МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

ЗВІТ ПРО ВИКОНАННЯ

ІНДИВІДУАЛЬНОГО ДОМАШНЬОГО ЗАВДАННЯ

із дисципліни: «Програмування мовою Python»

на тему «Використання бібліотеки SciPy для розв’язання задач чисельними методами»

Виконав студент групи ІН-11 Штогрін В.О.

Варіант № 6

Перевірила Парфененко Ю.В

Суми 2022

**ЗМІСТ**

[**ВСТУП** 3](#_Toc122626017)

[**1.** **РОЗВ’ЯЗАННЯ ЗАДАЧ ЧИСЕЛЬНИМИ МЕТОДАМИ З ВИКОРИСТАННЯМ БІБЛІОТЕК SCIPY ТА NUMPY** 4](#_Toc122626018)

[1.1 Модулі SciPy та NumPy 4](#_Toc122626019)

[1.2 Порівняння особливостей 9](#_Toc122626020)

[**2 РОЗВ’ЯЗАННЯ ЗАДАЧІ ЧИСЕЛЬНИМИ МЕТОДАМИ З ВИКОРИСТАННЯМ БІБЛІОТЕКИ SCIPY** 11](#_Toc122626021)

[2.1 Постановка задачі 11](#_Toc122626022)

[2.2 Алгоритм розв’язання задачі 12](#_Toc122626023)

[2.3 Програмна реалізація 12](#_Toc122626024)

[2.4 Текст програми 14](#_Toc122626025)

[2.5 Алгоритм роботи програми 18](#_Toc122626026)

[2.6 Контрольний приклад 19](#_Toc122626027)

[**ВИСНОВКИ** 27](#_Toc122626028)

[**СПИСОК ЛІТЕРАТУРИ** 28](#_Toc122626029)

# **[ВСТУП](#_ВСТУП_1)**

Python— це потужна мова програмування, якою легко оволодіти. Вона має ефективні структури даних високого рівня та простий, але ефективний підхід до об'єктно-орієнтованого програмування. Елегантний синтаксис Python, динамічна обробка типів, а також те, що це інтерпретована мова, роблять його ідеальним для написання скриптів та швидкої розробки прикладних програм у багатьох галузях на більшості платформ[[1]](https://uk.wikibooks.org/wiki/%D0%9F%D1%96%D0%B4%D1%80%D1%83%D1%87%D0%BD%D0%B8%D0%BA_%D0%BC%D0%BE%D0%B2%D0%B8_Python/%D0%92%D1%81%D1%82%D1%83%D0%BF).

Ще однією перевагою Python є його стандартна бібліотека, яка встановлюється разом з Python і містить готові інструменти для роботи з операційною системою, веб-сторінками, базами даних, різними форматами даних, для побудови графічного інтерфейсу програм тощо[[2]](https://pythonguide.rozh2sch.org.ua/).

Чисельні методи – методи наближеного або точного розв’язування задач прикладної математики, які ґрунтуються на побудові послідовності дій над скінченною множиною чисел. Згідно основних вимог чисельні методи мають бути стійкими та збіжними. Основне питання теорії чисельних методів: отримання чисельних методів, які задовольняють вимоги високої точності, стійкості та економічності. Однак, для підвищення якості оцінок необхідна велика кількість даних, обсяг обчислень може виявитися дуже великим. Чисельні методи націлені на скорочення кількості обчислень при збереженні якості результатів. До найбільш ефективних чисельних методів в цій галузі відносяться методи, які застосовують швидке перетворення Фур’є. Для розв’язування задач апроксимації та обчислення функцій різних класів застосовують чисельні методи інтерполювання, найменших квадратів, ортогоналізації, врівноваження значень, умовної мінімізації тощо[[3]](http://ism.lpnu.ua/uk/content/chyselni-metody-v-kompyuternyh-naukah-tom-2).

Тому у роботі буде розглянуто порівняння бібліотек Python, таких як: SciPy та NumPy, для розв’язання задач чисельними методами.

# **[РОЗВ’ЯЗАННЯ ЗАДАЧ ЧИСЕЛЬНИМИ МЕТОДАМИ З ВИКОРИСТАННЯМ БІБЛІОТЕК SCIPY ТА NUMPY](#_РОЗВ’ЯЗАННЯ_ЗАДАЧ_ЧИСЕЛЬНИМИ)**

## [Модулі SciPy та NumPy](#_Модулі_Bokeh_та)

Суспільство розпочало важливий період свого розвитку – еру інформатизації. Для вирішення теоретичних і практичних задач, що виникають під час діяльності людини у різних галузях науки, техніки та виробництва з метою звільнення людини від надмірного інтелектуального навантаження, значний ефект отримуємо за рахунок використання обчислювальної техніки при умові достатнього програмного забезпечення та ефективного його використання. Програмування обчислювальної техніки займає важливу роль в сьогоденні[[4]](http://ism.lpnu.ua/uk/content/chyselni-metody-v-kompyuternyh-naukah-tom-2). Тому розглянемо деякі бібліотеки Python, які можуть нам в цьому допомогти.

