МІНІСТУЕРСТВО ОСВІТИ І НАУКИ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

ЗВІТ

з виконання лабораторної роботи №4

з дисципліни «Функціональне програмування»

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

Виконав: Перевірив:

Ковалик Вадим Валерійович Міхаль Олег Пилипович

Харків 2024

* 1. Мета роботи

Продовження практичного відпрацювання використання елементів і підходів функціонального програмування в комбінації з процедурними імперативними складовими програми.

* 1. Порядок виконання роботи
     1. Завдання:

Вивчити принципи роботи з функціями вищого порядку у Python. Також потрібно ознайомитися з комбінацією процедурних та функціональних підходів до програмування. А також потрібно реалізувати систему переходів із зарядкою та розрядкою станів у процедурному та функціональному стилях.

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

Лістинг 4.1 – Правильно налагоджена програма, яка виконує зарядку переходів

import random

import math

from copy import copy

with (open("protocol-20151209.txt", "a") as file):

file.write("==== Protocol ====\n")

np = 3

nt = 2

P = [1, 0, 0]

T = [0, 0]

P2T = [[1, 0, 0], [0, 1, 0]]

T2P = [[0, 1, 0], [1, 0, 1]]

file.write(f"P = {P}\n")

file.write(f"T = {T}\n")

file.write(f"P2T = {P2T}\n")

file.write(f"T2P = {T2P}\n")

file.write("==== charging transitions ====\n")

file.write(f"i j P2T[i][j] P[j] flag1 flag2 T[i]\n")

flag1 = 0

flag2 = 0

for i in range(nt):

for j in range(np):

if P2T[i][j] > 0:

if P[j] > 0:

flag1 = 1

elif P[j] == 0:

flag2 = 1

file.write(f"{i} {j} {P2T[i][j]} {P[j]} {flag1} {flag2} {T[i]}\n")

if flag1 == 1:

if flag2 == 0:

for j in range(np):

if P2T[i][j] > 0:

P[j] -= 1

T[i] = 1

file.write(f"{i} {j} {P2T[i][j]} {P[j]} {flag1} {flag2} {T[i]}\n")

file.write("==== end of charging transitions ====\n")

file.write("===== The END =====\n")

print("Protocol is generated")

z = 0

while z != "":

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

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

Зображення, що містить текст, Шрифт, знімок екрана

Автоматично згенерований опис

Рисунок 4.1 – Вивід програми

Зображення, що містить текст, знімок екрана, Шрифт, число

Автоматично згенерований опис

Рисунок 4.2 – Результат виконання

* + - 1. Доповнена програми згідно коментаря

Лістинг 4.2 – Робота системи переходів із зарядкою та розрядкою станів

import random

import math

from copy import copy

with (open("protocol-20151209.txt", "a") as file):

file.write("==== Protocol ====\n")

np = 3

nt = 2

P = [1, 0, 0]

T = [0, 0]

P2T = [[1, 0, 0], [0, 1, 0]]

T2P = [[0, 1, 0], [1, 0, 1]]

file.write(f"P = {P}\n")

file.write(f"T = {T}\n")

file.write(f"P2T = {P2T}\n")

file.write(f"T2P = {T2P}\n")

file.write("==== charging transitions ====\n")

file.write(f"i j P2T[i][j] P[j] flag1 flag2 T[i]\n")

flag1 = 0

flag2 = 0

for i in range(nt):

for j in range(np):

if P2T[i][j] > 0:

if P[j] > 0:

flag1 = 1

elif P[j] == 0:

flag2 = 1

file.write(f"{i} {j} {P2T[i][j]} {P[j]} {flag1} {flag2} {T[i]}\n")

if flag1 == 1:

if flag2 == 0:

for j in range(np):

if P2T[i][j] > 0:

P[j] -= 1

T[i] = 1

file.write(f"{i} {j} {P2T[i][j]} {P[j]} {flag1} {flag2} {T[i]}\n")

file.write("==== end of charging transitions ====\n")

file.write("==== discharging transitions ====\n")

flag1 = 0

flag2 = 0

for i in range(nt):

for j in range(np):

if T2P[i][j] > 0:

flag1 = 1

if T[i] > 0:

flag2 = 1

file.write(f"{i} {j} {P2T[i][j]} {P[j]} {flag1} {flag2} {T[i]}\n")

if flag1 == 1:

if flag2 == 1:

