МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Кафедра інформаційних систем та мереж

Лабораторна робота №3

з дисципліни

СПЕЦІАЛІЗОВАНІ МОВИ ПРОГРАМУВАННЯ

на тему

Розробка ASCII ART генератора для візуалізації текстових даних

Виконала:

ст. гр. РІ-31

Оксана ЛЕСЮК

Прийняв:

Сергій Сергійович ЩЕРБАК

Львів-2024

**Мета роботи:** Створення додатка Генератора ASCII-арту.

**Завдання:**

План роботи

Завдання 1: Введення користувача Створіть Python-програму, яка приймає введення користувача для слова або фрази, яку треба перетворити в ASCII-арт.

Завдання 2: Бібліотека ASCII-арту Інтегруйте бібліотеку ASCII-арту (наприклад, pyfiglet або art) у вашу програму для генерації ASCII-арту з введення користувача

Завдання 3: Вибір шрифту Дозвольте користувачам вибирати різні стилі шрифтів для свого ASCII-арту. Надайте список доступних шрифтів та дозвольте їм вибрати один.

Завдання 4: Колір тексту Реалізуйте опцію вибору користувачем кольору тексту для їхнього ASCII-арту. Підтримуйте основний вибір кольорів (наприклад, червоний, синій, зелений).

Завдання 5: Форматування виводу Переконайтеся, що створений ASCII-арт правильно відформатований та вирівнюється на екрані для зручності читання.

Завдання 6: Збереження у файл Додайте функціональність для збереження створеного ASCII-арту у текстовому файлі, щоб користувачі могли легко завантажувати та обмінюватися своїми творіннями.

Завдання 7: Розмір ARTу Дозвольте користувачам вказувати розмір (ширина і висота) ASCII-арту, який вони хочуть створити. Масштабуйте текст відповідно.

Завдання 8: Вибір символів Дозвольте користувачам вибирати символи, які вони хочуть використовувати для створення ASCII-арту (наприклад, '@', '#', '\*', тощо).

Завдання 9: Функція попереднього перегляду Реалізуйте функцію попереднього перегляду, яка показує користувачам попередній перегляд їхнього ASCII-арту перед остаточним збереженням.

Завдання 10: Інтерфейс, зрозумілий для користувача Створіть зручний для користувача інтерфейс командного рядку для додатка, щоб зробити його інтуїтивно зрозумілим та легким у використанні.

Висновки: Виконуючи ці завдання, студенти створять універсальний Генератор ASCII-арту, який дозволить користувачам налаштовувати свої творіння з різними шрифтами, кольорами, розмірами та символами. Проект надасть практичний досвід роботи з введенням користувача, зовнішніми бібліотеками, роботою з файлами та дизайном інтерфейсу користувача в Python.

**Хід роботи**

**AsciiArtGenerator.py:**

class AsciiArtGenerator:

"""Generates ASCII art based on user-defined settings."""

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

self.settings = settings

def generate\_art(self, phrase):

"""Generate ASCII art with current settings."""

art\_engine = Figlet(font=self.settings.font, width=self.settings.size[0])

art = art\_engine.renderText(phrase)

if self.settings.symbols:

art = self.\_replace\_symbols(art, self.settings.symbols[0])

return self.settings.color + art

@staticmethod

def \_replace\_symbols(art, symbol):

"""Replace non-whitespace characters in the ASCII art with a symbol."""

return ''.join(symbol if char not in ('\n', ' ') else char for char in art)

**AsciiArtSettings.py**:

class AsciiArtSettings:

"""Stores settings for ASCII art generation."""

def \_\_init\_\_(self, font="slant", size=(80, 25), symbols=None, color=Fore.RESET):

self.font = font

self.size = size

self.symbols = symbols

self.color = color

**FileManager.py:**

class FileManager:

"""Handles saving and loading ASCII art."""

@staticmethod

def save\_art(folder\_path, art):

file\_name = input("Enter a file name to save: ")

file\_path = os.path.join(folder\_path, f"{file\_name}.txt")

with open(file\_path, "w", encoding="utf-8") as file:

file.write(art)

print(f"Art saved to {file\_path}")

@staticmethod

def load\_art(folder\_path):

file\_name = input("Enter the file name to load: ")

file\_path = os.path.join(folder\_path, f"{file\_name}.txt")

try:

with open(file\_path, "r", encoding="utf-8") as file:

print(file.read())

except FileNotFoundError:

print(f"File {file\_path} not found.")

**SettingsManager.py:**

class SettingsManager:

"""Manages user interaction for configuring ASCII art settings."""

@staticmethod

def select\_font():

fonts = Figlet().getFonts()

print("\nAvailable fonts:")

for idx, font in enumerate(fonts, 1):

print(f"{idx}. {font}")

choice = int(input("Choose a font number: ")) - 1

return fonts[choice]

@staticmethod

def set\_size():

width = int(input("Enter width: "))

height = int(input("Enter height: "))

return width, height

@staticmethod

def set\_symbols():

regular\_symbol = input("Enter the replacement symbol: ")

shadow\_symbol = input("Enter the shadow symbol (optional): ") or None

return regular\_symbol, shadow\_symbol

@staticmethod

def set\_color():

print("\nAvailable colors:")

for idx, color in enumerate(Fore.\_\_dict\_\_.keys(), 1):

print(f"{idx}. {color}")

choice = int(input("Choose a color number: ")) - 1

color\_name = list(Fore.\_\_dict\_\_.keys())[choice]

return Fore.\_\_dict\_\_[color\_name]

**Посилання на гіт-репозиторій:**

https://github.com/lesiukoksana/SMP-labs

**Висновок:**

У ході виконання лабораторної роботи було розроблено додаток "Генератор ASCII-арту", який дозволяє користувачам створювати персоналізовані ASCII-арт зображення. Реалізація програми дала змогу закріпити знання з роботи із зовнішніми бібліотеками Python, налаштування введення користувача, збереження даних у файл, а також із принципами об'єктно-орієнтованого програмування. Особливу увагу було приділено гнучкості та адаптивності коду завдяки модульній архітектурі програми, що складається з окремих класів для генерації арту, налаштувань, та управління файлами.