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:

ст. гр. РІ-31

ЛАЗАР В.С.

Прийняв:

ЩЕРБАК С.С.

Львів-2024

**Мета роботи:**

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

**Хід роботи:**

**Завдання 1: Проектування класів**

Розробити структуру класів для генератора 3D ASCII-арту. Визначити основні компоненти, атрибути та методи, необхідні для програми.

**Завдання 2: Введення користувача**

Створити 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**

Визначити 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**

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

**Завдання 5: Відображення ASCII-арту**

Написати 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**

Створити 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**

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

**Завдання 8: Варіанти кольорів**

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

**Завдання 9: Збереження та експорт**

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

**Завдання 10: Розширені функції**

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

**Код програми:**

"""The console module of this lab work"""

from Data.Shared.classes.validators import Validators

class Console:

"""The console class of this lab work"""

instance = None

def \_\_new\_\_(cls):

if cls.instance is None:

cls.instance = super(Console, cls).\_\_new\_\_(cls)

return cls.instance

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

self.main()

self.ascii = None

def main(self):

"""The main menu of this lab work"""

self.create\_shape()

while True:

prompt = input("1 - Create a new shape\n"

"2 - Change shape's size\n"

"3 - Move shape\n"

"4 - Change shape's color\n"

"5 - Print shape\n"

"6 - Export shape\n"

"Your choice: ")

if not Validators.main\_prompt(prompt, self):

return

def create\_shape(self):

"""The UI for the shape creation"""

while True:

shape\_prompt = input("Select shape:\n"

"1 - Cube\n"

"2 - Pyramid\n"

"3 - Sphere\n"

"Your choice: ")

try:

Validators.validate\_value(shape\_prompt, 0, 4)

except ValueError as e:

print(e)

continue

size\_prompt = input("Enter shape's size (5-100): ")

try:

Validators.validate\_value(size\_prompt, 4, 101)

size = int(size\_prompt)

except ValueError as e:

print(e)

continue

self.ascii = Validators.create\_shape(shape\_prompt, size)

print("Shape was created successfully")

return

def change\_size(self):

"""The UI for the shape's size changing"""

prompt = input("Change shape's new size (5-100): ")

try:

Validators.validate\_value(prompt, 4, 101)

size = int(prompt)

except ValueError as e:

print(e)

return

self.ascii.change\_size(size)

print("Shape's size was changed successfully")

def move\_shape(self):

"""The UI for moving the shape"""

prompt\_x = input("Move shape by (x) (-10-10): ")

try:

Validators.validate\_value(prompt\_x, -10, 10)

x = int(prompt\_x)

except ValueError as e:

print(e)

return

prompt\_y = input("Move shape by (y) (-10-10): ")

try:

Validators.validate\_value(prompt\_y, -10, 10)

y = int(prompt\_y)

except ValueError as e:

print(e)

return

prompt\_z = input("Move shape by (z) (-10-10): ")

try:

Validators.validate\_value(prompt\_z, -10, 10)

z = int(prompt\_z)

except ValueError as e:

print(e)

return

self.ascii.move(x, y, z)

print("Shape was moved successfully")

def change\_color(self):

"""The UI for changing the shape's color"""

color\_prompt = input("Choose the color of your shape:\n"

"1 - Red\n"

"2 - Green\n"

"3 - Yellow\n"

"4 - Blue\n"

"5 - Magenta\n"

"6 - Cyan\n"

"7 - Light gray\n"

"8 - Random\n"

"0 - Default\n"

"Your choice: ")

try:

Validators.validate\_color(color\_prompt, self)

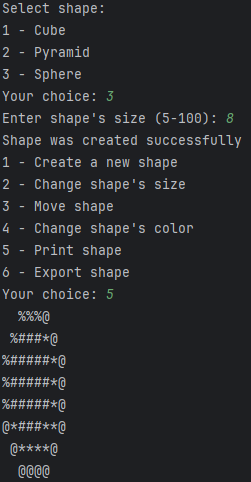
except ValueError as e:

print(e)

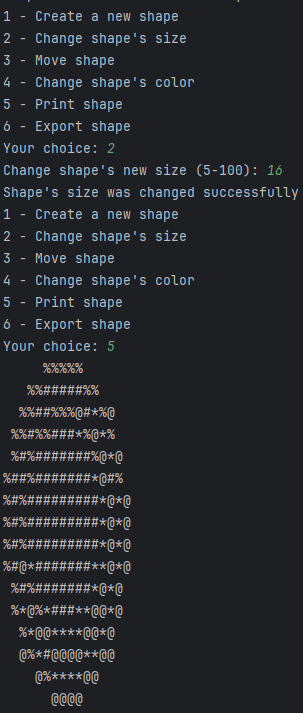
return

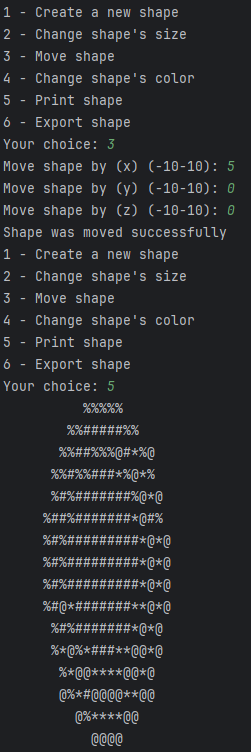
print("Shape's color changed successfully")

На рис. 1-4 зображено результат виконання програми:

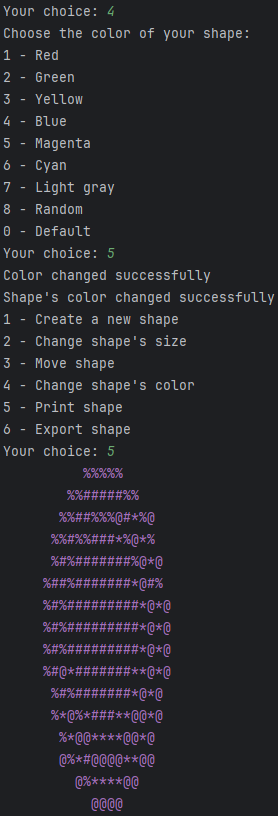


*Рис. 1. Приклад генерації та виведення ASCII-арту*

*  
Рис. 2. Приклад зміни розміру ASCII-арту*



*Рис. 3. Приклад зміни позиції арту*



*Рис. 4. Приклад зміни кольору арту*

Посилання на Github: [PaperGlit/Python\_Lab\_5](https://github.com/PaperGlit/Python_Lab_5)

**Висновок:**

Виконуючи ці завдання, я створив високорівневий об'єктно-орієнтований генератор 3D ASCII-арту, який дозволить користувачам проектувати, відображати та маніпулювати 3D-фігурами в ASCII-арті. Цей проект надав мені глибоке розуміння об'єктно-орієнтованого програмування і алгоритмів графіки, сприяв творчому підходу до створення ASCII-арту.