

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'єктивно-орієнтоване проектування
програм для мобільних систем»

Тема: «Структурування програм з використанням
функцій»

ХАІ.301. 174. 322. 3 ЛР

Виконав студент гр. 322

_____ Діхтяренко М.Г.

(підпис, дата)

(П.І.Б.)

Перевірів

_____ к.т.н., доц. О. В. Гавриленко

(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. 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. Завдання наведено в табл.1.

Завдання 2. 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`. Завдання представлено в табл.2.

ВИКОНАННЯ РОБОТИ

Завдання 1. Вирішення задачі 20:

Описати функцію `TriangleP (a, h)`, що знаходить периметр рівнобедреного трикутника по його основі a і висоті h , проведеної до основи (a і h - речові). За допомогою цієї функції знайти периметри трьох трикутників, для яких дані підстави і висоти. Для знаходження збоку b трикутника використовувати теорему Піфагора: $b^2 = (a / 2)^2 + h^2$.

Алгоритм вирішення завдання:

1. Початок.

2. Введення даних:

- Встановити кількість трикутників (у прикладі це 3).
- Для кожного трикутника:
 - Вивести номер трикутника.
 - Ввести основу трикутника (a).
 - Ввести висоту трикутника (h).

3. Обчислення бічної сторони (b):

- Використати формулу теореми Піфагора

4. Обчислення периметра трикутника:

- Периметр трикутника обчислюється за формулою:
 $P = a + 2b$

5. Виведення результату:

- Вивести значення периметра трикутника (P) з округленням до 2-х десяткових знаків.

6. Перехід до наступного трикутника:

- Повторити обчислення для наступного трикутника, поки всі трикутники не будуть оброблені.

7. Завершення алгоритму:

- Завершити виконання програми.

Лістинг коду вирішення задачі наведено в дод. А (стор. 7). Екран роботи програми показаний в дод.Б на рис. 1 (стор. 9).

Завдання 2. Вирішення задачі 13:

У текстовому файлі задана матриця розміру $M \times N$. У кожній її рядку визначення кількості елементів, менших середнього арифметичного всіх елементів цього рядка. Знайти обернену матрицю як добуток транспонованою на визначник.

Алгоритм вирішення завдання:

1. Початок.
2. Завантаження даних:
 - Ввести шлях до файлу з матрицею.
 - Зчитати матрицю з файлу (елементи матриці розділені пробілами).
3. Виведення матриці:
 - Вивести зчитану матрицю на екран.
4. Обробка рядків матриці:
 - Для кожного рядка матриці:
 - Обчислити середнє арифметичне значення елементів у рядку
 - Порахувати кількість елементів у рядку
 - Вивести для кожного рядка:
 - Середнє значення
 - Кількість елементів менших
5. Перевірка на квадратність матриці:
 - Якщо кількість рядків не дорівнює кількості стовпців:

- Вивести повідомлення, що обернену матрицю знайти неможливо (матриця не квадратна).
 - Перейти до завершення програми.
6. Перевірка визначника матриці:
- Обчислити визначник матриці
 - Якщо $\text{determinant} = 0$:
 - Вивести повідомлення, що обернену матрицю знайти неможливо (визначник дорівнює 0).
 - Перейти до завершення програми.
7. Обчислення оберненої матриці:
- Знайти транспоновану матрицю.
 - Помножити транспоновану матрицю на визначник, щоб отримати обернену матрицю.
 - Вивести обернену матрицю.
8. **Кінець програми.**

Лістинг коду вирішення задачі наведено в дод. А (стор. 7). Екран роботи програми показаний в дод.Б на рис. 1 (стор. 9).

ВИСНОВКИ

В результаті виконання Лабораторної роботи було вивчено теоретичний матеріал із синтаксису визначення і виклику функцій та особливостей послідовностей у Python, а також документацію бібліотеки `numpy`; отримано навички реалізації бібліотеки функцій з параметрами, що структурують вирішення завдань «згори – до низу».

ДОДАТОК А

Лістинг коду програми до задач 1 та 2

```

import math

def TriangleP(a, h):
    """
        Обчислює периметр рівнобедреного трикутника за основою a і
        висотою h.
        :param a: Основа трикутника
        :param h: Висота, проведена до основи
        :return: Периметр трикутника
    """
    # Знаходимо бічну сторону за теоремою Піфагора
    b = math.sqrt((a / 2) ** 2 + h ** 2)
    perimeter = a + 2 * b
    return perimeter

def task1():
    """
        Введення основи і висоти трикутника, обчислення периметрів.
    """
    try:
        n = 3 # Кількість трикутників
        for i in range(n):
            print(f"Трикутник {i + 1}:")
            a = float(input("Введіть основу (a): "))
            h = float(input("Введіть висоту (h): "))
            result = TriangleP(a, h)
            print(f"Периметр трикутника: {result:.2f}")
    except ValueError:
        print("Помилка: введіть коректні числові значення.")

import numpy as np

def process_matrix(file_path):
    """
        Обробляє матрицю:
        1. Рахує кількість елементів у кожному рядку, менших за
        середнє арифметичне.
        2. Обчислює обернену матрицю.
        :param file_path: Шлях до текстового файлу з матрицею
        :return: None
    """

