МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

КАФЕДРА ПРИКЛАДНОЇ МАТЕМАТИКИ І ІНФОРМАТИКИ

Практична робота №4

з дисципліни «Методи та системи штучного інтелекту»

на тему «РОБОТА З ФАЙЛАМИ ТА ОРГАНІЗАЦІЯ ПРОЕКТІВ»

Варіант 4

Виконав:

студент гр. КІБ-21

Андрій Севастьянов

Перевірив:

асистент кафедри ПМІ

Андрій НІКІТЕНКО

Луцьк – 2024

**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.

**Завдання:**

Перша частина

Для довідника реалізованого згідно власного варіанту у практичній роботі No3, створити програму з підтримкою функцій роботи з файлами, а саме:

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

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)

Друга частина

У даній практичній роботі необхідно виконати розбиття функцій з практичної роботи No2 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.  
  
Створити функцію для вирішення наступного 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:

Програмний код першої частини:  
*# Словник для зберігання даних про транспортні компанії  
'''  
companies = {  
 "Нова Пошта": {  
 "vehicles": 1000,  
 "cost\_per\_km": 1.5,  
 "address": "вул. Київська, 12",  
 "max\_weight": 30,  
 },  
 "Укрпошта": {  
 "vehicles": 500,  
 "cost\_per\_km": 1.2,  
 "address": "вул. Шевченка, 24",  
 "max\_weight": 20,  
 },  
 "Делівері": {  
 "vehicles": 300,  
 "cost\_per\_km": 1.8,  
 "address": "вул. Франка, 36",  
 "max\_weight": 50,  
 },  
}  
'''*companies = {}  
def save(filename):  
 f = open(filename, "w")  
  
 for company\_name, company\_info in companies.items():  
 f.write(company\_name + '\n')  
 f.write(str(company\_info["vehicles"]) + '\n')  
 f.write(str(company\_info["cost\_per\_km"]) + '\n')  
 f.write(company\_info["address"] + '\n')  
 f.write(str(company\_info["max\_weight"]) + '\n')  
 f.close()  
  
def load(filename):  
 with open(filename, "r") as f:  
 while True:  
 company\_name = f.readline().strip() *# читаємо рядок з видаленням всього непотрібного в тому числі і переносів* if not company\_name: *# Якщо прочитали порожній рядок, то вже кінець файлу* break  
 vehicles = int(f.readline().strip())  
 cost\_per\_km = float(f.readline().strip())  
 address = f.readline().strip()  
 max\_weight = float(f.readline().strip())  
  
 companies[company\_name] = {  
 "vehicles": vehicles,  
 "cost\_per\_km": cost\_per\_km,  
 "address": address,  
 "max\_weight": max\_weight  
 }  
 return companies  
  
load("mf.txt")  
  
  
*# Валідація*def get\_int\_input(prompt):  
 while True:  
 try:  
 value = int(input(prompt))  
 return value  
 except ValueError:  
 print("Невірний формат. Введіть число.")  
  
def get\_float\_input(prompt):  
 while True:  
 try:  
 value = float(input(prompt))  
 return value  
 except ValueError:  
 print("Невірний формат. Введіть число з плаваючою комою.")  
  
*# Функція для додавання нової компанії*def add\_company():  
 name = input("Введіть назву компанії: ")  
 vehicles = get\_int\_input("Введіть кількість автомобілів: ")  
 cost\_per\_km = get\_float\_input("Введіть вартість 1 км перевезення: ")  
 address = input("Введіть адресу: ")  
 max\_weight = get\_int\_input("Введіть максимальну допустиму вагу: ")  
  
 companies[name] = {  
 "vehicles": vehicles,  
 "cost\_per\_km": cost\_per\_km,  
 "address": address,  
 "max\_weight": max\_weight,  
 }  
  
*# Функція для видалення компанії*def remove\_company():  
 name = input("Введіть назву компанії для видалення: ")  
  
 if name in companies:  
 del companies[name]  
 print(f"Компанія {name} успішно видалена.")  
 else:  
 print(f"Компанія {name} не знайдена.")  
  
