мІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

нАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «лЬВІВСЬКА ПОЛІТЕХНІКА»

ІНСТИТУТ КОМП’ЮТЕРНИХ НАУК ТА ІНОРМАЦІЙНИХ ТЕХНОЛОГІЙ

|  |
| --- |
|  |



ЗВІТ

Про виконання лабораторної роботи № 4

З дисципліни

«Спеціалізовані мови програмування»

на тему «Розробка ASCII ART генератора для візуалізації 2D-фігур»

Виконала:

студентка гр. ІТ-31

Проців Р.В.

Прийняв:

доц. каф. ІСМ

Щербак С.С.

Львів – 2023

**Мета роботи**: Створення Генератора ASCII-арту без використання зовнішніх бібліотек

**Хід роботи**:

**Завдання 1**: Введення користувача

Створіть програму Python, яка отримує введення користувача щодо слова або фрази, яку вони хочуть перетворити в ASCII-арт.

**Завдання 2**: Набір символів

Визначте набір символів (наприклад, '@', '#', '\*', тощо), які будуть використовуватися для створення ASCII-арту. Ці символи будуть відображати різні відтінки.

**Завдання 3**: Розміри Art-у

Запитайте у користувача розміри (ширина і висота) ASCII-арту, який вони хочуть створити. Переконайтеся, що розміри в межах керованого діапазону

**Завдання 4**: Функція генерації Art-у

Напишіть функцію, яка генерує ASCII-арт на основі введення користувача, набору символів та розмірів. Використовуйте введення користувача, щоб визначити, які символи використовувати для кожної позиції в Art-у.

**Завдання 5**: Вирівнювання тексту

Реалізуйте опції вирівнювання тексту (ліво, центр, право), щоб користувачі могли вибирати, як їх ASCII-арт розміщується на екрані.

**Завдання 6:** Відображення мистецтва

Відобразіть створений ASCII-арт на екрані за допомогою стандартних функцій друку Python.

**Завдання 7**: Збереження у файл

Додайте можливість зберігати створений ASCII-арт у текстовий файл, щоб користувачі могли легко завантажувати та обмінюватися своїми творіннями.

**Завдання 8**: Варіанти кольорів

Дозвольте користувачам вибирати опції кольорів (чорно-білий, відтінки сірого) для свого ASCII-арту.

**Завдання 9**: Функція попереднього перегляду

Реалізуйте функцію попереднього перегляду, яка показує користувачам попередній перегляд їх ASCII-арту перед остаточним збереженням

**Завдання 10**: Інтерфейс, зрозумілий для користувача

Створіть інтерфейс для користувача у командному рядку, щоб зробити програму легкою та інтуїтивно зрозумілою для використання.

**Програмний код**:

class UserInteraction:

@staticmethod

def select\_color():

color\_choice = input("Select text color (white, gray): ").lower()

if color\_choice in ['white', 'gray']:

return color\_choice

else:

return None

import os

class ArtGenerator:

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

self.ascii\_dict = ascii\_dict

self.settings = settings

def get\_valid\_text(self):

while True:

text = input("Enter a word or phrase: ")

if all(c in self.ascii\_dict or c == ' ' for c in text):

return text

else:

print("Error. The input contains invalid characters. Please enter a valid phrase.")

def print\_ascii\_art(self, text, color):

output\_lines = [""] \* 6

for symbol in text:

if symbol == ' ':

for i in range(6):

output\_lines[i] += ' '

elif symbol in self.ascii\_dict:

symbol\_lines = self.ascii\_dict[symbol]

for i in range(6):

output\_lines[i] += symbol\_lines[i]

if len(output\_lines[0]) > self.settings.get\_max\_width():

raise ValueError("Error. Maximum width is too small for the input phrase.")

if self.settings.get\_alignment() == 'center':

output\_lines = [line.center(self.settings.get\_max\_width()) for line in output\_lines]

elif self.settings.get\_alignment() == 'right':

output\_lines = [line.rjust(self.settings.get\_max\_width()) for line in output\_lines]

colored\_output\_lines = [self.apply\_color(line, color) for line in output\_lines]

for line in colored\_output\_lines:

print(line)

return output\_lines

def apply\_color(self, text, color):

color\_codes = {'white': '\u001b[97m', 'gray': '\u001b[90m'}

reset\_color = '\u001b[0m'

return color\_codes[color] + text + reset\_color

def save\_to\_text\_file(self, output\_lines, output\_file\_name):

result\_folder = "result"

file\_path = os.path.join(result\_folder, output\_file\_name)

with open(file\_path, 'w') as file:

for line in output\_lines:

file.write(line + '\n')

class Settings:

def \_\_init\_\_(self, max\_width, alignment):

self.max\_width = max\_width

self.alignment = alignment

def get\_max\_width(self):

return self.max\_width

def get\_alignment(self):

return self.alignment

from classes.art\_generator import ArtGenerator

from classes.art\_settings import Settings

from classes.interactions import UserInteraction

from classes.ascii\_art import AsciiArt

from input\_handling import get\_string\_input, get\_integer\_input, get\_yes\_no\_input

import os

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

script\_dir = os.path.dirname(os.path.abspath(\_\_file\_\_))

symbol\_set = {

"@": os.path.join(script\_dir,"art\_symbols/dog.txt"),

"\*": os.path.join(script\_dir,"art\_symbols/asterisk.txt"),

}

while True:

selected\_symbol = get\_string\_input("Select a symbol ( @, \*): ")

if selected\_symbol not in symbol\_set:

print("Error! Invalid symbol.")

continue

file\_path = symbol\_set[selected\_symbol]

ascii\_art = AsciiArt.load\_ascii\_art(file\_path)

while True:

color = UserInteraction.select\_color()

if color is None:

continue

alignment = get\_string\_input("Select alignment (left, center, right): ").lower()

if alignment not in ["left", "center", "right"]:

print("Error! Invalid alignment.")

continue

max\_width = get\_integer\_input("Enter maximum width: ")

try:

settings = Settings(max\_width, alignment)

art\_generator = ArtGenerator(ascii\_art, settings)

text = art\_generator.get\_valid\_text()

print("The result of ASCII art:")

art = art\_generator.print\_ascii\_art(text, color)

save\_choice = get\_yes\_no\_input("Do you want to save the ASCII art to a text file? (yes/no): ")

if save\_choice == "yes":

output\_file\_name = get\_string\_input("Enter the file name to save to: ")

art\_generator.save\_to\_text\_file(art, output\_file\_name)

print("ASCII art saved to", output\_file\_name)

continue\_choice = get\_yes\_no\_input("Do you want to continue drawing ASCII art? (yes/no): ")

if continue\_choice != "yes":

exit()

except ValueError as e:

print(e)

**GitHub Repository:** <https://github.com/fxxwol/python/tree/main/lab_4>

**Висновок:** в результаті виконання даної лабораторної роботи створено генератор ASCII-арту без використання зовнішніх бібліотек