ОСВІТИ І НАУКИ УКРАЇНИ
Національний аерокосмічний університет ім. М. Є. Жуковського
«Харківський авіаційний інститут»

Кафедра систем управління літальними апаратами

Лабораторна робота № 5

з дисципліни «Об'єктно-орієнтоване проектування СУ»

Тема: «Реалізація класу і робота з об'єктами»

ХАІ.301 . 3.320.5 ЛР

Виконав студент гр. 320

_____Семеняга Ігор___
(підпис, дата) (П.І.Б.)

Перевірив

_____ к.т.н., доц. О. В. Гавриленко
(підпис, дата) (П.І.Б.)

2024

МЕТА РОБОТИ

Застосувати теоретичні знання з основ роботи з бібліотекою tkinter на мові Python, навички використання бібліотеки matplotlib, а також об'єктно-орієнтований підхід до проектування програм, і навчитися розробляти скрипти для інженерних додатків з графічним інтерфейсом.

ПОСТАНОВКА ЗАДАЧІ

Завдання 1. Описати клас, який реалізує графічний інтерфейс користувача для вирішення розрахункової задачі згідно варіанту (див. табл.1) і скрипт для роботи з об'єктом цього класу. Зазначена у задачі функція повинна бути окремим методом класу.

Завдання 2. Розробити скрипт із графічним інтерфейсом, що виконує наступні функції:

A. установка початкових значень параметрів для побудови графіка (змінні Tkinter)

B. створення текстового файлу з двома стовпцями даних: аргумент і значення функції відповідно до варіанту (див. табл.2). Роздільник в кожному рядку файлу: для парних варіантів – ';', для непарних – '#';

C. зчитування з файлу масивів даних;

D. підрахунок і відображення мінімального / максимального значення аргументу / функції у зчитаних масивах;

E. відображення масивів даних за допомогою пакета matplotlib у вигляді графіка функції в декартовій системі координат з назвою функції, позначенням осей, оцифруванням і сіткою;

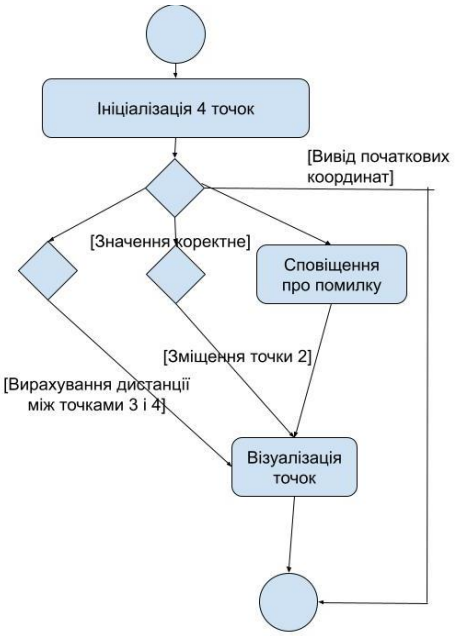
F. заголовок вікна повинен містити текст текст: lab # - <# групи> -v <# варіанту> - <прізвище> - <ім'я>, наприклад:

lab4_2-320-v01-Ivanov-Ivan

№	Рекурентний вираз	Поч. умови	Параме-три	Фіз. сенс
2	$y[k+2] = \left(2 - \frac{2 \cdot \xi \cdot T_0}{T}\right) \cdot y[k+1] + \left(\frac{2 \cdot \xi \cdot T_0}{T} - 1 - \frac{T_0^2}{T^2}\right) \cdot y[k] + \frac{K \cdot T_0^2}{T^2} U$	U [0] = 0.1 рад / с, y [0] == y [1] = 0	$T = 0{,}1$ $K = 3$ $\xi = 0{,}2$	y – υ, рад U – δ_B, рад

ВИКОНАННЯ РОБОТИ

Завдання 1. Вирішення задачі 1 (13)



Описати функцію IsPrime(N) логічного типу, що повертає True, якщо цілий параметр N (> 1) є простим числом, і False в іншому випадку (число, більше 1, називається простим, якщо воно не має додатних дільників, крім 1 і самого себе). Даний набір із 10 цілих чисел, більших за 1. За допомогою функції IsPrime знайти кількість простих чисел у цьому наборі.

