Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Лабораторна робота №6

з дисципліни
«Об’єктно-орієнтоване програмування»

Виконала:

студентка групи ІМ-43
Хубеджева Єлизавета Павлівна
номер у списку групи: 26

Перевірив:

Порєв В. М.

Київ 2025

Завдання

Так як варіант 26, то Ж mod 4 = 2:

- Lab6:

1. Користувач вводить значення n, Min, Max у діалоговому вікні.

2. Програма викликає програми Object2, 3 і забезпечує обмін повідомленнями для передавання та отримання інформації.

- Object2:

1. Створює вектор n дробових (double) чисел у діапазоні Min – Max

2. Показує числові значення у декількох стовпчиках та рядках у власному головному вікні

3. Записує дані в Clipboard Windows у текстовому форматі

- Object3:

1. Зчитує дані з Clipboard Windows

2. Виконує сортування масиву чисел і відображає його у декількох стовпчиках та рядках у власному головному вікні

Тексти програм:

Lab6.py:

```
import tkinter as tk
from tkinter import messagebox
import subprocess
import sys
import os
import json
import time
import socket
import threading

HOST = '127.0.0.1'
PORT_OBJ2 = 65432
PORT_OBJ3 = 65433

proc_obj2 = None
proc_obj3 = None
script_dir = os.path.dirname(os.path.abspath(__file__))

def is_running(proc):
    if proc is None: return False
    if proc.poll() is not None: return False
    return True

def is_port_open(port):
    with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
        s.settimeout(0.5)
        result = s.connect_ex((HOST, port))
        return result == 0

def run_workflow_thread(n_val, min_val, max_val):
    global proc_obj2, proc_obj3

    print(f"--- Початок роботи: n={n_val}, range=[{min_val}, {max_val}]
---")

    if is_port_open(PORT_OBJ2):
        print("Manager: Object2 вже запущено")
    else:
```

```
        print("Manager: Запуск Object2")
        proc_obj2 = subprocess.Popen([sys.executable,
os.path.join(script_dir, 'Object2.py')])
        time.sleep(0.5)


    if is_port_open(PORT_OBJ3):
        print("Manager: Object3 вже запущено ")
    else:
        print("Manager: Запуск Object3")
        proc_obj3 = subprocess.Popen([sys.executable,
os.path.join(script_dir, 'Object3.py')])
        time.sleep(0.5)


    try:
        print(f"Manager: Надсилання параметрів до Object2")
        with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
            s.connect((HOST, PORT_OBJ2))


            msg_data = json.dumps({"n": n_val, "min": min_val, "max":
max_val})
            s.sendall(msg_data.encode('utf-8'))


            response = s.recv(1024).decode('utf-8')


            if response == "DONE":
                print("Manager: Object2 повідомив про завершення.")
            else:
                print("Manager: Помилка від Object2.")
                return
    except ConnectionRefusedError:
        messagebox.showerror("Помилка", "Object2 не відповідає (порт
закритий).")
        return
    except Exception as e:
        print(f"Помилка комунікації з Object2: {e}")
        return


    try:
        print(f"Manager: Надсилання команди START до Object3")
        with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
            s.connect((HOST, PORT_OBJ3))
```

```
            s.sendall(b"START")

            response = s.recv(1024).decode('utf-8')

            if response == "DONE":
                print("Manager: Object3 повідомив про завершення.")
    except Exception as e:
        print(f"Помилка комунікації з Object3: {e}")


def open_input_dialog():
    dialog = tk.Toplevel(root)
    dialog.title("Введення параметрів")
    dialog.geometry("300x200")
    dialog.grab_set()

    tk.Label(dialog, text="Введіть параметри:", font=("Arial", 10,
"bold")).pack(pady=10)
    frame_d = tk.Frame(dialog)
    frame_d.pack(pady=5)

    tk.Label(frame_d, text="n:").grid(row=0, column=0, padx=5, pady=5)
    d_entry_n = tk.Entry(frame_d, width=10); d_entry_n.grid(row=0,
column=1); d_entry_n.insert(0, "10")

    tk.Label(frame_d, text="Min:").grid(row=1, column=0, padx=5, pady=5)
    d_entry_min = tk.Entry(frame_d, width=10); d_entry_min.grid(row=1,
column=1); d_entry_min.insert(0, "1.0")

    tk.Label(frame_d, text="Max:").grid(row=2, column=0, padx=5, pady=5)
    d_entry_max = tk.Entry(frame_d, width=10); d_entry_max.grid(row=2,
column=1); d_entry_max.insert(0, "100.0")

    def on_submit():
        try:
            n = int(d_entry_n.get())
            mn = float(d_entry_min.get())
            mx = float(d_entry_max.get())
            dialog.destroy()
            threading.Thread(target=run_workflow_thread, args=(n, mn,
mx), daemon=True).start()
        except ValueError:
```

```
            messagebox.showerror("Помилка", "Введіть коректні числа")

    tk.Button(dialog, text="Виконати", command=on_submit, bg="lightblue",
width=15).pack(pady=15)


def send_exit_command(port):
    try:
        with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
            s.settimeout(0.5)
            s.connect((HOST, port))
            s.sendall(b"EXIT")
            print(f"Manager: Надіслано EXIT на порт {port}")
    except (ConnectionRefusedError, socket.timeout):
        pass
    except Exception as e:
        print(f"Помилка при закритті порту {port}: {e}")

def on_closing():
    global proc_obj2, proc_obj3

    if is_running(proc_obj2):
        proc_obj2.terminate()
    else:
        send_exit_command(PORT_OBJ2)

    if is_running(proc_obj3):
        proc_obj3.terminate()
    else:
        send_exit_command(PORT_OBJ3)

    root.destroy()

root = tk.Tk()
root.title("Lab6 Manager")
root.geometry("450x300+50+50")

main_menu = tk.Menu(root)
root.config(menu=main_menu)
file_menu = tk.Menu(main_menu, tearoff=0)
main_menu.add_cascade(label="Файл", menu=file_menu)
```

```
file_menu.add_command(label="Вихід", command=on_closing)
main_menu.add_command(label="Порахувати", command=open_input_dialog)

root.protocol("WM_DELETE_WINDOW", on_closing)
root.mainloop()
```

