ПРАКТИЧНА РОБОТА №4. РОБОТА З ФАЙЛАМИ ТА ОРГАНІЗАЦІЯ ПРОЕКТІВ

Мета роботи – вивчити та засвоїти базові принципи роботи з файлами, модулями та пакетами. Ознайомитися з існуючими режимами доступу, а також з особливостями використання різних методів і функцій для роботи з файлами. Дізнатися про особливості та способи підключення модулів та організації пакетів.

4.1 Загальні відомості

4.1.1 Відкриття файлу

Відкрити файл можна за допомогою функції open:

```
open(name[, mode[, buffering]])
```

Функція повертає файловий об'єкт. Обов'язковий тільки перший аргумент. Якщо інші параметри відсутні, файл буде доступний на читання. Таблиця режимів (mode) функції open:

'r' - читання.
'w' - запис.
'a' - додавання.
'b' - бінарний режим.
'+' - читання / запис.

Режим '+' може бути доданий до решти режимів. За замовчуванням python відкриває файли в текстовому режимі. Для відкриття файлу в бінарному режимі на читання можна додати 'rb'. Третій параметр встановлює розмір буферизації при роботі з файлом. За замовчуванням він вимкнений, і читання / запис йде безпосередньо з диска на диск. Для включення буфера третій параметр повинен бути відмінним від нуля.

4.1.2 Базові файлові методи

У python багато об'єктів є файлами: стандартне введення sys.stdin, стандартний вивід sys.stdout, об'єкти, що відкриваються функцією urllib.urlopen і т.д.

Запис в файл:

```
3]: f = open('my_file', 'w')
    f.write('Hello, ')
    f.write('World!')
    f.close()
```

Читання:

```
[5]: f = open('my_file', 'r')
     s=f.read(5)
     print(s, end="")
     s=f.read()
     print(s, end="")

Hello, World!
```

За замовчуванням метод read() читає дані послідовно по порядку, від початку і до кінця файлу. Для довільного доступу до файлу є функція seek:

```
[ ]: seek(offset[, whence])
```

offset - зміщення в байтах відносно початку файлу;

whence - за замовчуванням дорівнює нулю, вказує на те, що зміщення береться щодо початку файлу.

```
[7]: f = open(r'my_file', 'w')
     f.write('01234567890123456789')
     f.seek(5)
     f.write('Hello, World!')
     f.close()
     f = open(r'my_file')
     s=f.read()
     print(s, end="")

01234Hello, World!89
```

Функція tell () повертає поточну позицію файлу.

4.1.3 Порядкова робота з файлами

Зазвичай ми маємо справу з текстовими файлами. Прочитати один рядок:

```
file.readline()
```

Функція readline () без параметра читає весь рядок, наявність параметра вказує функції максимальне число символів рядка, яке буде прочитано.

Прочитати всі рядки і повернути список рядків:

```
file.readlines()
```

Записати рядки в файл:

```
file.writelines()
```

Приклад. Прочитати файл і записати його вміст в інший файл:

```
[18]: f = open(r'my_file')
lines = f.readlines()
f.close()
lines[0] = "This is a my_file2 \n" # изменяем 1-ю строку
f = open(r'my_file2','w')
f.writelines(lines)
f.close()
f = open(r'my_file2','r')
s=f.readlines()
print(s)
f.close()
['This is a my_file2 \n']
```

4.1.4 Закриття файлу

Для закриття файлу є метод close ( ). Зазвичай файл закривається сам після того, як ви виходите з програми, але файли потрібно закривати вручну з кількох причин.

Пітон може буферізувати запис в файл даних, що може привести до несподіваних ефектів і виникнення помилок.

У операційної системи є обмеження на число одночасно відкритих файлів.

При доступі до файлу з різних місць одночасно і на читання, і на запис необхідно синхронізувати файлові операції. Буферизація запису може привести до того, що запис вже стався, а даних в файлі ще немає.

