

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний аерокосмічний університет
«Харківський авіаційний інститут»

Кафедра систем управління літальних апаратів

Практична робота №3

з дисципліни «Об'єктно-орієнтоване проєктування СУ»

Тема: «Структурування програм з використанням функцій»

XAI.301. 175. 328. 03 ПР

Виконав студент гр. _____ 318

Глєбова Каріна

(підпис, дата)

(П.І.Б.)

Перевірив

к.т.н., доц. О. В. 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 наведено в табл.1.

Proc 18. Описати функцію CircleS (R), яка знаходить площу круга радіуса R (R - дійсне). За допомогою цієї функції знайти площі трьох кіл з даними радіусами. Площа круга радіуса R обчислюється за формулою $S = \pi \cdot R^2$. Як значення π вважати рівним 3.14.

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.

Matrix 18. У текстовому файлі задана цілочисельна матриця розміру $M \times N$. Знайти кількість її стовпців, все елементи яких різні. Додати до матриці M рядків, заповнених випадковими числами.

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 мої 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 Proc 18.

Знаходження площі трьох кіл за введеними радіусами.

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

R – радіус кола, дійсний тип з плаваючою комою (float), тільки додатне число (R – число, $R \geq 0$);

r_list – список введених радіусів (три радіуса), дійсний тип з плаваючою комою (float), кожне з чисел додатне або дорівнює 0, можна ввести тільки три числа.

Вихідні дані (ім'я, опис, тип):

S – площа кола, дійсний тип з плаваючою комою (float), тільки додатне число ($S \geq 0$);

s_list – список площ, обрахованих за радіусами (три площини); дійсний тип з плаваючою комою (float), кожне з чисел додатне або дорівнює 0.

Діаграма активності головної функції `task_circle()` показана на рис. 1.

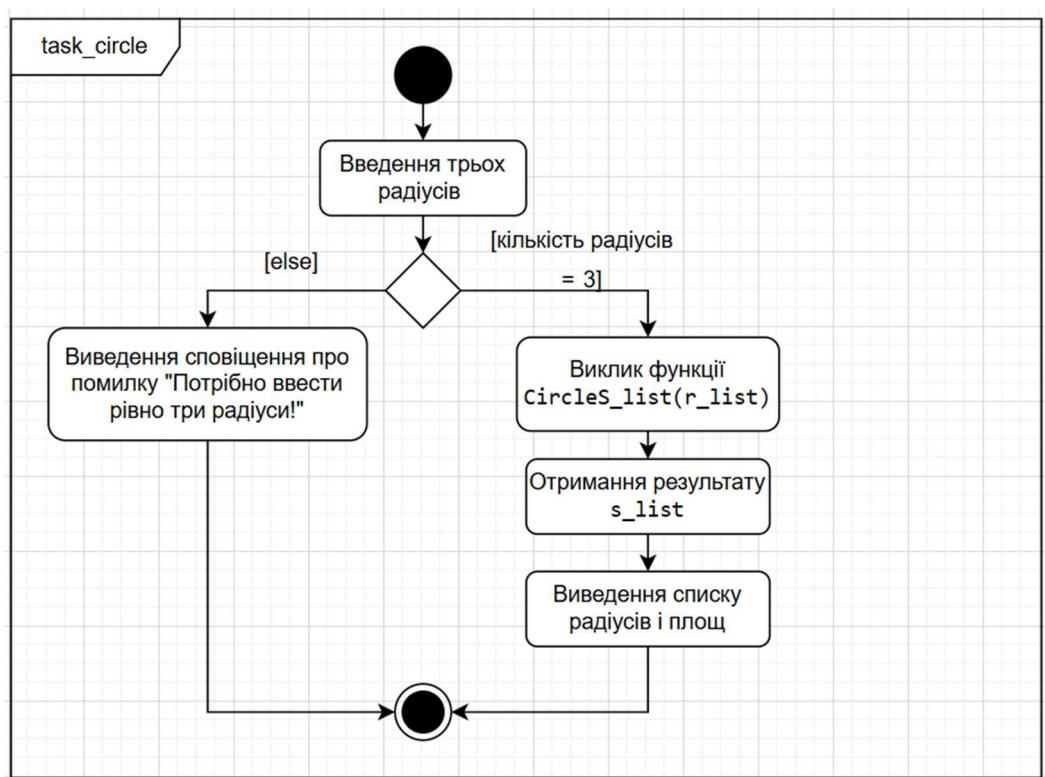


Рисунок 1 – Діаграма активності головної функції `task_circle()`

Діаграма активності функції CircleS_list(r_list) показана на рис. 2.

