МІНІСТЕРСТВО 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 програмування СУ»

Тема:«Розробка графічного інтерфейсу для розрахункових завдань і побудови графіків»

Виконав студент гр.
_____321__Новіков Олексій

Перевірив
к.т.н., доц. О. В. Гавриленко
ас. В. О. Білозерський

2023

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.

Func5. Описати функцію TrianglеP(a, h), що знаходить периметр рівнобедреного трикутника з його основи a та висоті h, проведеної до основи (a та h — дійсні). За допомогою цієї функції знайти периметри трьох трикутників, для яких дано основи та висоти. Для знаходження бічної сторони трикутника b використовувати теорему Піфагора: b2 = (a/2)2 + h2.

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;

19	$y[k+1] = \left(1 - \frac{T_0}{T}\right) \cdot y[k] + \frac{T_0}{T} \cdot K \cdot U$	U [0] = 2 Вт, y [0] = 0	$T = 0{,}7$ $K = 2{,}8$	$y - T, K$ $U -$ Qн, Вт

Лістинг коду

main

Підключення створенних вікон

```
import tkinter
from task1 import TriangleCalculator
from task2 import Task2Window

# словник для швидкого доступу до відповідної функції виконання
task_window_dict = {
    "1": (TriangleCalculator, "Lab5_1-321-v5-Novikov-Oleksiy", "300x200"),
    "2": (Task2Window, "Lab5_2-321-v19-Novikov-Oleksiy", "600x300")
}


# Основна функція
def main():
    choice = input("Please, choose the task 1-2 (0-EXIT): ")
    while choice != "0":
        # якщо даний ключ є у словнику
```

```
        if choice in task_window_dict.keys():
            # Створення відповідного вікна
            application = tkinter.Tk()
            window_class, window_name, window_size = task_window_dict.get(choice)
            window = window_class(application)
            application.geometry(window_size)
            application.title(window_name)
            application.mainloop()
        else:
            print("Wrong task number!")
        choice = input("Please, choose the task again (0-EXIT): ")


if __name__ == '__main__':
    main()
        Task1
import tkinter
from tkinter import messagebox

class TriangleCalculator(tkinter.Frame):
    """Graphical user interface and logic for calculating the perimeter of an isosceles triangle"""

    def __init__(self, parent):
        super().__init__(parent)
        self.pack(fill=tkinter.BOTH, expand=1)
        self.grid_rowconfigure(0, weight=1)
        self.grid_rowconfigure(1, weight=1)
        self.grid_columnconfigure(0, weight=1)
        self.grid_columnconfigure(1, weight=1)
        self.grid_columnconfigure(2, weight=1)

        self.lb1 = tkinter.Label(self, text="Enter base length a:")
```

```
        self.lb2 = tkinter.Label(self, text="Enter height h:")
        self.lb3 = tkinter.Label(self, text="Perimeter:")
        self.lb4 = tkinter.Label(self, text="sm")
        self.a_entr = tkinter.Entry(self)
        self.h_entr = tkinter.Entry(self)
        self.btn1 = tkinter.Button(self, text="Calculate perimeter",
command=self.calc_perimeter)
        self.p_str = tkinter.StringVar()
        self.result_label = tkinter.Label(self, textvariable=self.p_str)

        self.lb1.grid(row=0, column=0, sticky=tkinter.NSEW)
        self.a_entr.grid(row=0, column=1, sticky=tkinter.NSEW)
        self.lb2.grid(row=1, column=0, sticky=tkinter.NSEW)
        self.h_entr.grid(row=1, column=1, sticky=tkinter.NSEW)
        self.btn1.grid(row=2, column=0, columnspan=2,
sticky=tkinter.NSEW)
        self.lb3.grid(row=3, column=0, sticky=tkinter.NSEW)
        self.result_label.grid(row=3, column=1, sticky=tkinter.NSEW)
        self.lb4.grid(row=3, column=2, sticky=tkinter.NSEW)

    def calc_perimeter(self):
        try:
            a = float(self.a_entr.get())
            h = float(self.h_entr.get())
        except ValueError:
            messagebox.showerror("Data ERROR", "Side and height must be
numbers!")
            self.a_entr.delete(0, tkinter.END)
            self.h_entr.delete(0, tkinter.END)
        else:
            if a < 0 or h < 0:
                a = abs(a)
                h = abs(h)
                self.a_entr.delete(0, tkinter.END)
```

```
            self.h_entr.delete(0, tkinter.END)
            self.a_entr.insert(tkinter.END, str(a))
            self.h_entr.insert(tkinter.END, str(h))
            messagebox.showinfo("Data Warning", "Negative values changed to positive")

        b = (a / 2) ** 2 + h ** 2
        b = b ** 0.5  # square root
        P = a + 2 * b
        self.p_str.set(f"{P:.2f}")
    Task2
    # Для графічного інтерфейсу
import tkinter
from tkinter import messagebox
from tkinter.filedialog import askopenfile
# Для малювання графіка
from pylab import *
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
from matplotlib.figure import Figure
from PIL import Image, ImageTk


class Task2Window(tkinter.Frame):
    """Клас MainWindow, що наслідує Frame"""

    def __init__(self, parent):
        """Настройка графічного інтерфейсу"""
        super().__init__(parent)
        # Розтягнути фрейм
        self.pack(fill=tkinter.BOTH, expand=1)
        # Розтягнути сітку
        self.grid_rowconfigure(0, weight=1)
        self.grid_rowconfigure(1, weight=1)
        self.grid_rowconfigure(2, weight=1)
```