Для повної впевненості в закритті файлу можна використовувати блок try / finally:

```
[ ]: try:
        # Тут іде запис до файлу
     finally:
        file.close()
```

Можна також використовувати менеджер контексту, який в будь-якому випадку закриє файл:

```
with open("my_file") as somefile:
    do_something(somefile)
```

Якщо ви все ж не хочете закривати файл, то синхронізувати розрахований на багато користувачів доступ до файлу на читання / запис можна за допомогою функції flush (), яка актуалізує всі операції запису на диск. При цьому можливе блокування файлу на читання.

4.1.5 Ітерація

Ітерація по файлу є базовою операцією і має безліч варіантів. Використання функції read () для байтового читання:

```
[25]: f = open("my_file")
      while True:
          char = f.read(1)
          if not char: break
          print(char,end="")
      f.close()
```

```
01234Hello, World!89
```

Прогресивне читання текстових файлів і функція readline ():

```
[26]: f = open("my_file")
      while True:
          line = f.readline()
          if not line: break
          print(line,end="")
      f.close()
```

```
01234Hello, World!89
```

4.1.6 Бінарні файли

Стандартний модуль struct дозволяє перетворювати об'єкти в структури C у вигляді рядків в бінарному форматі і назад. Дані в рядку розташовуються відповідно до рядку формату. Ці можливості можуть бути використані для читання і збереження в двійковому форматі.

Функції цього модуля:

```
pack(format, value1, value2 ...)
```

Повертає рядок, що містить значення value1 ..., упаковані відповідно до формату. Кількість і тип аргументів повинні відповідати значенням, які вимагає рядок формату format.

```
unpack(format, string)
```

Розпаковує рядок string відповідно до формату format і повертає кортеж об'єктів.

```
calcsize(format)
```

Повертає розмір структури (тобто довжину рядка), що відповідає формату format.

Перед символом формату може йти число, що позначає кількість повторень. Наприклад, рядок формату '4h' повністю еквівалентна рядку 'hhhh'. Символи пропуску між символами формату ігноруються, проте символи пропуску між числом і символом формату не допускаються.

Число перед символом формату 's' інтерпретується як довжина рядка, а не число повторень. Тобто '10s' позначає рядок з 10 символів, в той час як '10c' - 10 раз по одному символу.

Можна змінити порядок проходження байтів вручну:

```
< - little-endian
> - big-endian
```

У наступному прикладі ми пакуємо в структуру два числа - ціле і float, рядок з п'яти символів, зберігаємо в бінарний файл, а потім витягуємо з файлу:

[19]:
```
from struct import *
out = open("123.bin", "wb")
f = "if5s"
data = pack(f, 24, 12.48,b"12345")
out.write(data)
out.close()
input = open("123.bin", "rb")
data = input.read()
input.close()
format = "if5s"                    # one integer
value,value2,value3 = unpack(format, data) # note the ',' in 'value,':
print(value)
print(value2)
print(value3)
print(calcsize(format))
```

```
24
12.479999542236328
b'12345'
13
```

4.2 Базові поняття про модулі

Модулі виконують як мінімум три важливі функції:

- Повторне використання коду: такий код може бути завантажений багато разів у багатьох місцях;
- Управління адресним простором: модуль – це високорівнева організація програм, це пакет імен, який позбавляє вас від конфліктів. Кожен об'єкт «проживає» свій цикл всередині свого модуля, тому модуль - це засіб для угруповання системних компонентів;
- Глобалізація сервісів і даних: для реалізації об'єкта, який використовується в багатьох місцях, досить написати один модуль, який слід імпортувати.

Python дозволяє помістити класи, функції або дані в окремий файл і використовувати їх в інших програмах. Такий файл називається модулем. Об'єкти з модуля можуть бути імпортовані в інші модулі. Файл утворюється шляхом додавання до імені модуля розширення .py. При імпорті модуля інтерпретатор шукає файл з ім'ям my_module.py спочатку в поточному каталозі, потім в каталогах, зазначених у змінній оточення PYTHONPATH, потім в залежних від платформи шляхах за замовчуванням, а також в спеціальних файлах з розширенням '.pth', які лежать в стандартних каталогах. Програміст

може внести зміни в PYTHONPATH і в '.pth', додавши туди свій шлях. Каталоги, в яких здійснюється пошук, можна подивитися в змінної sys.path.

