МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Український державний університет імені Михайла Драгоманова

Факультет математики, інформатики та фізики

Кафедра інформаційних технологій та програмування

**Звіт**

з лабораторної роботи №9

«Лінійні алгоритми мовою Python»

з дисципліни «Програмування»

Виконав:

студент ІІІ курсу групи 31І

Бритвич О.В.

Перевірила:

викладач Устименко О.Б.

Оцінка\_\_\_\_\_\_\_\_\_\_\_

Дата \_\_\_\_\_\_\_\_\_\_\_\_\_

Київ – 2024

**Зміст**

[Мета роботи 3](#_Toc150170352)

[1 Постановка задачі 4](#_Toc150170353)

[2 Основна частина 4](#_Toc150170354)

[2.1 Опис вхідних та вихідних даних 4](#_Toc150170355)

[2.1 Блок-схема 8](#_Toc150170356)

[Висновки 9](#_Toc150170357)

[Список літератури 10](#_Toc150170358)

[Додатки 11](#_Toc150170359)

# Мета роботи

Мета лабораторної роботи – скласти програму мовою Python, яка реалізує класичний алгоритм сортування стовпців двовимірного масиву методом обміну. Розмірність масиву та всі елементи генеруються за допомогою випадкових чисел.

# 1 Постановка задачі

1. Задача №1.

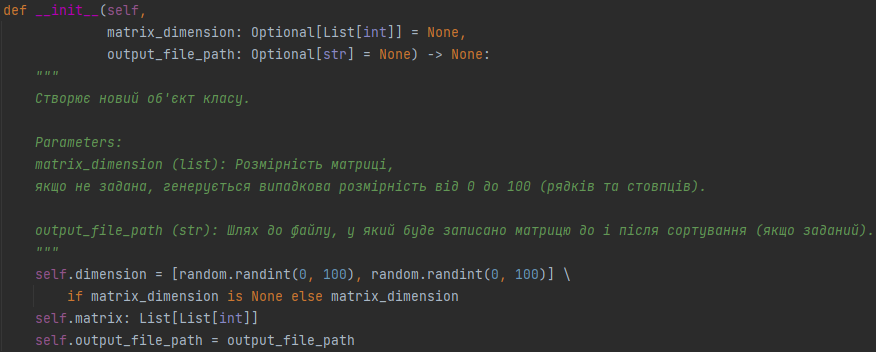
Скласти програму мовою Python, яка реалізує класичний алгоритм сортування стовпців двовимірного масиву методом обміну.

# 2 Основна частина

Задача №1

При написанні цієї програми використовувався стиль написання ООП, був створений основний клас "TDCBSI (Two Dimensional Column Bubble Sort Implementation)", в якому були розміщені всі необхідні сабметоди, далі перелік цих методів:

1. "\_\_init\_\_()" – конструктор класу, створює новий об’єкт класу.



1. "\_bubble\_column\_sort()" – відповідає за сортування матриці (двомірного масиву).

**Аналіз складності:**

1. Цикл по стовпцях (for col in range(cols)):

Цей цикл виконується cols разів (де cols - кількість стовпців).

1. Зовнішній цикл для бульбашкового сортування (for i in range(rows)):

Цей цикл виконується rows разів (де rows - кількість рядків).

1. Внутрішній цикл для бульбашкового сортування (for j in range(0, rows - i - 1)):

В гіршому випадку цей цикл виконується приблизно rows разів на початку і зменшується на кожну ітерацію зовнішнього циклу.

**Оцінка часу виконання:**

Для кожного стовпця:

1. Зовнішній цикл виконується rows разів.
2. Внутрішній цикл для кожної ітерації зовнішнього циклу в середньому виконується rows/2 разів (грубо кажучи).

Таким чином, час виконання для одного стовпця можна оцінити як:

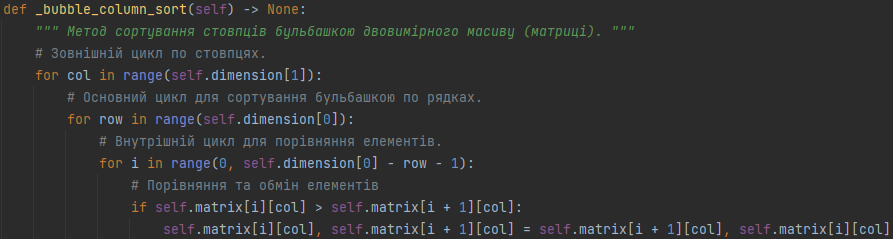
𝑂(𝑟𝑜𝑤𝑠×𝑟𝑜𝑤𝑠)=𝑂(𝑟𝑜𝑤𝑠2)O(rows×rows)=O(rows2)

Оскільки таких стовпців cols, то загальна часова складність алгоритму буде:

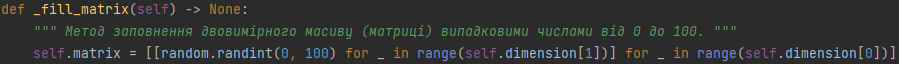
𝑂(𝑐𝑜𝑙𝑠×𝑟𝑜𝑤𝑠2)O(cols×rows2)

**Просторова складність:**

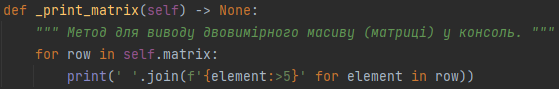
Просторова складність даного алгоритму є: 𝑂(1), оскільки алгоритм сортує масив на місці без використання додаткової пам'яті пропорційної розміру вхідних даних.