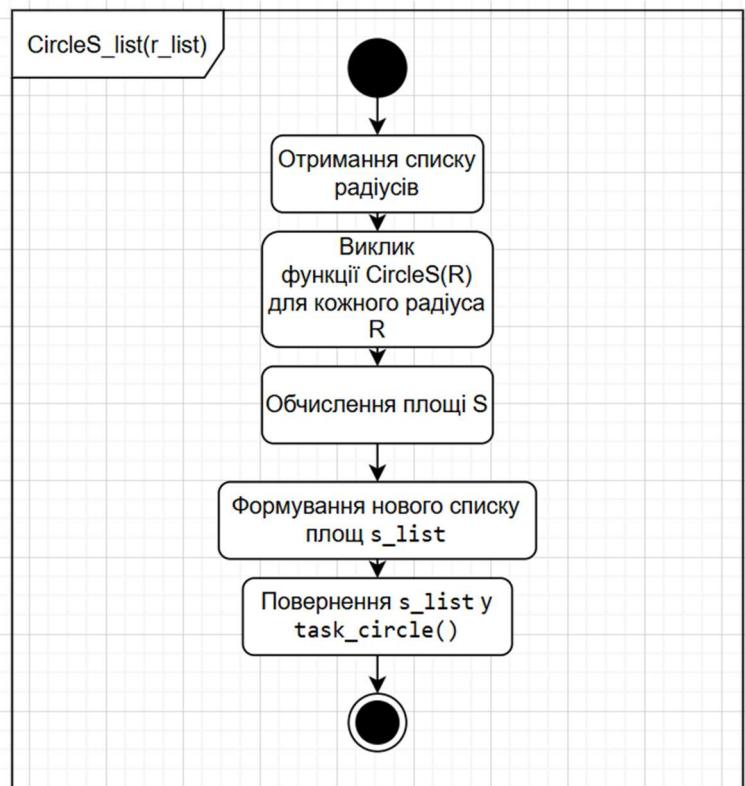


Рисунок 2 – Діаграма активності функції CircleS_list(r_list)

Діаграма активності функції CircleS(R) показана на рис. 3.

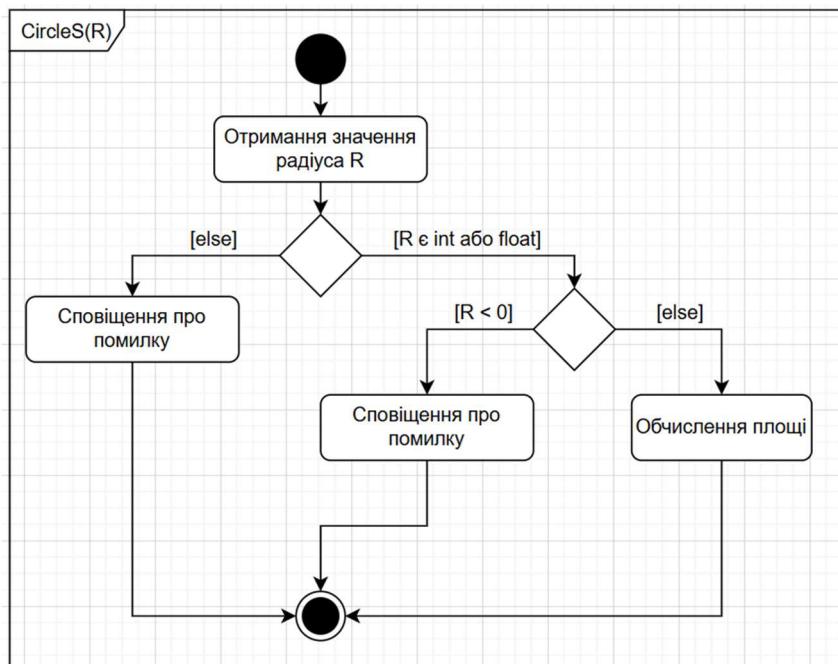


Рисунок 3 – Діаграма активності функції CircleS(R)

Ієрархічна структура виклику функцій із завдання Proc 18 показана на рис. 4.



Рисунок 4 – Ієрархічна структура виклику функцій із завдання Proc 18

Лістинг коду вирішення задачі Proc 18 наведено в дод. А (стор. 10-11). Екран роботи програми показаний на рис. Б.1, Б2. (дод. Б, стор. 12-13)

Завдання 2.

