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»

Виконала студентка

групи РІ-32

Копейка Х.А.

Прийняв:

Титиш О.І.

Львів – 2024

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

**План роботи**

Завдання 1: Проектування класів

Розробіть структуру класів для вашого генератора 3D ASCII-арту. Визначте основні компоненти, атрибути та методи, необхідні для програми.

Завдання 2: Введення користувача

Створіть методи у межах класу для введення користувача та вказання 3D-фігури, яку вони хочуть намалювати, та її параметрів (наприклад, розмір, кольори).

Завдання 3: Представлення фігури

Визначте структури даних у межах класу для представлення 3D-фігури. Це може включати використання списків, матриць або інших структур даних для зберігання форми фігури та її властивостей.

Завдання 4: Проектування з 3D в 2D

Реалізуйте метод, який перетворює 3D-представлення фігури у 2D-представлення, придатне для ASCII-арту.

Завдання 5: Відображення ASCII-арту

Напишіть метод у межах класу для відображення 2D-представлення 3D-фігури як ASCII-арту. Це може включати відображення кольорів і форми за допомогою символів ASCII.

Завдання 6: Інтерфейс, зрозумілий для користувача

Створіть зручний для користувача командний рядок або графічний інтерфейс користувача (GUI) за допомогою об'єктно-орієнтованих принципів, щоб дозволити користувачам спілкуватися з програмою.

Завдання 7: Маніпуляція фігурою

Реалізуйте методи для маніпулювання 3D-фігурою, такі масштабування або зміщення, щоб надавати користувачам контроль над її виглядом.

Завдання 8: Варіанти кольорів

Дозвольте користувачам вибирати варіанти кольорів для їхніх 3D ASCII-арт-фігур. Реалізуйте методи для призначення кольорів різним частинам фігури.

Завдання 9: Збереження та експорт

Додайте функціональність для зберігання згенерованого 3D ASCII-арту у текстовий файл

Завдання 10: Розширені функції

Розгляньте можливість додавання розширених функцій, таких як тінь, освітлення та ефекти перспективи, для підвищення реалізму 3D ASCII-арту.

**Код програми**

AsciiArtGenerator.py

from colorama import Back, Style  
  
class AsciiArtGenerator:  
 def add\_shadow(self, ascii\_art):  
 shadow = []  
 for line in ascii\_art:  
 shadow.append(Back.BLACK + line + Style.RESET\_ALL)  
 return ascii\_art + [''] + shadow  
  
 def add\_lighting(self, ascii\_art):  
 lighting = []  
 for line in ascii\_art:  
 lighting.append(Back.WHITE + line + Style.RESET\_ALL)  
 return lighting

Shape3D.py

from colorama import Fore, Style  
class Figure3D:  
 @staticmethod  
 def is\_appropriate\_character(character):  
 return len(character) == 1  
  
class Cube(Figure3D):  
 def \_\_init\_\_(self, length, character, color\_position):  
 self.length = length  
 self.character = character  
 self.color\_position = color\_position  
  
 def get\_3d\_representation(self, scale=1.0):  
 l = int(self.length \* scale)  
 color = self.get\_color()  
 return [  
 f"{color} +{'-' \* l}+{Style.RESET\_ALL}",  
 f"{color} /{' ' \* l}/ |{Style.RESET\_ALL}",  
 f"{color} +{'-' \* l}+ +{Style.RESET\_ALL}",  
 f"{color} |{' ' \* l}| /{Style.RESET\_ALL}",  
 f"{color} |{' ' \* l}|/{Style.RESET\_ALL}",  
 f"{color} +{'-' \* l}+{Style.RESET\_ALL}"  
 ]  
  
 def get\_2d\_representation(self):  
 color = self.get\_color()  
 return [  
 f"{color}{self.character \* self.length}{Style.RESET\_ALL}",  
 f"{color}{self.character} {self.character}{Style.RESET\_ALL}",  
 f"{color}{self.character \* self.length}{Style.RESET\_ALL}"  
 ]  
  
 def get\_color(self):  
 colors = [Fore.RED, Fore.GREEN, Fore.BLUE]  
 return colors[self.color\_position]

FileManager.py

class FileManager:  
 @staticmethod  
 def save\_to\_file(filename, content):  
 try:  
 with open(filename, "w") as file:  
 file.write(content)  
 print(f"Content saved to {filename}")  
 except PermissionError:  
 print("You do not have permission to write to the file!")  
 except FileNotFoundError:  
 print("The file does not exist!")