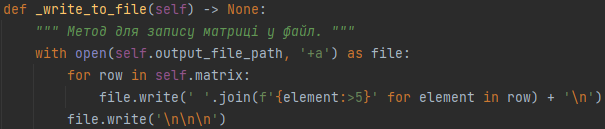
1. "\_fill\_matrix()" – відповідає за заповнення матриці (двомірного масиву) випадковими числами від 0 до 100.



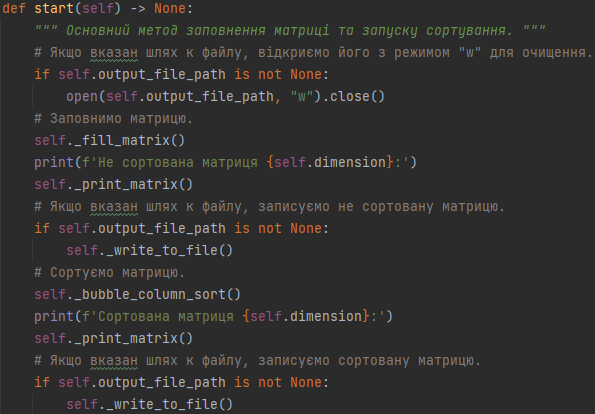
1. "\_print\_matrix()" – відповідає за вивід матриці (двомірного масиву) у консоль.



1. "\_write\_to\_file()" – відповідає за запис матриці у файл.



1. "start()" – відповідає за почерговий запуск усіх методів.



## 2.1 Опис вхідних та вихідних даних

Вхідні дані:

Якщо потрібно вказати точний розмір матриці, вказуємо "matrix\_dimension". Наприклад, якщо вказати "matrix\_dimension" як [9, 10], тоді розмір матриці буде 9 рядків на 10 стовбців.

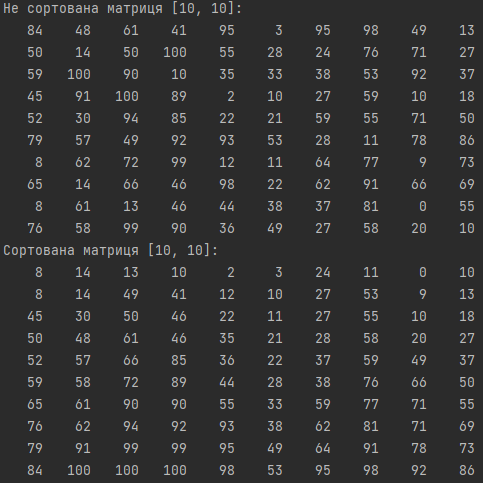


Якщо потрібно, щоб при виконанні програми був створений файл із записаними сортованої та не сортованої матрицями - вказуємо шлях до файлу у "output\_file\_path".



Вихідні дані:

Якщо "output\_file\_path" задано, після виконання файлу, буду створено файл із сортованою та не сортованою матрицями. Також у консолі будуть виведенні обидві матриці.



## 2.1 Блок-схема

Початок

Виводимо сортовану матрицю у консоль

Створення обєкту TDCBSI

Ні

Кінець

Записуємо сортовану матрицю у файл

Був задан

output\_file\_path?

Так / Ні

Так

Записуємо не сортовану матрицю у файл

Був задан

output\_file\_path?

Так / Ні

Так

Запуск методу

\_bubble\_column\_sort()

Ні

Виводимо не сортовану матрицю у консоль

Ні

Запуск методу

\_fill\_matrix()

Так

Створюємо / очищаємо вихідний файл.

Був задан

output\_file\_path?

Так / Ні

Запуск методу start()

Визначення розмірності матриці

# Висновки

У даній лабораторній роботі була створена програма мовою Python, яка реалізує класичний алгоритм сортування стовпців двовимірного масиву методом обміну.

Програма працює коректно. Були розроблені розширенні функції, а саме: можливість вказувати розмірність двомірного масиву (матриці), та вказувати шлях до файлу, у котрий будуть виводитися сортована та не сортована матриці.

