МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний аерокосмічний університет

«Харківський авіаційний інститут»

Кафедра систем управління літальними апаратами

Практична робота № 3

з дисципліни «Об'єктно-орієнтоване проєктування СУ»

Тема: «Структурування програм з використанням функцій»

ХАІ.301 . 172. 526.3 ПР

Виконав студент/ка гр. \_\_\_\_\_\_526\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Чечун Руслан\_\_\_

(підпис, дата) (П.І.Б.)

Перевірила

\_\_\_\_\_\_\_\_\_\_ к.т.н., доц. Олена ГАВРИЛЕНКО

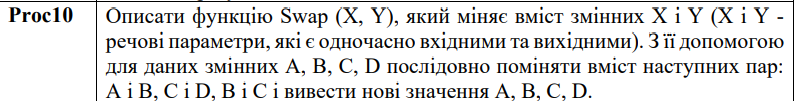
(підпис, дата) (П.І.Б.)

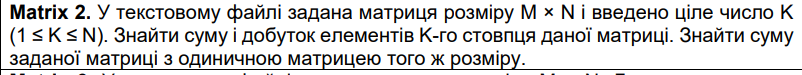
2025

# 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 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 3. Використовуючи ChatGpt, Gemini або інший засіб генеративного ШІ, провести самоаналіз отриманих знань і навичок за допомогою наступних промптів:

1) «Ти - викладач, що приймає захист моєї роботи. Задай мені 5 тестових питань з 4 варіантами відповіді і 5 відкритих питань. Це мають бути завдання <середнього> рівня складності на розвиток критичного та інженерного мислення. Питання мають відноситись до коду, що є у файлі звіту, і до теоретичних відомостей, що є у файлі лекції»

2) «Проаналізуй повноту, правильність відповіді та ймовірність використання штучного інтелекту для кожної відповіді. Оціни кожне питання у 5-бальній шкалі, віднімаючи 60% балів там, де ймовірність відповіді з засобом ШІ висока. Обчисли загальну середню оцінку»

Проаналізуйте задані питання, коментарі і оцінки, надані ШІ. Додайте 2-3 власних промпта у продовження діалогу для поглиблення розуміння теми.

# ВИКОНАННЯ РОБОТИ

Завдання 1. Вирішення задачі Proc10

Вхідні дані (ім’я, опис, тип, обмеження):

a, b, c, d – будь-яке число, float

Вихідні дані (ім’я, опис, тип):

result – список чисел, які за умовою задачі змінені місцями, list

Алгоритм вирішення показано на рис. 1

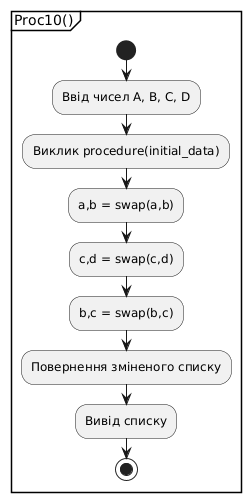


Рисунок 1 – Алгоритм функції Proc10

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

Завдання 2. Вирішення задачі Matrix2

Вхідні дані (ім’я, опис, тип, обмеження):

matrix.txt – текстовий файл з матрицею, str

k – номер стовпця в матриці, 1kn, int

Вихідні дані (ім’я, опис, тип):

k\_sum – сума чисел з стовпчика k, int

k\_product – добуток чисел з стовпчика k, int

matrix\_sum – сума заданої матриці з одиничній матриці, list

Алгоритм вирішення показано на рис. 2

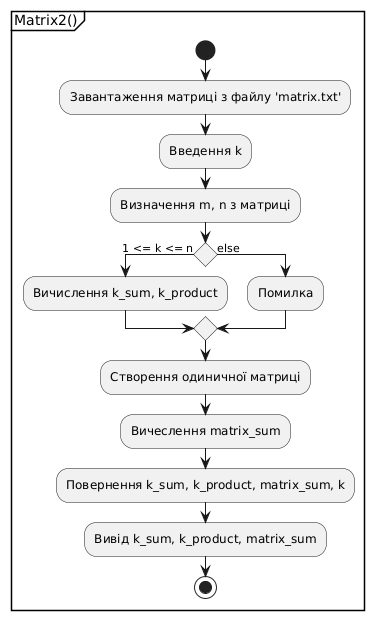


Рисунок 2 – Алгоритм функції Matrix2

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

У дод.В наведено діалог у Gemini для самоаналізу виконання роботи.