Великі програми, як правило, складаються з стартового файлу - файлу верхнього рівня, і набору файлів-модулів. Головний файл займається контролем програми. У той же час модуль - це не тільки фізичний файл. Модуль являє собою колекцію компонентів. У цьому сенсі модуль - це простір імен, - namespace, і всі імена всередині модуля ще називаються атрибутами - такими, наприклад, як функції і змінні.

4.3 Імпорт модулів

Якщо запустити в каталозі, в якому лежить даний модуль (наприклад, my_module.py), інтерпретатор:

```
>>> python
```

і потім зробити імпорт модуля:

```
>>> import my_module
```

то ми отримуємо доступ до всіх функцій, які в модулі визначені:

```
>>> my_module.func1()
>>> my_module.func2()
...
```

Для більш короткого запису можна створити локальну змінну:

```
>>> f1 = my_module.func1
```

Другий варіант імпорту - взяття безпосередньо імені без імені модуля:

```
>>> from my_module import func1, func2
>>> func1()
```

Третій варіант імпорту - включення всіх імен, визначених в модулі:

```
>>> from my_module import *
>>> func1()
```

Приклад. Імпорт на основі from має таку особливість, що він робить імпортовані атрибути read-only:

```
>>> from small import x, y
>>> x = 42
```

В даному випадку x - це локальна змінна, в той час як змінні x, y в самому модулі small не змінюються:

```
>>> import small
>>> small.x = 42
```

тут **x** - глобальна змінна.

Щоб уникнути непорозумінь import краще без from в тих випадках, коли один і той же модуль використовується в декількох місцях. Оскільки модуль завантажується один раз, для його повторного завантаження можна використовувати функцію reload ().

Кожен модуль має власний простір імен, що є глобальною областю видимості для всіх визначених у ньому функцій. Для того щоб змінні цього модуля не потрапили в конфлікт з іншими глобальними іменами або іншими модулями, потрібно використовувати префікс: _ім'я_модуля _._ ім'я_змінної.

Модулі можуть імпортувати інші модулі. Зазвичай інструкцію import розташовують на початку модуля або програми.

4.4 Створення власного модуля

Створимо файл mymodule.py, в якій визначимо якісь функції:

```
def hello():
    print('Hello, world!')

def fib(n):
    a = b = 1
    for i in range(n - 2):
        a, b = b, a + b
    return b
```

Тепер в цій же папці створимо інший файл, та підключимо у ньому створений модуль.

[2]:
```
import My_module
My_module.hello()
print(My_module.fib(10))
```

```
Hello, world!
55
```

4.5 Пакети

Пакети – спосіб структурування просторів імен модулів на основі файлової системи. Пакетна організація дає всі зручності з управління великою кількістю файлів. Пакетний імпорт робить код більш читальним і значно спрощує пошук. Якщо весь код структурований в одному рутовому каталозі, все, що потрібно додати в PYTHONPATH - це рутовий каталог.

Так само, як застосування модулів робить безпечним використання глобального простору імен авторами різних модулів, застосування пакетів робить безпечним використання імен модулів авторами багатомодульних пакетів.

Наприклад, є пакет, який лежить в кореневій папці TCP. У ньому лежать два підкаталогу - Server і Client:

```
TCP/
    _init_.py
    main.py

    Server/
          _init_.py
          tcp.py
          server.py
          lib.py
    Client/
          _init_.py
          tcp.py
          client.py
          lib.py
```

Файл _init_.py необхідний для того, щоб інтерпретатор розпізнав каталог, як що містить пакет. Зазвичай це порожній файл. Тоді імпорт індивідуальних модулів пакета може бути таким:

```
>>> import TCP.Server.lib
>>> import TCP.Client.lib
```

Посилання на функцію повинно бути повним:

```
>>> import TCP.Server.lib.connect()
```

Можна зробити альтернативне завантаження:

```
>>> from TCP.Server import lib as server_lib
>>> from TCP.Client import lib as client_lib
>>> server_lib.connect()
>>> client_lib.connect()
```

Тут замість lib може бути підставлений модуль, підпакету або ім'я, визначене в TCP.Server - тобто це може бути функція, клас чи змінна.

