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 проектування СУ»

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

ХАІ.301. 173. 3-91АВ/С. 5 ЛР

Виконав студент гр. \_\_\_\_\_\_3-91АВ/С\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Опанасюк Олександр\_\_\_

(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:

А. 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> - - , наприклад: lab4\_2-320-v01-Ivanov-Ivan

# ВИКОНАННЯ РОБОТИ

Завдання 1 – Розрахунок площі кіл

Вхідні дані (ім’я, опис, тип, обмеження):

r1 – радіус першого кола; тип: float; допустимі значення: [0, ∞).

r2 – радіус другого кола; тип: float; допустимі значення: [0, ∞).

r3 – радіус третього кола; тип: float; допустимі значення: [0, ∞).

Вихідні дані (ім’я, опис, тип):

area1 – площа першого кола; тип: float.

area2 – площа другого кола; тип: float.

area3 – площа третього кола; тип: float.

Текстові повідомлення – у разі помилок введення або виведення результатів.

Алгоритм вирішення показано нижче (рис.1):

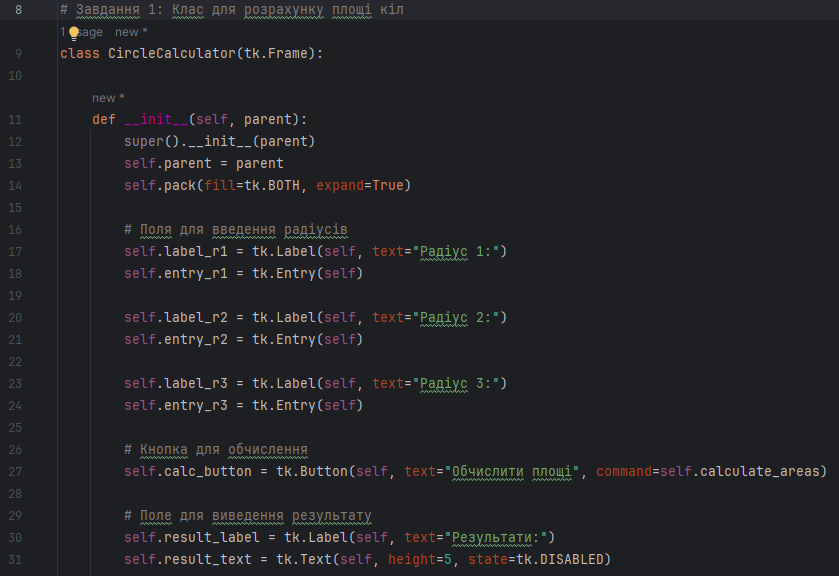
* Користувач вводить значення радіусів (r1, r2, r3).
* Перевіряється, чи є радіуси невід'ємними числами. У разі помилки виводиться повідомлення.
* Для кожного радіуса обчислюється площа за формулою:

Площа кола = π ⋅

* Результати обчислень виводяться на екран.

Реалізовано клас CircleCalculator з наступними методами:

* Введення значень радіусів через графічний інтерфейс.
* Перевірка на коректність даних.
* Розрахунок площі кіл.
* Виведення результатів у текстове поле.



