МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Кафедра інформаційних систем та мереж



Лабораторна робота №5

з дисципліни Спеціалізовані мови програмування

на тему

Розробка ASCII ART генератора для візуалізації 3D-фігур

Виконав:

студент групи РІ-21сп

Андрій ПІГУЛЯК

Львів – 2024

**Мета виконання лабораторної роботи:** Cтворення додатка для малювання 3D-фігур у ASCII-арті на основі об’єктно - орієнтованого підходу та мови Python

**Хід роботи**

1. Спроєктувати структуру класів для генератора 3D ASCII-арту з основними компонентами, атрибутами та методами.
2. Реалізувати методи для введення користувача: вибір фігури та її параметрів (розмір, кольори).
3. Визначити структури даних для представлення 3D-фігури (списки, матриці тощо).
4. Написати метод для перетворення 3D-представлення фігури в 2D для ASCII-арту.
5. Реалізувати метод для відображення 2D-представлення 3D-фігури у вигляді ASCII-арту.
6. Створити зрозумілий інтерфейс користувача (CLI або GUI) для взаємодії з програмою.
7. Додати методи для масштабування та зміщення фігури.
8. Забезпечити вибір кольорів для різних частин фігури.
9. Реалізувати збереження 3D ASCII-арту у текстовий файл.
10. Додати розширені функції: тінь, освітлення, ефекти перспективи.
11. **Опис і структура програми:**
12. **Реалізація базового класу Shape:**
13. Базовий клас визначає основні методи:
    * get\_vertices — для отримання координат вершин фігури.
    * get\_edges — для отримання списку ребер, які з'єднують вершини.
    * draw — для візуалізації фігури у форматі ASCII-арту.
14. Клас включає підтримку обертання (rotate), проєктування (project) та малювання ліній (draw\_line).
15. **Створення підкласів Cube і Pyramid:**

* Cube: представляє куб з визначеними вершинами і ребрами.
* Pyramid: представляє піраміду з вершиною та основою у вигляді прямокутника.

1. **Додано підтримку кольорів через JSON-файл:**

* Конфігураційний файл config.py містить набір кольорів, які доступні для вибору користувачем.

1. **Розроблено модульну структуру:**

* Класи фігур зберігаються в директорії shapes.
* Утиліти, такі як завантаження кольорів, обробка файлів.

5. **Реалізація функціональності збереження ASCII-арту до файлу:**

* Користувач може зберегти зображення фігури без кольорового оформлення в текстовий файл.

1. **Функціонал:**
2. **Візуалізація фігур**: Програма відображає фігури в ASCII-арт форматі з можливістю обертання, зміни масштабу та кольору.
3. **Підтримка кольорового оформлення**: Користувач може вибрати один із доступних кольорів для фігури.
4. **Система обертання**: Реалізовано обертання фігури в просторі навколо осей X та Y.
5. **Збереження у файл**: ASCII-арт без кольорів можна зберегти у текстовий файл.

**3. Код програми**

**Основні функціональні модулі:**

1. shape.py   
   Базовий клас відповідає за генерацію:

import numpy as np  
  
class Shape:  
 def \_\_init\_\_(self, size, color):  
 self.size = size  
 self.color = color  
 self.reset\_color = "\033[0m"  
  
 def get\_vertices(self):  
 raise NotImplementedError("Метод get\_vertices() повинен бути реалізований у підкласах.")  
  
 def get\_edges(self):  
 raise NotImplementedError("Метод get\_edges() повинен бути реалізований у підкласах.")  
  
 def draw(self, angle\_x, angle\_y):  
 vertices = self.get\_vertices()  
 edges = self.get\_edges()  
  
 rotated\_vertices = [self.rotate(v, angle\_x, angle\_y) for v in vertices]  
 projected\_vertices = [self.project(v) for v in rotated\_vertices]  
  
 canvas = [[' ' for \_ in range(40)] for \_ in range(20)]  
 for start, end in edges:  
 x1, y1 = projected\_vertices[start]  
 x2, y2 = projected\_vertices[end]  
 char = '+' if isinstance(self, Cube) and start in {0, 4} and end in {1, 5} else '#'  
 self.draw\_line(canvas, x1, y1, x2, y2, char)  
  
 return "\n".join("".join(line) for line in canvas)