P[j] += T2P[i][j]

for k in range(np):

if T2P[i][k] > 0:

P[k] += T2P[i][k]

T[i] = 0

if P[k] == 1:

P[k] = 0

elif P[k] == 2:

P[k] = 1

file.write(f"{i} {j} {P2T[i][j]} {P[j]} {flag1} {flag2} {T[i]}\n")

file.write("==== end of discharging transitions ====\n")

file.write("===== The END =====\n")

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 4.3 – Результат виконання коду

Зображення, що містить текст, знімок екрана, програмне забезпечення, Мультимедійне програмне забезпечення

Автоматично згенерований опис

Рисунок 4.4 – Протокол, який відображає зарядку та розрядку переходів

* + - 1. Переписання програми у функціональному стилі

Лістинг 4.3 – Відтворення програми у функціональному вигляді

import random

import math

from copy import copy

def initialize\_system():

np = 3

nt = 2

P = [1, 0, 0]

T = [0, 0]

P2T = [[1, 0, 0], [0, 1, 0]]

T2P = [[0, 1, 0], [1, 0, 1]]

return np, nt, P, T, P2T, T2P

def write\_initial\_data(file, P, T, P2T, T2P):

file.write("==== Protocol ====\n")

file.write(f"P = {P}\n")

file.write(f"T = {T}\n")

file.write(f"P2T = {P2T}\n")

file.write(f"T2P = {T2P}\n")

def charge\_transitions(file, np, nt, P, T, P2T):

file.write("==== charging transitions ====\n")

file.write(f"i j P2T[i][j] P[j] flag1 flag2 T[i]\n")

flag1 = 0

flag2 = 0

for i in range(nt):

flag1, flag2 = 0, 0

for j in range(np):

if P2T[i][j] > 0:

if P[j] > 0:

flag1 = 1

elif P[j] == 0:

flag2 = 1

file.write(f"{i} {j} {P2T[i][j]} {P[j]} {flag1} {flag2} {T[i]}\n")

if flag1 == 1:

if flag2 == 0:

for j in range(np):

if P2T[i][j] > 0:

P[j] -= 1

T[i] = 1

file.write(f"{i} {j} {P2T[i][j]} {P[j]} {flag1} {flag2} {T[i]}\n")

file.write("==== end of charging transitions ====\n")

def discharge\_transitions(file, np, nt, P, T, T2P, P2T):

file.write("==== discharging transitions ====\n")

flag1 = 0

flag2 = 0

for i in range(nt):

for j in range(np):

if T2P[i][j] > 0:

flag1 = 1

if T[i] > 0:

flag2 = 1

file.write(f"{i} {j} {P2T[i][j]} {P[j]} {flag1} {flag2} {T[i]}\n")

if flag1 == 1:

if flag2 == 1:

P[j] += T2P[i][j]

for k in range(np):

if T2P[i][k] > 0:

P[k] += T2P[i][k]

T[i] = 0

if P[k] == 1:

P[k] = 0

elif P[k] == 2:

P[k] = 1

file.write(f"{i} {j} {P2T[i][j]} {P[j]} {flag1} {flag2} {T[i]}\n")

file.write("==== end of discharging transitions ====\n")

def main():

np, nt, P, T, P2T, T2P = initialize\_system()

with open("protocol-20151209.txt", "a") as file:

write\_initial\_data(file, P, T, P2T, T2P)

charge\_transitions(file, np, nt, P, T, P2T)

discharge\_transitions(file, np, nt, P, T, T2P, P2T)

file.write("===== The END =====\n")

print("Protocol is generated")

while input("Press ENTER for exit ...") != "":

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

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

main()

