МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»
Кафедра інформаційних систем та мереж

ЗВІТ

про виконання лабораторної роботи № 3
«Розробка ASCII ART генератора для візуалізації текстових даних»
з дисципліни "Спеціалізовані мови програмування”

Виконала:
ст. гр. ІТ-32,
Троцько О. М.
Прийняв:
Щербак С. С.

ЛЬВІВ – 2023

Мета: створення додатка Генератора ASCII-арту.

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 твореннями.

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.

Код програми:

```
# data_from_console.py
"""
Module: data_from_console

This module contains functions that prompt the user to enter various parameters from the console.
"""
import pyfiglet
import termcolor
from shared.input_handler import InputHandler
from shared.settings import get_lab_settings

settings = get_lab_settings("lab3")
FIGLET_FONT_SIZES = settings["figlet_font_sizes"]

def get_font_from_console():
    """
    Prompts the user to enter a font name from the available fonts.

    Returns:
        str: The selected font name.
    """
    available_fonts = pyfiglet.FigletFont.getFonts()
    print("Font sizes: ", FIGLET_FONT_SIZES)
    font = InputHandler().get_one_of_list_input("Enter font name", available_fonts)
    return font

def get_width_from_console():
```

```
    """
    Prompts the user to enter the width.

    Returns:
        int: The entered width.
    """
    width = InputHandler().get_int_input("Enter width")
    return width

def get_symbol_from_console():
    """
    Prompts the user to enter a symbol.

    Returns:
        str: The entered symbol.
    """
    symbol = InputHandler().get_one_char_input("Enter symbol (e.g. '@', '#', '*')")
    return symbol

def get_color_from_console():
    """
    Prompts the user to enter a color name from the available colors.

    Returns:
        str: The selected color name.
    """
            color   =   InputHandler().get_one_of_list_input("Enter   color   name",
termcolor.COLORS)
    return color

def get_text_from_console():
    """
    Prompts the user to enter a text.

    Returns:
        str: The entered text.
    """
    text = InputHandler().get_str_input("Enter text")
    return text
```

```
# figlet_generator.py
"""
Module: figlet_generator

This module contains the FigletGenerator class that generates figlet art based on
provided text and settings.
"""
from pyfiglet import Figlet
from termcolor import colored
from UI.menu import Menu
from UI.menu_item import Item
from shared.settings import get_lab_settings
from shared.file_handler import FileHandler
from classes.lab3.figlet.figlet_settings import FigletSettings
from classes.lab3.console_reader.data_from_console import get_text_from_console

settings = get_lab_settings("lab3")
ART_PATH = settings["art_path"]

class FigletGenerator:
    """
    A class that generates figlet art based on the provided text and settings.
    """

    def __init__(self, text=None):
        """
        Initialize the FigletGenerator object.

        Args:
        - text: The text to generate art from (default: None)
        """
        self.__text = text
        self.__settings = FigletSettings()
        self.__figlet = None

    def menu(self):
        """
        Displays the menu options for the Figlet Generator.
        """
        main_menu = Menu("\nMenu")
```

```
    main_menu.add_item(Item('1', 'Generate art', self.generate_art))
    main_menu.add_item(Item('2', 'Change settings', self.change_settings))
    main_menu.add_item(Item('3', 'Preview', self.view_art ))
    main_menu.add_item(Item('4', 'Save art', self.save_art))
    main_menu.add_item(Item('5', 'View saved art', self.view_saved_art))
    main_menu.add_item(Item('0', 'Exit'))

    main_menu.run()

def get_text(self):
    """
    Get the current text.

    Returns:
    - The current text.
    """
    return self.__text

def get_settings(self):
    """
    Get the current settings.

    Returns:
    - The current settings.
    """
    return self.__settings

def get_figlet(self):
    """
    Get the current figlet art.

    Returns:
    - The current figlet art.
    """
    return self.__figlet

def set_text(self, text):
    """
    Set the text.
```

```
        Args:
        - text: The text to set.
        """
        self.__text = text

    def set_settings(self, figlet_settings):
        """
        Set the settings.

        Args:
        - settings: The settings to set.
        """
        self.__settings = figlet_settings

    def set_figlet(self, figlet):
        """
        Set the figlet art.

        Args:
        - figlet: The figlet art to set.
        """
        self.__figlet = figlet

    def generate_art(self):
        """
        Generate the figlet art based on the current text and settings.
        """
        self.__text = get_text_from_console()
                        self.__figlet   =   Figlet(font=self.__settings.get_font(),
width=self.__settings.get_width())
        self.__figlet = self.__figlet.renderText(self.__text)

        if self.__settings.get_symbol() is not None:
            self.modify_symbols(self.__settings.get_symbol())

        print("\nArt is generated!")

    def modify_symbols(self, symbol):
        """
        Modify the symbols in the figlet art.
```

```
    Args:
    - symbol: The symbol to replace the existing symbols with.
    """
    for char in self.__figlet:
        if char != '\n' and char != ' ':
            self.__figlet = self.__figlet.replace(char, symbol)

def change_settings(self):
    """
    Change the settings for generating the figlet art.
    """
    self.__settings.settings_menu()

