента з відповідним ім'ям.
* Виведення поточної інформації: – Якщо студент знайдений, виводиться його поточна інформація (ім'я, телефон, Gmail, група).
* choice = input(“What data do you want to change (name|phone|gmail|group): “) – Запитує, яку саме інформацію користувач хоче змінити.
* match choice: – Використовується конструкція match для вибору відповідного поля для зміни.
* Введення нових даних: – У випадку вибору конкретного поля, запитує нову інформацію і оновлює відповідне поле у словнику студента.
* case "name": – Якщо обрано поле "name", запитує нове ім'я студента.
* newName = input("Enter a new name: ") – Запитує у користувача нове ім'я студента.
* list.remove(elem) – Видаляє старий запис студента зі списку.
* newItem = {"name": newName, "phone": elem["phone"], "gmail": elem["gmail"], "group": elem["group"]} – Створює новий словник студента з новим ім'ям та старими даними.
* insertPosition = 0 – Ініціалізує позицію для вставки нового елемента.
* for item in list: – Проходить через список для визначення позиції нового елемента.
* if newName > item["name"]: – Перевіряє, чи нове ім'я повинно йти перед існуючим ім'ям.
* list.insert(insertPosition, newItem) – Вставляє новий елемент у відповідне місце в списку.
* case "phone": – Якщо обрано поле "phone", запитує новий номер телефону.
* newPhone = input("Enter a new phone: ") – Запитує у користувача новий номер телефону.
* elem["phone"] = newPhone – Оновлює номер телефону студента.
* case "gmail": – Якщо обрано поле "gmail", запитує нову електронну адресу.
* newGmail = input("Enter a new gmail: ") – Запитує у користувача нову електронну адресу.
* elem["gmail"] = newGmail – Оновлює електронну адресу студента.
* case "group": – Якщо обрано поле "group", запитує нову групу.
* newGroup = input("Enter a new group: ") – Запитує у користувача нову групу.
* elem["group"] = newGroup – Оновлює групу студента.
* case \_: – Якщо вибір не збігається з жодним з варіантів, виводиться повідомлення.
* print("You didn't choose from the proposed options") – Виводить повідомлення, якщо користувач зробив неправильний вибір.
* printAllList() – Виклик функції, що виводить інформацію про студентів.
* else: – Якщо ім'я студента не знайдено, виводиться повідомлення “Name not found”.
* return – Завершує виконання функції.

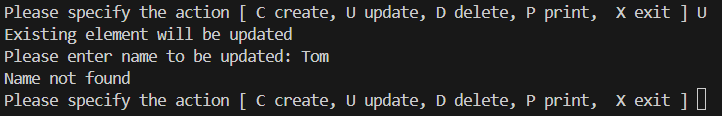


Результат роботи коду:

Змінимо ім’я студента вибравши в меню “U” – update.



Спробуємо ввести ім’я якого немає в списку.