Object2.py:

```
import tkinter as tk
import random
import json
import socket
import threading

HOST = '127.0.0.1'
PORT = 65432

def server_listener():

    with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
        s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        s.bind((HOST, PORT))
        s.listen()

        while True:
            try:
                conn, addr = s.accept()
                with conn:
                    data = conn.recv(1024)
                    if not data: continue

                    message = data.decode('utf-8')

                    if message == "EXIT":
                        print("Отримано команду EXIT. Завершення
роботи.")
                        root.after(0, root.destroy)
                        break

                    try:
                        params = json.loads(message)
                        done_event = threading.Event()
                        root.after(0, lambda: perform_generation(params,
done_event))
                        done_event.wait()
                        conn.sendall(b"DONE")
                    except json.JSONDecodeError:
```

```
                        pass

            except Exception as e:
                print(f"Server Error: {e}")
                break

def perform_generation(params, event):
    try:
        n = params['n']
        mn = params['min']
        mx = params['max']

        numbers = [round(random.uniform(mn, mx), 2) for _ in range(n)]

        text_area.delete('1.0', tk.END)
        formatted_text = ""
        for i, num in enumerate(numbers):
            formatted_text += f"{num:^10}"
            if (i + 1) % 5 == 0:
                formatted_text += "\n"
        text_area.insert(tk.END, formatted_text)

        root.clipboard_clear()
        root.clipboard_append(json.dumps(numbers))
        root.update()

        print("Object2: Згенеровано і записано.")
    except Exception as e:
        print(f"Error in gen: {e}")
    finally:
        event.set()

root = tk.Tk()
root.title("Object 2")
root.geometry("450x300+530+50")

tk.Label(root, text="Object 2 (Generator)", font=("Arial", 13,
"bold")).pack(pady=5)
text_area = tk.Text(root, height=12, width=60, font=("Consolas", 10))
text_area.pack(pady=5, padx=10)
```

```
t = threading.Thread(target=server_listener, daemon=True)
t.start()

root.mainloop()
```