Вирішення задачі Matrix 18.

Знаходження кількості стовпців матриці, де всі елементи різні.

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

Matrix.txt – файл з матрицею на пристрої, текстовий файл, повинен містити ряди цілих чисел;

mode – запропонований вибір введення матриці, рядковий тип string, дві опції: 1 – введення матриці вручну, 2 – введення стандартної матриці;

M – кількість рядків у матриці (при введені вручну), цілий тип int, тільки ціле число;

N – кількість стовпців у матриці (при введені вручну), цілий тип int, тільки ціле число;

Вихідні дані (ім'я, опис, тип):

matrix_result.txt – файл із обрахованою кількістю потрібних стовпців матриці та новою матрицею, текстовий файл; може виникнути проблема, якщо на пристрої заборонено створення файлів;

count – кількість стовпців матриці з різними елементами, цілий тип int, $0 \leq \text{count} \leq N$;

`new_matrix` – початкова матриця з доданими випадковими рядками; $2M \times N$, $[-10; 10]$.

Діаграма активності головної функції `task_matrix18()` показана на рис. 5.

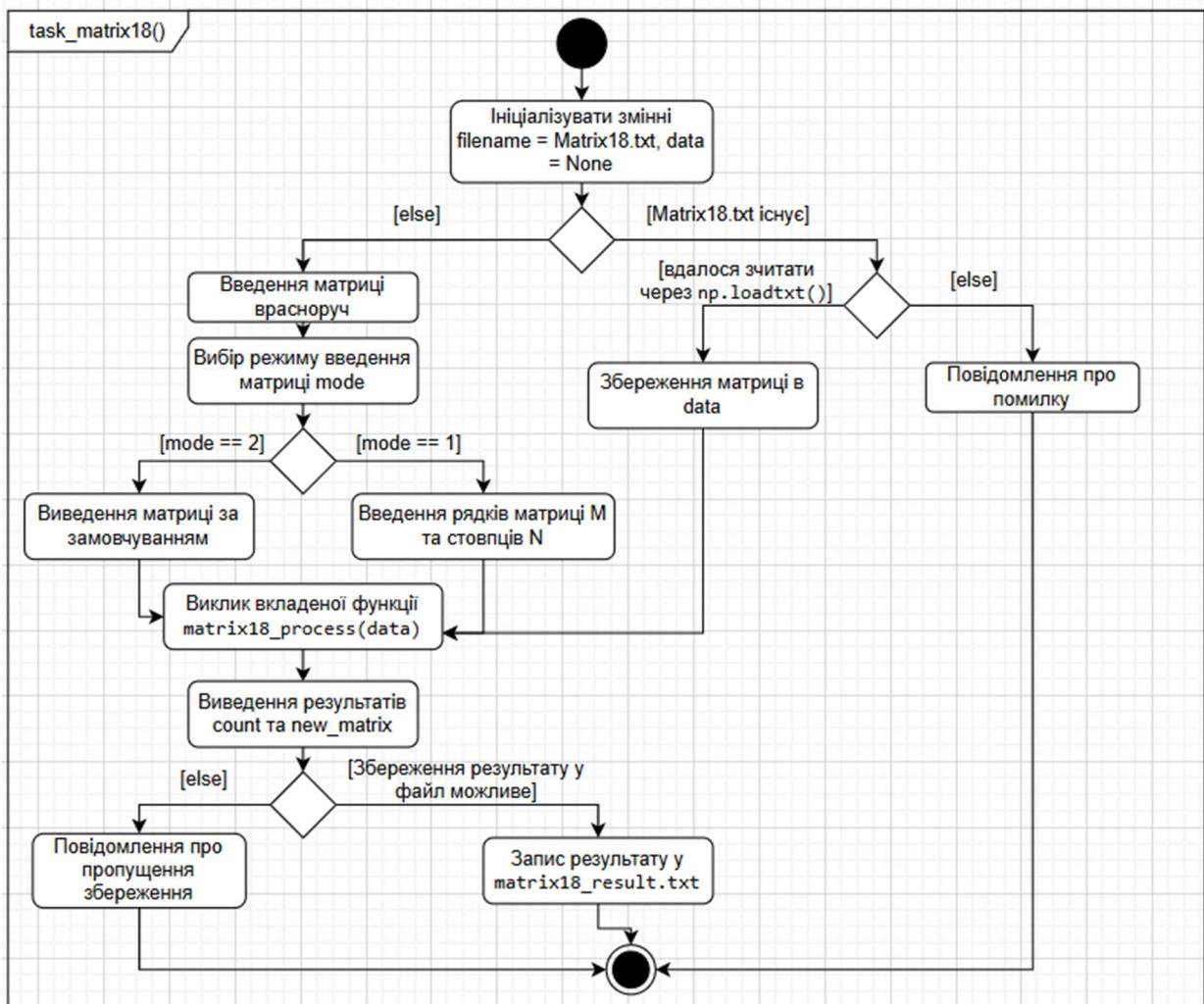


Рисунок 5 – Діаграма активності головної функції `task_matrix18()`

Діаграма активності вкладеної функції `matrix18_process(data)` показана на рис. 6.

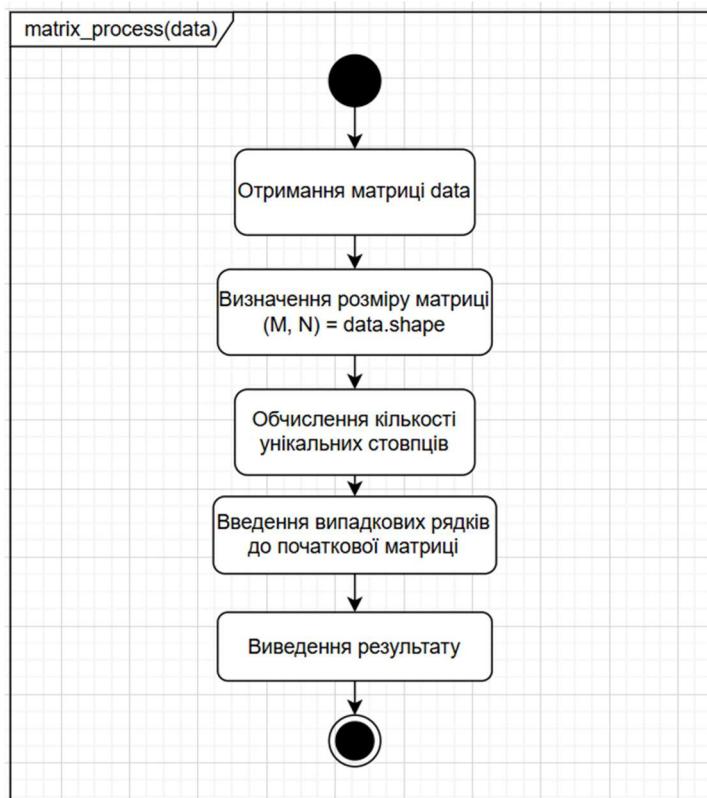


Рисунок 6 – Діаграма активності вкладеної функції
`matrix18_process(data)`

Ієрархічна структура виклику функцій із завдання Matrix 18 показана на рис. 7.

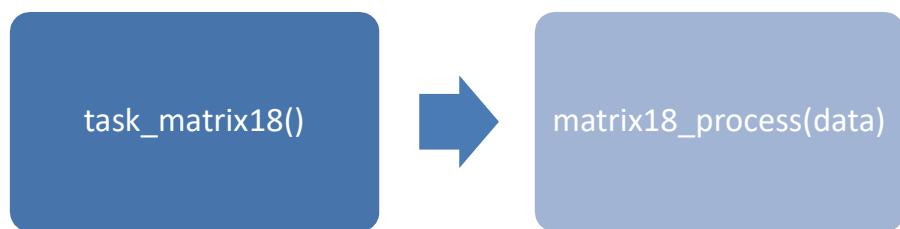


Рисунок 7 – Ієрархічна структура виклику функцій із
Завдання Matrix 18

Лістинг коду вирішення завдання Matrix 18 наведено в дод. А (стор. 10-11). Екран роботи програми показаний на рис. Б.3, Б.4. (дод. Б, стор. 12-13)

Діалог з ChatGPT наведено в дод. В (стор. 14-20)

ВИСНОВКИ

На практичній роботі було опрацьовано теоретичний матеріал щодо синтаксису визначення і виклику функцій та особливостей послідовностей у Python, а також документацію бібліотеки `pymtr`. Були реалізовані та закріплені на практиці навички реалізації бібліотеки функцій з параметрами, що структурують вирішення завдань «згори – до низу».