1. **art\_file\_manager.py**  
   Додає можливість зберігати ASCII-арт:

import os  
  
def save\_ascii\_art\_to\_file(ascii\_art):  
 if not os.path.exists("ARTTXT"):  
 os.makedirs("ARTTXT")  
  
 while True:  
 filename = input("Введіть ім'я файлу для збереження (без розширення): ").strip()  
 if filename:  
 filepath = os.path.join("ARTTXT", f"{filename}.txt")  
  
 try:  
 with open(filepath, "w") as file:  
 file.write(ascii\_art)  
 print(f"Ваш ASCII-арт збережено у файл: {filepath}")  
 break  
 except Exception as e:  
 print(f"Помилка при збереженні файлу: {e}. Спробуйте ще раз.")  
 else:  
 print("Ім'я файлу не може бути порожнім. Будь ласка, введіть ім'я файлу.")

1. **user\_color.py**  
   Реалізує функцію збереження ASCII-арту у файл:   
   def load\_colors\_from\_json():  
    config\_path = os.path.join(os.path.dirname(\_\_file\_\_), '..', 'config', 'config.json')  
    try:  
    with open(config\_path, 'r') as file:  
    data = json.load(file)  
    colors = data.get("colors", {})  
    default\_color\_name = data.get("default\_color", "white")  
    color = colors.get(default\_color\_name, colors.get("white", "\033[37m"))  
    return colors, color  
    except FileNotFoundError:  
    print("Файл config.json не знайдено. Використовується колір за замовчуванням (білий).")  
    return {"white": "\033[37m"}, "\033[37m"
2. **user\_input.py**Фрагмент обробки користувацького вводу:

def get\_shape\_type():  
 shape\_type = input("Яку фігуру ви хочете побудувати? (cube/pyramid): ").strip().lower()  
 while shape\_type not in ["cube", "pyramid"]:  
 print("Невірний вибір. Будь ласка, виберіть 'cube' або 'pyramid'.")  
 shape\_type = input("Яку фігуру ви хочете побудувати? (cube/pyramid): ").strip().lower()  
 return shape\_type

**4.** **Приклад роботи:**

**Приклад 1:** Настройка слова для ASCII-арту

A white background with black text

Description automatically generated

Рис. 1 Вибір слова «Andrew»

**Приклад 2:** Налаштування куба

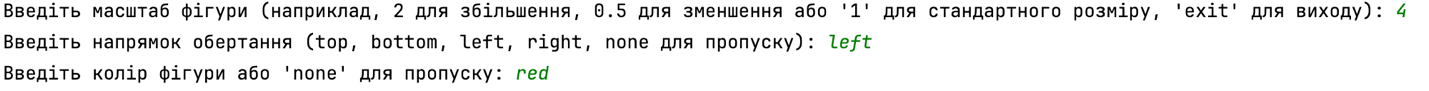


Рис. 2 Результат налаштування

**Приклад 3:** Відображення куба арта

A group of red hashtags

Description automatically generated

Рис. 3 Результат куба

**Приклад 4:** Зміна налаштувань

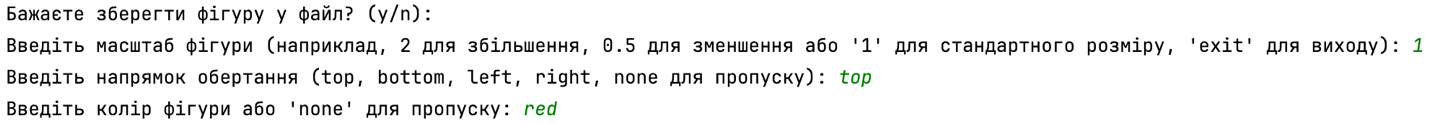
****

Рис. 4Результат зміни налаштувань куба

**Приклад 5:** Фінанльний результат куба

A group of red symbols

Description automatically generated  
Рис. 5 Результат куба

**Приклад 6:** Збереження куба

A close up of a number

Description automatically generated

Рис. 6 Збереження куба в «txt» файл

**Приклад 7:** Збережений куб

**A group of black symbols

Description automatically generated**

Рис. 8 Збережений куб в «txt» файлі

**Приклад 8:** Побудований піраміда

A black and white image of a letter

Description automatically generated

Рис. 8 Збережна піраміда в «txt» файлі

**Висновок:** Під час виконання роботи було розроблено програму, яка дозволяє відображати 3D-фігури в текстовому форматі з підтримкою інтерактивного управління. Завдяки використанню об'єктно-орієнтованого підходу та модульної структури код вийшов гнучким і легко розширюваним.