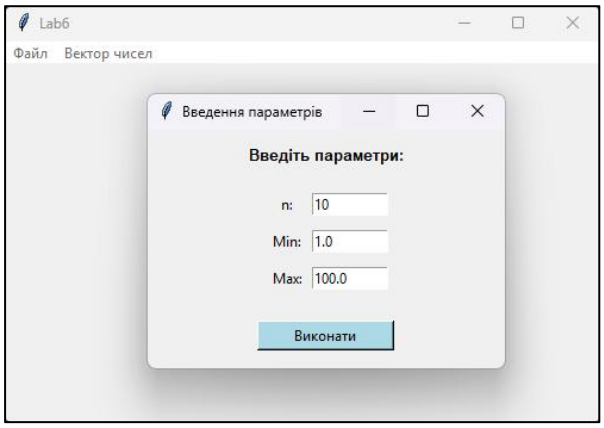
Object3.py:

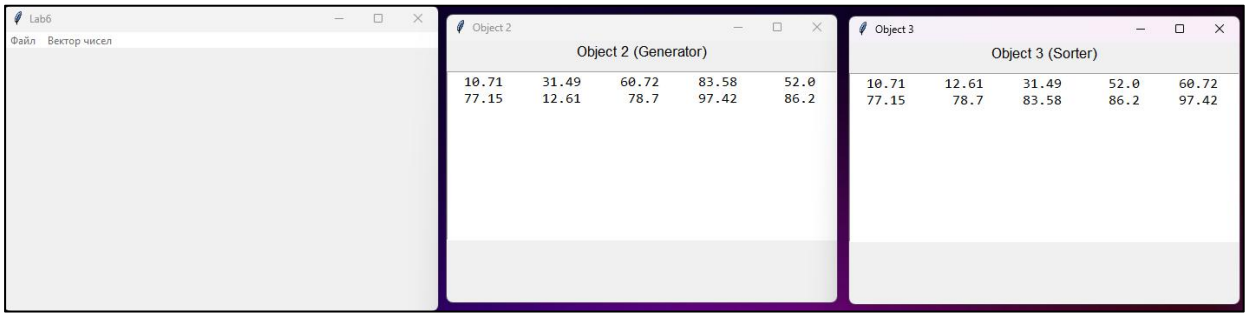
```
import tkinter as tk
import json
import socket
import threading

HOST = '127.0.0.1'
PORT = 65433

def server_listener():
    with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
        s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        s.bind((HOST, PORT))
        s.listen()

        while True:
            try:
                conn, addr = s.accept()
                with conn:
                    cmd = conn.recv(1024).decode('utf-8')

                    if cmd == "EXIT":
                        print("Отримано команду EXIT. Завершення
роботи.")
                        root.after(0, root.destroy)
                        break

                    if cmd == "START":
                        done_event = threading.Event()
                        root.after(0, lambda:
perform_sorting(done_event))
                        done_event.wait()
                        conn.sendall(b"DONE")

            except Exception as e:
                print(f"Server Error: {e}")
                break

def perform_sorting(event):
    try:
```

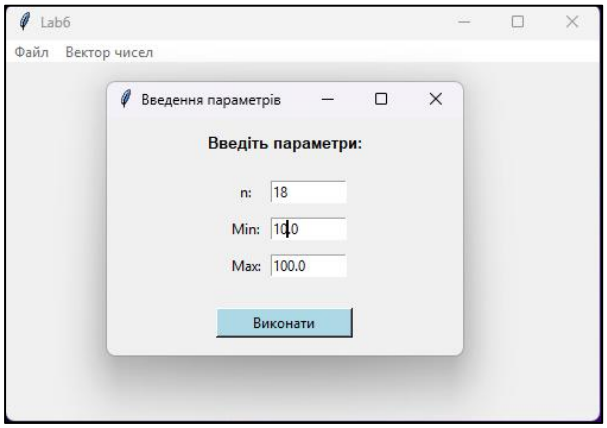
```
            content = root.clipboard_get()
            numbers = json.loads(content)

            numbers.sort()

            text_area.delete('1.0', tk.END)
            formatted_text = ""
            for i, num in enumerate(numbers):
                formatted_text += f"{num:^10}"
                if (i + 1) % 5 == 0:
                    formatted_text += "\n"
            text_area.insert(tk.END, formatted_text)

            print("Object3: Відсортовано.")
        except Exception as e:
            print(f"Error in sort: {e}")
        finally:
            event.set()

root = tk.Tk()
root.title("Object 3")
root.geometry("450x300+1000+50")

tk.Label(root, text="Object 3 (Sorter)", font=("Arial", 13,
"bold")).pack(pady=5)
text_area = tk.Text(root, height=12, width=60, font=("Consolas", 10))
text_area.pack(pady=5, padx=10)

t = threading.Thread(target=server_listener, daemon=True)
t.start()

root.mainloop()
```

