Міністерство освіти і науки України

Національний університет «Львівська політехніка»

******

Звіт про виконання лабораторної роботи №5  
з дисципліни  
“Спеціалізовані мови програмування”  
на тему  
“Розробка ASCII ART генератора для візуалізації 3D-фігур ”

Виконала:  
студентка групи РІ-31,  
Пятницька Вікторія

Прийняв:

Щербак С.С.

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

**Хід 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:**

***base\_figure.py****:*

from abc import ABC, abstractmethod

class BaseFigure(ABC):

def \_\_init\_\_(self, size: int):

self.size = size

@abstractmethod

def get\_two\_d\_art(self):

pass

@abstractmethod

def get\_three\_d\_art(self):

pass

***cube.py:***

from lab5.classes.base\_figure import BaseFigure

class Cube(BaseFigure):

def \_\_init\_\_(self, side\_a: int):

super().\_\_init\_\_(side\_a)

self.side\_a = side\_a

def get\_two\_d\_art(self):

result = ""

for side\_a\_pixel in range(self.side\_a):

result += "\* "

for side\_b\_pixel in range(self.side\_a - 2):

if side\_a\_pixel % self.side\_a == 0 or side\_a\_pixel + 1 == self.side\_a:

result += "\* "

else:

result += " "

result += "\*\n"

return result

def get\_three\_d\_art(self):

result = ""

art\_size = int(self.side\_a + self.side\_a / 2)

side\_side\_size = int(self.side\_a / 1.5)

space\_counter = 0

for x in range(art\_size):

if x == 0:

result += "\* "

elif x < side\_side\_size:

result += "\* " + space\_counter \* " " + "\* "

space\_counter += 1

elif side\_side\_size <= x <= art\_size - side\_side\_size:

result += "\* " + (side\_side\_size - 2) \* " " + "\* "

space\_counter = 1

elif x < art\_size - 1:

result += space\_counter \* " " + "\* " + (art\_size - x - 2) \* " " + "\* "

space\_counter += 1

else:

result += space\_counter \* " " + "\* "

for y in range(self.side\_a - 2):

if x == 0 or x == side\_side\_size - 1 or x == art\_size - 1:

result += "\* "

else:

result += " "

result += "\*\n"

return result

***threeD\_art.py:***

import os

from lab5.classes.cube import Cube

from termcolor import colored

class ThreeDArtGenerator:

def \_\_init\_\_(self):

self.art = None

self.color = 'green'

self.direction = True

def choose\_shape(self):

while True:

try:

shape = input("Choose a shape (1 - Cube): ")

if shape == '1':

size = int(input("Enter size (minimum 3): "))

if size < 3:

print("Size must be at least 3.")

else:

self.art = Cube(size)

break

else:

print("Invalid choice. Try again.")

except ValueError:

print("Invalid input, please enter a number for the size and shifts.")

def change\_size(self):

while True:

try:

size = int(input("Enter a new size (minimum 3): "))

if size < 3:

print("Size must be at least 3.")

elif self.art:

self.art.size = size

break

else:

print("Please select a shape first.")

except ValueError:

print("Invalid input, please enter a number for the size.")

def get\_available\_colors(self):

return ['red', 'green', 'yellow', 'blue', 'magenta', 'cyan', 'white']

def choose\_color(self):

while True:

print("Available colors:")

for i, color in enumerate(self.get\_available\_colors()):

print(f"{i + 1}. {color}")

try:

color\_choice = int(input("Choose a color by number: "))

if 1 <= color\_choice <= len(self.get\_available\_colors()):

self.color = self.get\_available\_colors()[color\_choice - 1]

break

else:

print("Invalid choice, please select again.")

except ValueError:

print("Invalid input, please enter a number.")

def change\_direction(self):

self.direction = not self.direction

print("Direction successfully changed.")

def convert\_to\_2d(self):

if self.art:

return self.art.get\_two\_d\_art()

else:

print("Please select a shape first.")

return ""

def display\_ascii\_art(self):

if self.art:

projection = self.convert\_to\_2d()

if projection:

print("2D representation of the shape:")

for line in projection.splitlines():

print(colored(line, self.color))

def get\_art(self) -> str:

if not self.art:

return ""

art\_lines = self.art.get\_three\_d\_art().splitlines()

return "\n".join([colored(line, self.color) for line in art\_lines])

def print\_art(self):

if self.art:

print(self.get\_art())

else:

print("Please select a shape first.")

def save\_to\_file(self, filename="ascii\_art"):

if self.art:

filename += '.txt'

while os.path.exists(filename):

choice = input(f"The file '{filename}' already exists. Overwrite it? (Enter 'yes' to overwrite, any other character to cancel): ").strip().lower()

if choice == 'yes':

break

else:

filename = input("Enter a new filename: ").strip() + '.txt'

with open(filename, 'w') as file:

file.write(self.get\_art())

print(f"ASCII art saved to {filename}")

else:

print("No art to save. Please select a shape first.")

def run(self):

while True:

print("\nMain Menu")

print("1. Generate 3D Art")

print("2. Exit")

choice = input("Enter your choice (1-2): ").strip()

if choice == '1':

self.choose\_shape()

self.choose\_color()

self.print\_art()

converted\_to\_2d = False

while not converted\_to\_2d:

convert\_choice = input("Do you want to convert this 3D art to 2D? (Enter 'yes' or any other character for no): ").strip().lower()

if convert\_choice == 'yes':

self.display\_ascii\_art()

converted\_to\_2d = True

else:

print("Skipping 2D conversion.")

break

saved\_to\_file = False

while not saved\_to\_file:

save\_choice = input("Do you want to save the ASCII art to a file? (Enter 'yes' or any other character for no): ").strip().lower()

if save\_choice == 'yes':

filename = input("Enter filename: ")

self.save\_to\_file(filename)

saved\_to\_file = True

else:

print("Skipping save.")

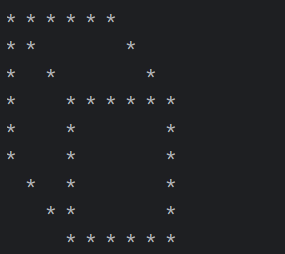
break

elif choice == '2':

break

else:

print("Invalid choice, please select a number between 1 and 2.")

**

*Рис.1.Приклад виконання програми*

**Висновки:** внаслідок виконання лабораторної роботи я створила для малювання 3D-фігур у ASCII-арті на основі об’єктно - орієнтованого підходу та мови Python.