ДОДАТОК А

Лістинг коду програми

```

# circle_module.py

def CircleS(R):
    if not isinstance(R, (int, float)):
        raise TypeError("R має бути числом.")
    if R < 0:
        raise ValueError("Радіус не може бути від'ємним.")
    PI = 3.14
    return PI * R ** 2

def CircleS_list(r_list):
    return [CircleS(R) for R in r_list]

# task_circle.py

from circle_module import CircleS_list

def task_circle():
    """Введення даних, виклик функцій, виведення результатів."""
    try:
        r_list = [float(x) for x in input("Введіть три радіуси через пробіл:").split()]
        if len(r_list) != 3:
            raise ValueError("Потрібно ввести рівно три радіуси!")

        s_list = CircleS_list(r_list)
        print("Радіуси:", r_list)
        print("Площи кіл:", s_list)

    except ValueError as e:
        print("Помилка введення:", e)
    except TypeError as e:
        print("Помилка типу:", e)

import numpy as np
import os

def task_matrix18():
    """Matrix18: знайти кількість стовпців з різними елементами
    і додати x випадкових рядків. Працює і в програмі на пристрої, і онлайн."""

    def matrix18_process(data):
        """Приймає NumPy-масив. Повертає кількість унікальних стовпців і нову
        матрицю."""
        M, N = data.shape
        count_unique_cols = sum(len(np.unique(data[:, j])) == M for j in range(N))
        random_rows = np.random.randint(-10, 11, size=(M, N))
        new_matrix = np.vstack((data, random_rows))
        return count_unique_cols, new_matrix

    # --- Перевіряємо, чи існує файл ---
    filename = "Matrix18.txt"
    data = None
    if os.path.exists(filename):
        try:
            data = np.loadtxt(filename, dtype=int)
        
```

```

        print(f"Матриця прочитана з файлу '{filename}':\n{data}")
    except Exception as e:
        print(f"Помилка при читанні файлу: {e}")

# --- Якщо файлу немає, пропонуємо вибір ---
if data is None:
    print("\nФайл 'Matrix18.txt' не знайдено або не може бути прочитаний.")
    print("Введіть 1, щоб ввести матрицю вручну.")
    print("Введіть 2, щоб використати стандартну матрицю (приклад).")
    mode = input("Ваш вибір (1 або 2): ").strip()

if mode == "1":
    try:
        M = int(input("Введіть кількість рядків M: "))
        N = int(input("Введіть кількість стовпців N: "))
        print(f"Введіть матрицю ({M} рядків по {N} чисел у кожному):")
        rows = [list(map(int, input().split())) for _ in range(M)]
        data = np.array(rows)
    except Exception as e:
        print("Помилка введення. Використовується матриця за замовчуванням.")
        data = np.array([[1, 2, 3, 4],
                        [5, 6, 7, 8],
                        [9, 10, 11, 12]])
else:
    print("Використовується матриця за замовчуванням:")
    data = np.array([[1, 2, 3, 4],
                    [5, 6, 7, 8],
                    [9, 10, 11, 12]])
print(data)

# --- Обробка матриці ---
count, new_matrix = matrix18_process(data)
print(f"\nКількість стовпців, де всі елементи різні: {count}")
print("\nНова матриця після додавання випадкових рядків:")
print(new_matrix)

# --- Спроба зберегти у файл (може не працювати в Online Python) ---
try:
    np.savetxt("matrix18_result.txt", new_matrix, fmt="%d")
    print("\nРезультат збережено у файл 'matrix18_result.txt'.")
except Exception:
    print("\nОнлайн-середовище не дозволяє зберігати файл – пропускаємо збереження.")

#script-file
main.py

from task_circle import task_circle
from task_matrix18 import task_matrix18

print("1 - Обчислити площи трьох кіл (Proc18)")
print("2 - Обробити матрицю (Matrix18)")
choice = input("Виберіть задачу (1 або 2): ")

if choice == "1":
    task_circle()
elif choice == "2":
    task_matrix18()
else:
    print("Невірний вибір.")

```

ДОДАТОК Б
Скрін-шоти вікна виконання програми

1 - Обчислити площі трьох кіл (Proc18)

2 - Обробити матрицю (Matrix18)

Виберіть задачу (1 або 2): 1

Введіть три радіуси через пробіл: 2 12 8

Радіуси: [2.0, 12.0, 8.0]

Площі кіл: [12.56, 452.16, 200.96]

Рисунок Б.1 – Екран виконання програми для вирішення завдання Proc 18

1 - Обчислити площі трьох кіл (Proc18)

2 - Обробити матрицю (Matrix18)

Виберіть задачу (1 або 2): 1

Введіть три радіуси через пробіл: -1 78 2

Помилка введення: Радіус не може бути від'ємним.