У цій роботі буде описано різницю між NumPy та SciPy в Python. NumPy та SciPy є дуже важливими бібліотеками в Python. Вони мають широкий спектр функцій і контрастних операцій.

NumPy це скорочення від Numerical Python, тоді SciPy це абревіатура від Scientific Python. Обидва є модулями Python і використовуються для виконання операцій з даними[[5]](https://www.educative.io/answers/what-is-the-difference-between-numpy-and-scipy-in-python).

SciPy – це бібліотека Python з відкритим вихідним кодом, призначена для вирішення наукових та математичних проблем. Вона побудована на базі NumPy та дозволяє керувати даними, а також візуалізувати їх за допомогою різних високорівневих команд[[6]](https://pythonru.com/biblioteki/scipy-python).

SciPy використовують фахівці з Data Science , Big Data , аналітики даних , а також математики та вчені:

* для складних математичних розрахунків, які важко здійснити вручну або за допомогою калькулятора;
* проведення наукових досліджень, де потрібне використання просунутої математики;
* глибокого аналізу даних, інтерполяції та інших методів роботи з інформацією;
* машинного навчання та створення моделей штучного інтелекту, прогнозування та побудови моделей;
* формування двовимірних та тривимірних графіків, які можна потім візуалізувати (вже за допомогою інших бібліотек).

Особливості SciPy

* Безкоштовне поширення

Вихідний код бібліотеки відкритий, вона розповсюджується безкоштовно за ліцензією на BSD. Це означає, що SciPy можна завантажувати, користуватися в особистих чи професійних цілях, переглядати вихідний код функції. Модулі, написані за допомогою SciPy, можна вбудовувати у пропрієтарне, закрите та платне програмне забезпечення.

* Низький поріг входу

Працювати зі SciPy зможе новачок у Python. Для використання бібліотеки не обов'язково бути професійним розробником, але важливо мати хороший математичний апарат і розумітися на наукових обчисленнях. На офіційному сайті представлена ​​документація SciPy англійською мовою.

* Швидке виконання коду

Python критикують за повільну роботу щодо трудомістких обчислень. Тому розробники SciPy написали бібліотеку не лише на ньому. Складні обчислення, що потребують високої швидкості роботи, реалізовані на Fortran, C та C++. Тому код на SciPy швидко виконується і забирає багато ресурсів[[7]](https://blog.skillfactory.ru/glossary/scipy/).

NumPy — це бібліотека мови Python, яка додає підтримку великих багатовимірних масивів і матриць, разом із великою бібліотекою високорівневих (і дуже швидких) математичних функцій для операцій із цими масивами[[8]](https://pythonworld.ru/numpy/1.html).

Пакет NumPy є незамінним помічником Python. Він тягне на собі аналіз даних, машинне навчання та наукові обчислення , а також суттєво полегшує обробку векторів та матриць. Деякі провідні пакети Python використовують NumPy як основний елемент своєї інфраструктури. Крім можливості розібрати по кісточках числові дані, вміння працювати з NumPy дає значну перевагу при налагодженні складніших сценаріїв бібліотек[[9]](https://python-scripts.com/numpy).

Ця бібліотека має кілька важливих особливостей, які зробили її популярним інструментом. По-перше, вихідний код у вільному доступі зберігається на GitHub, тому NumPy називають open-source модулем для Python. По-друге, бібліотека написана мовами C та Fortran. Це компільовані мови (мови програмування, текст яких перетворюється на машинний код — набір інструкцій для конкретного типу процесора. Перетворення відбувається за допомогою спеціальної програми-компілятора, завдяки якому обчислення компілюваними мовами відбуваються швидше), на яких обчислення виробляються набагато швидше та ефективніше, ніж мовами, що інтерпретуються (мови програмування, які не заточені під конкретний тип процесора і можуть бути запущені на різних типах пристроїв). До цих мов належить і сам Python[[10]](https://blog.skillfactory.ru/glossary/numpy/).

Приклад роботи бібліотеки SciPy[[11]](https://www.edureka.co/blog/scipy-tutorial/):

Приклад 1

import numpy as np

import matplotlib.pyplot as plt

from scipy import interpolate

x = np.arange(5, 20)

y = np.exp(x/3.0)

f = interpolate.interp1d(x, y)

x1 = np.arange(6, 12)

y1 = f(x1) # use interpolation function returned by `interp1d`

plt.plot(x, y, 'o', x1, y1, '--')

plt.show()

