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

Ольга КЛІЩЕВСЬКА

Прийняв:

Сергій ЩЕРБАК

Львів-2024

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

**Хід роботи**

**Завдання:**

Створіть Python-програму, яка приймає введення користувача для слова або Інтегруйте бібліотеку ASCII-арту (наприклад, pyfiglet або art) у вашу програму для генерації ASCII-арту з введення користувача. Дозвольте користувачам вибирати різні стилі шрифтів для свого ASCII-арту. Надайте список доступних шрифтів та дозвольте їм вибрати один. Реалізуйте опцію вибору користувачем кольору тексту для їхнього ASCII-арту. Підтримуйте основний вибір кольорів (наприклад, червоний, синій, зелений). Переконайтеся, що створений ASCII-арт правильно відформатований та вирівнюється на екрані для зручності читання. Додайте функціональність для збереження створеного ASCII-арту у текстовому файлі, щоб користувачі могли легко завантажувати та обмінюватися своїми творіннями. Реалізуйте функцію попереднього перегляду, яка показує користувачам попередній перегляд їхнього ASCII-арту перед остаточним збереженням.

import pyfiglet

from termcolor import colored

import shutil

import textwrap

def get\_available\_fonts():

return [

"doom",

"isometric2",

"3-d", "3d-ascii",

"3d\_diagonal",

"banner3-D",

"starwars",

"slant",

"5lineoblique",

"block"

]

def generate\_ascii\_art(text, font, color, width, height, align):

try:

figlet = pyfiglet.Figlet(font=font, width=width)

ascii\_art = figlet.renderText(text)

if align == 'center':

ascii\_art\_custom = '\n'.join(textwrap.fill(line, width=width, subsequent\_indent='', initial\_indent=' '\*((width-len(line))//2)) for line in ascii\_art.splitlines())

elif align == 'right':

ascii\_art\_custom = '\n'.join(textwrap.fill(line, width=width, subsequent\_indent='', initial\_indent=' '\*(width-len(line))) for line in ascii\_art.splitlines())

else:

ascii\_art\_custom = ascii\_art

lines = ascii\_art\_custom.splitlines()

if len(lines) > height:

ascii\_art\_custom = '\n'.join(lines[:height])

return colored(ascii\_art\_custom, color)

except Exception as e:

print(f"Error rendering with font '{font}': {e}")

return None

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

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

file.write(ascii\_art)

def main():

print("Welcome to ASCII Art Generator!")

text = input("Enter the text (you can include symbols) you want to convert into ASCII art: ")

fonts = get\_available\_fonts()

print("\nAvailable fonts:")

for i, font in enumerate(fonts[:10], 1):

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

while True:

try:

font\_choice = int(input(f"Choose a font (1-{min(10, len(fonts))}): "))

font = fonts[font\_choice - 1]

break

except (ValueError, IndexError):

print("Invalid choice. Please select a valid font number.")

colors = ["white", "orange", "red", "blue", "yellow", "magenta", "cyan"]

print("\nAvailable colors: " + ", ".join(colors))

color = input(f"Choose a color ({', '.join(colors)}): ").lower()

if color not in colors:

print(f"Invalid color, defaulting to white.")

color = "white"

alignments = ['left', 'center', 'right']

print("\nText placement options: left, center, right")

align = input(f"Choose alignment ({', '.join(alignments)}): ").lower()

if align not in alignments:

print(f"Invalid alignment, defaulting to left.")

align = 'left'

terminal\_width = shutil.get\_terminal\_size().columns

print(f"\nDefault terminal width is {terminal\_width}.")

width = input(f"Enter custom width or press Enter to use terminal width ({terminal\_width}): ")

if width.isdigit():

width = int(width)

else:

width = terminal\_width

terminal\_height = shutil.get\_terminal\_size().lines

print(f"\nDefault terminal height is {terminal\_height}.")

height = input(f"Enter custom height or press Enter to use terminal height ({terminal\_height}): ")

if height.isdigit():

height = int(height)

else:

height = terminal\_height

print("\nPreview of your ASCII art:")

ascii\_art\_preview = generate\_ascii\_art(text, font, color, width, height, align)

if ascii\_art\_preview:

print(ascii\_art\_preview)

save\_option = input("\nDo you want to save this ASCII art to a file? (yes/no): ").lower()

if save\_option == 'yes':

filename = input("Enter the filename (without extension): ") + ".txt"

save\_to\_file(ascii\_art\_preview, filename)

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

else:

print("ASCII art not saved.")

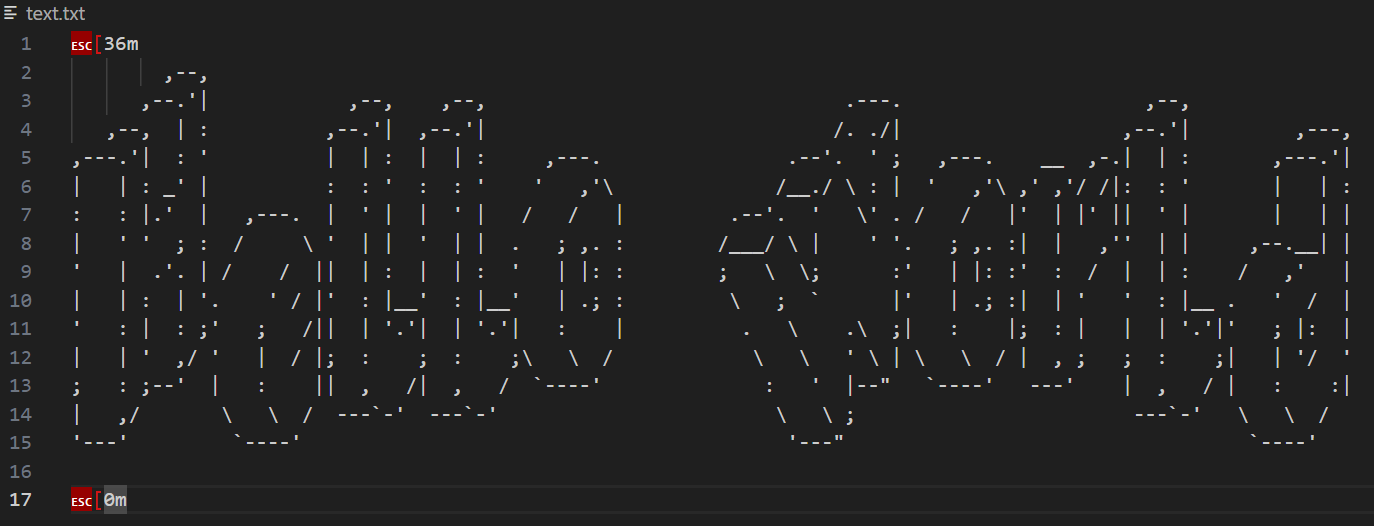
else:

print("Failed to generate ASCII art. Please try again.")

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

main()

На рис. 1 зображено результат роботи програми.



*Рис. 1. Результат виконання завдання 1*

Посилання на GitHub: https://github.com/olia28/SPL.git

**Висновок:** під час виконання лабораторної роботи, я створила універсальний Генератор ASCII-арту, який дозволить користувачам налаштовувати свої творіння з різними шрифтами, кольорами, розмірами та символами. Проект надав практичний досвід роботи з введенням користувача, зовнішніми бібліотеками, роботою з файлами та дизайном інтерфейсу користувача в Python.