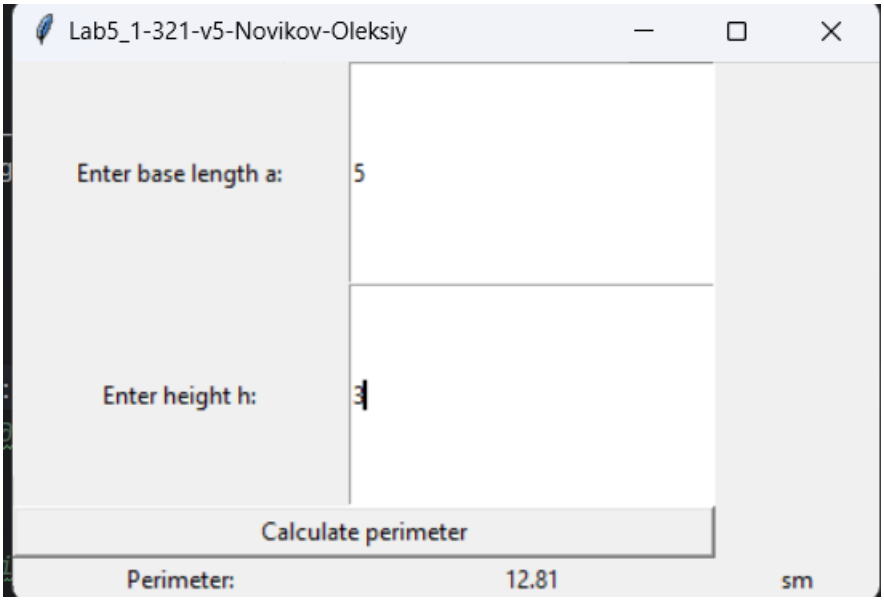
```
        self.grid_columnconfigure(0, weight=1)
        self.grid_columnconfigure(1, weight=1)
        self.grid_columnconfigure(2, weight=1)
        self.grid_columnconfigure(3, weight=1)
        # Створення віджетів (зображення виразу та поле для введення N)
        self.img = ImageTk.PhotoImage(file='image.png')
        self.lb_image = tkinter.Label(self, image=self.img)
        self.lb1 = tkinter.Label(self, text="N = ")
        self.N_entr = tkinter.Entry(self)
        # Створення віджетів (4 командні кнопки)
        self.but1 = tkinter.Button(self, text="Create file", command=self.create_file)
        self.but2 = tkinter.Button(self, text="Open file", command=self.open_file)
        self.but3 = tkinter.Button(self, text="Show content", command=self.show_msg)
        self.but4 = tkinter.Button(self, text="Show plot", command=self.show_plot)
        # Розміщення віджетів в сітці основного вікна
        self.lb_image.grid(row=0, column=0, columnspan=2, sticky=tkinter.NSEW)
        self.lb1.grid(row=0, column=2, sticky=tkinter.NSEW)
        self.N_entr.grid(row=0, column=3, sticky=tkinter.NSEW)
        self.but1.grid(row=1, column=0, sticky=tkinter.NSEW)
        self.but2.grid(row=1, column=1, sticky=tkinter.NSEW)
        self.but3.grid(row=1, column=2, sticky=tkinter.NSEW)
        self.but4.grid(row=1, column=3, sticky=tkinter.NSEW)
        self.text1 = ""  # вміст файлу

    def create_file(self):
        """Розрахунок значень функції і збереження результатів у файл"""
        try:
```