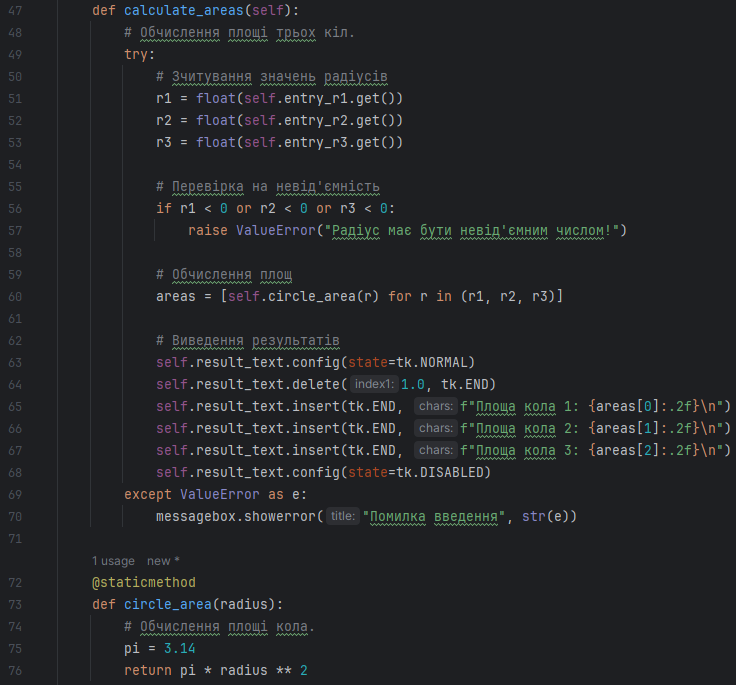


Рисунок 1 – Алгоритм вирішення завдання 1

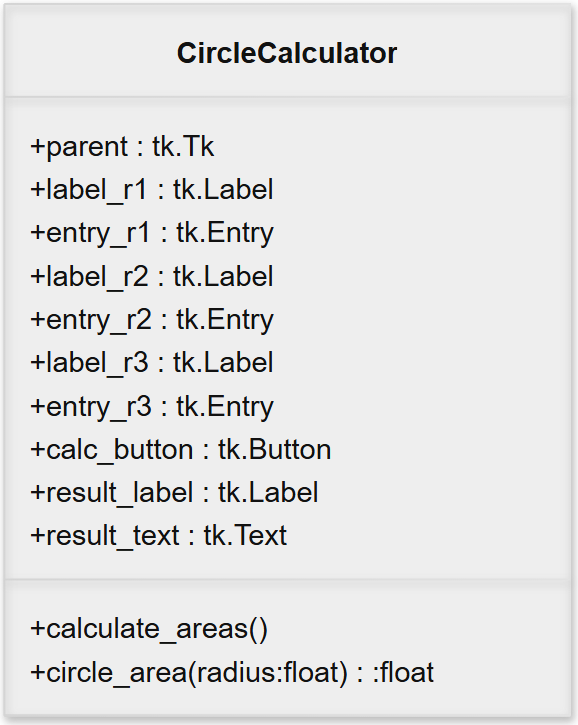


Рисунок 1.1 – Діаграма класів для завдання 1

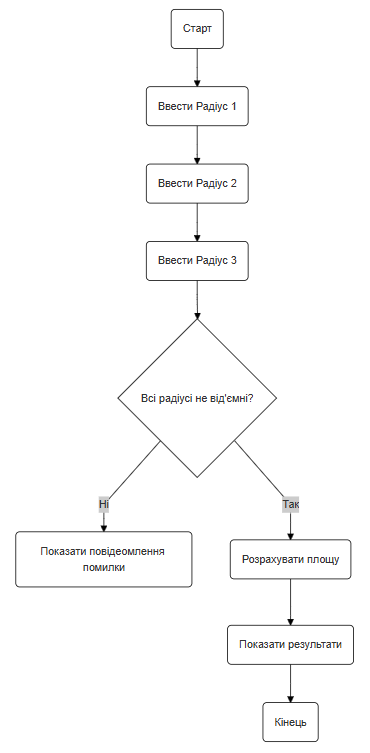


Рисунок 1.2 – Діаграма активності для завдання 1

Завдання 2 – Робота з графіками та файлами

**Вхідні дані (ім’я, опис, тип, обмеження):**

t – час у секундах; тип: float; допустимі значення: [0, 1] (з кроком 0.01).

y – значення функції y = 3.14⋅sin(2πt); тип: float.