Скріншоти:


Lab6
Файл
Вектор чисел


Lab6
Файл
Вектор чисел
Введення параметрів
Введіть параметри:
n:
10
Min:
1.0
Max:
100.0
Виконати


Lab6
Файл
Вектор чисел
Object 2
Object 2 (Generator)
10.71 31.49 60.72 83.58 52.0
77.15 12.61 78.7 97.42 86.2
Object 3
Object 3 (Sorter)
10.71 12.61 31.49 52.0 60.72
77.15 78.7 83.58 86.2 97.42


Lab6
Файл
Вектор чисел
Введення параметрів
Введіть параметри:
n:
18
Min:
100
Max:
100.0
Виконати

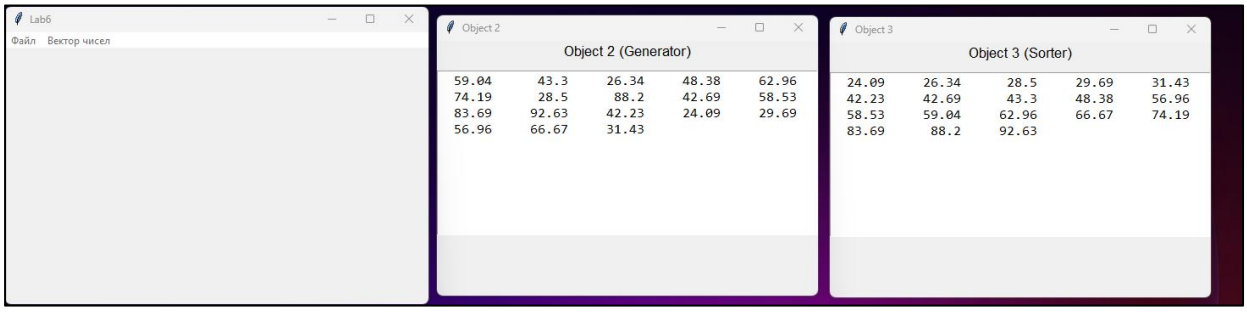
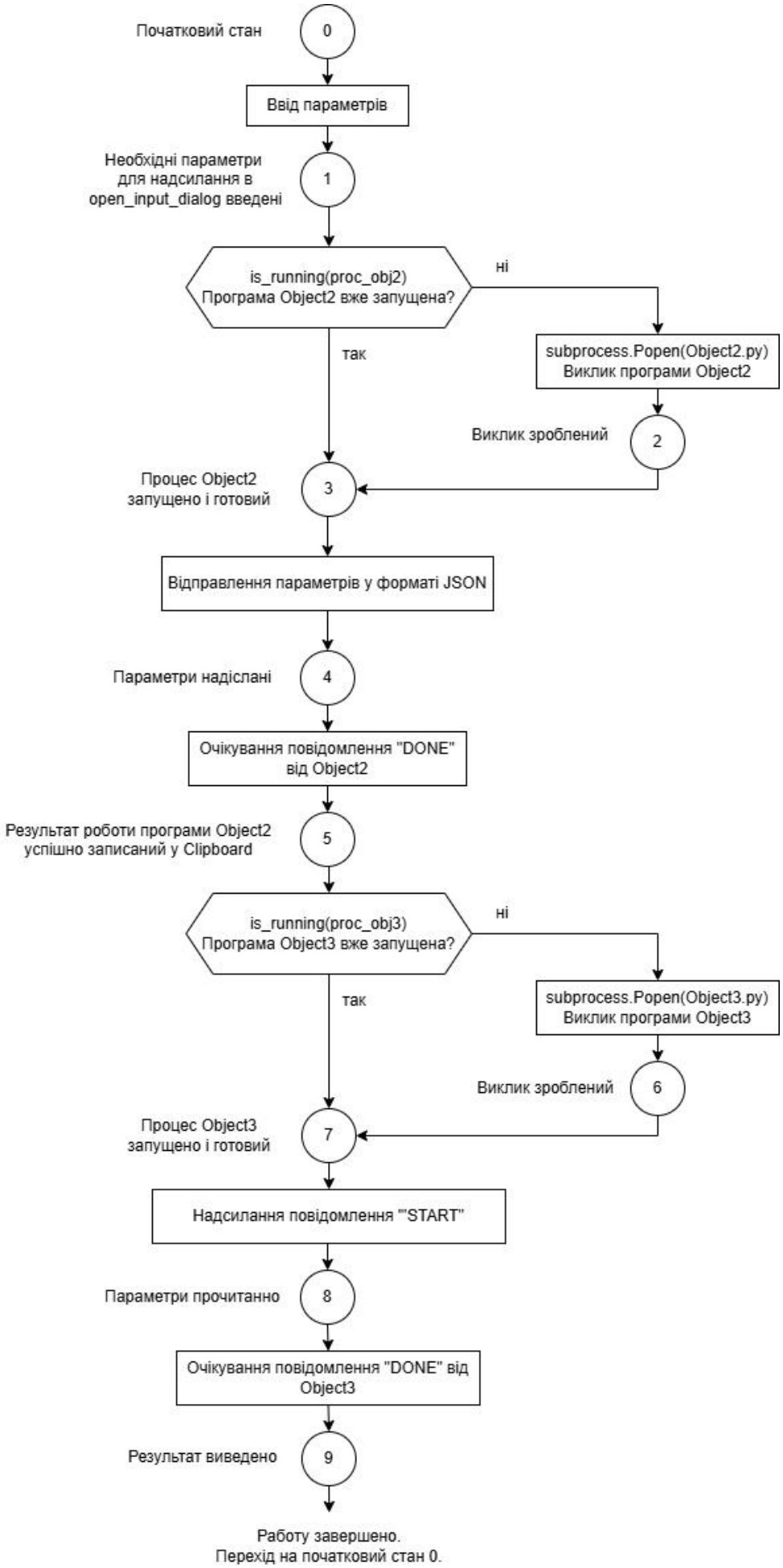