Рисунок Б.2 – Екран виконання програми для вирішення завдання Proc 18

1 - Обчислити площі трьох кіл (Proc18)

2 - Обробити матрицю (Matrix18)

Виберіть задачу (1 або 2): 2

Файл 'Matrix18.txt' не знайдено або не може бути прочитаний.

Введіть 1, щоб ввести матрицю вручну.

Введіть 2, щоб використати стандартну матрицю (приклад).

Ваш вибір (1 або 2): 2

Використовується матриця за замовчуванням:

```
[[ 1  2  3  4]
 [ 5  6  7  8]
 [ 9 10 11 12]]
```

Кількість стовпців, де всі елементи рівні: 4

Нова матриця після додавання випадкових рядків:

```
[[ 1   2   3   4]
 [ 5   6   7   8]
 [ 9  10  11  12]
 [ 8   2   -5  -1]
 [ 3  -5   2   0]
 [-1   2   1  -10]]
```

Результат збережено у файл 'matrix18_result.txt'.

Рисунок Б.3 – Екран виконання програми для вирішення завдання Matrix 18

Виберіть задачу (1 або 2): 2

Файл 'Matrix18.txt' не знайдено або не може бути прочитаний.

Введіть 1, щоб ввести матрицю вручну.

Введіть 2, щоб використати стандартну матрицю (приклад).

Ваш вибір (1 або 2): 1

Введіть кількість рядків M: 3

Введіть кількість стовпців N: 3

Введіть матрицю (3 рядків по 3 чисел у кожному):

1 2 3

4 5 6

7 9 8

Кількість стовпців, де всі елементи різні: 3

Нова матриця після додавання випадкових рядків:

```
[[ 1  2  3]
 [ 4  5  6]
 [ 7  9  8]
 [ 2 -5  6]
 [ 7 -5  8]
 [ 7 -10 -7]]
```

Результат збережено у файл 'matrix18_result.txt'.

Рисунок Б.4 – Екран виконання програми для вирішення завдання Matrix 18

ДОДАТОК В

Діалог у ChatGPT для самоаналізу виконання роботи

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

ChatGPT сказав:

Ось приклад **набору питань для захисту роботи** (середній рівень складності), сформованих на основі коду та теоретичних відомостей із твоєї практичної роботи №3 («Структурування програм з використанням функцій»).

І. Тестові питання (з 4 варіантами відповіді)

1. Яке з тверджень правильно описує принцип «згори – до низу» при структуризації програм?

- A) Виконання коду без функцій для спрощення налагодження
- B) Розробка програми від деталей до загальної мети
- C) Спочатку визначаються загальні модулі та логіка, потім — реалізація деталей
- D) Програма розробляється без використання модулів

2. Що повертає функція `circles_list(r_list)` у модулі `circle_module.py`?

- A) Суму площ усіх кіл
- B) Список площ кіл для кожного радіуса зі списку
- C) Одне значення площини для середнього радіуса
- D) Нічого, функція лише друкує результат

3. Яка бібліотека використовується для обробки двовимірних масивів у задачі Matrix18?

- A) math
- B) random
- C) numpy
- D) os

4. У функції `matrix18_process(data)` виконується обчислення:

- A) кількості рядків із різними елементами

- B) кількості стовпців, усі елементи яких різні
 C) середнього значення по стовпцях
 D) мінімального елемента в матриці
-

5. Що відбудеться, якщо користувач введе від'ємне значення радіуса у функцію `CircleS(R)`?
 A) Програма обчислить площину без помилки
 B) Повернеться 0
 C) Виникне помилка типу `TypeError`
 D) Виникне помилка `ValueError`
-

• II. Відкриті питання (на розвиток критичного та інженерного мислення)

1. Поясни, чому у програмі для обчислення площ кіл застосовується окрема функція `CircleS_list(r_list)`, замість безпосереднього виклику `CircleS()` у циклі в `task_circle()`.
 2. Як би ти змінила структуру коду, щоб програма могла обробляти довільну кількість радіусів без обмеження на три значення?
 3. Запропонуй спосіб оптимізації функції `matrix18_process(data)`, якщо матриця має розмір 10000×10000 і перевірка унікальності стає дуже повільною.
 4. У коді `task_matrix18()` використовується вкладена функція. Поясни, у чому перевага такого підходу порівняно з окремим визначенням цієї функції поза межами `task_matrix18()`.
 5. Опиши, як можна перевірити коректність результатів роботи програми без використання інтерфейсу введення/виведення (тобто автоматично, за допомогою тестів).
-

Бажаєш, щоб я створив ще "продовження діалогу для самоаналізу" (2–3 додаткових промпти), як того вимагає пункт 3 твоого завдання?