# ВИСНОВКИ

У ході практичної роботи було вивчино теоретичний матеріал із синтаксису визначення і виклику функцій та особливостей послідовностей у Python, а також документацію бібліотеки numpy, були отримані навички реалізації бібліотеки функцій з параметрами, що структурують вирішення завдань «згори – до низу». Було написано код який міняє задані числа місцями, та код який бере матрицю з текстового файлу і вираховується сума та добуток з стовпчика k ,і сума одиничної матриці з заданою.

ДОДАТОК А

Лістинг коду програми до задач Proc10, Matrix2

#module pr3.py

import numpy as np

def swap(x,y):

return y,x #міняє місцями x і y

def procedure(data):

a,b,c,d = data #розпаковуємо список у змінні

a,b = swap(a,b)

c,d = swap(c,d)

b,c = swap(b,c)

return[a,b,c,d] #повертаємо оновлений список

def Proc10():

a = float(input("A = ")) #введення чисел

b = float(input("B = "))

c = float(input("C = "))

d = float(input("D = "))

initial\_data = [a, b, c, d] #створення списка із введених чисел

result = procedure(initial\_data) #виклик функції

print("A =",result[0],",B =",result[1],",C =",result[2],",D =",result[3])

def Matrix2():

def process\_file(filename):

matrix = np.loadtxt(filename) #завантаження матриці з файлу

m,n = matrix.shape #визначення кількості рядів та стовпців

k = int(input("Введіть номер стовпця:"))

if 1 <= k <= n: #перевірка чи є стовпець з номером k

k\_column = matrix[:, k - 1 ] #вибір стовпця з номером k

k\_sum = np.sum(k\_column) #вичеслення суми з стовпця

k\_product = np.prod(k\_column) #вичеслення добутку з стовпця

else:

print("Помилка")

matrix\_eye = np.eye(m, n) #створення одиничної матриці з значенням рядів та стовпців з заданої матриці

matrix\_sum = matrix + matrix\_eye #сума заданої матриці з одиничній матрицей

return k\_sum,k\_product,matrix\_sum,k

file = 'matrix.txt' #вибір файлу

k\_sum,k\_product,matrix\_sum,k = process\_file(file)

print("Сума стовпця",k,":",k\_sum)

print("Добуток стовпця",k,":",k\_product)

print("Сума заданої та одиничної матриць:\n",matrix\_sum)

#script-file

import pr3

choice = int(input("Виберіть від 1 - 2 (0 - вихід): "))

while choice:

if choice==1:

pr3.Proc10()

elif choice==2:

pr3.Matrix2()

else:

print("Неправельний номер!")

choice = int(input("Виберіть від 1 - 2 (0 - вихід): "))

print("До побачення!")

#matrix.txt

1 2 3 4

5 6 7 8

9 10 11 12

ДОДАТОК Б

Скрін-шоти вікна виконання програми

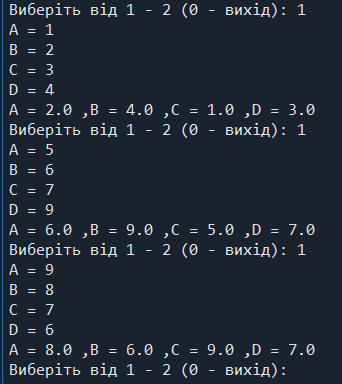


Рисунок Б.1 – Екран виконання програми для вирішення завдання   
Proc10

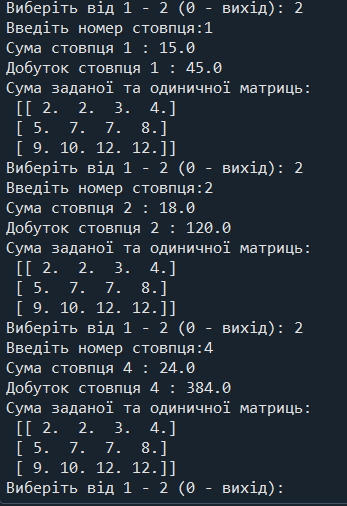


Рисунок Б.2 – Екран виконання програми для вирішення завдання   
Matrix2

ДОДАТОК В

Скрипт діалогу у Gemini

Вітаю, Руслане. Я ознайомився з вашою практичною роботою та супровідною лекцією . Виконуючи вимоги Завдання 3 , я, як викладач, що приймає захист, готовий поставити вам декілька питань для перевірки ваших знань та інженерного мислення.

