МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний аерокосмічний університет ім. М. Є. Жуковського

«Харківський авіаційний інститут»

Кафедра систем управління літальними апаратами

Лабораторна робота № 3

«Об'єктно-орієнтоване програмування авіаційно-транспортних систем»

Тема: "Структурування програм з використанням функцій"

ХАІ.301.173.3-92ав(і).3ЛР

Виконав студентка гр. \_\_\_\_3-92ав(і)\_\_\_\_\_\_\_

Микитас Анна Олексіївна (П.І.Б.)

Перевірив:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ к.т.н., доц. О. В. Гавриленко

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ас.  В. О. Білозерський

(підпис, дата) (П.І.Б.)

2023

# МЕТА РОБОТИ

# Вивчити теоретичний матеріал із синтаксису визначення і виклик функцій т 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 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.

# ВИКОНАННЯ РОБОТИ

Завдання 1. Proc5

Вхідні дані:

x1, y1, x2, y2: Координати протилежних вершин прямокутника.

Опис: Координати двох протилежних вершин прямокутника на площині.

Тип: int.

Обмеження: Немає обмежень на значення координат.

Вихідні дані:

P, S: Периметр та площа прямокутника.

Опис: Розраховані значення периметру та площі прямокутника.

Тип: float.

Алгоритм вирішення показано нижче

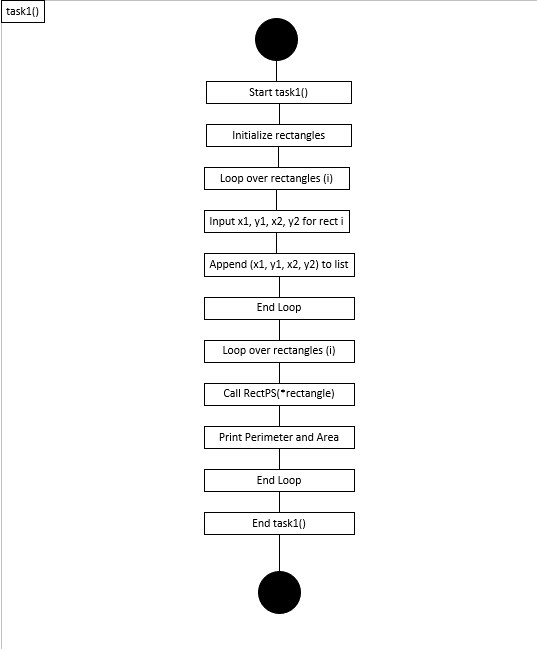


Рисунок 1 – Алгоритм роботи функції task1()

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

Завдання 2. Matrix15

Функція process\_matrix:

Вхідні дані:

file\_name: Назва файлу, з якого буде зчитана матриця.

Опис: Назва текстового файлу з інформацією про матрицю.

Тип: str.

Обмеження: Файл повинен існувати та мати коректний формат.

Вихідні дані :

Немає явно визначених вихідних даних, але функція виводить результати обробки матриці та її елементів.

Функція find\_row:

Вхідні дані:

matrix: Матриця.

Опис: Матриця, для якої шукається рядок з однаковою кількістю додатних і від'ємних елементів.

Тип: numpy.ndarray.

Вихідні дані:

row\_number:Номер рядка з однаковою кількістю додатних і від'ємних елементів.

Опис: Номер рядка з однаковою кількістю додатних і від'ємних елементів.

Тип: int.

Функція divide\_matrix()

Вхідні дані:

matrix: Матриця.

Опис: Матриця, для якої обчислюється поелементна частка на випадкову матрицю того ж розміру.

Тип: numpy.ndarray.

Вихідні дані:

result\_matrix: Результат поелементної частки матриці на випадкову матрицю того ж розміру.

Опис: Матриця, отримана в результаті поелементної частки вхідної матриці на випадкову матрицю того ж розміру.

Тип: numpy.ndarray.

Алгоритм вирішення показано нижче

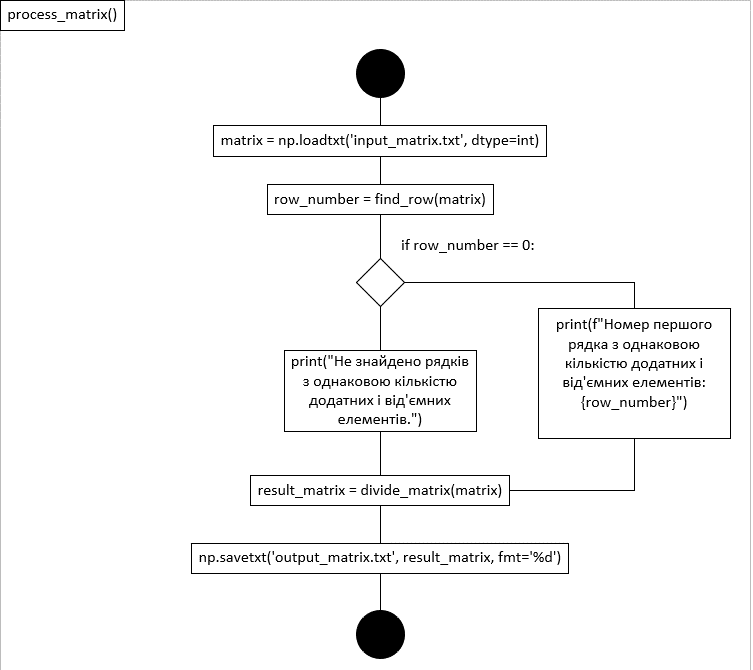


Рисунок 2 – Алгоритм роботи функції process\_matrix()

Лістинг коду вирішення задачі наведено в дод. А (стор.7). Екран роботи програми показаний на рис. Б.2. та рис. Б.3-4.