**Вихідні дані (ім’я, опис, тип):**

* Файл з парами значень t і y, збережений у текстовому форматі.
* Графік функції y = 3.14⋅sin(2πt) побудований на основі даних із файлу.
* Текстові повідомлення – підтвердження успішного створення або зчитування файлу, а також попередження про відсутність даних.

**Алгоритм вирішення показано нижче:**

**-** Генерація значень:

- Масив часу t із 101 точкою в межах [0, 1] з кроком 0.01.

- Масив значень функції y = 3.14⋅sin(2πt); Запис даних до текстового файлу у форматі t;y.

- Зчитування даних із файлу.

- Побудова графіка функції за зчитаними даними:

- По осі x відкладається t.

- По осі y відкладається відповідне значення y.

- Відображення графіка на Canvas у графічному інтерфейсі

**Реалізовано клас** GraphApp **з наступними методами:**

- create\_file() – створення файлу з обчисленими значеннями.

- open\_file() – зчитування даних із файлу.

- plot\_graph() – побудова графіка функції.

Обидва завдання реалізовані в одному графічному інтерфейсі з можливістю перемикання між функціоналом.

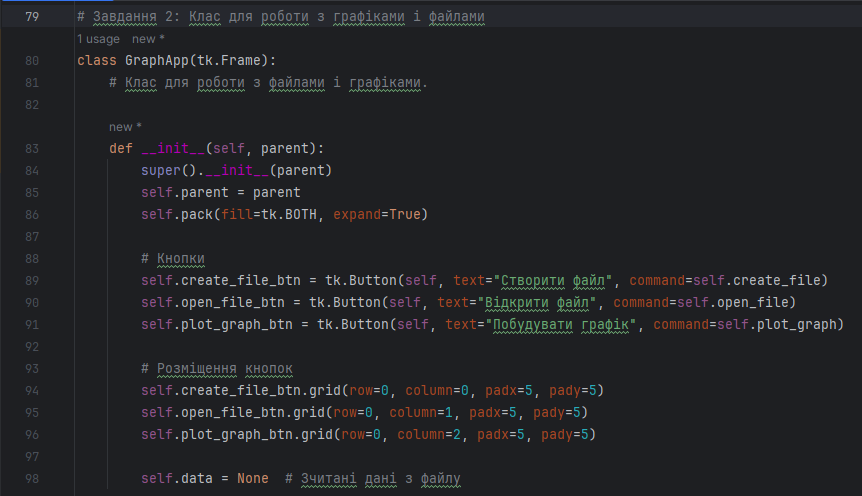




Рисунок 2 - Робота з графіками і файлами

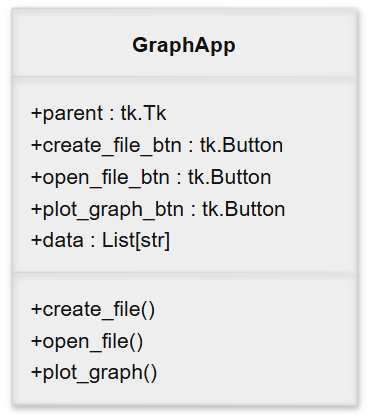


Рисунок 2.1 – Діаграма класів для завдання 2

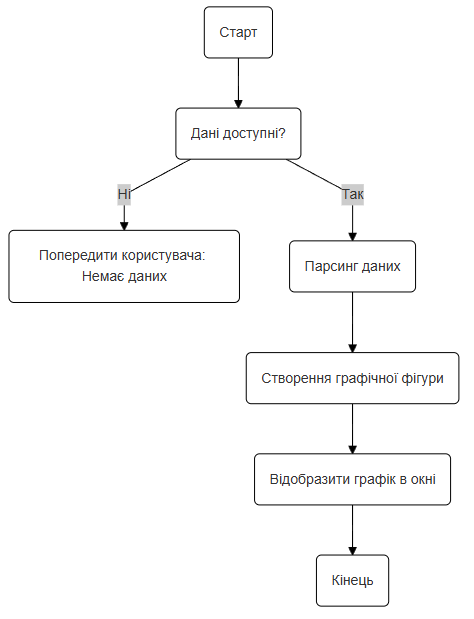


Рисунок 2.2 – Діаграма активності для завдання 2

# ВИСНОВКИ

Було вивчено основи роботи з графічним інтерфейсом у Python за допомогою бібліотеки Tkinter, а також засоби візуалізації даних за допомогою Matplotlib. Закріплено на практиці методи створення програм із дружнім інтерфейсом для розрахунків та обробки даних. Відпрацьовано роботу з текстовими файлами, обчислення математичних функцій та побудову графіків, що дозволило отримати навички інтеграції різних бібліотек у Python для вирішення інженерних задач.

ДОДАТОК А

Лістинг коду програми до задач

№1: Розрахунок площі кіл, №2: Робота з графіками та файлами

<

import tkinter as tk

from tkinter import messagebox, filedialog

from matplotlib.backends.backend\_tkagg import FigureCanvasTkAgg

from matplotlib.figure import Figure

import math

# Завдання 1: Клас для розрахунку площі кіл

class CircleCalculator(tk.Frame):