Що стосується варіанту з імпортом:

```
>>> from TCP import *
```

то в кореневому __init__.py може бути визначений список __all__, в якому перераховуються модулі, які імпортуються в цьому випадку. наприклад:

```
__all__ = ["Server", "Client"]
```

4.6 Варіанти завдання

Для довідника реалізованого згідно власного варіанту у практичній роботі №3, створити програму з підтримкою функцій роботи з файлами, а саме:

- Збереження довідника до файлу.
- Читання довідника з файлу.
- Збереження змін у довіднику.

Таблиця 4.1 – Варіанти завдань

№	Завдання
1.	Довідник - **«Аеропорт»** Поля - **[**ПІБ] [Рейс] [Клас] [Місце] [Вартість квитка]
	Вивести білети з вартістю нижче ніж середня вартість квитка
2.	Довідник - **«Пошта»** Поля - **[**ID відправлення] [Відправник] [Одержувач] [Адреса] [Вага]
	Вивести відправлення з вагою більшою за N (N вводити з клавіатури)
3.	Довідник - **«Квітковий магазин»** Поля - **[**Назва квітки] [Кількість на складі] [Вартість за шт.] [Дата поставки] [Термін зберігання]
	Вивести усі квіти залишок яких на складі більший за N (N вводити з клавіатури)

№	Завдання
4.	Довідник - **«Транспортні компанії»** Поля - [Назва] [Кількість Авто] [Вартість 1км перевезення] [Адреса] [Макс. допустима вага]
	Вивести компанії яких Макс. допустима вага більша за N (N вводити з клавіатури)
5.	Довідник - **«Футбольний Матч»** Поля - [Команда 1] [Команда 2] [Рахунок] [Попереджень] [Видаленнь]
	Вивести матчі у яких суддя діставав картку більше ніж N разів (N вводити з клавіатури)
6.	Довідник - **«Магазин Техніки»** Поля - **[**ID товару] [Назва] [Вартість за шт.] [Кількість на складі] [Термін гарантії (місяців, або років)]
	Вивести усі товари гарантія на які більша за N (N вводити з клавіатури)
7.	Довідник - **«Beauty bloggers»** Поля - **[**Нікнейм] [Назва каналу] [Посилання] [Вік] [Кількість підписчиків]
	Вивести N найпопулярніших блогерів за зростанням віку (N вводити з клавіатури)
8.	Довідник - **«Кінолог»** Поля - **[**Порода] [Середня вага] [Середній мак. Вік] [Регіон розповсюдження] [Середня вартість]
	Вивести усі породи у яких середня вага менше за N, а середній вік більший за M (M-N ввести з клавіатури)
9.	Довідник - **«Бабусині заготовки»** Поля - **[**Назва] [Об'єм] [Рік] [Вид] [Термін придатності]
	Вивести усі заготовки зроблені до N року (N ввести з клавіатури)
10.	Довідник - **«Художня галерея»** Поля - **[**Назва полотна] [Автор] [Рік] [Розмір] [Ціна]
	Вивести усі полотна певного автора (автора вводити з клавіатури)
11.	Довідник - **«Ремон взутя»** Поля - **[**ID взуття] [дата прийому] [Вид роботи] [Тел. власника] [Ціна]
	Вивести усі прийняті пари взуття за номером телефону власника (номер телефону вводити з клавіатури)
12.	Довідник - **«Розклад занять»** Поля - **[**Група] [День тижня] [№ пари] [Аудиторія] [Предмет]
	Вивести усі пари певної групи (Групу вводити з клавіатури)
13.	Довідник - **«Бібліотека»** Поля - **[**Назва] [Автор] [Видавництво] [Тираж] [Рік]
	Вивести усі товари певного видавництва (видавництво вводити з клавіатури)