Lab6
Файл
Вектор чисел
Object 2
Object 2 (Generator)
59.04 43.3 26.34 48.38 62.96
74.19 28.5 88.2 42.69 58.53
83.69 92.63 42.23 24.09 29.69
56.96 66.67 31.43
Object 3
Object 3 (Sorter)
24.09 26.34 28.5 29.69 31.43
42.23 42.69 43.3 48.38 56.96
58.53 59.04 62.96 66.67 74.19
83.69 88.2 92.63

Схема послідовності надсилання-обробки повідомлень:



Висновок

В ході даної лабораторної роботи було запрограмовано програмну систему з множини окремих незалежних програм, керованих повідомленнями. В результаті було створено три незалежні програми, які взаємодіють через TCP/IP сокети.

Файл з основною програмою має назву Lab6.py. У ньому реалізовано головне вікно менеджера з меню, де можна обрати пункт "Порахувати" для виклику діалогового вікна введення даних. При натисканні кнопки "Виконати" запускається окремий потік для автоматичної обробки, що не блокує інтерфейс користувача.

Lab6 виконує перевірку доступності додатків через перевірку відкритих портів (65432 для Object2 та 65433 для Object3). Якщо програми не запущені, він автоматично їх запускає за допомогою subprocess.Popen(). Після цього Lab6 встановлює TCP-з'єднання з Object2 та надсилає йому параметри (n, min, max) у форматі JSON. Object2 відповідає "DONE" після завершення обробки.

Програма Object2 працює як TCP-сервер на порту 65432. Вона постійно слухає вхідні з'єднання в окремому потоці. При отриманні параметрів вона генерує вектор n дробових чисел у заданому діапазоні, відображає їх у власному вікні у декількох стовпчиках та рядках (по 5 чисел у рядку) та зберігає результат у буфері обміну у форматі JSON.

Після успішного завершення Object2, Lab6 встановлює з'єднання з Object3 на порту 65433 та надсилає команду "START". Програма Object3 також працює як TCP-сервер у окремому потоці. При отриманні команди "START" вона зчитує дані з буферу обміну, сортує масив чисел та відображає відсортований вектор у власному вікні у декількох стовпчиках та рядках.

Для коректного завершення роботи всієї системи Lab6 надсилає команди "EXIT" обом програмам при закритті, що гарантує їх коректне завершення. Усі вікна автоматично розташовуються на екрані для зручного перегляду результатів.

Таким чином, було реалізовано надійну систему з трьох незалежних програм, які обмінюються повідомленнями через TCP/IP сокети, що забезпечує швидку та стабільну комунікацію між процесами.