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

Виконав:

ст. гр. РІ-21сп

Павлусь В.П

Прийняв:

Шербак С.С.

Львів-2024

**Мета лабораторної роботи:** Створення додатка для генерації ASCII-арту

**Хід 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.

**Виконання роботи**

**Папка Classes:**

**Вміст файлу art\_operations.py:**

from Classes.error\_handler import SymbolError

class ArtOperations:

def \_\_init\_\_(self, ascii\_art):

self.ascii\_art = ascii\_art

def scale\_art(self, width, height):

if width < 1 or height < 1:

raise ValueError("Error: Width and height must be 1 or more!")

if not isinstance(width, int) or not isinstance(height, int):

raise ValueError("Error: Width and height must be positive integer!")

lines = self.ascii\_art.split('\n')

scaled\_art = ""

for line in lines:

scaled\_art += ''.join([char \* width for char in line]) + '\n' \* height

return scaled\_art

def change\_symbol(self, color\_choice, color\_manager):

while True:

change\_symbol = input("Want to change the symbol? (y/n): ").strip().lower()

if change\_symbol in ['y']:

new\_symbol = input("Enter new symbol: ")

if not new\_symbol:

raise SymbolError(new\_symbol)

self.ascii\_art = self.choose\_custom\_symbols(self.ascii\_art, new\_symbol)

ascii\_art\_colored = color\_manager.get\_colored\_art(self.ascii\_art, color\_choice)

print("\nASCII-art with new symbol:")

print(ascii\_art\_colored)

elif change\_symbol in ['n', ""]:

break

else:

print("Incorrect input: Enter 'y' or 'n'.")

def choose\_custom\_symbols(self, ascii\_art, symbol):

new\_art = ""

for char in ascii\_art:

if char == ' ':

new\_art += ' '

elif char != '\n':

new\_art += symbol

else:

new\_art += char

return new\_art

**Вміст файлу ascii\_art.py:**

import art

from Classes.art\_operations import ArtOperations

class AsciiArtGenerator:

def \_\_init\_\_(self, text, font, symbol=None):

self.text = text

self.font = font

self.symbol = symbol

self.ascii\_art = ""

def generate\_art(self):

self.ascii\_art = art.text2art(self.text, font=self.font)

if self.symbol:

self.ascii\_art = self.add\_spaces\_between\_letters(self.ascii\_art)

self.ascii\_art = self.choose\_custom\_symbols(self.ascii\_art, self.symbol)

return self.ascii\_art

def choose\_custom\_symbols(self, ascii\_art, symbol):

new\_art = ""

for char in ascii\_art:

if char == ' ':

new\_art += ' '

elif char != '\n':

new\_art += symbol

else:

new\_art += char

return new\_art

def add\_spaces\_between\_letters(self, ascii\_art):

lines = ascii\_art.split('\n')

spaced\_art = ""

for line in lines:

spaced\_line = ""

for char in line:

spaced\_line += char + ' '

spaced\_art += spaced\_line.rstrip() + '\n'

return spaced\_art

def resize\_art(self):

art\_operations = ArtOperations(self.ascii\_art)

while True:

try:

width = int(input("Enter new width (1 or more): "))

height = int(input("Enter new height (1 or more): "))

return art\_operations.scale\_art(width, height)

except ValueError:

print("Please, enter positive integers!")

def change\_symbol(self, color\_choice, color\_manager):

art\_operations = ArtOperations(self.ascii\_art)

art\_operations.change\_symbol(color\_choice, color\_manager)

**Вміст файлу color\_utils.py:**

from colorama import Fore

from Classes.error\_handler import ColorError

class ColorManager:

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

self.color\_mapping = {

"red": Fore.RED,

"green": Fore.GREEN,

"blue": Fore.BLUE,

"yellow": Fore.YELLOW,

"white": Fore.WHITE

}

def get\_colored\_art(self, ascii\_art, color\_choice):

if color\_choice in self.color\_mapping:

return self.color\_mapping[color\_choice] + ascii\_art + Fore.RESET

else:

return ascii\_art

def select\_color(self):

while True:

color\_choice = input("Enter a text color (red, green, blue, yellow, white): ").strip().lower()

if color\_choice in self.color\_mapping:

return color\_choice

else:

raise ColorError(color\_choice)

**Вміст файлу error\_handler.py:**

class FontError(Exception):

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

self.message = f"Error: Incorrect font!'{font}'."

super().\_\_init\_\_(self.message)

class ColorError(Exception):

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

self.message = f"Error: Incorrect color!'{color}'."

super().\_\_init\_\_(self.message)

class SymbolError(Exception):

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

self.message = "Error: Symbol cannot be empty!"

super().\_\_init\_\_(self.message)

**Вміст файлу file\_manager.py:**

from pathlib import Path

class FileManager:

@staticmethod

def save\_to\_file(ascii\_art):

data\_dir = Path('/Users/admin/Desktop/ДЗ/3 семестр/Спеціалізовані мови програмування/Lab3/data')

data\_dir.mkdir(parents=True, exist\_ok=True)

filename = input("Enter name of file to save (without the extension): ") + '.txt'

file\_path = data\_dir / filename

with file\_path.open('w') as f:

f.write(ascii\_art)

print(f"ASCII-art was saved in file {file\_path}")