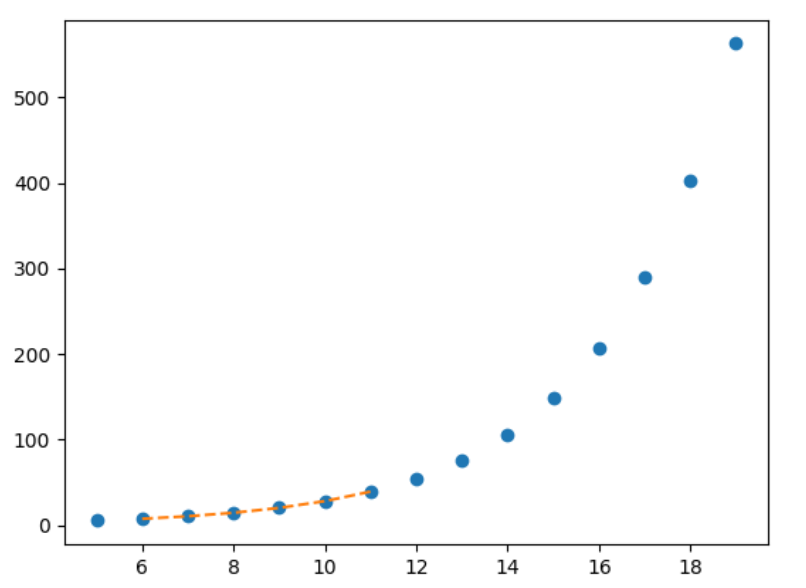


Рисунок 1.1 - Приклад роботи SciPy

Приклад 2

import numpy as np

from scipy.signal import chirp, spectrogram

import matplotlib.pyplot as plt

t = np.linspace(6, 10, 500)

w = chirp(t, f0=4, f1=2, t1=5, method='linear')

plt.plot(t, w)

plt.title("Linear Chirp")

plt.xlabel('time in sec)')

plt.show()

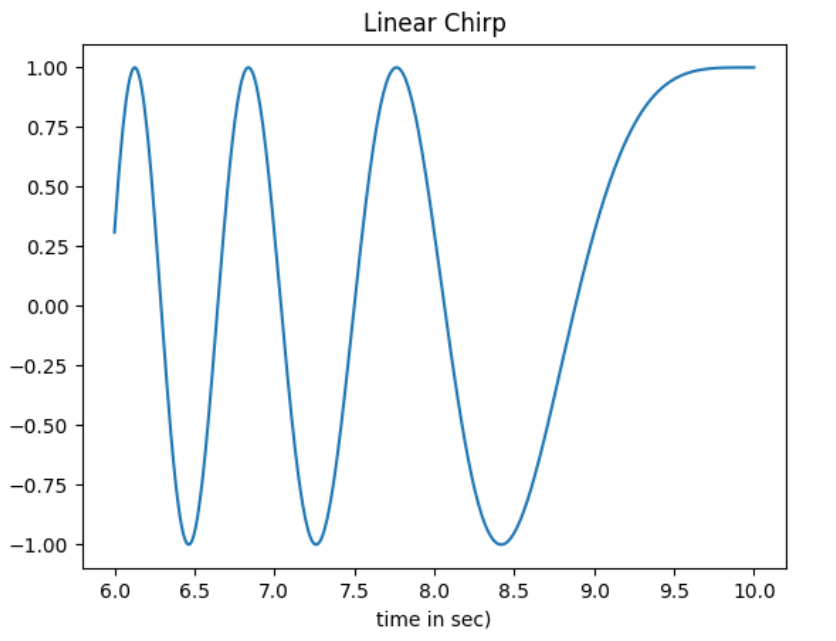


Рисунок 1.2 - Приклад роботи SciPy

Приклад роботи бібліотеки NumPy[[12]](https://www.edureka.co/blog/python-numpy-tutorial/):

Приклад 1

import numpy as np

import matplotlib.pyplot as plt

x= np.arange(0,3\*np.pi,0.1)

y=np.tan(x)

plt.plot(x,y)

plt.show()

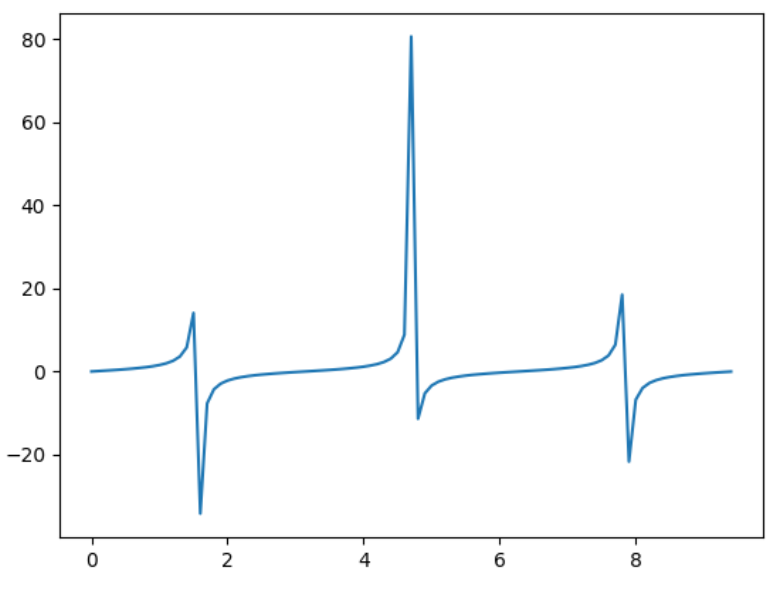


Рисунок 1.3 - Приклад роботи NumPy

Приклад 2

import numpy as np

import matplotlib.pyplot as plt

x= np.arange(0,3\*np.pi,0.1)

y=np.sin(x)

plt.plot(x,y)

plt.show()