# Список літератури

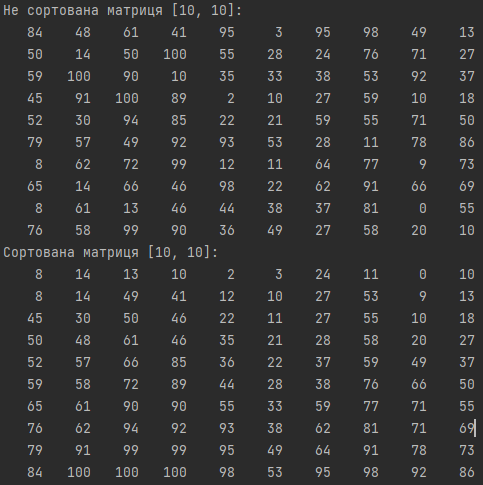
1. Вивчаємо Python, В двох томах. 5-е видання. Марк Лутц (комплект з двох книг) – [Режим доступу] : <https://leader-books.com.ua/komp-yuterna-literatura/vivchayemo-python-v-dvoh-tomah-5-e-vidannya-mark-lutc-komplekt-z-dvoh-knig?gclid=CjwKCAiAgeeqBhBAEiwAoDDhnxzTW_ZFWceIMmEQLdDM0XPtB2uG9tUdKMSjvmw6BX7zHEoS49h0gxoC_skQAvD_BwE>
2. Форум з технічних питань – [Режим доступу] : <https://stackoverflow.com/>

# Додатки

Текст програми №1

import random  
from typing import List, Optional  
  
  
class TDCBSI:  
 *"""  
 TDCBSI - Two Dimensional Column Bubble Sort Implementation.  
 Даний клас реалізує класичний алгоритм сортування стовпців двовимірного масиву методом обміну.  
 Розмірність масиву та всі елементи генеруються за допомогою випадкових чисел.  
 """* def \_\_init\_\_(self,  
 matrix\_dimension: Optional[List[int]] = None,  
 output\_file\_path: Optional[str] = None) -> None:  
 *"""  
 Створює новий об'єкт класу.  
  
 Parameters:  
 matrix\_dimension (list): Розмірність матриці,  
 якщо не задана, генерується випадкова розмірність від 0 до 100 (рядків та стовпців).  
  
 output\_file\_path (str): Шлях до файлу, у який буде записано матрицю до і після сортування (якщо заданий).  
 """* self.dimension = [random.randint(0, 100), random.randint(0, 100)] \  
 if matrix\_dimension is None else matrix\_dimension  
 self.matrix: List[List[int]]  
 self.output\_file\_path = output\_file\_path  
  
 def \_bubble\_column\_sort(self) -> None:  
 *""" Метод сортування стовпців бульбашкою двовимірного масиву (матриці). """* # Зовнішній цикл по стовпцях.  
 for col in range(self.dimension[1]):  
 # Основний цикл для сортування бульбашкою по рядках.  
 for row in range(self.dimension[0]):  
 # Внутрішній цикл для порівняння елементів.  
 for i in range(0, self.dimension[0] - row - 1):  
 # Порівняння та обмін елементів  
 if self.matrix[i][col] > self.matrix[i + 1][col]:  
 self.matrix[i][col], self.matrix[i + 1][col] = self.matrix[i + 1][col], self.matrix[i][col]  
  
 def \_fill\_matrix(self) -> None:  
 *""" Метод заповнення двовимірного масиву (матриці) випадковими числами від 0 до 100. """* self.matrix = [[random.randint(0, 100) for \_ in range(self.dimension[1])] for \_ in range(self.dimension[0])]  
  
 def \_print\_matrix(self) -> None:  
 *""" Метод для виводу двовимірного масиву (матриці) у консоль. """* for row in self.matrix:  
 print(' '.join(f'{element:>5}' for element in row))  
  
 def \_write\_to\_file(self) -> None:  
 *""" Метод для запису матриці у файл. """* with open(self.output\_file\_path, '+a') as file:  
 for row in self.matrix:  
 file.write(' '.join(f'{element:>5}' for element in row) + '\n')  
 file.write('\n\n\n')  
  
 def start(self) -> None:  
 *""" Основний метод заповнення матриці та запуску сортування. """* # Якщо вказан шлях к файлу, відкриємо його з режимом "w" для очищення.  
 if self.output\_file\_path is not None:  
 open(self.output\_file\_path, "w").close()  
 # Заповнимо матрицю.  
 self.\_fill\_matrix()  
 print(f'Не сортована матриця {self.dimension}:')  
 self.\_print\_matrix()  
 # Якщо вказан шлях к файлу, записуємо не сортовану матрицю.  
 if self.output\_file\_path is not None:  
 self.\_write\_to\_file()  
 # Сортуємо матрицю.  
 self.\_bubble\_column\_sort()  
 print(f'Сортована матриця {self.dimension}:')  
 self.\_print\_matrix()  
 # Якщо вказан шлях к файлу, записуємо сортовану матрицю.  
 if self.output\_file\_path is not None:  
 self.\_write\_to\_file()  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 TDCBSI(output\_file\_path='./output.txt').start()

Результат роботи програми №1



Поява нового файлу з матрицями