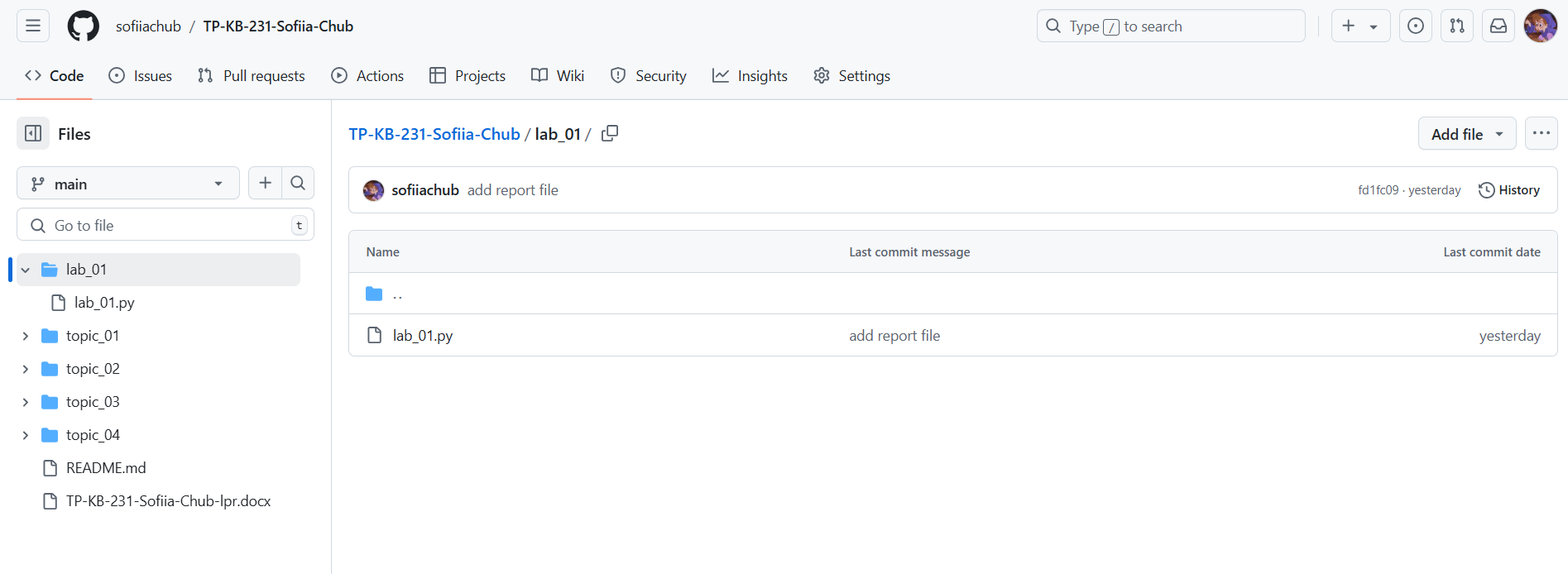


Повний код програми

|  |
| --- |
| list = [  # added parameters "gmail" and "group" in list  {"name":"Bob", "phone":"0631234567", "gmail":"bobby@gmail.com", "group":"cs-44"},  {"name":"Emma", "phone":"0631234567", "gmail":"emmy@gmail.com", "group":"cs-51"},  {"name":"Jon", "phone":"0631234567", "gmail":"johnny@gmail.com", "group":"cs-63"},  {"name":"Zak", "phone":"0631234567", "gmail":"zakky@gmail.com", "group":"cs-90"}  ]    def printAllList():  for elem in list:  # added parameters "gmail" and "group" for print all list  strForPrint = "Student name is " + elem["name"] + ", Phone is " + elem["phone"] + ", Gmail is " + elem["gmail"] + ", Group is " + elem["group"]  print(strForPrint)  return    def addNewElement():  # added parameters "gmail" and "group" for add new element  name = input("Pease enter student name: ")  phone = input("Please enter student phone: ")  gmail = input("Please enter student gmail: ")  group = input("Please enter student group: ")  newItem = {"name": name, "phone": phone, "gmail": gmail, "group": group}    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    def deleteElement():  name = input("Please enter name to be delated: ")  deletePosition = -1  for item in list:  if name == item["name"]:  deletePosition = list.index(item)  break    if deletePosition == -1:  print("Element was not found")  else:  print("Dele position " + str(deletePosition))  del list[deletePosition]  return    def updateElement():  # the date change function is implemented  name = input("Please enter name to be updated: ")    studentNames = [elem["name"] for elem in list]    if name in studentNames:  for elem in list:  if elem["name"] == name:  print("Current information about the student")  strForPrint = "Name: " + elem["name"] + "\n Phone: " + elem["phone"] + "\n Gmail: " + elem["gmail"] + "\n Group: " + elem["group"]  print(strForPrint)    choice = input("What data do you want to change (name|phone|gmail|group): ")    match choice:  case "name":  newName = input("Enter a new name: ")    list.remove(elem)  newItem = {"name": newName, "phone": elem["phone"], "gmail": elem["gmail"], "group": elem["group"]}    insertPosition = 0  for item in list:  if newName > item["name"]:  insertPosition += 1  else:  break  list.insert(insertPosition, newItem)    case "phone":  newPhone = input("Enter a new phone: ")  elem["phone"] = newPhone  case "gmail":  newGmail = input("Enter a new gmail: ")  elem["gmail"] = newGmail  case "group":  newGroup = input("Enter a new group: ")  elem["group"] = newGroup  case \_:  print("You didn't choose from the proposed options")  break  else:  print("Name not found")    printAllList()    return    def main():  while True:  chouse = input("Please specify the action [ C create, U update, D delete, P print, X exit ] ")  match chouse:  case "C" | "c":  print("New element will be created:")  addNewElement()  printAllList()  case "U" | "u":  print("Existing element will be updated")  updateElement()  case "D" | "d":  print("Element will be deleted")  deleteElement()  case "P" | "p":  print("List will be printed")  printAllList()  case "X" | "x":  print("Exit()")  break  case \_:  print("Wrong chouse")    main() |

Посилання на github: <https://github.com/sofiiachub/TP-KB-231-Sofiia-Chub/tree/main/lab_01>

Знімок екрану з посилання на github:



**Висновки:** Використовуючи теоретичне підґрунтя про складні структури даних списки, словники, кортежі, та використовуючи існуючий код, доробила програму додавши функціонал зміни інформації про студента. Також розширила функції виводу інформації та створення нового студента новими даними. Познайомилась та зрозуміла роботу анонімної функції lambda. Засвоїла вивчений матеріал та використала його на практиці.