**Вміст файлу fonts.py:**

from Classes.error\_handler import FontError

class FontManager:

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

self.fonts = [

"dot", "shadow", "banner", "dancing", "slant",

"straight", "block", "caligraphy", "stellar", "ghost", "3d"

]

def display\_fonts(self):

print("\nAvailable fonts:")

print(", ".join(self.fonts))

def select\_font(self):

self.display\_fonts()

font\_choice = input("Enter a font name from the list: ").strip()

if font\_choice not in self.fonts:

raise FontError(f"Font '{font\_choice}' not found. Please, try again!")

return font\_choice

**Папка UI:**

**Вміст файлу user\_interface.csv:**

from Classes.art\_operations import ArtOperations

from Classes.ascii\_art import AsciiArtGenerator

from Classes.color\_utils import ColorManager

from Classes.file\_manager import FileManager

from Classes.fonts import FontManager

from Classes.error\_handler import FontError, ColorError, SymbolError

def main():

print("\nASCII-ART Generator\n")

while True:

print("Select option:")

print("1 - Create new ASCII-art")

print("2 - Exit")

choice = input("Select 1 or 2: ")

if choice == '1':

text = input("Enter text for create ASCII-art: ")

try:

font\_manager = FontManager()

font\_choice = font\_manager.select\_font()

symbol = input("Enter symbol for create ASCII-art: ")

art\_generator = AsciiArtGenerator(text, font\_choice, symbol)

ascii\_art = art\_generator.generate\_art()

color\_manager = ColorManager()

color\_choice = color\_manager.select\_color()

ascii\_art\_colored = color\_manager.get\_colored\_art(ascii\_art, color\_choice)

print("\nASCII-art with your symbol:")

print(ascii\_art\_colored)

art\_operations = ArtOperations(ascii\_art)

art\_operations.change\_symbol(color\_choice, color\_manager)

while True:

resize\_option = input("Want to resize ASCII-art? (y/n): ").strip().lower()

if resize\_option in ['y']:

resized\_art = art\_generator.resize\_art()

ascii\_art\_colored = color\_manager.get\_colored\_art(resized\_art, color\_choice)

print("\nASCII-art with new size:")

print(ascii\_art\_colored)

elif resize\_option in ['n', '']:

break

else:

print("Incorrect input: Enter 'y' or 'n'.")

save\_option = input("Save ASCII-art in file? (y/n): ").strip().lower()

if save\_option in ['y']:

FileManager.save\_to\_file(ascii\_art)

except (FontError, ColorError, SymbolError) as e:

print(e)

except ValueError as e:

print(e)

elif choice == '2':

print("Exit")

break

else:

print("Wrong choice. Please, select 1 or 2.")

**Вміст файлу run.py:**

from UI.user\_interface import main

if \_\_name\_\_ == "\_\_main\_\_":

main()

Результат виконання, а саме генерація ASCII-арту та збереження його у файл зображено на рис. 1 та рис. 2.

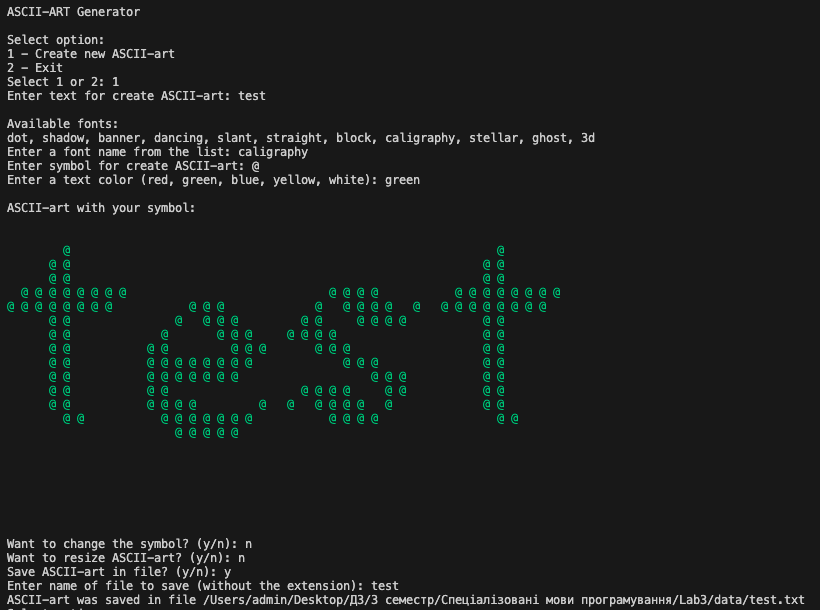


Рис. 1 – Генерація ASCII-арту



Рис. 2 – Зображення арту у файлі

**Висновок:** У ході виконання ЛР я створив універсальний Генератор ASCII-арту, який дозволить користувачам налаштовувати свої творіння з різними шрифтами, кольорами, розмірами та символами.