*# Функція для редагування інформації про компанію*def edit\_company():  
 while True:  
 name = input("Введіть назву компанії для редагування: ")  
 if name in companies:  
 company = companies[name]  
  
 vehicles = get\_int\_input("Введіть нову кількість автомобілів (або Enter для пропуску): ")  
 if vehicles:  
 company["vehicles"] = vehicles  
 cost\_per\_km = get\_float\_input("Введіть нову вартість 1 км перевезення (або Enter для пропуску): ")  
 if cost\_per\_km:  
 company["cost\_per\_km"] = cost\_per\_km  
 address = input("Введіть нову адресу (або Enter для пропуску): ")  
 if address:  
 company["address"] = address  
 max\_weight = get\_int\_input("Введіть нову максимальну допустиму вагу (або Enter для пропуску): ")  
 if max\_weight:  
 company["max\_weight"] = max\_weight  
 print(f"Інформація про компанію {name} успішно оновлена.")  
 break  
 else:  
 print(f"Компанія {name} не знайдена.")  
  
  
*# Функція для демонстрації всіх компаній*def show\_companies():  
 for name, company in companies.items():  
 print(f"--- {name} ---")  
 print(f"Кількість автомобілів: {company['vehicles']}")  
 print(f"Вартість 1 км перевезення: {company['cost\_per\_km']}")  
 print(f"Адреса: {company['address']}")  
 print(f"Максимальна допустима вага: {company['max\_weight']}")  
  
*# Функція для пошуку компаній з максимальною допустимою вагою більше N*def find\_companies\_by\_max\_weight():  
 while True:  
 try:  
 n = get\_int\_input("Введіть мінімальну допустиму вагу: ")  
 break  
 except ValueError:  
 print("Невірний формат. Введіть ціле число більше 0.")  
  
 filtered\_companies = []  
 for name, company in companies.items():  
 if company["max\_weight"] >= n:  
 filtered\_companies.append((name, company))  
  
 if filtered\_companies:  
 print(f"Знайдено {len(filtered\_companies)} компаній з максимальною допустимою вагою більше {n}:")  
 for name, company in filtered\_companies:  
 print(f"- {name}")  
 else:  
 print(f"Компаній з максимальною допустимою вагою більше {n} не знайдено.")  
  
  
*# Меню для роботи з довідником*while True:  
 print("-" \* 20)  
 print("Меню довідника \"Транспортні компанії\"")  
 print("-" \* 20)  
 print("1. Додати нову компанію")  
 print("2. Видалити компанію")  
 print("3. Редагувати інформацію про компанію")  
 print("4. Переглянути всі компанії")  
 print("5. Знайти компанії за максимальною допустимою вагою")  
 print("6. Зберегти зміни")  
 print("0. Вихід")  
 print("-" \* 20)  
  
 choice = input("Введіть номер пункту меню: ")  
  
 if choice == "1":  
 add\_company()  
 elif choice == "2":  
 remove\_company()  
 elif choice == "3":  
 edit\_company()  
 elif choice == "4":  
 show\_companies()  
 elif choice == "5":  
 find\_companies\_by\_max\_weight()  
 elif choice =="6":  
 save("mf.txt")  
 elif choice == "0":  
 print("Дякуємо за використання довідника!")  
 break  
 else:  
 print("Невірний номер пункту меню.")

Програмний код другої частини:  
main.py  
from lab2 import \*  
  
numb = [1, -2, 3, -4, 5, -6, 7, -8, 9, -10, 11, -12, 13, -14]  
print("Масив до обробки:\t\t" + str(numb))  
result = module1.swap\_negatives(numb)  
print("Масив після обробки:\t" + str(result))  
  
arr = [64, 34, 25, 12, 22, 11, 90]  
print("Масив до сортування:", arr)  
module2.descending\_insertion\_sort(arr)  
print("Масив після сортування:", arr)  
  
*# Створити анонімну функцію для обчислення функції f = ((3x)\(y^4))-2z*result = module3.lmbd(2, 3, 4)  
print("Результат = " + str(result))

