МІНІСТЕРСТВО ОСВІТИ І НАУКИ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

ЗВІТ

з виконання лабораторної роботи №3

з дисципліни «Функціональне програмування»

за темою «Функції вищого порядку. Знайомство з можливостями функціонального програмування мови Python»

Виконав: Перевірив:

Ковалик Вадим Валерійович Міхаль Олег Пилипович

Харків 2024

* 1. Мета роботи

Практичне відпрацювання прийомів використання елементів і підходів функціонального програмування у складі та в комбінації з процедурними імперативними складовими програми.

* 1. Порядок виконання роботи
     1. Завдання:

На даній лабораторній роботі ми повинні запустити програму, перевірити її працездатність та переконатися, що вона видає протокол. Після цього слід дописати програму у процедурному стилі. Завершальним кроком є переписання програми з використанням функціонального стилю.

* + 1. Хід роботи
       1. Перевірка працездатності заданої програми

Запускаємо програму та перевіряємо її працездатність та переконуємося, що вона видає протокол.

Лістинг 3.1 – Код програми для перевірки працездатності

import random

Protocol\_name = "Protocol\_name.txt"

file = open(Protocol\_name, "a")

file.write("======== Beginning of the protocol " + Protocol\_name + " ========\n")

print("======== Beginning of the protocol " + Protocol\_name + " ========")

nn = 20

file.write("nn = " + str(nn) + "\n")

print("nn = " + str(nn))

Q = [0] \* nn

file.write("Q (initial) = " + str(Q) + "\n")

print("Q (initial) = " + str(Q))

for i in range(0, nn):

Q[i] = int(random.random() \* 255)

file.write("Q (filled) = " + str(Q) + "\n")

print("Q (filled) = " + str(Q))

MaxQ = max(Q)

MinQ = min(Q)

file.write("MaxQ = " + str(MaxQ) + ", MinQ = " + str(MinQ) + "\n")

print("MaxQ = " + str(MaxQ) + ", MinQ = " + str(MinQ))

file.write("======== End of protocol " + Protocol\_name + " ========\n")

print("======== End of protocol " + Protocol\_name + " ========")

file.close()

print("Protocol is generated")

z = 0

while z != "":

print("=======================================")

z = input("Press ENTER for exit ... ")

Зображення, що містить текст, знімок екрана, Шрифт

Автоматично згенерований описРисунок 3.1 – Результат виконання в консолі

Зображення, що містить текст, Шрифт, знімок екрана

Автоматично згенерований опис

Рисунок 3.2 – Вигляд файлу Protocol\_name

* + - 1. Доповнення програми в процедурному стилі

Розширюємо програму, використовуючи процедурний стиль, дотримуючись коментаря, що міститься в її заключній частині.

Лістинг 3.2 – Доповнена програма згідно коментаря

import random

Protocol\_name = "Protocol\_name.txt"

file = open(Protocol\_name, "a")

file.write("======== Beginning of the protocol " + Protocol\_name + " ========\n")

print("======== Beginning of the protocol " + Protocol\_name + " ========")

nn = 20

file.write("nn = " + str(nn) + "\n")

print("nn = " + str(nn))

Q = [0] \* nn

file.write("Q (initial) = " + str(Q) + "\n")

print("Q (initial) = " + str(Q))

for i in range(0, nn):

Q[i] = int(random.random() \* 255)

file.write("Q (filled) = " + str(Q) + "\n")

print("Q (filled) = " + str(Q))

MaxQ = max(Q)

MinQ = min(Q)

file.write("MaxQ = " + str(MaxQ) + ", MinQ = " + str(MinQ) + "\n")

print("MaxQ = " + str(MaxQ) + ", MinQ = " + str(MinQ))

range\_step = (MaxQ - MinQ) / 5

histogram = [0] \* 5

for value in Q:

index = min(int((value - MinQ) / range\_step), 4)

histogram[index] += 1

file.write("Histogram = " + str(histogram) + "\n")

print("Histogram = " + str(histogram))

file.write("======== End of protocol " + Protocol\_name + " ========\n")

print("======== End of the protocol " + Protocol\_name + " ========")

file.close()

print("Protocol is generated")

z = 0

while z != "":

print("=======================================")

z = input("Press ENTER for exit ... ")

Зображення, що містить текст, знімок екрана, Шрифт

Автоматично згенерований опис

Рисунок 3.3 – Результат виконання розширеної програми

Зображення, що містить текст, Шрифт, знімок екрана

Автоматично згенерований опис

Рисунок 3.4 – Вигляд файлу Protocol\_name

* + - 1. Переписання програми у функціональному стилі

Переписуємо програму з максимальним використанням функціонального стилю відповідно до принципів.