Вхідні дані (ім’я, опис, тип, обмеження):
numbers_input = input("Введіть числа через пробіл: ")

Вихідні дані (ім’я, опис, тип):
print(f"Кількість простих чисел у наборі: {prime_count}") int

Алгоритм вирішення

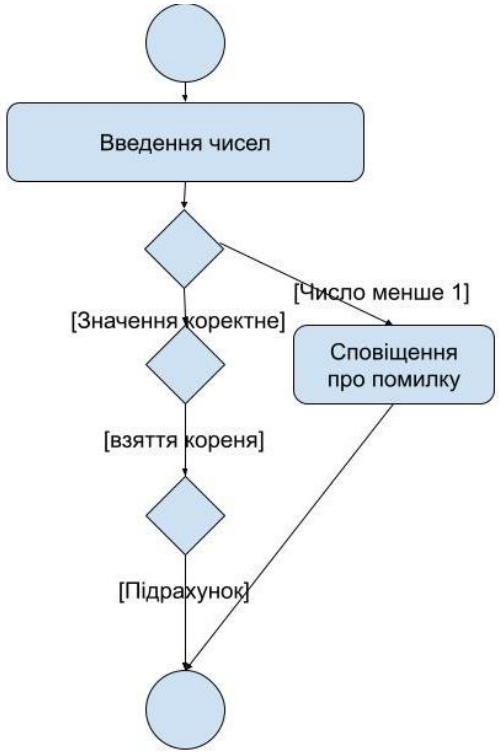


Рисунок 1 – Алгоритм роботи завдання

Завдання 2. Вирішення задачі 2 (4)

Розробити скрипт із графічним інтерфейсом, що виконує наступні функції:

A. установка початкових значень параметрів для побудови графіка (змінні Tkinter)

B. створення текстового файлу з двома стовпцями даних: аргумент значення функції відповідно до варіанту (див. табл.2). Роздільник в кожному рядку файлу: для парних варіантів – ';', для непарних – '#';

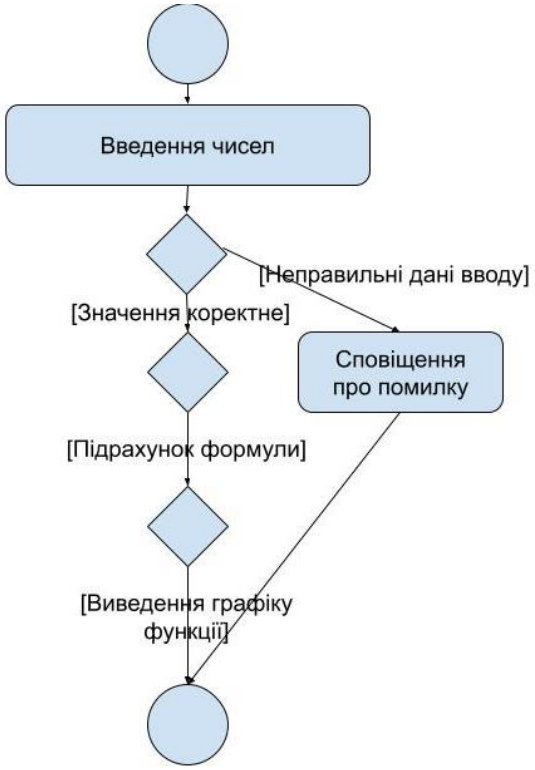
C. зчитування з файлу масивів даних;

D. підрахунок і відображення мінімального / максимального значення аргументу / функції у зчитаних масивах;

E. відображення масивів даних за допомогою пакета matplotlib у вигляді графіка функції в декартовій системі координат з назвою функції, позначенням осей, оцифруванням і сіткою;

№	Рекурентний вираз	Поч. умови	Параме-три	Фіз. сенс
2	$y[k+2] = \left(2 - \frac{2 \cdot \xi \cdot T_0}{T}\right) \cdot y[k+1] + \left(\frac{2 \cdot \xi \cdot T_0}{T} - 1 - \frac{T_0^2}{T^2}\right) \cdot y[k] + \frac{K \cdot T_0^2}{T^2} U$	U [0] = 0.1 рад / с, y [0] == y [1] = 0	$T = 0{,}1$ $K = 3$ $\xi = 0{,}2$	y – υ, рад U – $\delta_{\text{В}}$, рад

Вхідні дані (ім’я, опис, тип, обмеження):
ttk.Label(frame, text="T (період):") float
ttk.Label(frame, text="K (коефіцієнт):") float
ttk.Label(frame, text="tau (постійна часу):") float
ttk.Label(frame, text="N (кількість точок):") float
Вихідні дані (ім’я, опис, тип):
plot_graph(t, y, f"Графік функції y(t), T={T}, K={K}, tau={tau}") str
lbl_results = ttk.Label(frame, text="", foreground="blue", padding="10")



Алгоритм вирішення

Рисунок 2 – Алгоритм роботи завдання 2

Лістинг коду вирішення задачі наведено в дод. А (стор. 7). Екран роботи програми показаний на рис. Б.8.