def \_\_init\_\_(self, parent):

super().\_\_init\_\_(parent)

self.parent = parent

self.pack(fill=tk.BOTH, expand=True)

# Поля для введення радіусів

self.label\_r1 = tk.Label(self, text="Радіус 1:")

self.entry\_r1 = tk.Entry(self)

self.label\_r2 = tk.Label(self, text="Радіус 2:")

self.entry\_r2 = tk.Entry(self)

self.label\_r3 = tk.Label(self, text="Радіус 3:")

self.entry\_r3 = tk.Entry(self)

# Кнопка для обчислення

self.calc\_button = tk.Button(self, text="Обчислити площі", command=self.calculate\_areas)

# Поле для виведення результату

self.result\_label = tk.Label(self, text="Результати:")

self.result\_text = tk.Text(self, height=5, state=tk.DISABLED)

# Розміщення віджетів

self.label\_r1.grid(row=0, column=0, sticky=tk.W, padx=5, pady=5)

self.entry\_r1.grid(row=0, column=1, padx=5, pady=5)

self.label\_r2.grid(row=1, column=0, sticky=tk.W, padx=5, pady=5)

self.entry\_r2.grid(row=1, column=1, padx=5, pady=5)

self.label\_r3.grid(row=2, column=0, sticky=tk.W, padx=5, pady=5)

self.entry\_r3.grid(row=2, column=1, padx=5, pady=5)

self.calc\_button.grid(row=3, column=0, columnspan=2, pady=10)

self.result\_label.grid(row=4, column=0, sticky=tk.W, padx=5)

self.result\_text.grid(row=5, column=0, columnspan=2, padx=5, pady=5)

def calculate\_areas(self):

# Обчислення площі трьох кіл

try:

# Зчитування значень радіусів

r1 = float(self.entry\_r1.get())

r2 = float(self.entry\_r2.get())

r3 = float(self.entry\_r3.get())

# Перевірка на невід'ємність

if r1 < 0 or r2 < 0 or r3 < 0:

raise ValueError("Радіус має бути невід'ємним числом!")

# Обчислення площ

areas = [self.circle\_area(r) for r in (r1, r2, r3)]

# Виведення результатів

self.result\_text.config(state=tk.NORMAL)

self.result\_text.delete(1.0, tk.END)

self.result\_text.insert(tk.END, f"Площа кола 1: {areas[0]:.2f}\n")

self.result\_text.insert(tk.END, f"Площа кола 2: {areas[1]:.2f}\n")

self.result\_text.insert(tk.END, f"Площа кола 3: {areas[2]:.2f}\n")

self.result\_text.config(state=tk.DISABLED)

except ValueError as e:

messagebox.showerror("Помилка введення", str(e))