№	Завдання
14.	Довідник - **«Орнітолог»** Поля - **[**Назва виду] [Сімейство] [Кількість особин] [Регіон розповсюдження] [Середня вартість]
	Вивести усіх птахів з вартістю більше N (N ввести з клавіатури)
15.	Довідник - **«Вокзал»** Поля - **[**Пункт відбуття] [Пункт Прибуття] [Маршрут] [Місце] [Вартість квитка]
	Розрахувати середню вартість квитка
16.	Довідник - **«Контакти»** Поля - **[**Ім'я] [Телефон] [Вік]
	Розрахувати середній вік

У даній практичній роботі необхідно виконати розбиття функцій з практичної роботи №2 на модулі з наступним об'єднанням отриманих модулів в один загальний пакет. Пакет необхідно підключити в окремому файлі програми і продемонструвати роботу з кожним модулем пакета.

Таблиця 4.2 – Варіанти завдань

№	Завдання
1.	Створити функцію для вирішення наступного завдання: Масив X=(x1,x2,...,xn) містить велику кількість нульових елементів. Визначити положення і розмір найбільш довгої серії таких елементів.
	Створити функцію сортування масиву за зростанням.
	Створити анонімну функцію для обчислення функції: $f = 3x * y^2 - 9z$
2.	Створити функцію для вирішення наступного завдання: Заданий масив X=(x1,x2,...,xn), в якому можуть бути однакові числа. Знайти максимальний і мінімальний елементи серед неповторюваних чисел.
	Створити функцію сортування в порядку убування.
	Створити анонімну функцію для обчислення функції: $f = 5x * y^3 + 7z$
3.	Створити функцію для вирішення наступного завдання: З масиву чисел X=(x1,x2,...,xn) вилучити всі парні за значенням елементи.
	Створити функцію сортування масиву за зростанням.
	Створити анонімну функцію для обчислення функції: $f = 8x * y^2 - 9z$
4.	Створити функцію для вирішення наступного завдання:

№	Завдання
	У масиві X=(x1,x2,...,xn) поміняти місцями перший і другий негативні елементи, третій і четвертий негативні елементи тощо.
	Створити функцію сортування в порядку убування.
	Створити анонімну функцію для обчислення функції: $$f = \frac{3x}{y^4} - 2z$$
5.	Створити функцію для вирішення наступного завдання: Елементи масиву X = (x1,x2,...,xn) – це послідовність цифр цілого числа. Переставити цифри числа у зворотному порядку
	Створити функцію сортування масиву за зростанням.
	Створити анонімну функцію для обчислення функції: $$f = x - y^2 * 4z$$
6.	Створити функцію для вирішення наступного завдання: Відомо, що в целочисельному масиві X=(x1,x2,...,xn) три і тільки три числа, що є рівними між собою. Знайти ці числа
	Створити функцію сортування в порядку убування.
	Створити анонімну функцію для обчислення функції: $$f = 2x + y^2 - 5z$$
7.	Створити функцію для вирішення наступного завдання: За однократний перегляд масиву знайти його максимальний позитивний елемент X_{max}
	Створити функцію сортування масиву за зростанням.
	Створити анонімну функцію для обчислення функції: $$f = 7x * y^3 * 2z$$
8.	Створити функцію для вирішення наступного завдання: Перетворити масив X, розташувавши спочатку його негативні, а потім позитивні елементи, зберігши при цьому в групі негативних та позитивні елементів їх вихідний відносний порядок.
	Створити функцію сортування в порядку убування.
	Створити анонімну функцію для обчислення функції: $$f = 11x - y^{-2} + 6z$$
9.	Створити функцію для вирішення наступного завдання: У масиві X=(x1,x2,...,xn) поміняти місцями перший і другий позитивні елементи, третій і четвертий позитивні елементи тощо.
	Створити функцію сортування масиву за зростанням.
	Створити анонімну функцію для обчислення функції: $$f = 6x + y^3 * 8z$$
10.	Створити функцію для вирішення наступного завдання: Заданий масив цілих чисел X=(x1,x2,...,xn). Сформувати масив Y=(y1,y2,...,ym), помістивши в нього в порядку убування всі позитивні числа, що входять у масив X.
	Створити функцію сортування масиву за зростанням.