\_\_init\_\_.py  
\_\_all\_\_ = ["module1", "module2", "module3"]

module1.py  
def swap\_negatives(arr):  
 fl = False  
 first\_min\_ind = 0  
 for index in range(len(arr)):  
 if arr[index] < 0 and fl == False:  
 *# finding the first minimum* first\_min\_ind = index  
 fl = True  
 elif arr[index] < 0 and fl == True:  
 *# swap* arr[index], arr[first\_min\_ind] = arr[first\_min\_ind], arr[index]  
 *# normal* fl = False  
 return arr

module2.py

def descending\_insertion\_sort(arr):  
 *# починаємо з другого* for i in range(1, len(arr)):  
 key = arr[i] *# key = наш елемент для якого ми шукаємо місце  
 # находимо правильне місце для вставки елемента* j = i - 1  
 while j >= 0 and arr[j] < key:  
 arr[j + 1] = arr[j]  
 j -= 1  
 arr[j + 1] = key  
 return arr

module3.py

lmbd = lambda x, y, z: ((3\*x) / (y\*\*4)) - 2\*z

Скріншот роботи програмного коду першої частини:

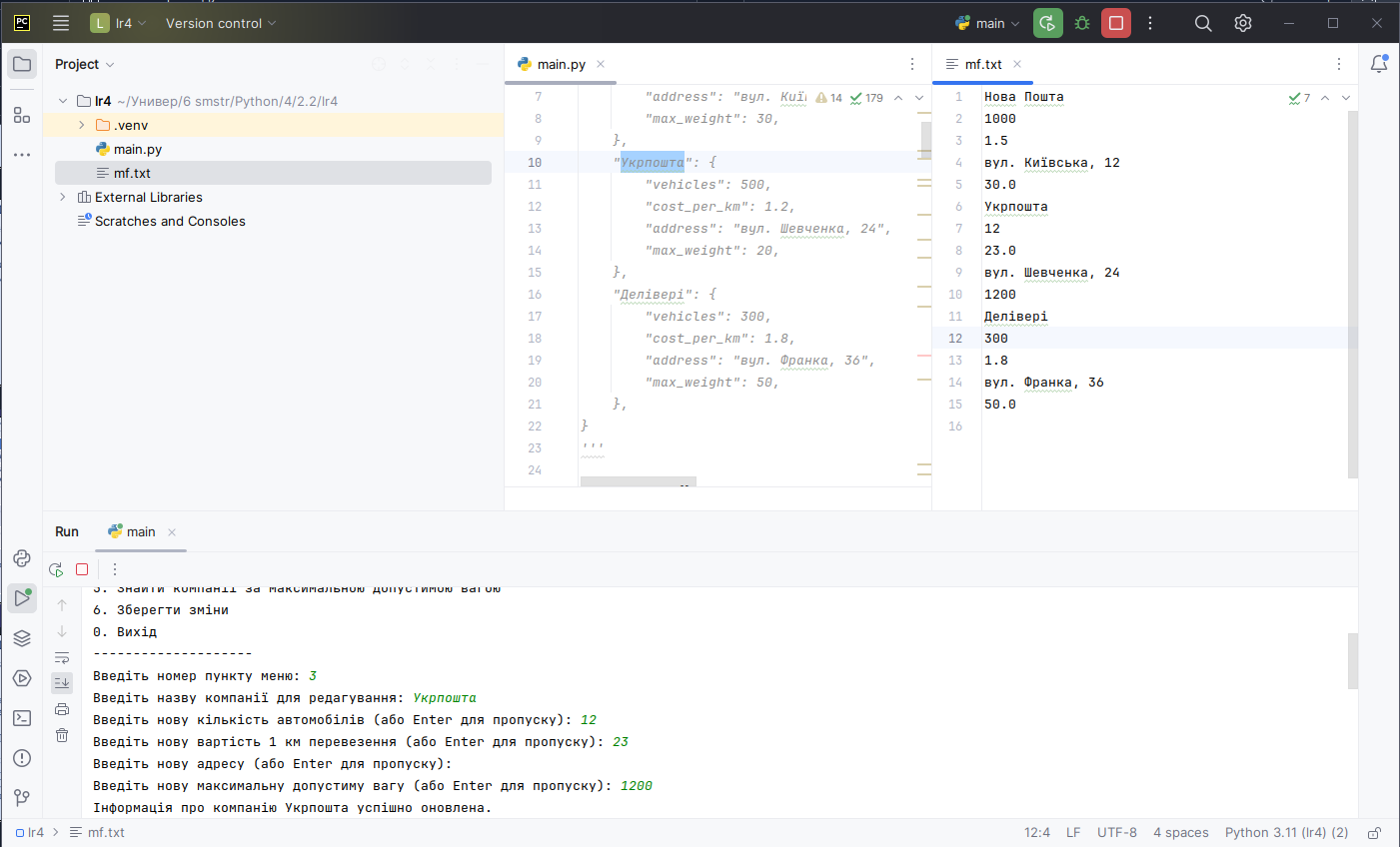


Рисунок 1 – робота першої частини

Скріншот роботи програмного коду другої частини:

