МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний аерокосмічний університет ім. М. Є. Жуковського
«Харківський авіаційний інститут»

Кафедра систем управління літальними апаратами

Лабораторна робота № 3

з дисципліни «Об'єктно-орієнтоване програмування СУ»

Тема: «Структурування програм з використанням функцій»

ХАІ.301 .174. 312ст.3 ЛР

Виконав студент гр. _____312ст_____

_____Твердохліб Максим Анатолійович
(підпис, дата) (П.І.Б.)

Перевірив
_____ к.т.н., доц. О. В.
Гавриленко
_____ ас. В. О.
Білозерський
(підпис, дата) (П.І.Б.)

2024

МЕТА РОБОТИ

Вивчити теоретичний матеріал із синтаксису визначення і виклику функцій та особливостей послідовностей у Python, а також документацію бібліотеки numpy; отримати навички реалізації бібліотеки функцій з параметрами, що структурують вирішення завдань «згори – до низу»

ПОСТАНОВКА ЗАДАЧІ

Завдання 1. (Proc) 17 Описати функцію відповідно до варіанту. Для виклику функції (друга частина задачі) описати іншу функцію, що на вході має список вхідних даних і повертає список вихідних даних. Введення даних, виклик функції та виведення результатів реалізувати в третій функції без параметрів.

Proc17	Описати функцію RootCount (A, B, C) цілого типу, яка визначає кількість коренів квадратного рівняння A · x2 + B · x + C = 0 (A, B, C - речові параметри, A ≠ 0). З її допомогою знайти кількість коренів для кожного з трьох квадратних рівнянь з даними коефіцієнтами. Кількість коренів визначати за значенням дискримінанту: D = B2 - 4 · A · C.

Завдання 2. Matrix 9 Розробити дві вкладені функції для вирішення задачі обробки двовимірних масивів відповідно до варіанту: зовнішня – без параметрів, внутрішня має на вході ім’я файлу з даними, на виході – підраховані параметри матриці (перша частина задачі) та перетворену матрицю (друга частина задачі). Для обробки масивів використати функції бібліотеки numpy.

Matrix 9. У текстовому файлі задана матриця розміру M × N. Знайти номер її рядки з найбільшою сумою елементів і вивести даний номер, а також значення максимальної суми. Відсортувати задану матрицю по рядках по зростанню.

Виконання роботи

Завдання 1. Вирішення задачі Proc 17

Вхідні дані:

Ім’я	Опис	Тип	Обмеження
A	Коефіцієнт при x^2, перший параметр квадратного рівняння	float	A≠0
B	Коефіцієнт при x, другий параметр квадратного рівняння	float	немає
C	Вільний член, третій параметр квадратного рівняння	float	немає
equations	Список рівнянь у вигляді кортежів, кожен кортеж містить (A, B, C)	list[tuple[float]]	Довжина списку: 3 рівняння

Вихідні дані:

Ім’я	Опис	Тип
count	Кількість коренів для одного рівняння	int
results	Список кількостей коренів для кожного рівняння	list[int]

Алгоритм вирішення показано на рис. 1

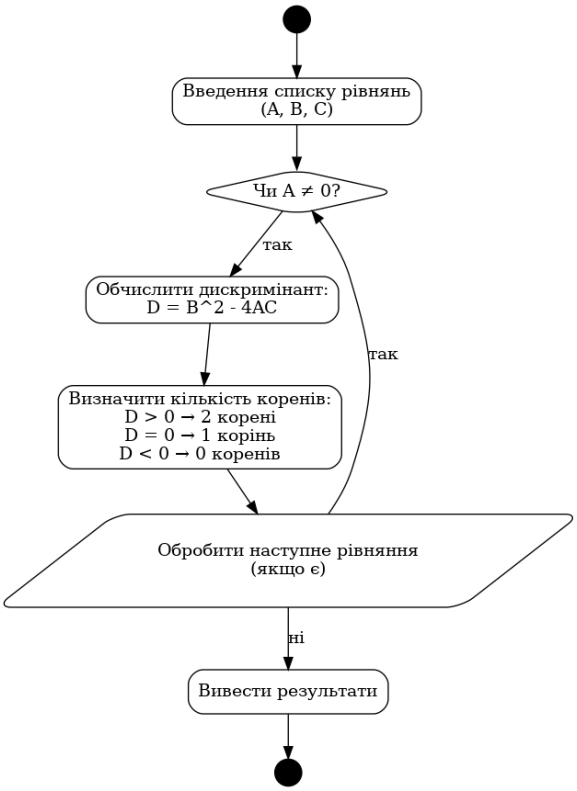

Рисунок 1 – Алгоритм вирішення завдання 1

Лістинг коду вирішення задачі наведено в дод. А. стор.10. Екран роботи програми показаний на рис. Б. стор. 11