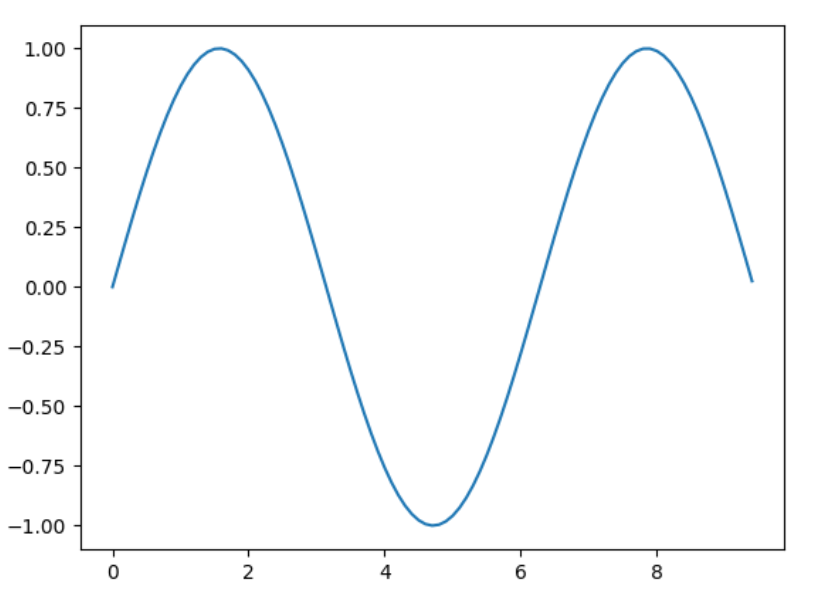


Рисунок 1.4 - Приклад роботи NumPy

## 1.2 [Порівняння особливостей](#_1.2_Порівняння_особливостей)

Основні відмінності між бібліотеками:

* у SciPy набагато більше функцій та методів, ніж у NumPy;
* NumPy орієнтована на базові обчислення та просту роботу з матрицями, SciPy призначена для глибокого наукового аналізу;
* NumPy не має додаткових залежностей, разом із бібліотекою не потрібно нічого встановлювати. SciPy вимагає встановлення NumPy для коректної роботи.

SciPy розширює можливості NumPy, а деякі часті дії в ній реалізовані як окремі функції, тому бібліотека значно спрощує роботу зі складними завданнями з використанням просунутої математики. При цьому для ряду простих завдань вона надмірна. Буде достатньо NumPy чи інших бібліотек[[13]](https://blog.skillfactory.ru/glossary/scipy/).

Таблиця 1.1 - Порівняльна таблиця модулів SciPy та NumPy [[14]](https://www.educative.io/answers/what-is-the-difference-between-numpy-and-scipy-in-python).

|  |  |  |
| --- | --- | --- |
| **Критерій** | **SciPy** | **NumPy** |
| **Тип операцій** | Виконує базові операції, такі як сортування, індексування тощо. Здебільшого використовується під час роботи з наукою про дані та статистичними концепціями. | Використовується для складних операцій, таких як алгебраїчні функції, різні чисельні алгоритми тощо. |
| **Функції** | Містить різноманітні функції, але вони не визначені детально | Містить докладні версії функцій, як-от лінійна алгебра, які повністю представлені. |
| **Масиви** | Масиви NumPy — це багатовимірні масиви об’єктів одного типу, тобто однорідні. | SciPy не має таких концепцій масиву, оскільки він більш функціональний. Він не має обмежень однорідності. |
| **Базова мова створення та швидкість** | NumPy написаний мовою C , тому має вищу швидкість обчислень. | SciPy написаний на Python , тому має нижчу швидкість виконання, але широку функціональність. |

# **2** [**РОЗВ’ЯЗАННЯ ЗАДАЧІ ЧИСЕЛЬНИМИ МЕТОДАМИ З ВИКОРИСТАННЯМ БІБЛІОТЕКИ SCIPY**](#_2__)

## 2.1 [Постановка задачі](#_2.1_Постановка_задачі)

Необхідно зробити програму, яка аналізує зібрану інформацію для подальшого його застосування.

Дана програма виконана на тему «Вартість перельотів між країнами ЄС» дані збиралися з сайту онлайн купівлі квитків(<https://www.omio.com.ua/>) та самостійно аналізувались, після чого заносилися до відповідних файлів (рисунок 2.1). Програма створювалася в [JupiterNotebook](https://jupyter.org/try-jupyter/retro/notebooks/?path=notebooks/Intro.ipynb).