@staticmethod

def circle\_area(radius):

# Обчислення площі кола

pi = 3.14

return pi \* radius \*\* 2

# Завдання 2: Клас для роботи з графіками і файлами

class GraphApp(tk.Frame):

# Клас для роботи з файлами і графіками.

def \_\_init\_\_(self, parent):

super().\_\_init\_\_(parent)

self.parent = parent

self.pack(fill=tk.BOTH, expand=True)

# Кнопки

self.create\_file\_btn = tk.Button(self, text="Створити файл", command=self.create\_file)

self.open\_file\_btn = tk.Button(self, text="Відкрити файл", command=self.open\_file)

self.plot\_graph\_btn = tk.Button(self, text="Побудувати графік", command=self.plot\_graph)

# Розміщення кнопок

self.create\_file\_btn.grid(row=0, column=0, padx=5, pady=5)

self.open\_file\_btn.grid(row=0, column=1, padx=5, pady=5)

self.plot\_graph\_btn.grid(row=0, column=2, padx=5, pady=5)

self.data = None # Зчитані дані з файлу

def create\_file(self):

# Створення текстового файлу з даними.

try:

# Генерація даних

t = [i \* 0.01 for i in range(101)] # 101 точка

y = [3.14 \* math.sin(2 \* math.pi \* t[i]) for i in range(len(t))]

lines = [f"{t[i]:.2f};{y[i]:.2f}\n" for i in range(len(t))]

# Збереження файлу

file = filedialog.asksaveasfile(mode='w', defaultextension=".txt",

filetypes=(("Text files", "\*.txt"), ("All files", "\*.\*")))

if file:

file.writelines(lines)

file.close()

messagebox.showinfo("Успіх", "Файл успішно створено!")

except Exception as e:

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

def open\_file(self):

# Зчитування даних із файлу

try:

file = filedialog.askopenfile(mode='r', filetypes=(("Text files", "\*.txt"), ("All files", "\*.\*")))

if file:

self.data = [line.strip().split(';') for line in file.readlines()]

file.close()

messagebox.showinfo("Успіх", "Дані успішно зчитано!")

except Exception as e:

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

def plot\_graph(self):

# Побудова графіка

if not self.data:

messagebox.showwarning("Попередження", "Дані відсутні!")

return

try:

x = [float(row[0]) for row in self.data]

y = [float(row[1]) for row in self.data]

# Побудова графіка

fig = Figure(figsize=(5, 4))

ax = fig.add\_subplot(111)

ax.plot(x, y, label="Графік функції", color="blue")

ax.set\_title("Графік функції")

ax.set\_xlabel("Час t")

ax.set\_ylabel("Значення y")

ax.grid(True)

ax.legend()

# Відображення на Canvas

canvas = FigureCanvasTkAgg(fig, master=self)

canvas.get\_tk\_widget().grid(row=1, column=0, columnspan=3, padx=5, pady=5)

canvas.draw()

except Exception as e:

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

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

root = tk.Tk()

root.title("lab5\_3-91AVS-v01-Opanasiuk\_Oleksandr")

# Перемикач між завданнями

tab\_control = tk.Frame(root)

tab\_control.pack(side=tk.TOP, fill=tk.BOTH, expand=True)

calc\_app = CircleCalculator(tab\_control)

graph\_app = GraphApp(tab\_control)

root.mainloop()