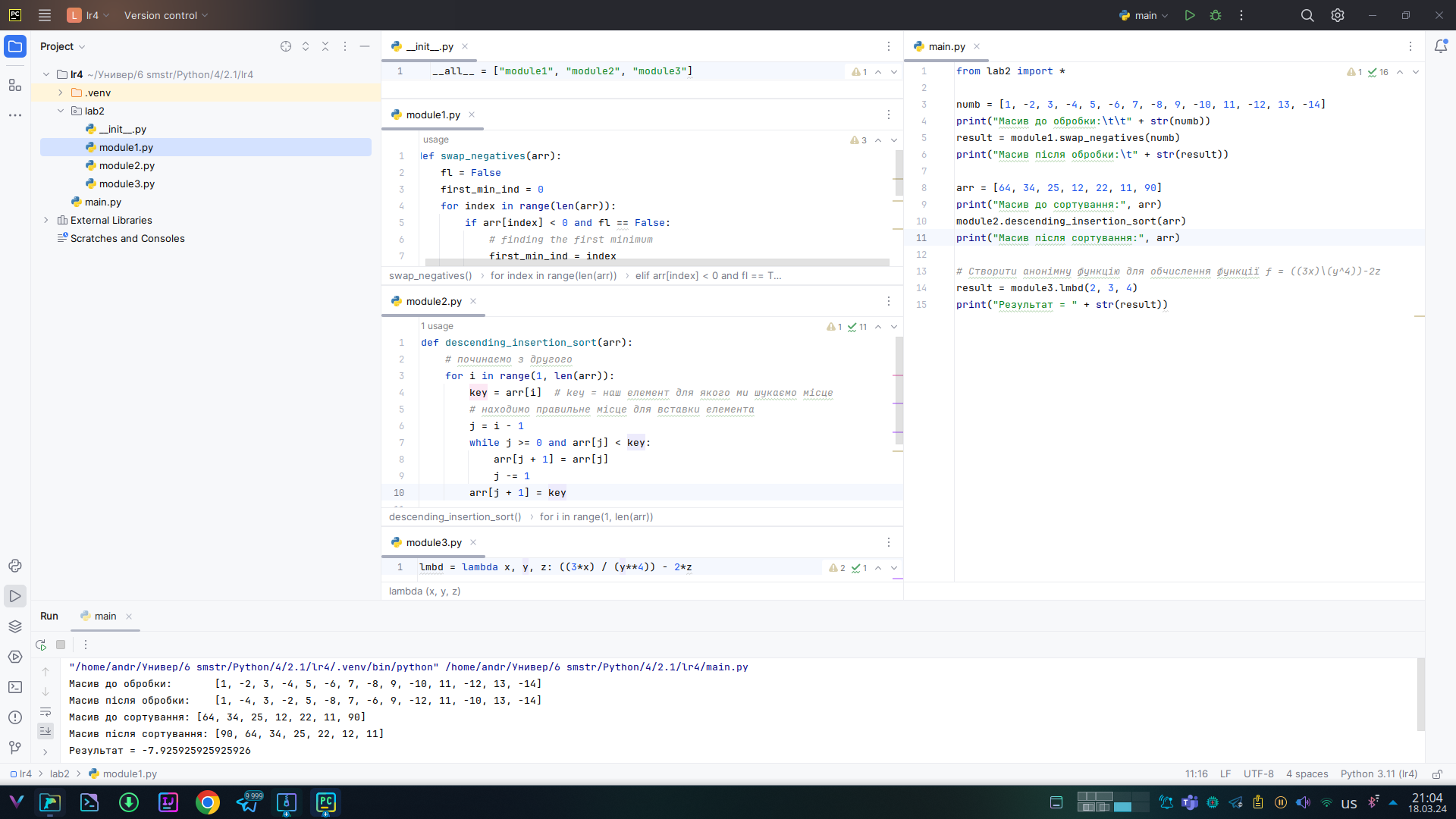


Рисунок 2 – робота другої частини

Висновок:  
 Ми вивчили та засвоїли базові принципи роботи з файлами, модулями та пакетами. Ознайомилися 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.