def view_art(self):
    """
    View the generated figlet art.
    """
    if self.__figlet is None:
        print("No art to preview")
    else:
        print(colored(self.__figlet, self.__settings.get_color()))

def save_art(self):
    """
    Save the generated figlet art to a file.
    """
    if self.__figlet is None:
        print("No art to save")
        return
    saved_file = FileHandler(ART_PATH)
    saved_file.write_to_file(self.__figlet)

def view_saved_art(self):
    """
    View the saved figlet art from a file.
    """
    saved_file = FileHandler(ART_PATH)
    saved_file.read_from_file()
```

```
# figlet_settings.py
"""
A module that defines the FigletSettings class representing the settings for a Figlet program.
"""
from UI.menu import Menu
from UI.menu_item import Item
from  classes.lab3.console_reader.data_from_console  import  get_font_from_console, get_width_from_console, get_symbol_from_console, get_color_from_console
from shared.settings import get_lab_settings

settings = get_lab_settings("lab3")
DEFAULT_FIGLET_SETTINGS = settings["default_figlet_settings"]
DEFAULT_FONT = DEFAULT_FIGLET_SETTINGS["font"]
DEFAULT_WIDTH = DEFAULT_FIGLET_SETTINGS["width"]
DEFAULT_COLOR = DEFAULT_FIGLET_SETTINGS["color"]

class FigletSettings():
    """
    Class representing the settings for a Figlet program.
    """

    def __init__(self):
        """
        Initialize the FigletSettings object with default settings.
        """
        self._font = DEFAULT_FONT
        self._width = DEFAULT_WIDTH
        self._symbol = None
        self._color = DEFAULT_COLOR

    def set_font(self, font):
        """
        Set the font for the Figlet program.

        Args:
            font (str): The font to set.
        """
        self._font = font
```

```
def set_width(self, width):
    """
    Set the width for the Figlet program.

    Args:
        width (int): The width to set.
    """
    self._width = width

def set_symbol(self, symbol):
    """
    Set the symbol for the Figlet program.

    Args:
        symbol (str): The symbol to set.
    """
    self._symbol = symbol

def set_color(self, color):
    """
    Set the color for the Figlet program.

    Args:
        color (str): The color to set.
    """
    self._color = color

def get_font(self):
    """
    Get the current font for the Figlet program.

    Returns:
        str: The current font.
    """
    return self._font

def get_width(self):
    """
    Get the current width for the Figlet program.
```

```
    Returns:
        int: The current width.
    """
    return self._width

def get_symbol(self):
    """
    Get the current symbol for the Figlet program.

    Returns:
        str: The current symbol.
    """
    return self._symbol

def get_color(self):
    """
    Get the current color for the Figlet program.

    Returns:
        str: The current color.
    """
    return self._color

def get_settings(self):
    """
    Get the current settings for the Figlet program.

    Returns:
        dict: A dictionary containing the current settings.
    """
    figlet_settings = {
        "font": self._font,
        "width": self._width,
        "symbol": self._symbol,
        "color": self._color,
    }
    return figlet_settings

def print_settings(self):
    """
```

```
    Print the current settings for the Figlet program.
    """
    print("\nSettings:")
    for key, value in self.get_settings().items():
        print(f"{key}: {value}")

def settings_menu(self):
    """
    Display the settings menu for the Figlet program.
    """
    settings_menu = Menu("\nSettings Menu")
    settings_menu.set_color('grey')
    settings_menu.add_item(Item('1', 'View Settings', self.print_settings))
    settings_menu.add_item(Item('2', 'Change Font', self.change_font))
    settings_menu.add_item(Item('3', 'Change Width', self.change_width))
    settings_menu.add_item(Item('4', 'Change Symbol', self.change_symbol))
    settings_menu.add_item(Item('5', 'Change Color', self.change_color))
    settings_menu.add_item(Item('0', 'Back'))

    settings_menu.run()

def change_font(self):
    """
    Change the font for the Figlet program.
    """
    new_font = get_font_from_console()
    self._font = new_font
    print("\nFont was changed to", self._font)

def change_width(self):
    """
    Change the width for the Figlet program.
    """
    new_width = get_width_from_console()
    print(new_width)
    self._width = new_width
    print("\nWidth was changed to", self._width)

def change_symbol(self):
    """
```

```
        Change the symbol for the Figlet program.
        """
        new_symbol = get_symbol_from_console()
        self._symbol = new_symbol
        print("\nSymbol was changed to", self._symbol)

    def change_color(self):
        """
        Change the color for the Figlet program.
        """
        new_color = get_color_from_console()
        self._color = new_color
        print("\nColor was changed to", self._color)
```

```
# runner.py
"""
Module: run_figlet_generator

Module provides a simple script to run the Figlet text generator for Lab 3.

"""

from classes.lab3.figlet.figlet_generator import FigletGenerator

def run():
    """
    Initializes and runs the Figlet text generator.
    """
    figlet_generator = FigletGenerator()
    figlet_generator.menu()
```

GitHub Repository: https://github.com/trolchiha/SPL-labs.git

Висновок: під час виконання лабораторної роботи навчилася створювати універсальний Генератор ASCII-арту, який дозволить користувачам налаштовувати свої творіння з різними шрифтами, кольорами, розмірами та символами.