>

ДОДАТОК Б

Скрін-шоти вікна виконання програми

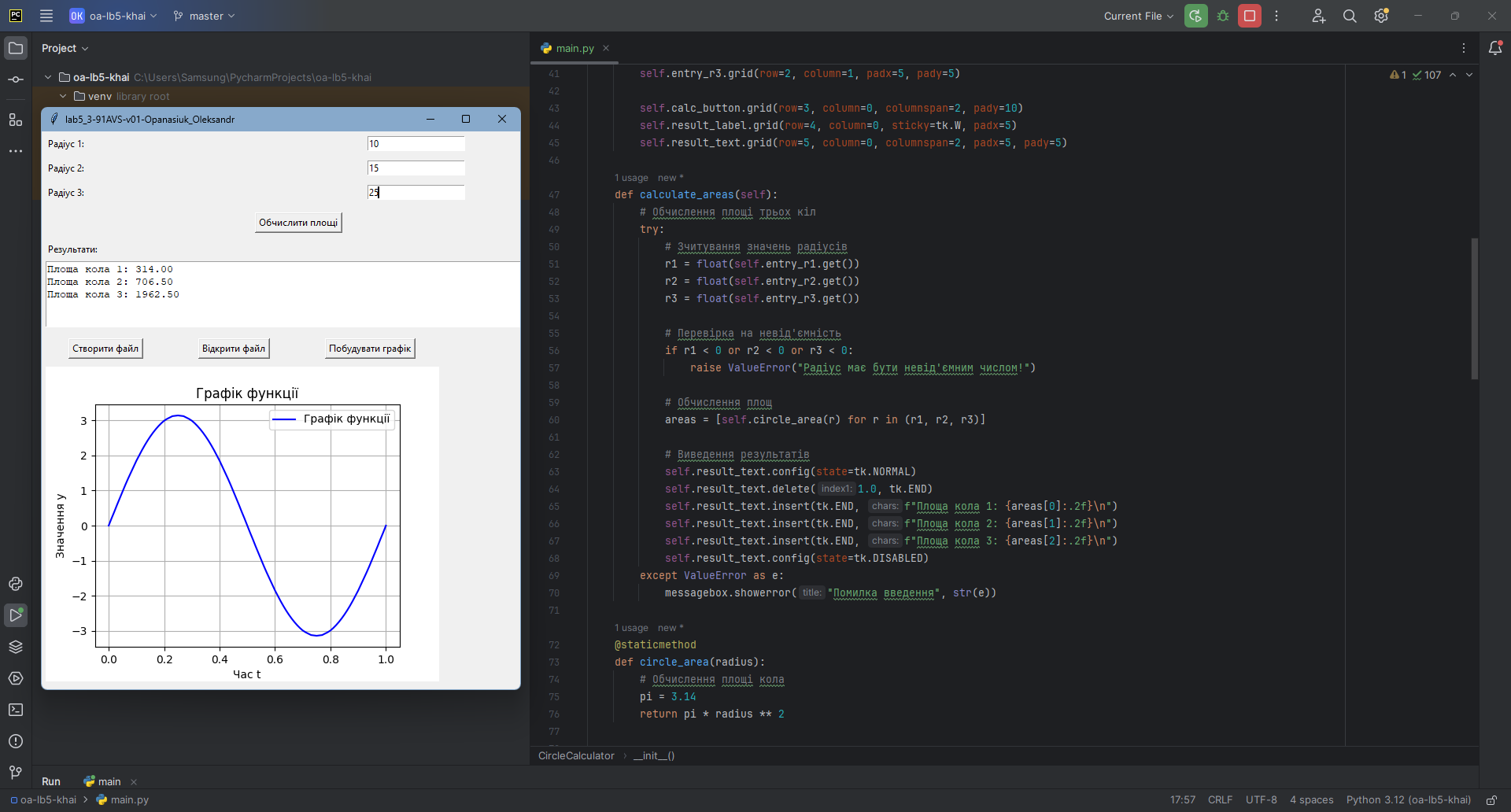


Рисунок Б.1 – Екран виконання програми для вирішення завдання   
Завдання 1: Розрахунок площі кіл