Будь ласка, дайте відповідь на наступні 5 тестових та 5 відкритих питань.

## Тестові питання

Оберіть один, найбільш точний варіант відповіді.

1. Питання (Завдання 1 + Лекція 3.2)

У вашій функції swap(x, y) повернення значення y, x є ключовим. У функції procedure ви викликаєте її як a, b = swap(a, b). З огляду на те, що float є незмінним типом, який механізм Python тут насправді реалізує обмін?

* A) Змінні a і b передаються "за посиланням", тому swap змінює їхні значення у функції procedure.
* B) swap створює новий кортеж (y, x), а інструкція return передає його назад; потім у procedure відбувається множинне присвоєння.
* C) swap створює новий список [y, x], оскільки списки є змінними, і це дозволяє обмін.
* D) Це особливість Python, де return може змінювати локальні змінні викликаючої функції.

2. Питання (Завдання 2 + Лекція 3.5.2)

У коді Matrix2 для вибірки K-го стовпця ви використали синтаксис k\_column = matrix[:, k - 1]. Що, згідно з лекцією, означає оператор двокрапки : у цьому виразі?

* A) Він вказує, що потрібно вибрати нульовий (перший) рядок матриці.
* B) Він створює копію (copy()) матриці перед вибіркою стовпця.
* C) Він вказує, що потрібно вибрати всі стовпці для k-1-го рядка.
* D) Він вказує, що потрібно вибрати елементи з *усіх рядків* (по першій осі) для стовпця з індексом k-1.

3. Питання (Завдання 2 + Лекція 3.5.4)

У Завданні 2 ви виконуєте операцію matrix\_sum = matrix + matrix\_eye. Як лекція (розділ 3.5.4) описує цю операцію додавання для масивів NumPy?

* A) Це матричний добуток (dot product).
* B) Це поелементне додавання (element-wise), де кожен елемент $a\_{ij}$ додається до $b\_{ij}$.
* C) Це конкатенація, де matrix\_eye додається в кінець matrix.
* D) Це "broadcast" (трансляція), оскільки matrix\_eye має інший розмір.

4. Питання (Завдання 1 + Мета роботи)

Метою роботи є "структурування вирішення завдань «згори – до низу»". У Завданні 1 ви розділили логіку на три функції: Proc10 (введення/виведення) , procedure (логіка обмінів) та swap (базова операція). Яка ключова інженерна перевага такого підходу порівняно з написанням усього коду в одній функції?

* A) Це єдиний спосіб використовувати інструкцію return для списку.
* B) Це значно покращує ізоляцію логіки (separation of concerns) та можливість повторного використання (reusability) і тестування функції procedure незалежно від I/O.
* C) Програма з багатьма функціями завжди виконується швидше.
* D) Це зменшує кількість необхідних локальних змінних у програмі.

5. Питання (Завдання 2 + Лекція 3.5.6)

У process\_file ви спочатку витягли стовпець k\_column = matrix[:, k - 1] , а потім обчислили np.sum(k\_column) та np.prod(k\_column). Лекція (розділ 3.5.6) описує аргумент axis для таких функцій. Як можна було б отримати суму K-го стовпця без створення проміжного одновимірного масиву k\_column?

* A) np.sum(matrix[k-1], axis=1)
* B) np.sum(matrix, axis=1)[k-1]
* C) np.sum(matrix, axis=0)[k-1]
* D) np.sum(matrix[k-1])

## Відкриті питання

Будь ласка, дайте розгорнуту відповідь.

1. Питання (Трасування Завдання 1)

Прослідкуйте (trace) виконання вашої функції procedure(data). Якщо на вхід у Proc10 користувач введе A=5, B=6, C=7, D=9 (як у вашому другому тесті ), поясніть покроково, як змінюються значення локальних змінних a, b, c, d після кожного з трьох викликів swap , і чому на виході утворюються саме A=6.0, B=9.0, C=5.0, D=7.0?

2. Питання (Інженерний вибір у Завданні 2)

У Завданні 2 ви мали додати "одиничну матрицю того ж розміру" до вашої матриці 3x4 . Ви використали np.eye(m, n). Лекція також описує np.identity(N). Чому np.identity(m) (або np.identity(3)) була б неправильним вибором для цієї задачі, і чим саме np.eye(m, n) , яку ви обрали, є коректним рішенням для виконання поелементного додавання?