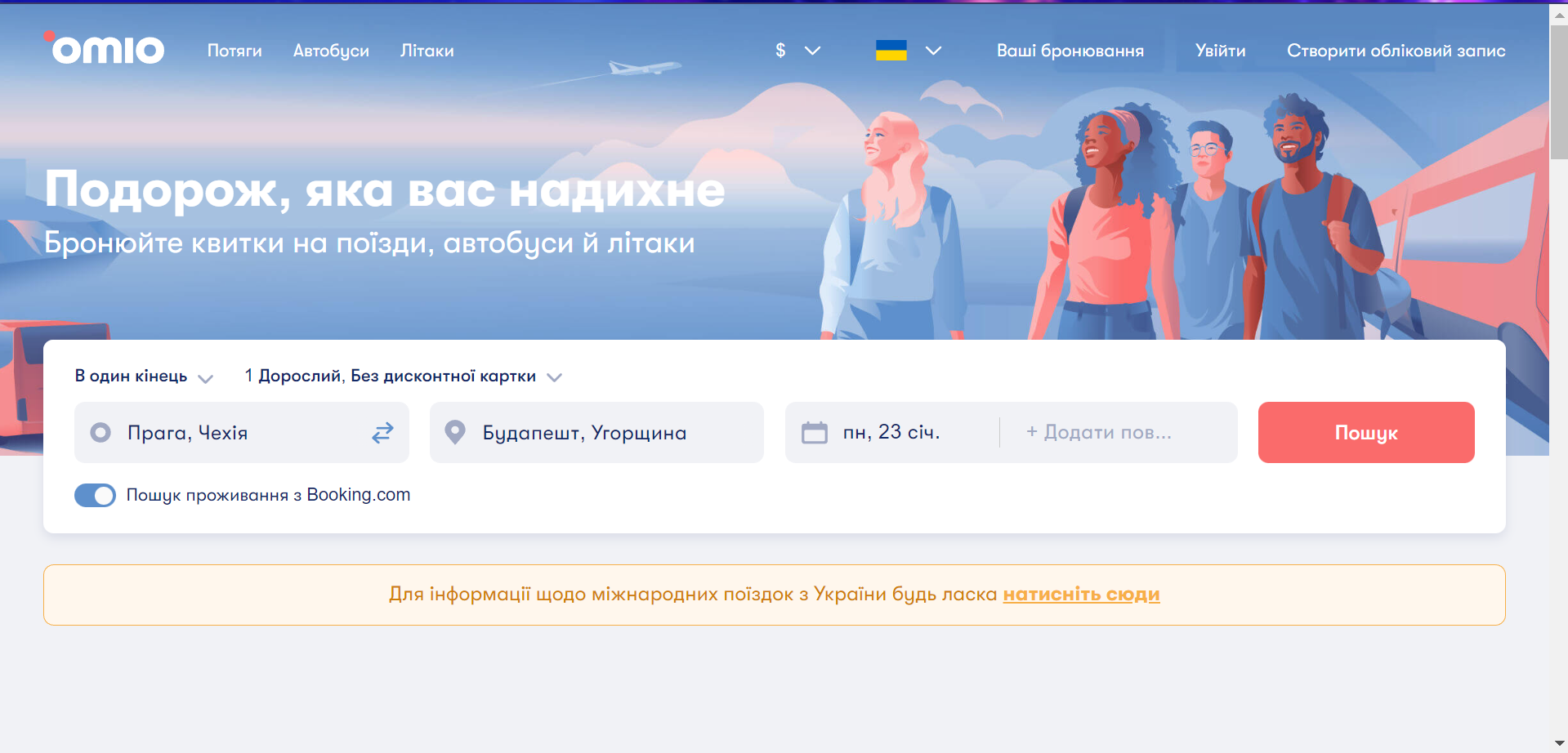


Рисунок 2.1 - Приклад вигляду сайту, з якого буде збиратися інформація

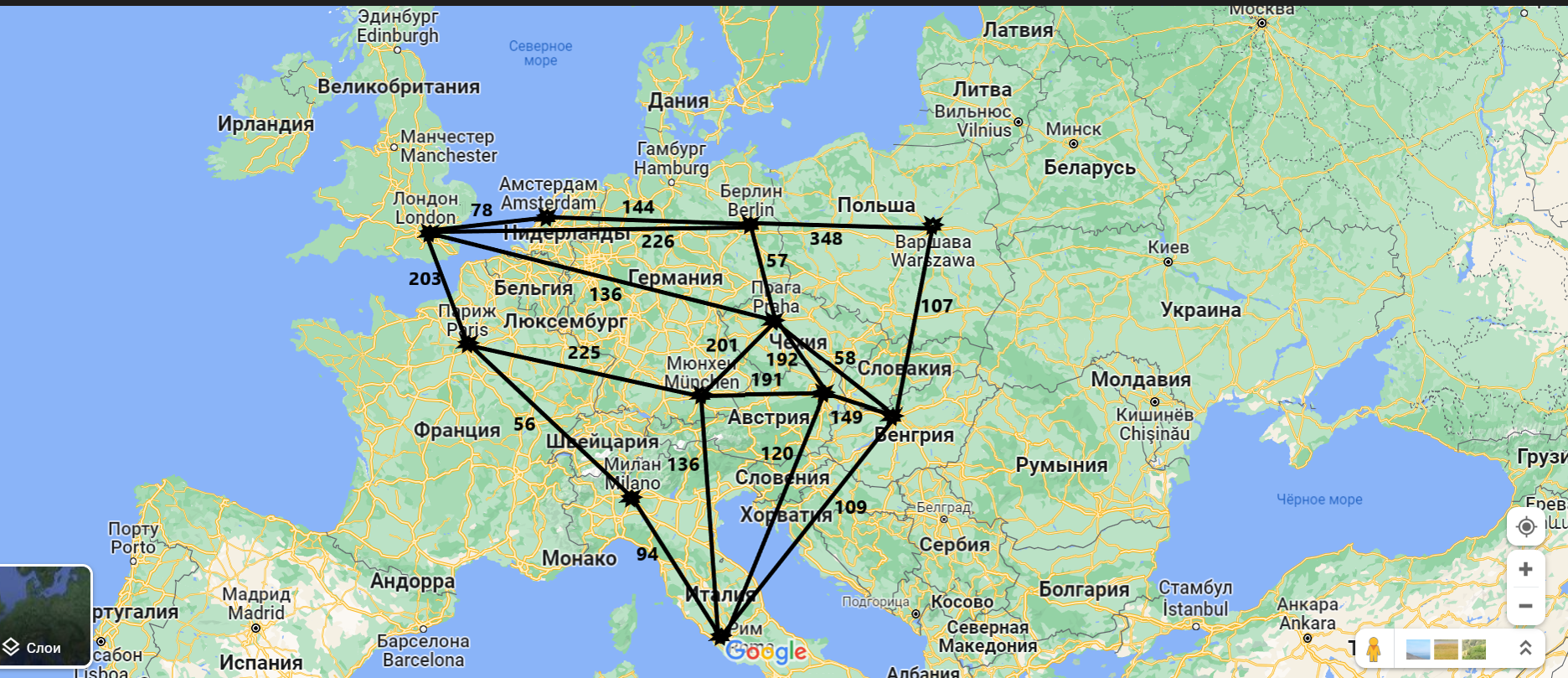


Рисунок 2.2 - Приклад вигляду побудови графа польотів

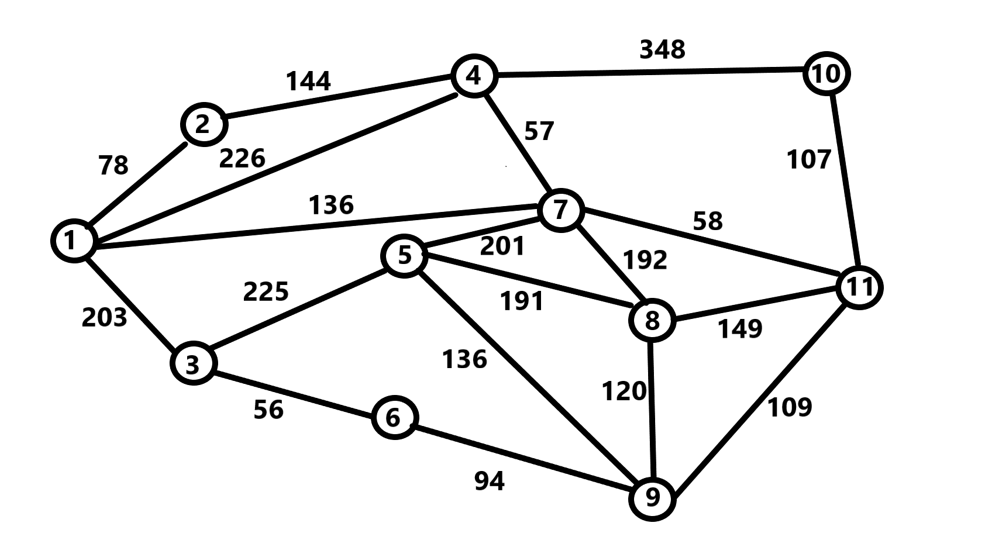


Рисунок 2.3 – Граф для створення файлів

## 2.2 [Алгоритм розв’язання задачі](#_2.2_Алгоритм_розв’язання)

1. Аналіз даних на сайті
2. Групування даних з сайту та занесення їх до таблиці файлів.
3. Зберегти ці дані та відкрити у програмі
4. Побудувати математичні моделі за допомогою бібліотеки SciPy
5. Застосувати чисельні методи для аналізу та застосуванню певних методів бібліотеки
6. Зберегти програму та скопіювати у звіт