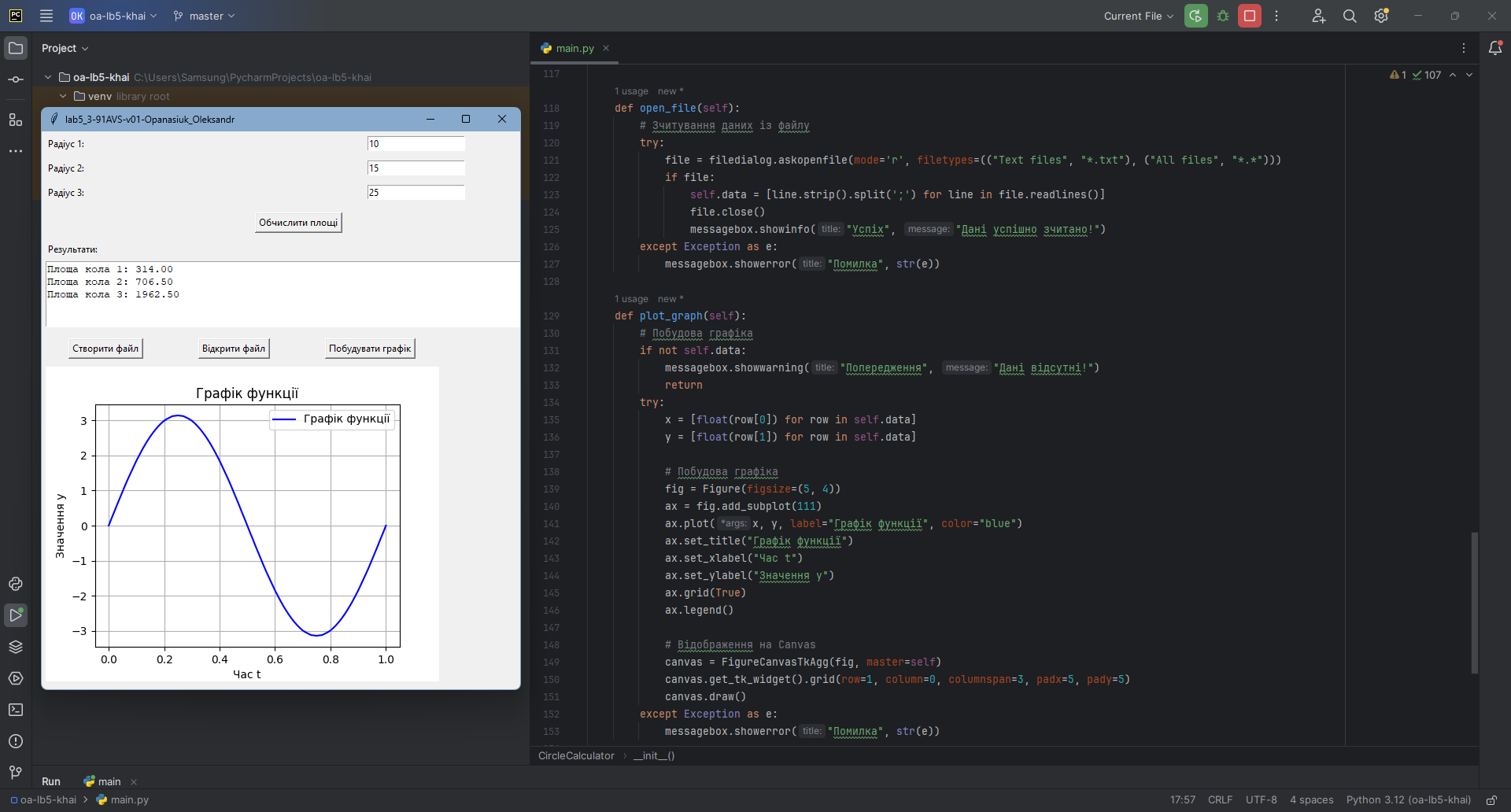


Рисунок Б.2 – Екран виконання програми для вирішення завдання   
Завдання 2: Робота з графіками та файлами