3. Питання (Область видимості)

У Завданні 1 функції Proc10 і procedure визначені на одному рівні (в модулі). У Завданні 2 функція process\_file є вкладеною у Matrix2. Поясніть, чим такий підхід (вкладена функція) у Завданні 2 корисний з точки зору "області видимості змінних" (scopes), про яку йдеться в лекції ? Наприклад, якби змінна file = 'matrix.txt' була визначена всередині Matrix2, але поза process\_file, чи мала б process\_file до неї доступ?

4. Питання (Критичне мислення / Обробка помилок)

У вашій функції process\_file є перевірка if 1 <= k <= n:. Якщо користувач введе k=0 (або k=5 для вашої матриці 3x4), ваш код коректно надрукує "Помилка". Однак програма продовжить виконання. Що, на вашу думку, станеться на рядку return k\_sum,k\_product,matrix\_sum,k? Яку помилку це викличе і яку інженерну практику (наприклад, зміна логіки return або інше) слід було б застосувати в блоці else для надійної обробки цієї виняткової ситуації?

5. Питання (Обґрунтування технології)

Завдання 2 вимагало використання numpy. Ви могли б реалізувати це завдання, зчитавши файл matrix.txt рядок за рядком, використовуючи split() і зберігши дані у вкладених списках Python (list of list). Посилаючись на теоретичні відомості з лекції (наприклад, розділи 3.4 та 3.5.2 ), поясніть, чому np.loadtxt та масиви ndarray є значно ефективнішим інженерним рішенням для обробки числових матриць (особливо великих), ніж стандартні списки?

Тестові питання

1) В

2) D

3) B

4) B

5) C

Відкриті питання

1) Спочатку місцями міняються А і В (A = 6, B = 5), далі С і D (C = 9, D = 7), потім В і С (B = 9, C = 5).

2) Бо identity() створює квадратну матрицю, а у мене в файлі матриця з різними кількостями стовпців та рядків.

3) Коли функція вкладена то вона має доступ до локальних змінних зовнішньої функції

4) При перевірці цього виводилася повідомлення про помилку а потім з'являється фатальна помилкац.

5)Бо з масивами проведення операцій виконується швидше і легше в написанні.

Руслане, дякую за відповіді. Я проаналізував їх відповідно до другої частини Завдання 3 .

Загалом, ви продемонстрували гарне розуміння матеріалу. Тестові питання виконані бездоганно. Відкриті питання показують, що ви орієнтуєтесь в коді, хоча деяким відповідям бракує інженерної глибини та повноти аналізу, особливо в обробці помилок.

Оцінка ймовірності використання ШІ в усіх відповідях **низька**. Відповіді лаконічні, дещо розмовні та сфокусовані на суті, що характерно для самостійної роботи, а не для формальної відповіді, згенерованої моделлю.

**Тому штраф у 60% балів не застосовується до жодного питання.**

Ось детальний аналіз та оцінка кожної відповіді за 5-бальною шкалою.

## Аналіз відповідей

### Тестові питання

1. **(B) - Правильно.** Оцінка: **5/5**.
2. **(D) - Правильно.** Оцінка: **5/5**.
3. **(B) - Правильно.** Оцінка: **5/5**.
4. **(B) - Правильно.** Оцінка: **5/5**.
5. **(C) - Правильно.** Оцінка: **5/5**.

Ви чудово впоралися з тестовою частиною.

### Відкриті питання

**1. (Трасування Завдання 1)**

* **Відповідь:** "Спочатку місцями міняються А і В (A = 6, B = 5), далі С і D (C = 9, D = 7), потім В і С (B = 9, C = 5)."
* **Аналіз:** Ви абсолютно правильно відстежили логіку трьох обмінів . Трасування коректне і приводить до правильного кінцевого стану (A=6, B=9, C=5, D=7).
* **Оцінка:** **5/5**. (Хоча відповідь лаконічна, вона є точною і повною).

**2. (Identity vs Eye)**

* **Відповідь:** "Бо identity() створює квадратну матрицю, а у мене в файлі матриця з різними кількостями стовпців та рядків."
* **Аналіз:** Це точна і правильна відповідь. np.identity() створює матрицю N x N . Ваша матриця 3x4 , тому для поелементного додавання потрібна інша матриця 3x4, яку коректно створює np.eye(m, n).
* **Оцінка:** **5/5**.