ВИСНОВКИ

Було застосовано теоретичні знання щодо роботи з бібліотекою Tkinter на Python, використання matplotlib та об'єктно-орієнтованого підходу для проектування програм. Також були здобуті навички розробки скриптів для інженерних додатків з графічними інтерфейсами.

ДОДАТОК А

Лістинг коду програми до задач №11

```
import tkinter as tk
from tkinter import ttk, filedialog, messagebox
import numpy as np
import matplotlib.pyplot as plt


def main_menu():
    while True:
        print("\nМеню:")
        print("1. Завдання 1")
        print("2. Завдання 2")
        print("3. Вихід")

        choice = input("Оберіть опцію (1-3): ")

        if choice == "1":

            def IsPrime(N):
                """
                Функція перевіряє, чи є число N простим.
                Повертає True, якщо число просте, і False в іншому випадку.
                """
                if N <= 1:
                    return False
                for i in range(2, int(N**0.5) + 1):
                    if N % i == 0:
                        return False
                return True

            # Введення чисел від користувача
            try:
                numbers_input = input("Введіть числа через пробіл: ")
                numbers = list(map(int, numbers_input.split()))
            except ValueError:
                print("Будь ласка, введіть тільки цілі числа через пробіл.")
                continue

            # Підрахунок кількості простих чисел
            prime_count = sum(1 for num in numbers if IsPrime(num))

            # Результат
            print(f"Кількість простих чисел у наборі: {prime_count}")

        elif choice == "2":



            # Функція для обчислення значень за формулою
```

```
            def calculate_function(params, N):
                T, K, tau = params["T"], params["K"], params["tau"]
                T0 = 2 * T / N  # Крок часу
                t = np.linspace(0, 2 * T, N)  # Масив часу
                y = np.zeros_like(t)  # Масив для результатів

                # Рекурсивний підхід
                for k in range(2, len(t)):
                    y[k] = (2 - (2 * T0 / tau)) * y[k - 1] - (1 - (T0 / tau)) *
y[k - 2] + K * (T0 / tau)

                return t, y


            # Збереження даних у файл
            def save_to_file(t, y, separator):
                filename = filedialog.asksaveasfilename(defaultextension=".txt",
filetypes=[("Text files", "*.txt")])
                if not filename:
                    return
                try:
                    with open(filename, "w") as f:
                        for ti, yi in zip(t, y):
                            f.write(f"{ti:.5f}{separator}{yi:.5f}\n")
                    messagebox.showinfo("Успіх", f"Дані збережено у файл:
{filename}")
                except Exception as e:
                    messagebox.showerror("Помилка", f"Не вдалося зберегти файл:
{e}")


            # Побудова графіка
            def plot_graph(t, y, title):
                plt.figure(figsize=(8, 6))
                plt.plot(t, y, label="y(t)", color="blue")
                plt.title(title)
                plt.xlabel("Час t, сек")
                plt.ylabel("Значення y(t)")
                plt.grid(True)
                plt.legend()
                plt.show()


            # Функція для запуску обчислень
            def run_calculations():
                try:
                    T = float(entry_T.get())
                    K = float(entry_K.get())
                    tau = float(entry_tau.get())
                    N = int(entry_N.get())
                    if N <= 0:
```

```
                    raise ValueError("Кількість точок має бути більше 0.")

                params = {"T": T, "K": K, "tau": tau}
                t, y = calculate_function(params, N)

                # Відображення мінімальних і максимальних значень
                min_t, max_t = np.min(t), np.max(t)
                min_y, max_y = np.min(y), np.max(y)
                lbl_results["text"] = f"Мінімальне t: {min_t:.2f},
Максимальне t: {max_t:.2f}\n" \
                                      f"Мінімальне y: {min_y:.2f},
Максимальне y: {max_y:.2f}"

                # Побудова графіка
                plot_graph(t, y, f"Графік функції y(t), T={T}, K={K},
tau={tau}")

            except ValueError as e:
                messagebox.showerror("Помилка", f"Невірні дані: {e}")


        # Інтерфейс Tkinter
        root = tk.Tk()
        root.title("lab5 - 320 - v01 - Семеняга Ігор")

        # Поля введення
        frame = ttk.Frame(root, padding="10")
        frame.grid(row=0, column=0, sticky=(tk.W, tk.E, tk.N, tk.S))

        ttk.Label(frame, text="T (період):").grid(row=0, column=0,
sticky=tk.W)
        entry_T = ttk.Entry(frame, width=10)
        entry_T.grid(row=0, column=1)

        ttk.Label(frame, text="K (коефіцієнт):").grid(row=1, column=0,
sticky=tk.W)
        entry_K = ttk.Entry(frame, width=10)
        entry_K.grid(row=1, column=1)

        ttk.Label(frame, text="tau (постійна часу):").grid(row=2, column=0,
sticky=tk.W)
        entry_tau = ttk.Entry(frame, width=10)
        entry_tau.grid(row=2, column=1)

        ttk.Label(frame, text="N (кількість точок):").grid(row=3, column=0,
sticky=tk.W)
        entry_N = ttk.Entry(frame, width=10)
        entry_N.grid(row=3, column=1)

        # Кнопки
```

```
            btn_calculate = ttk.Button(frame, text="Обчислити",
command=run_calculations)
            btn_calculate.grid(row=4, column=0, columnspan=2)

            btn_save = ttk.Button(frame, text="Зберегти у файл", command=lambda:
save_to_file(t, y, ";"))
            btn_save.grid(row=5, column=0, columnspan=2)

            # Поле для результатів
            lbl_results = ttk.Label(frame, text="", foreground="blue",
padding="10")
            lbl_results.grid(row=6, column=0, columnspan=2)

            root.mainloop()


        elif choice == "3":
            print("Вихід з програми.")
            break

        else:
            print("Неправильний вибір. Спробуйте ще раз.")

if __name__ == "__main__":
    main_menu()
```