Лістинг 3.3 – Використання функціонального стилю

import random

def initialize\_protocol\_file(filename):

file = open(filename, "a")

file.write("======== Beginning of the protocol " + filename + " ========\n")

print("======== Beginning of the protocol " + filename + " ========")

return file

def close\_protocol\_file(file, filename):

file.write("======== End of protocol " + filename + " ========\n")

print("======== End of protocol " + filename + " ========")

file.close()

def generate\_random\_array(size, max\_value=255):

return [int(random.random() \* max\_value) for \_ in range(size)]

def calculate\_histogram\_bounds(min\_val, max\_val, num\_bins=5):

range\_step = (max\_val - min\_val) / num\_bins

return [min\_val + i \* range\_step for i in range(1, num\_bins)]

def populate\_histogram(array, bounds):

histogram = [0] \* (len(bounds) + 1)

for value in array:

for i, bound in enumerate(bounds):

if value < bound:

histogram[i] += 1

break

else:

histogram[-1] += 1

return histogram

def main\_protocol(filename="Protocol\_name.txt", array\_size=20):

file = initialize\_protocol\_file(filename)

Q = generate\_random\_array(array\_size)

file.write("Q = " + str(Q) + "\n")

print("Q = " + str(Q))

MaxQ = max(Q)

MinQ = min(Q)

file.write("MaxQ = " + str(MaxQ) + ", MinQ = " + str(MinQ) + "\n")

print("MaxQ = " + str(MaxQ) + ", MinQ = " + str(MinQ))

bounds = calculate\_histogram\_bounds(MinQ, MaxQ)

histogram = populate\_histogram(Q, bounds)

file.write("Histogram = " + str(histogram) + "\n")

print("Histogram = " + str(histogram))

close\_protocol\_file(file, filename)

print("Protocol is generated")

main\_protocol()

Зображення, що містить текст, знімок екрана, Шрифт

Автоматично згенерований опис

Рисунок 3.5 – Результат функціональної програми

Зображення, що містить текст, Шрифт, знімок екрана

Автоматично згенерований опис

Рисунок 3.6 – Вигляд файлу

* + - 1. Додавання графічного інтерфейсу за допомогою бібліотеки tkinter

Цей інтерфейс дозволяє користувачеві запускати процес створення протоколу, налаштувати розмір масиву та переглядати результат безпосередньо в програмі.

Лістинг 3.4 – Код створеного графічного інтерфейсу

import random

import tkinter as tk

from tkinter import messagebox, scrolledtext

def generate\_random\_array(size, max\_value=255):

return [int(random.random() \* max\_value) for \_ in range(size)]

def calculate\_histogram\_bounds(min\_val, max\_val, num\_bins=5):

range\_step = (max\_val - min\_val) / num\_bins

return [min\_val + i \* range\_step for i in range(1, num\_bins)]

def populate\_histogram(array, bounds):

histogram = [0] \* (len(bounds) + 1)

for value in array:

for i, bound in enumerate(bounds):

if value < bound:

histogram[i] += 1

break

else:

histogram[-1] += 1

return histogram

def generate\_protocol(array\_size, text\_widget):

Q = generate\_random\_array(array\_size)

MaxQ = max(Q)

MinQ = min(Q)

bounds = calculate\_histogram\_bounds(MinQ, MaxQ)

histogram = populate\_histogram(Q, bounds)

protocol\_text = (

f"======== Beginning of the protocol ========\n"

f"Q = {Q}\n"

f"MaxQ = {MaxQ}, MinQ = {MinQ}\n"

f"Histogram = {histogram}\n"

f"======== End of protocol ========\n"

)

text\_widget.delete(1.0, tk.END)

text\_widget.insert(tk.END, protocol\_text)

def on\_generate():

try:

array\_size = int(entry\_size.get())

if array\_size <= 0:

raise ValueError("The size of the array must be greater than zero.")

generate\_protocol(array\_size, text\_output)

messagebox.showinfo("Protocol created", "The protocol is successfully created and displayed.")

except ValueError as e:

messagebox.showerror("Error", str(e))

root = tk.Tk()

root.title("Protocol generator")

frame = tk.Frame(root)

frame.pack(pady=10)

lbl\_size = tk.Label(frame, text="The size of the array:")

lbl\_size.pack(side=tk.LEFT)

entry\_size = tk.Entry(frame, width=10)

entry\_size.pack(side=tk.LEFT)

entry\_size.insert(0, "20")

btn\_generate = tk.Button(root, text="Create a protocol", command=on\_generate)

btn\_generate.pack(pady=5)

text\_output = scrolledtext.ScrolledText(root, width=50, height=15, wrap=tk.WORD)

text\_output.pack(pady=10)

root.mainloop()

Зображення, що містить текст, електроніка, знімок екрана, комп’ютер

Автоматично згенерований опис

Рисунок 3.7 – Вигляд програми

* 1. Висновок

У даній роботі було реалізовано генерацію протоколу на основі випадкових значень з визначенням мінімальних та максимальних значень, а також побудовою гістограми для візуалізації розподілу даних. Для зручності користувачів створено графічний інтерфейс за допомогою бібліотеки tkinter, що дозволяє налаштовувати параметри масиву та переглядати результати без редагування коду. Це демонструє ефективність автоматизації процесів аналізу даних та можливості, які відкриваються при використанні графічних інтерфейсів у програмуванні.