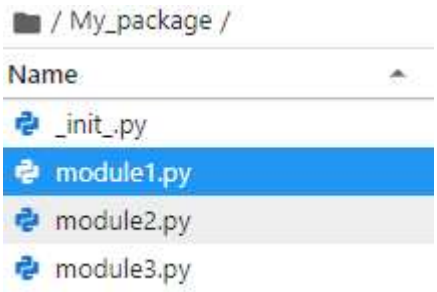
№	Завдання
	Створити анонімну функцію для обчислення функції: $f = \frac{2x}{y^1} + 5z$
11.	Створити функцію для вирішення наступного завдання: Заданий цілочисельний масив X=(x1,x2,...,xn), у якому можуть бути однакові числа. Підрахувати кількість повторюваних чисел у масиві.
	Створити функцію сортування масиву за зростанням.
	Створити анонімну функцію для обчислення функції: $f = 5x + \frac{y^2}{7z}$
12.	Створити функцію для вирішення наступного завдання: Виконати циклічне зрушення масиву X=(x1,x2,...,xn) на 5 елементів вліво.
	Створити функцію сортування в порядку убування.
	Створити анонімну функцію для обчислення функції: $f = 4x * y^2 - \frac{9}{z}$
13.	Створити функцію для вирішення наступного завдання: Виконати циклічне зрушення масиву X=(x1,x2,...,xn) на 3 елементів вправо.
	Створити функцію сортування масиву за зростанням.
	Створити анонімну функцію для обчислення функції: $f = 11x * \frac{y^3}{3} + 8z$
14.	Створити функцію для вирішення наступного завдання: Масив X=(x1,x2,...,xn) містить велику кількість нульових елементів. Визначити положення і розмір найбільш довгої серії ненульових елементів.
	Створити функцію сортування в порядку убування.
	Створити анонімну функцію для обчислення функції: $f = 2xz * \frac{y^2}{2}$
15.	Створити функцію для вирішення наступного завдання: Заданий масив цілих чисел X=(x1,x2,...,xn). Сформувати масив Y=(y1,y2,...,ym), помістивши в нього в порядку убування всі негативні числа, що входять у масив X.
	Створити функцію сортування масиву за зростанням.
	Створити анонімну функцію для обчислення функції: $f = 2(x - y) * \frac{z^2}{4}$

№	Завдання
16.	Створити функцію для вирішення наступного завдання: Заданий масив цілих чисел X=(x1,x2,...,xn). Сформувати масив Y=(y1,y2,...,ym), помістивши в нього в порядку убування всі різні (неповторювані) числа, що входять у масив X.
	Створити функцію сортування масиву за зростанням.
	Створити анонімну функцію для обчислення функції: $$f = 2(x - 4z) * \frac{y^{-2}}{5}$$

4.7 Приклад

Варіант - 16.

Структура пакету



Вміст файлу _init.py_

```
1  __all__ = ["module1","module2","module3"]
```

Головний файл програми

```
[20]: from My_package import *
mas=[1,2,2,4,5,8,4,3,18,11,11] # вхідний масив
x=5
y=6
z=3
print(module1.func1(mas))
print(module2.func2(x,y,z))
print(module3.func3(mas))
```

```
[18, 8, 5, 3, 1]
-0.07777777777777777
[1, 2, 2, 3, 4, 4, 5, 8, 11, 11, 18]
```

Варіант - 16.