ДОДАТОК Б

Скрін-шоти вікна виконання програми

```
Меню:
1. Завдання 1
2. Завдання 2
3. Вихід
Оберіть опцію (1-3): 1
Введіть числа через пробіл: 2 3 4 5
Кількість простих чисел у наборі: 3
```

Рисунок Б.1 – Екран виконання програми для вирішення завдання 1-№13

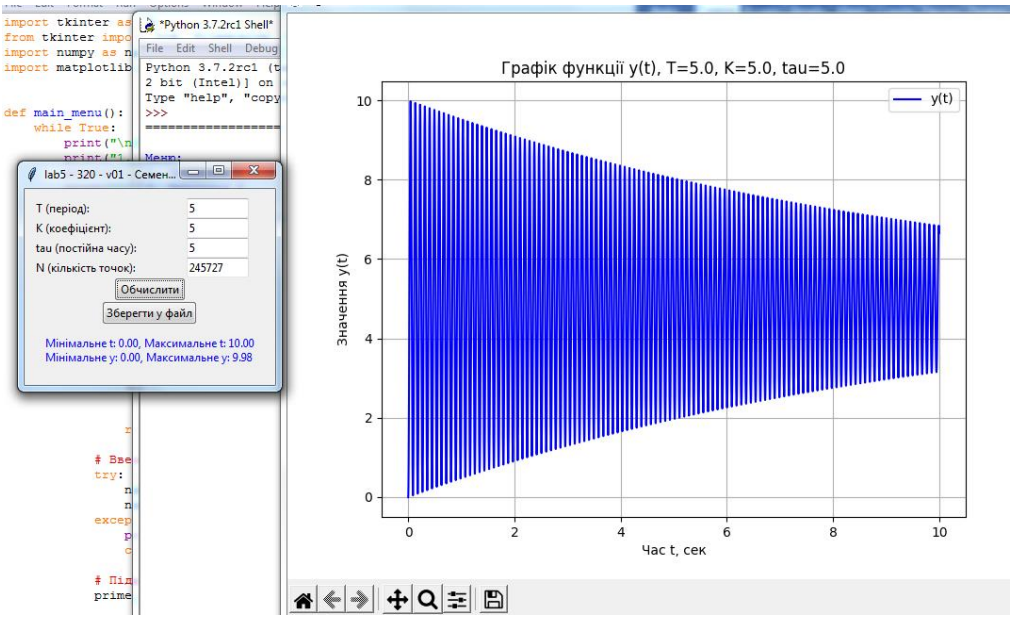


Рисунок Б.2– Екран виконання програми для вирішення завдання 2-№4