```

```

try:
    # Завантаження матриці з файлу
    matrix = np.loadtxt(file_path, delimiter=' ')
    print("Матриця:")
    print(matrix)

    # Кількість елементів у рядках, менших за середнє
арифметичне
    for i, row in enumerate(matrix):
        mean_value = np.mean(row)
        count_less_than_mean = np.sum(row < mean_value)
        print(f"Рядок {i + 1}: середнє = {mean_value:.2f},
менших за середнє = {count_less_than_mean}")

    # Перевірка на квадратність матриці
    if matrix.shape[0] != matrix.shape[1]:
        print("\nОбернену матрицю знайти неможливо, оскільки
матриця не квадратна.")
        return

    determinant = np.linalg.det(matrix)
    if determinant == 0:
        print("\nОбернену матрицю знайти неможливо, оскільки
визначник дорівнює 0.")
        return

    # Обчислення оберненої матриці
    transposed_matrix = matrix.T
    inverse_matrix = transposed_matrix * determinant
    print("\nОбернена матриця:")
    print(inverse_matrix)

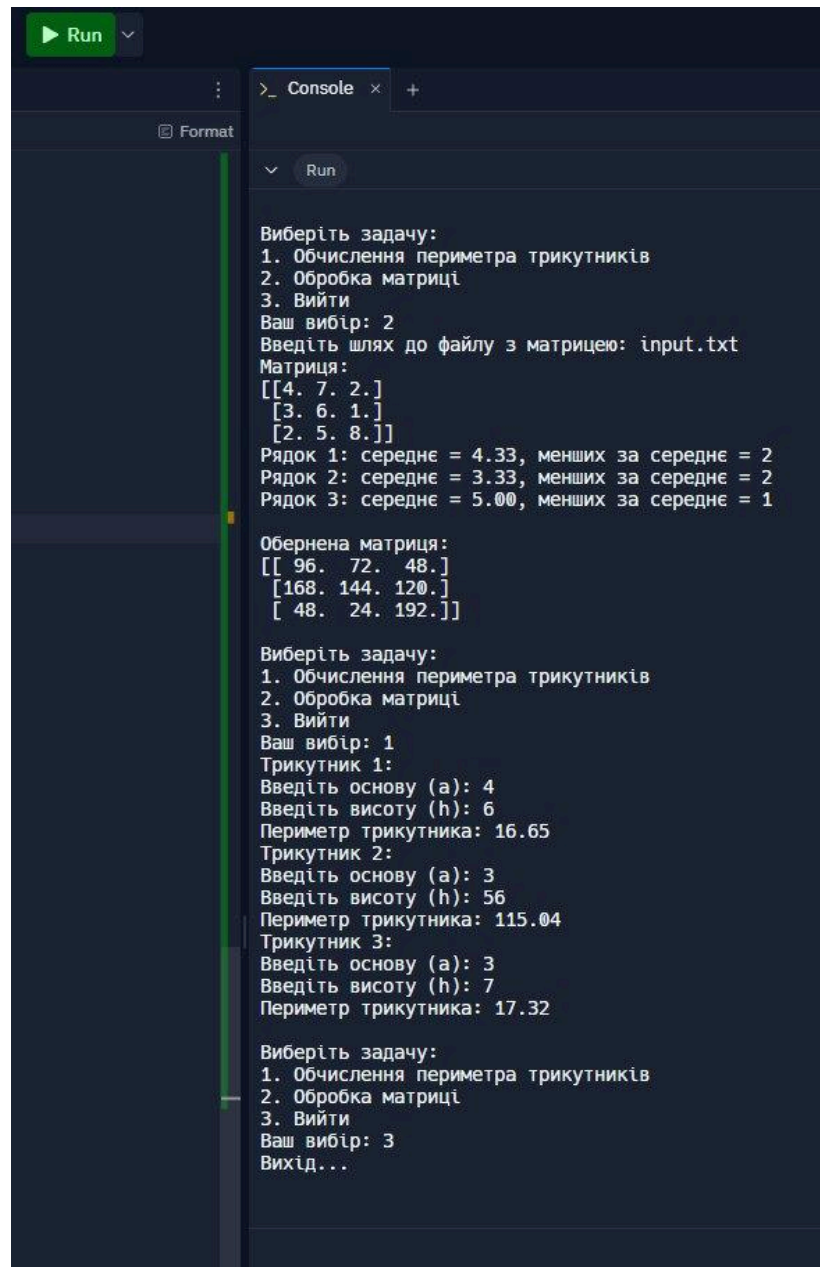
except Exception as e:
    print(f"Помилка: {e}")

def task2():
    """
    Виклик обробки матриці з файлу.
    """
    file_path = input("Введіть шлях до файлу з матрицею: ")
    process_matrix(file_path)

```


ДОДАТОК Б

Скрін-шот вікна виконання програми



```

Run
> Console x +
Format
Run
Виберіть задачу:
1. Обчислення периметра трикутників
2. Обробка матриці
3. Вийти
Ваш вибір: 2
Введіть шлях до файлу з матрицею: input.txt
Матриця:
[[4. 7. 2.]
 [3. 6. 1.]
 [2. 5. 8.]]
Рядок 1: середнє = 4.33, менших за середнє = 2
Рядок 2: середнє = 3.33, менших за середнє = 2
Рядок 3: середнє = 5.00, менших за середнє = 1

Обернена матриця:
[[ 96. 72. 48.]
 [168. 144. 120.]
 [ 48. 24. 192.]]

Виберіть задачу:
1. Обчислення периметра трикутників
2. Обробка матриці
3. Вийти
Ваш вибір: 1
Трикутник 1:
Введіть основу (a): 4
Введіть висоту (h): 6
Периметр трикутника: 16.65
Трикутник 2:
Введіть основу (a): 3
Введіть висоту (h): 56
Периметр трикутника: 115.04
Трикутник 3:
Введіть основу (a): 3
Введіть висоту (h): 7
Периметр трикутника: 17.32

Виберіть задачу:
1. Обчислення периметра трикутників
2. Обробка матриці
3. Вийти
Ваш вибір: 3
Вихід...
```

Рисунок 1 – Екран виконання програми для вирішення всіх завдань