Лістинг

[*]:
```
person = {}.fromkeys(['name', 'age', 'phone'])
person_list = list()
c=-1
def print_f(person_list):
    f = open('Person_List', 'w+')
    for i in range(len(person_list)):
        f.write("%10s" %person_list[i]['name'])
        f.write("%2d" %person_list[i]['age'])
        f.write("%15s" %person_list[i]['phone'])
        f.write("\n")
    f.close();
def read_f(person_list):
    f = open('Person_List', 'r+')
    i=0
    while True:
        s=f.read(10)
        if not s:
            break
        person_list+=[person.copy()]
        person_list[i]['name'] = s
        s=f.read(2)
        person_list[i]['age'] = int(s)
        s=f.read(15)
        person_list[i]['phone'] = s
        i+=1
        s=f.read(1)
    f.close()
def age(person_list):
    i=0
    s=0
    for i in range(len(person_list)):
        s+=person_list[i]['age']
    return s/len(person_list)
def add(person_list):
    person_list+=[person.copy()]
    person_list[-1]['name']=input("Введіть ім'я:  ")
    person_list[-1]['age']=int(input("Введіть вік:  "))
    person_list[-1]['phone']=input("Введіть телефон:  ")
    print_f(person_list)
def delete(person_list):
    print_p(person_list)
    buf=int(input("Введіть номер видаляємого запису:  "))
    del person_list[buf-1]
    print_f(person_list)
def print_p(person_list):
    for i in range(len(person_list)):
        print(i+1,"%10s" %person_list[i]['name'],"%2d" %person_list[i]['age'],"%15s" %person_list[i]['phone'])

read_f(person_list)
while c!=0:
    print("Меню")
    print("1. Додати запис")
    print("2. Переглянути усі записи")
    print("3. Підрахувати середній вік")
    print("4. Видалити запис")
    print("0. завершити роботу")
    c=int(input());
    if c==1:
        add(person_list)
    if c==2:
        i=0
        print_p(person_list)
    if c==3:
        print("Середній вік =",age(person_list))
    if c==4:
        delete(person_list)
```

Результат роботи

```
Меню
1. Додати запис
2. Переглянути усі записи
3. Підрахувати середній вік
4. Видалити запис
0. завершити роботу
 2
1      Petro 54      0959559959
Меню
1. Додати запис
2. Переглянути усі записи
3. Підрахувати середній вік
4. Видалити запис
0. завершити роботу
 1
Введіть ім'я:   Kurulo
Введіть вік:   35
Введіть телефон:   0958685478
Меню
1. Додати запис
2. Переглянути усі записи
3. Підрахувати середній вік
4. Видалити запис
0. завершити роботу
 2
1      Petro 54      0959559959
2     Kurulo 35      0958685478
Меню
1. Додати запис
2. Переглянути усі записи
3. Підрахувати середній вік
4. Видалити запис
0. завершити роботу
 4
1      Petro 54      0959559959
2     Kurulo 35      0958685478
Введіть номер видаляємого запису:   1
Меню
1. Додати запис
2. Переглянути усі записи
3. Підрахувати середній вік
4. Видалити запис
0. завершити роботу
 2
1     Kurulo 35      0958685478
```

4.8 Контрольні запитання

1. Опишіть існуючі режими доступу до файлу.
2. Вкажіть базові методи для роботи з файлами, та особливості їх застосування.
3. Наведіть основні функції для порядкової роботи з файлами.
4. Опишіть що таке бінарні файли, та основні особливості їх застосування.
5. Наведіть функції відкриття/закриття файлів.
6. Опишіть можливі ризики одночасного читання та запису до файлу.
7. Функції у Python, їх види та призначення.
8. Які аргументи може приймати функція, їх типи та кількість.
9. Анонімні функції, призначення та особливості використання.
10. Повертаємі значення функції, їх типи та кількість.
11. Функції зі змінною кількістю аргументів, призначення та особливості використання.
12. Стандартні модулі у Python, їх види та призначення.
13. Які існують способи звернення до модулів, наведіть приклади.
14. Опишіть процедуру створення власного модуля.
15. Опишіть механізм об’єднання модулів в пакети, сенс, принципи, призначення.
16. Файл _init.py_, для чого створюється та як використовується? Наведіть приклади.

4.9 Оформлення звіту

1) Титульний лист;
2) Мета роботи;
3) Номер варіанту та умови завдання;
4) Програмний код з коментарями;
5) Скріни роботи програми;
6) Висновки