# ВИСНОВКИ

У результаті проведеної лабораторної роботи з теми "Структурування програм з використанням функцій в Python" було вивчено принципи організації коду за допомогою функцій. У ході практичних завдань з розрахунку периметру та площі прямокутників, а також обробки матриці, учасники закріпили навички створення та виклику функцій, що сприяє покращенню структури програм та полегшує їх розуміння. Також отримано практичний досвід роботи з бібліотекою NumPy для обробки матриць та використано функції для зчитування та запису даних з файлів. Учасники лабораторної роботи отримали цінні навички організації коду та розв'язання завдань з використанням функцій, що робить їх код більш читабельним та підтримуваним.

ДОДАТОК А

Лістинг коду програми до задач 1-2

import numpy as np

def RectPS(x1, y1, x2, y2):

    """

    Calculate the perimeter P and area S of a rectangle

    with opposite vertices at (x1, y1) and (x2, y2).

    Returns P and S as a tuple.

    """

    length = abs(x2 - x1)

    width = abs(y2 - y1)

    P = 2 \* (length + width)

    S = length \* width

    return P, S

def task1():

    """

    Input data, call function RectPS, and output results.

    """

    rectangles = []

    for i in range(3):

        x1 = int(input(f"Enter x1 for rectangle {i + 1}: "))

        y1 = int(input(f"Enter y1 for rectangle {i + 1}: "))

        x2 = int(input(f"Enter x2 for rectangle {i + 1}: "))

        y2 = int(input(f"Enter y2 for rectangle {i + 1}: "))

        rectangles.append((x1, y1, x2, y2))

    for i, rectangle in enumerate(rectangles):

        P, S = RectPS(\*rectangle)

        print(f"Rectangle {i + 1} - Perimeter: {P}, Area: {S}")

def process\_matrix():

    # Зчитування матриці з файлу

    matrix = np.loadtxt('input\_matrix.txt', dtype=int)

    # Знаходження номеру рядка

    row\_number = find\_row(matrix)

    # Виведення результату

    if row\_number == 0:

        print("Не знайдено рядків з однаковою кількістю додатних і від'ємних елементів.")

    else:

        print(f"Номер першого рядка з однаковою кількістю додатних і від'ємних елементів: {row\_number}")

    # Знаходження поелементної частки

    result\_matrix = divide\_matrix(matrix)

    # Збереження результату у файл

    np.savetxt('output\_matrix.txt', result\_matrix, fmt='%d')

def find\_row(matrix):

    for i, row in enumerate(matrix):

        positive\_count = np.count\_nonzero(row > 0)

        negative\_count = np.count\_nonzero(row < 0)

        if positive\_count == negative\_count:

            return i + 1  # Повертаємо номер рядка (індекс + 1), бо індексація зазвичай починається з 0

    return 0  # Якщо не знайдено рядків з однаковою кількістю додатних і від'ємних елементів

def divide\_matrix(matrix):

    random\_matrix = np.random.randint(1, 10, size=matrix.shape)  # Заповнюємо випадковою матрицею

    # Захист від ділення на нуль

    random\_matrix[random\_matrix == 0] = 1

    return matrix / random\_matrix

Лістинг коду програми до script.py

import module

choice = int(input("Please, choose the task 1-2 (0-EXIT): "))

while choice:

    if choice==1:

        module.task1();

    elif choice==2:

        module.process\_matrix();

    else:

        print("Wrong task number!")

    choice = int(input("Please, choose the task again (0-EXIT): "))

print("Good by!")

ДОДАТОК Б

Скрін-шоти вікна виконання програми

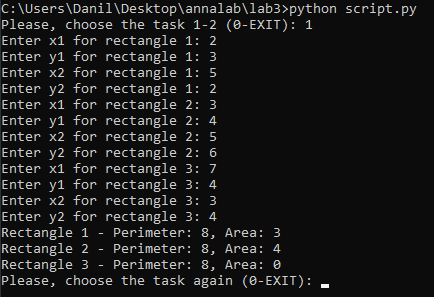


Рисунок Б.1 – Екран виконання програми для вирішення завдання 1



Рисунок Б.2 – Екран виконання програми для вирішення завдання   
2

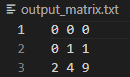
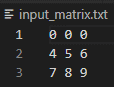


Рисунок Б.(3-4) – Файл вхідної і вихідної матриці.