## 2.3 [Програмна реалізація](#_2.3_Програмна_реалізація)

Таблиця 2.3.1 - Таблиця компонентів програми

|  |  |  |  |
| --- | --- | --- | --- |
| **Назва** | **Тип** | **Опис** | **Призначення** |
| import csv | Модуль | Модуль що створює настроювані об’єкти, подібні до таблиці, з елементів у файлі значень із розділеними символами (CSV). | Імпортування даних з csv файлу |
| import numpy as np | Метод | Наукова обчислювальна бібліотека, створена на основі мови програмування Python. | Створення типів даних за допомогою цього методу |
| import matplotlib.pyplot as plt | Метод | Набір функцій командного стилю, завдяки яким matplotlib працює як MATLAB | Створює та змінює фігури |
| from scipy import interpolate | Метод | Підпакет бібліотеки SciPy для об’єктів, що використовуються в інтерполяції. | Використання інтерполяції для різних даних |

Таблиця 2.3.3 – Таблиця символьних імен (змінних)

|  |  |  |
| --- | --- | --- |
| **Назва** | **Тип даних** | **Призначення** |
| final | int | Лічильник для створеного меню |
| index | int | Змінна для переходу по меню |
| schet | int | Змінна для перевірки |
| x, xnew | array | Масиви років |
| y\_mass, ynew | list | Ціна в $ |
| i | int | Лічильник циклу for |

## 2.4 [Текст програми](#_2.4_Текст_програми)

import csv

import numpy as np

import matplotlib.pyplot as plt

from scipy import interpolate

from scipy.sparse.csgraph import connected\_components

from scipy.sparse import csr\_matrix

from scipy.sparse.csgraph import shortest\_path

with open("data.csv") as data\_name: #відкриваємо файл "data.csv"

array = np.loadtxt(data\_name, delimiter=";") #завантажуємо дані з роздільником ';'

City = open('City.txt') #відкриваємо файл 'City.txt'

city = City.readlines() #завантажуємо дані порядково

city1 = [] #створюємо список

for word in city:

result = word.replace('\n', '') #замінюємо в списку city '\n' на ''

city1.append(result) # і заносимо ці дані до нового списку

graph = csr\_matrix(array) # Створюємо стиснуту матрицю нашого графу

dist\_matrix, predecessors = shortest\_path(csgraph=graph, directed=False, indices=0, return\_predecessors=True) #виконуємо пошук графа найкоротшого шляху

def Future(k): # створюємо функцію (де інтерполюємо дані) для полегшення виводу наступних даних

y\_mass = []

y\_mass.append(dist\_matrix[k])

x = np.arange(2022, 2027) #створюємо масив років

for i in range(1, len(x)):

y\_mass.append(y\_mass[i-1] \* 1.05)

f = interpolate.interp1d(x, y\_mass) #створюємо функцію інтерполяції списків

xnew = np.arange(2022, 2027) #створюємо масив років

ynew = f(xnew) # використовуємо функцію інтерполяції, яка повертає `interp1d`

plt.plot(x, y\_mass, 'o', xnew, ynew, 'r--') #створюємо графік підвищення цін на перельоти відносно років

plt.xlabel('Роки') # підписуємо вісь х

plt.ylabel('Ціна в $') # підписуємо вісь у

plt.grid(True) # виводимо сітку графіку

plt.show() #виводимо графік

#створюємо меню

print("\n\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\* ПРИВІТ! ВІТАЄМО ВАС НА НАШОМУ САЙТІ \*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*\n")

final = 1

while(final == 1):

print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n")

print("Якщо бажаєте вибрати місто в яке полетіти, то натисніть 1\n")

print("Якщо бажаєте вивести всі ціни на перельоти, то натисніть 2\n")

print("Якщо бажаєте вивести подорожчання цін перельоту до місту Amsterdam на наступні 5 років, то натисніть 3\n")

print("Якщо бажаєте вивести подорожчання цін перельоту до місту Paris на наступні 5 років, то натисніть 4\n")

print("Якщо бажаєте вивести подорожчання цін перельоту до місту Berlin на наступні 5 років, то натисніть 5\n")

print("Якщо бажаєте вивести подорожчання цін перельоту до місту Munchen на наступні 5 років, то натисніть 6\n")

print("Якщо бажаєте вивести подорожчання цін перельоту до місту Milano на наступні 5 років, то натисніть 7\n")

print("Якщо бажаєте вивести подорожчання цін перельоту до місту Praha на наступні 5 років, то натисніть 8\n")

print("Якщо бажаєте вивести подорожчання цін перельоту до місту Wien на наступні 5 років, то натисніть 9\n")

print("Якщо бажаєте вивести подорожчання цін перельоту до місту Rome на наступні 5 років, то натисніть 10\n")

print("Якщо бажаєте вивести подорожчання цін перельоту до місту Warszawa на наступні 5 років, то натисніть 11\n")

print("Якщо бажаєте вивести подорожчання цін перельоту до місту Budapest на наступні 5 років, то натисніть 12\n")

print("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n")