Завдання 2. Вирішення задачі Matrix 9
Вхідні дані:

Ім’я	Опис	Тип	Обмеження
filename	Ім'я текстового файлу з матрицею	str	Файл має містити M× N матрицю
matrix	Матриця розміру M×N	list[list[int]]	Розмір матриці: M>0, N>0

Вихідні дані:

Ім’я	Опис	Тип

max_row_num	Номер рядка з найбільшою сумою	int
max_sum	Максимальна сума елементів рядка	int
sorted_matrix	Матриця, відсортована по рядках	list[list[int]]

Алгоритм вирішення показано на рис. 1.2

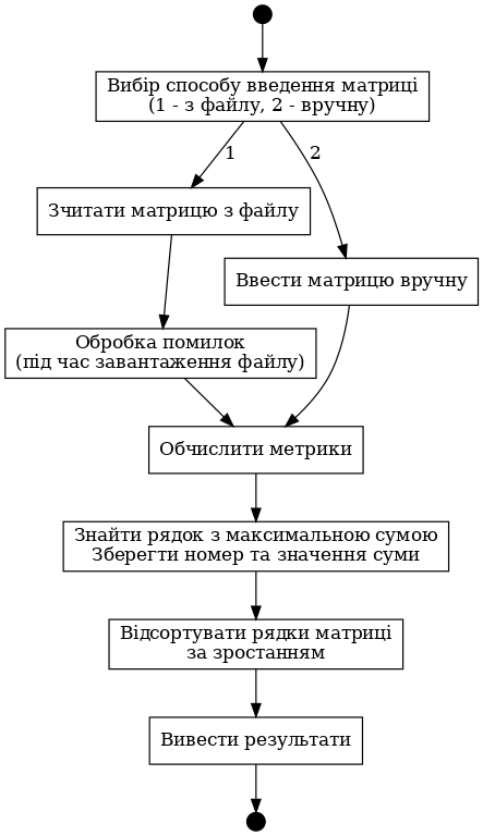

Рисунок 1.2 – алгоритм вирішення задачі 2

Лістинг коду задачі наведено в дод. А стор.10 . Екран роботи програми показаний на рис. Б.1 стор.11

Висновок:

Під час виконання лабораторної роботи ми ознайомилися з принципами структурування програм за допомогою функцій та опанували роботу з двовимірними масивами у Python за допомогою бібліотеки NumPy. Ми закріпили навички створення функцій для виконання конкретних завдань,

таких як обчислення суми та добутку елементів рядків матриці, а також визначення їхньої різниці з випадковою матрицею.

Додаток А

Лістинг коду програми до задачі Proc 17

```
def RootCount(A, B, C):

    """

    Функція для визначення кількості коренів квадратного рівняння A * x^2 + B
* x + C = 0

    Параметри:

    A, B, C - коефіцієнти рівняння (A ≠ 0)

    Повертає:

    Кількість коренів рівняння (0, 1, або 2)

    """

    if A == 0:

        raise ValueError("Коефіцієнт A не може дорівнювати 0")

    D = B**2 - 4 * A * C  # Дискримінант

    if D > 0:

        return 2  # Два корені

    elif D == 0:

        return 1  # Один корінь

    else:

        return 0  # Немає коренів

def ProcessEquations(equations):

    """

    Функція для обробки списку квадратних рівнянь.

    Параметри:

    equations - список кортежів, де кожний кортеж містить коефіцієнти (A, B,
C)
```

```
    Повертає:

    Список з кількістю коренів для кожного рівняння

    """

    results = []

    for A, B, C in equations:

        count = RootCount(A, B, C)

        results.append(count)

    return results

def MainFunction():

    """

    Головна функція для введення даних, виклику функцій та виведення
результатів

    """

    # Введення даних

    equations = [

        (1, -3, 2),  # D > 0

        (1, -2, 1),  # D == 0

        (1, 0, -1),  # D > 0

        (1, 2, 5)    # D < 0

    ]  # Приклад даних: список рівнянь у форматі (A, B, C)

    # Виклик функції для обробки списку рівнянь

    results = ProcessEquations(equations)

    # Виведення результатів

    print("\nРезультати:")

    for i, count in enumerate(results):

        print(f"Рівняння {i+1} ({equations[i][0]}x^2 + {equations[i][1]}x +
{equations[i][2]} = 0): Кількість коренів = {count}")

# Виклик головної функції

if __name__ == "__main__":
```

```
    MainFunction()
```