UserInput.py

from classes.Shape3D\_LAB5 import Figure3D  
  
  
def get\_character\_input():  
 while True:  
 character = input("Enter a character to represent in the shape: ")  
 if Figure3D.is\_appropriate\_character(character) is False:  
 print("You should have entered one character!")  
 else:  
 return character  
  
def get\_color\_position\_input(colors):  
 while True:  
 try:  
 color = int(input("Enter a number of color: "))  
 if color not in range(len(colors)):  
 print("You should have entered a color option which is available!")  
 else:  
 return color  
 except ValueError:  
 print("You should have entered an integer number!")  
  
def get\_length\_input():  
 while True:  
 try:  
 length = int(input("Enter a length: "))  
 if length <= 0:  
 print("You should have entered a length greater than 0!")  
 else:  
 return length  
 except ValueError:  
 print("You should have entered an integer number!")  
  
def get\_scale\_input():  
 while True:  
 try:  
 scale = float(input("Enter a scale for figure: "))  
 if scale <= 0:  
 print("You should have entered a scale greater than 0!")  
 else:  
 return scale  
 except ValueError:  
 print("You should have entered a float number!")

CLI.py

from classes.Shape3D\_LAB5 import Cube  
from BLL.AsciiArtGenerator\_LAB5 import AsciiArtGenerator  
from DAL.FileManager\_LAB5 import FileManager  
from Shared.UserInput\_LAB5 import get\_character\_input, get\_color\_position\_input, get\_length\_input, get\_scale\_input  
  
def display\_colors():  
 colors = ["Red", "Green", "Blue"]  
 for i, color in enumerate(colors):  
 print(f"{i}: {color}")  
  
def main():  
 is\_figure\_available = False  
 is\_3d\_representation\_available = False  
 representation\_3d\_file = "cube.txt"  
 colors = ["Red", "Green", "Blue"]  
 generator = AsciiArtGenerator()  
  
 while True:  
 print("1 - Create a cube")  
 print("2 - Display 2D")  
 print("3 - Save 3D")  
 print("4 - Add Shadow")  
 print("5 - Add Lighting")  
 print("0 - Exit")  
 option = str(input("Enter an option: "))  
  
 match option:  
 case "1":  
 character = get\_character\_input()  
 print("There are such colors available:")  
 display\_colors()  
 color\_position = get\_color\_position\_input(colors)  
 length = get\_length\_input()  
 scale = get\_scale\_input()  
 try:  
 figure = Cube(length, character, color\_position)  
 is\_figure\_available = True  
 representation\_3d = figure.get\_3d\_representation(scale=scale)  
 print("\n".join(representation\_3d))  
 is\_3d\_representation\_available = True  
 except ValueError as e:  
 print(e)  
 is\_figure\_available = False  
 case "2":  
 if is\_figure\_available:  
 representation\_2d = figure.get\_2d\_representation()  
 [print(item) for item in representation\_2d]  
 else:  
 print("There is no figure available!")  
 case "3":  
 if is\_3d\_representation\_available:  
 FileManager.save\_to\_file(representation\_3d\_file, "\n".join(representation\_3d))  
 else:  
 print("There is no figure available!")  
 case "4":  
 if is\_3d\_representation\_available:  
 shadow\_art = generator.add\_shadow(representation\_3d)  
 print("\n".join(shadow\_art))  
 else:  
 print("There is no figure available!")  
 case "5":  
 if is\_3d\_representation\_available:  
 lighting\_art = generator.add\_lighting(representation\_3d)  
 print("\n".join(lighting\_art))  
 else:  
 print("There is no figure available!")  
 case "0":  
 break  
 case \_:  
 print("Invalid option!")  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

**Результат виконання програми**

**Зображення, що містить текст, знімок екрана, Шрифт, число

Автоматично згенерований опис**

**Зображення, що містить текст, знімок екрана, Шрифт, схема

Автоматично згенерований опис** **Зображення, що містить текст, знімок екрана, Шрифт, число

Автоматично згенерований опис**

*Рисунок 1, 2, 3. Результати виконання*

Посилання на GitHub:

**Висновок:** Виконуючи лабораторну роботу, я створила високорівневий об'єктно-орієнтований генератор 3D ASCII-арту, який дозволяє користувачам проектувати, відображати та маніпулювати 3D-фігурами в ASCII-арті. Цей проект надав мені глибоке розуміння об'єктно-орієнтованого програмування і алгоритмів графіки, та сприяв творчому підходу до створення ASCII-арту.