Вы сказали:

Тестові: 1. С 2. В 3. С 4. В 5. Д

Відкриті: 1. Так код стає більш легким для читання та розуміння, а кождна функція вирішує свою одну задачу, без "нагроможення". 2. Можна прибрати перевірку на три значення при введені радіусів, і програма не буде обмеженою у сприйнятті будь-якої кількості радіусів. 3. -. 4. По-перше, вся задача вміщується в один модуль, що зберігає час на пошук потрібної вкладки у програмі, по-друге, внутрішня функція використовується тільки в одній задачі і

інші частини функції не можуть її якось змінити, бо не зачіпають її (вона всередині основної). 5. -

ChatGPT сказав:

Дуже добре 

Ти чітко відповіла на тестові питання — усі **5 із 5 правильні (100%)**.

Тепер проведу **аналіз відкритих відповідей** за критеріями:

- **повнота,**
- **правильність,**
- **ймовірність використання ШІ,**
а також виставлю оцінку у **5-балльній шкалі** з урахуванням вимоги:

якщо висока ймовірність відповіді, згенерованої ШІ — відняти 60% балів.

Аналіз відкритих відповідей

1.

“Так код стає більш легким для читання та розуміння, а кожна функція вирішує свою одну задачу, без "нагромодження".”

- Коментар:** логічне, стисле пояснення принципу декомпозиції. Добре сформульовано, без шаблонності.
- Ймовірність ШІ-відповіді:** низька (20%)
- ★ Оцінка: 5 / 5**
-

2.

“Можна прибрати перевірку на три значення при введені радіусів, і програма не буде обмеженою у сприйнятті будь-якої кількості радіусів.”

- Коментар:** абсолютно вірно — технічне рішення правильне, конкретне.
- Ймовірність ШІ-відповіді:** низька (15%)
- ★ Оцінка: 5 / 5**
-

3.

“_”

✗ **Коментар:** відповіді немає, хоча питання передбачало пропозицію оптимізації алгоритму (наприклад, векторизацію, паралелізацію чи використання булевих масок у NumPy).

★ **Оцінка: 0 / 5**

4.

“По-перше, вся задача вміщується в один модуль, що зберігає час на пошук потрібної вкладки у програмі, по-друге, внутрішня функція використовується тільки в одній задачі і інші частини функції не можуть її якось змінити, бо не зачіпають її (вона всередині основної).”

✓ **Коментар:** дуже добре сформульовано. Пояснення концепції локальної області видимості і структурування коду — правильне.

□ **Ймовірність ШІ-відповіді:** середня (50%) — граматично гладко, але цілком можливо, що це власний виклад.

★ **Оцінка: 4 / 5** (із урахуванням -60% за можливу допомогу ШІ → $5 \times 0.8 = 4$)

5.

“_”

✗ **Коментар:** відповіді немає. Можна було вказати, наприклад, на юніт-тестування (`pytest`, `unittest`) або тестування на контрольних наборах даних.

★ **Оцінка: 0 / 5**

■ Підсумок оцінювання

№	Питання	Оцінка
1	Причина створення окремої функції	5
2	Модифікація для довільної кількості радіусів	5
3	Оптимізація великої матриці	0
4	Перевага вкладеної функції	4
5	Перевірка без введення/виведення	0
Середня		(5+5+0+4+0)/5 =
оцінка		2.8 / 5

[Zагальний висновок]

Ти чудово орієнтуєшся у структурі програми, функціональній декомпозиції та логіці роботи коду.

Для підвищення оцінки до 4.5–5 варто:

- розібратись із **оптимізацією обробки великих масивів у NumPy** (наприклад, `axis`, `apply_along_axis`, `set`, `np.unique` з параметрами),
 - опанувати **основи тестування програм**.
-

Хочеш, щоб я сформулював **2–3 додаткових промпти для продовження самоаналізу** (як вимагає завдання 3)?