Лістинг коду до задачі 2 Matrix 9

```
import numpy as np


def calculate_matrix(matrix):
    # Обчислюємо суму елементів по рядках
    row_sums = np.sum(matrix, axis=1)
    # Обчислюємо добуток елементів по рядках
    row_products = np.prod(matrix, axis=1)


    print("Суми елементів рядків:", row_sums)
    print("Добутки елементів рядків:", row_products)


    # Створюємо випадкову матрицю того ж розміру
    random_matrix = np.random.randint(1, 10, size=matrix.shape)


    # Обчислюємо різницю між початковою і випадковою матрицями
    result_matrix = matrix - random_matrix
    print("Різниця початкової і випадкової матриць:\n", result_matrix)


    return row_sums, row_products, result_matrix


def process_matrix():
    def read_and_process_matrix(file_name=None):
        if file_name:
            # Завантажуємо матрицю з файлу
            try:
                matrix = np.loadtxt(file_name, dtype=int)
```

```
            except Exception as e:
                print(f"Помилка при завантаженні файлу: {e}")
                return None, None, None
        else:
            try:
                rows = int(input("Введіть кількість рядків матриці: "))
                cols = int(input("Введіть кількість стовпців матриці: "))
                matrix = []
                print("Введіть елементи матриці:")
                for i in range(rows):
                    row = list(map(int, input(f"Рядок {i + 1}: ").split()))
                    if len(row) != cols:
                        print("Кількість елементів у рядку не відповідає
вказаній кількості стовпців.")
                        return None, None, None
                    matrix.append(row)
                matrix = np.array(matrix)
            except ValueError:
                print("Помилка: введіть коректні цілі числа.")
                return None, None, None

        # Виклик функції обчислення параметрів
        return calculate_matrix(matrix)

    choice = input("Оберіть метод введення матриці (1 - з файлу, 2 - вручну):
")
    if choice == "1":
        filename = input("Введіть шлях до файлу з матрицею: ")
        row_sums, row_products, result_matrix =
read_and_process_matrix(filename)
```

```
    elif choice == "2":
        row_sums, row_products, result_matrix = read_and_process_matrix()
    else:
        print("Некоректний вибір. Спробуйте ще раз.")
        return

    if row_sums is not None:
        # Виведення результатів
        print(f"Суми елементів рядків: {row_sums}")
        print(f"Добутки елементів рядків: {row_products}")
        print(f"Різниця початкової і випадкової матриць:\n{result_matrix}")
# Виконуємо основну функцію
if __name__ == "__main__":
    process_matrix()
```

Додаток Б

Скріншоти вікна виконання роботи програми

```
Результати:
Рівняння 1 (1x^2 + -3x + 2 = 0): Кількість коренів = 2
Рівняння 2 (1x^2 + -2x + 1 = 0): Кількість коренів = 1
Рівняння 3 (1x^2 + 0x + -1 = 0): Кількість коренів = 2
Рівняння 4 (1x^2 + 2x + 5 = 0): Кількість коренів = 0
```

Рисунок Б.1 – Екран виконання програми для завдання 1 Proc 17

```
Оберіть метод введення матриці (1 - з файлу, 2 - вручну):
2
Введіть кількість рядків матриці:
3
Введіть кількість стовпців матриці:
3
Введіть елементи матриці:
Рядок 1:
4 6 7
Рядок 2:
9 3 5
Рядок 3:
```

```
Суми елементів рядків: [17 17 13]
Добутки елементів рядків: [168 135  60]
Різниця початкової і випадкової матриць:
 [[-4  2  1]
 [ 7 -5 -3]
 [ 5  0 -6]]
Суми елементів рядків: [17 17 13]
Добутки елементів рядків: [168 135  60]
Різниця початкової і випадкової матриць:
[[-4  2  1]
 [ 7 -5 -3]
 [ 5  0 -6]]
```

Рисунок Б.2 та Б.3 – Екран виконанн програми для завдання 2 Matrix 9