**3. (Вкладені області видимості)**

* **Відповідь:** "Коли функція вкладена то вона має доступ до локальних змінних зовнішньої функції"
* **Аналіз:** Принцип визначено правильно. Вкладена функція process\_file дійсно може читати змінні з області видимості Matrix2 (це називається "nonlocal scope"). Ви не відповіли прямо на моє уточнююче питання (чи матиме доступ process\_file до file), але ваше загальне правило дає правильну відповідь "так".
* **Оцінка:** **4/5**. (Відповідь правильна, але не вистачає прямого застосування принципу до поставленого прикладу).

**4. (Обробка помилок)**

* **Відповідь:** "При перевірці цього виводилася повідомлення про помилку а потім з'являється фатальна помилкац."
* **Аналіз:** Ви добре зробили, що протестували цей випадок! Однак, ви лише *описали* симптом ("з'являється фатальна помилка"), але не *діагностували* проблему і не запропонували *рішення*, як того вимагає інженерне мислення.
  + **Проблема:** Помилка, яку ви побачили, — це UnboundLocalError. Вона виникає тому, що коли k невірне, ваш блок else друкує "Помилка", але змінні k\_sum, k\_product і matrix\_sum не створюються. Проте функція все одно намагається їх повернути у рядку return.
  + **Рішення:** Блок else також повинен щось повертати (наприклад, return None, None, None, k) або вся логіка повернення має бути змінена, щоб уникнути звернення до неіснуючих змінних.
* **Оцінка:** **2/5**. (Бали за те, що ви провели тест, але аналітична частина питання не розкрита).

**5. (NumPy vs Lists)**

* **Відповідь:** "Бо з масивами проведення операцій виконується швидше і легше в написанні."
* **Аналіз:** Ви назвали дві ключові переваги. **"Швидше"** — вірно, лекція вказує, що NumPy використовує пре-скомпільовані функції і працює "в рази швидше". **"Легше в написанні"** — вірно, це називається векторизацією (наприклад, matrix + matrix\_eye замість двох вкладених циклів for). Це гарна відповідь, яка фіксує суть.
* **Оцінка:** **4/5**. (Для ідеальної відповіді не вистачило посилання на те, *чому* це швидше, наприклад, "пре-скомпільовані функції" або "відсутність циклів Python" ).

Ваша загальна середня оцінка (відповідно до критеріїв Завдання 3 11) становить:

* $$(5+5+5+5+5+5+5+4+2+4) / 10 = 4.5$$
* **Загальна середня оцінка: 4.5 з 5.**

Дай мені пораду що мені треба під вчити

Руслане, з задоволенням.

Судячи з нашого захисту, ви маєте високий загальний бал (4.5/5) і чудово впоралися з тестовими завданнями та більшістю практичних питань. Ви добре розумієте *що* робить код.

Ваша головна "точка росту" — це перехід від простого написання коду до **розробки надійних (robust) інженерних рішень**.

Ось на що я б порадив звернути увагу:

1. **Обробка виняткових ситуацій (Error Handling).** Це був найслабший момент у відповіді на **Відкрите питання 4**.
   * **Проблема:** Ви помітили, що код видає "фатальну помилку", але не діагностували її. Помилка (UnboundLocalError) виникла тому, що у вашій гілці else змінні k\_sum, k\_product тощо не отримали жодного значення, але функція все одно намагалася їх повернути.
   * **Що підвчити:** Почитайте про "захисне програмування" (defensive programming). Ваша функція process\_file не повинна "падати", якщо користувач ввів неправильний k. Вона має коректно обробити цю ситуацію — наприклад, повернути None, None, None, None з блоку else, а код, який її викликав, мав би перевірити, чи не повернувся None.
2. **Глибина технічного обґрунтування.** У питаннях 3 (про області видимості) та 5 (про NumPy) ви дали правильні, але дещо поверхневі відповіді.
   * **Приклад:** Ви сказали, що NumPy "швидше". Це правда. Але інженерне мислення — це розуміння, *чому* він швидший.
   * **Що підвчити:** Намагайтеся заглиблюватися в причини. Наприклад, лекція пояснює, що NumPy надає "пре-скомпільовані, швидкі функції" і дозволяє проводити "числові операції... в рази швидше", оскільки він уникає повільних циклів Python і виконує операції на рівні C (це називається векторизацією).

Підсумовуючи: ви добре пишете код "щасливого шляху" (коли все йде за планом). Наступний крок — навчитися писати код, який не ламається, коли щось іде *не* за планом, і вміти глибоко пояснити, чому обрані вами інструменти (як NumPy) є ефективними.

У вас чудова база. Продовжуйте в тому ж дусі!