Зображення, що містить текст, Шрифт, знімок екрана, типографія

Автоматично згенерований опис

Рисунок 4.5 – Результат виконання

Зображення, що містить текст, знімок екрана, Шрифт, дизайн

Автоматично згенерований опис

Рисунок 4.6 – Вигляд файлу протоколу

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

Лістинг 4.4 – Додатковий код для створення графічного інтерфесу

import tkinter as tk  
from tkinter import ttk, messagebox  
from copy import copy  
  
def initialize\_system():  
 np = 3  
 nt = 2  
 P = [1, 0, 0]  
 T = [0, 0]  
 P2T = [[1, 0, 0], [0, 1, 0]]  
 T2P = [[0, 1, 0], [1, 0, 1]]  
 return np, nt, P, T, P2T, T2P  
  
def write\_initial\_data(text\_widget, P, T, P2T, T2P):  
 text\_widget.insert(tk.END, "==== Protocol ====\n")  
 text\_widget.insert(tk.END, f"P = {P}\n")  
 text\_widget.insert(tk.END, f"T = {T}\n")  
 text\_widget.insert(tk.END, f"P2T = {P2T}\n")  
 text\_widget.insert(tk.END, f"T2P = {T2P}\n")  
  
def charge\_transitions(text\_widget, np, nt, P, T, P2T):  
 text\_widget.insert(tk.END, "==== charging transitions ====\n")  
 text\_widget.insert(tk.END, "i j P2T[i][j] P[j] flag1 flag2 T[i]\n")  
 for i in range(nt):  
 flag1, flag2 = 0, 0  
 for j in range(np):  
 if P2T[i][j] > 0:  
 if P[j] > 0:  
 flag1 = 1  
 elif P[j] == 0:  
 flag2 = 1  
 text\_widget.insert(tk.END, f"{i} {j} {P2T[i][j]} {P[j]} {flag1} {flag2} {T[i]}\n")  
 if flag1 == 1 and flag2 == 0:  
 for j in range(np):  
 if P2T[i][j] > 0:  
 P[j] -= 1  
 T[i] = 1  
 text\_widget.insert(tk.END, "==== end of charging transitions ====\n")  
  
def discharge\_transitions(text\_widget, np, nt, P, T, T2P):  
 text\_widget.insert(tk.END, "==== discharging transitions ====\n")  
 for i in range(nt):  
 flag1, flag2 = 0, 0  
 for j in range(np):  
 if T2P[i][j] > 0:  
 flag1 = 1  
 if T[i] > 0:  
 flag2 = 1  
 text\_widget.insert(tk.END, f"{i} {j} {T2P[i][j]} {P[j]} {flag1} {flag2} {T[i]}\n")  
 if flag1 == 1 and flag2 == 1:  
 for j in range(np):  
 if T2P[i][j] > 0:  
 P[j] += T2P[i][j]  
 T[i] = 0  
 text\_widget.insert(tk.END, "==== end of discharging transitions ====\n")  
  
def generate\_protocol():  
 np, nt, P, T, P2T, T2P = initialize\_system()  
 text\_widget.delete(1.0, tk.END)  
 write\_initial\_data(text\_widget, P, T, P2T, T2P)  
 charge\_transitions(text\_widget, np, nt, P, T, P2T)  
 discharge\_transitions(text\_widget, np, nt, P, T, T2P)  
 text\_widget.insert(tk.END, "===== The END =====\n")  
 messagebox.showinfo("Info", "Protocol generated successfully!")  
  
root = tk.Tk()  
root.title("Protocol Generator")  
root.geometry("600x600")  
frame = ttk.Frame(root, padding="10")  
frame.grid(row=0, column=0, sticky=(tk.W, tk.E, tk.N, tk.S))  
text\_widget = tk.Text(frame, wrap="word", width=70, height=30)  
text\_widget.grid(row=0, column=0, columnspan=2, sticky=(tk.W, tk.E))  
scrollbar = ttk.Scrollbar(frame, orient="vertical", command=text\_widget.yview)  
scrollbar.grid(row=0, column=2, sticky=(tk.N, tk.S))  
text\_widget['yscrollcommand'] = scrollbar.set  
button\_generate = ttk.Button(frame, text="Generate Protocol", command=generate\_protocol)  
button\_generate.grid(row=1, column=0, pady=10, sticky=(tk.W, tk.E))  
button\_exit = ttk.Button(frame, text="Exit", command=root.quit)  
button\_exit.grid(row=1, column=1, pady=10, sticky=(tk.W, tk.E))  
root.mainloop()

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

Автоматично згенерований опис

Рисунок 4.7 – Графічне відображення інтерфейсу

* 1. Висновок

На даній лабораторній роботі ми закріпили знання про функції вищого порядку та принципи функціонального програмування на мові Python. Також ми реалізували систему переходів мереж Петрі у функціональному та процедурному виглядах. Для зручності відображення інформації протоколу було додано простий графічний інтерфейс.