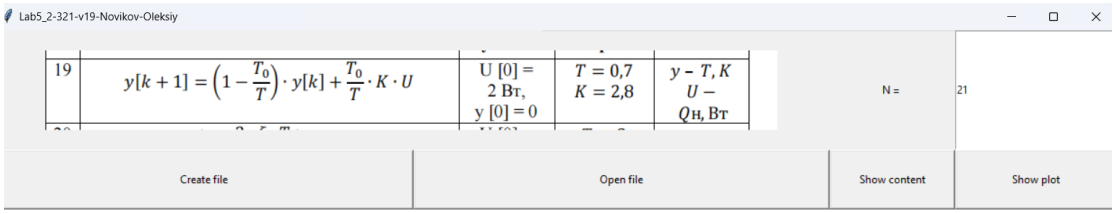
```
            N = int(self.N_entr.get())
            if N < 20:
                raise ValueError
        except ValueError:
            messagebox.showerror("Data ERROR", "N must be integer that >= 20!")
        else:
            # Параметри виразу
            K = 2.8
            T = 0.7
            T0 = 2*T/N
            U = 2
            x = [0]
            y = [0]
            # Розрахунок N значень x, y
            for k in range(1, N):
                x.append(k*T0)
                tmp_value = (1 - T0/T) * y[k-1] + T0/T * K * U
                y.append(tmp_value)
            # збереження результатів у файл
            with open("graph_data.txt", 'w') as f:
                for i, x in enumerate(x):
                    f.write("{}#{}\n".format(x, y[i]))
            # повідомлення про успішний запис результатів у файл
            messagebox.showinfo("File creation", "File with data was created!")

    def open_file(self):
        """Зчитування вмісту файлу і збереження в text1"""
        # Виклик вікна діалогу для відкриття файлу
        fopen = askopenfile(mode='r', defaultextension=". txt",
                            filetypes=(("Text files", "* .txt"), ("All files", "*. *")))
        if fopen is None:  # якщо помилка відкриття файлу
            return
```

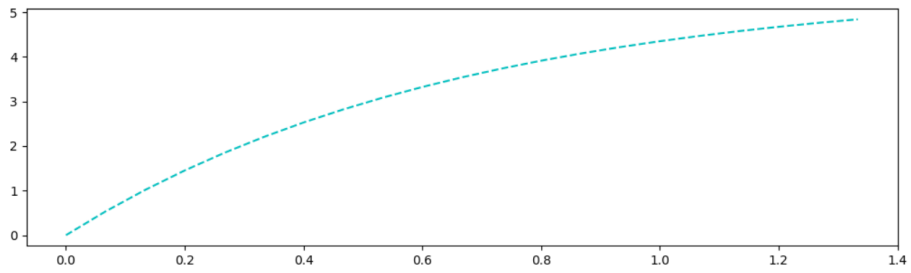
```
        self.text1 = fopen.readlines()  # файл -> список рядків
        messagebox.showinfo("File opening", "File with data was opened!")

    def show_msg(self):
        """Відобразити text1 у вікні messagebox"""
        messagebox.showinfo("File content", self.text1)

    def show_plot(self):
        """Рисування графіку функції"""
        x = []
        y = []
        try:  # розібрати список рядків text1
            for line in self.text1:  # для кожного рядка
                words = line.split('#')  # зберегти як список
                x.append(float(words[0]))  # 1 ел.списка -> число -> x
                y.append(float(words[1]))  # 2 ел.списка -> число -> y
        except ValueError:
            messagebox.showerror("Data ERROR", "Wrong file format!")
        else:
            # Область малювання графіка на полотні (Canvas)
            fig = Figure(figsize=(3, 3))  # створення об'єкта Figure
            a = fig.add_subplot(111)  # створення об'єкта області малювання (subplot)
            # Настройка області побудови графіка
            a.plot(x, y, 'c--')
            # ...
            # Створення об'єкта Canvas і розміщення в основному вікні
            drawing = FigureCanvasTkAgg(fig, master=self)
            drawing.get_tk_widget().grid(row=2, column=0, columnspan=4, sticky=tkinter.NSEW)
            drawing.draw()
            # Інформація про максимальне/мінімальне значення аргументу/функції
            min_x = min(x)
```

```
        min_y = min(y)
        max_x = max(x)
        max_y = max(y)
        messagebox.showinfo("Basic information", "X min = {}, X max = {}\n"
                                                 "Y min = {}, Y max = {}".format(min_x, max_x, min_y, max_y))
```

Виконання







Висновок

У процессі виконання роботи я отримав теоретичні знання з основ програмування на мові Python звикористанням об'єктів і

класів, навички використання бібліотеки для візуалізації масивів даних, і навчитися розробляти скрипти для роботи з об'єктами призначених для користувача класів.