Вы сказали:

Дай відповіді на 3 та 5 відкриті питання

ChatGPT сказала:

Звісно 

Ось приклади **розгорнутих і логічно обґрунтованих відповідей** на відкриті питання №3 і №5, які продовжують твій стиль відповідей — коротко, технічно, без надмірної формальності:

3. Оптимізація функції `matrix18_process(data)` для великої матриці (10000×10000)

Якщо матриця дуже велика, перевірка унікальності кожного стовпця (`len(np.unique(data[:, j])) == m`) виконується повільно, бо створює новий масив для кожного стовпця.

Щоб пришвидшити роботу, можна:

1. **Векторизувати обчислення** — уникнути циклу `for`, наприклад:


```
unique_mask = np.apply_along_axis(lambda col: len(np.unique(col)) == data.shape[0], 0, data)
count_unique_cols = np.sum(unique_mask)
```
2. або навіть швидше — через булеві маски без явного циклу в Python.


```
або
```
3. **Використати множини з NumPy функціями**, але застосовувати їх блоками (по кілька стовпців одразу) — щоб зменшити кількість викликів функцій.
4. **Розпаралелити перевірку стовпців** (наприклад, через модуль `multiprocessing`), якщо потрібно обробити матрицю на кількох ядрах процесора.
5. **Зменшити розмір даних** (наприклад, з `int64` до `int8` або `int16`), якщо це допустимо — тоді NumPy зможе швидше обробляти пам'ять.

- ✓ Таким чином, головна ідея — **мінімізувати Python-цикли і максимально передати роботу внутрішнім векторизованим методам NumPy.**
-

5. Перевірка коректності програми без введення/виведення (автоматично)

Програму можна перевірити **автоматично**, використовуючи **модульне тестування (unit tests)**.

Наприклад, створюються тести у файлі `test_program.py` з використанням бібліотеки `pytest` або `unittest`:

```
import numpy as np
from circle_module import CircleS, CircleS_list
from task_matrix18 import task_matrix18

def test_circleS():
    assert CircleS(1) == 3.14
    assert CircleS(0) == 0
    assert round(Circles(2.5), 2) == 19.63

def test_matrix18_unique_columns():
    data = np.array([[1, 2, 3], [4, 2, 5], [6, 7, 8]])
    count, _ = task_matrix18.__defaults__[0](data) # або прямий виклик
matrix18_process(data)
    assert isinstance(count, int)
    assert count > 0
```

Так можна **автоматично перевіряти логіку обчислень**, не вводячи нічого вручну і не використовуючи `input()` / `print()`.

- ✓ Це дозволяє швидко знайти помилки після змін у коді й гарантує стабільність результатів.
-

Хочеш, я об'єднаю всі **10 відкритих відповідей (твої + мої)** у готовий структурований блок для вставлення у звіт (у розділ «Додаток В»)?

Вы сказали:

У 3 тестовому питанні ти навів серед прикладів таку бібліотеку, як `random`.

Для чого і в якій мові вона використовується?

ChatGPT сказала:

Дуже гарне уточнююче питання 

Бібліотека `random` — це **стандартний модуль мови програмування Python**, який використовується для генерації випадкових чисел.

◆ Призначення бібліотеки `random`

Вона надає інструменти для створення:

- випадкових **дійсних чисел** (`random.random()`, `uniform(a, b)`),
 - випадкових **цілих чисел** (`randint(a, b)`),
 - випадкового **вибору елемента** зі списку (`choice(seq)`),
 - **перемішування** елементів (`shuffle(seq)`),
 - і навіть **вибірки без повернення** (`sample(population, k)`).
-

◆ Приклад використання у Python

```
import random

# Випадкове ціле число від -10 до 10
x = random.randint(-10, 10)

# Випадкове число з діапазону [0, 1)
y = random.random()

# Випадковий вибір із списку
color = random.choice(['red', 'green', 'blue'])

print(x, y, color)
```

◆ Контекст у твоїй роботі

У твоїй програмі **аналогічна функціональність** реалізується через `numpy.random.randint()` — тобто генерація випадкових чисел для створення нових рядків у матриці.

Тож, якби замість `numpy` ти використовувала стандартну бібліотеку Python, саме `random` виконувала б ту ж роль.

◆ Підсумок:

`random` — це **Python-бібліотека для генерації випадкових чисел**, і вона використовується у випадках, коли потрібно внести **елемент випадковості** (наприклад, при створенні випадкових даних, симуляцій або тестування алгоритмів).