index = int(input("Ваш вибір: "))

print("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n")

if(index == 1):

print("Ви можете полетіти в такі міста, як: ")

for i in range(1, len(dist\_matrix)):

print(city[i])

name = str(input("Введіть назву міста, в яке ви вирішили полетіти... "))

schet = 0

for i in range(1, len(dist\_matrix)):

if(city1[i] == name):

print("З міста ", city1[0], "до міста ", city1[i], " переліт коштує ", dist\_matrix[i], "$")

schet = 1

if(schet == 0):

print("Такого міста не існує")

print("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n")

final = int(input("Якщо бажаєте продовжити, то натисніть '1', якщо ні, то натисніть '0': "))

if(final == 0):

print("\n\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\* ДО ПОБАЧЕННЯ! \*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*\n")

if(index == 2):

for i in range(1, len(dist\_matrix)):

print("З міста ", city1[0], "до міста ", city1[i], " переліт коштує ", dist\_matrix[i], "$")

print("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n")

final = int(input("Якщо бажаєте продовжити, то натисніть '1', якщо ні, то натисніть '0': "))

if(final == 0):

print("\n\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\* ДО ПОБАЧЕННЯ! \*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*\n")

if(index > 2 and index < 13):

k = index - 2

print("Місто ", city1[k])

Future(k)

print("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n")

final = int(input("Якщо бажаєте продовжити, то натисніть '1', якщо ні, то натисніть '0': "))

if(final == 0):

print("\n\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\* ДО ПОБАЧЕННЯ! \*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*\n")

## 

## 2.5 [Алгоритм роботи програми](#_2.5_Алгоритм_роботи)

* Етап 1. Початок роботи програми
* Етап 2. Підключення відповідних бібліотек та модулів для роботи програми
* Етап 3. Спочатку відкриваємо всі файли з даними та заносимо їх до програми
* Етап 4. Використовуємо методи функцій бібліотеки SciPy
* Етап 5. Створюємо меню в програмі, для спілкування з користувачем
* Етап 6. Запускаємо програму та користуємося меню

**2.6** [**Контрольний приклад**](#_2.6_Контрольний_приклад)

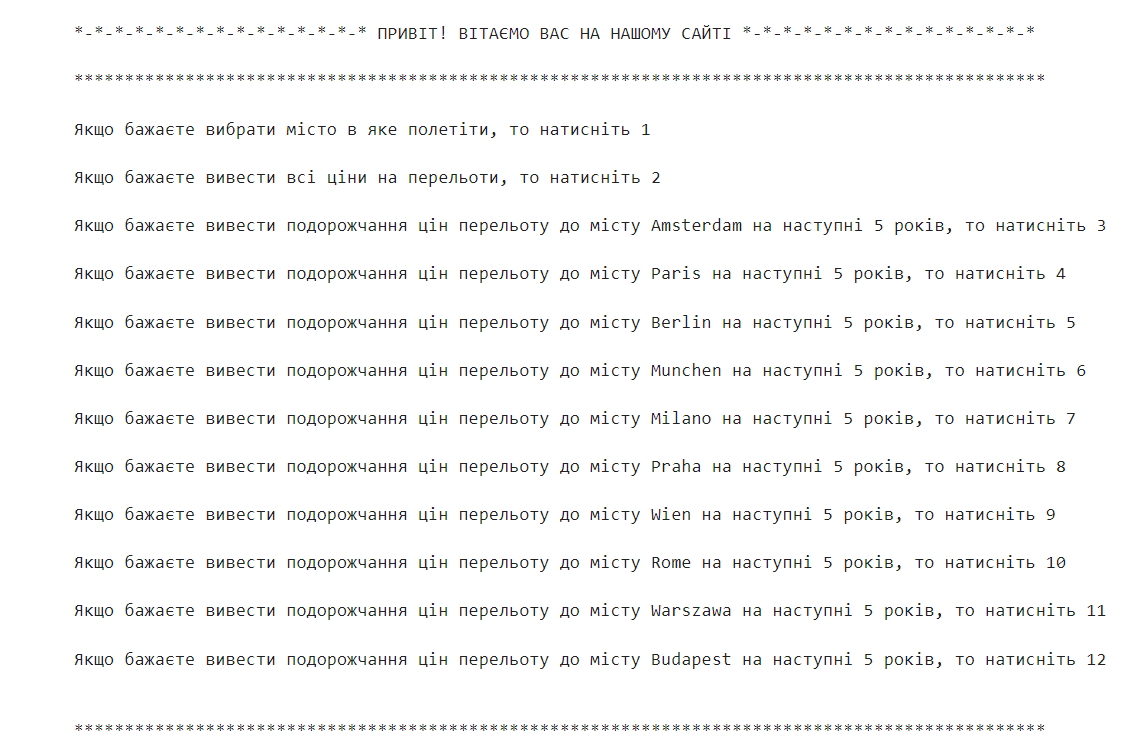


Рисунок 3.1 – Меню програми

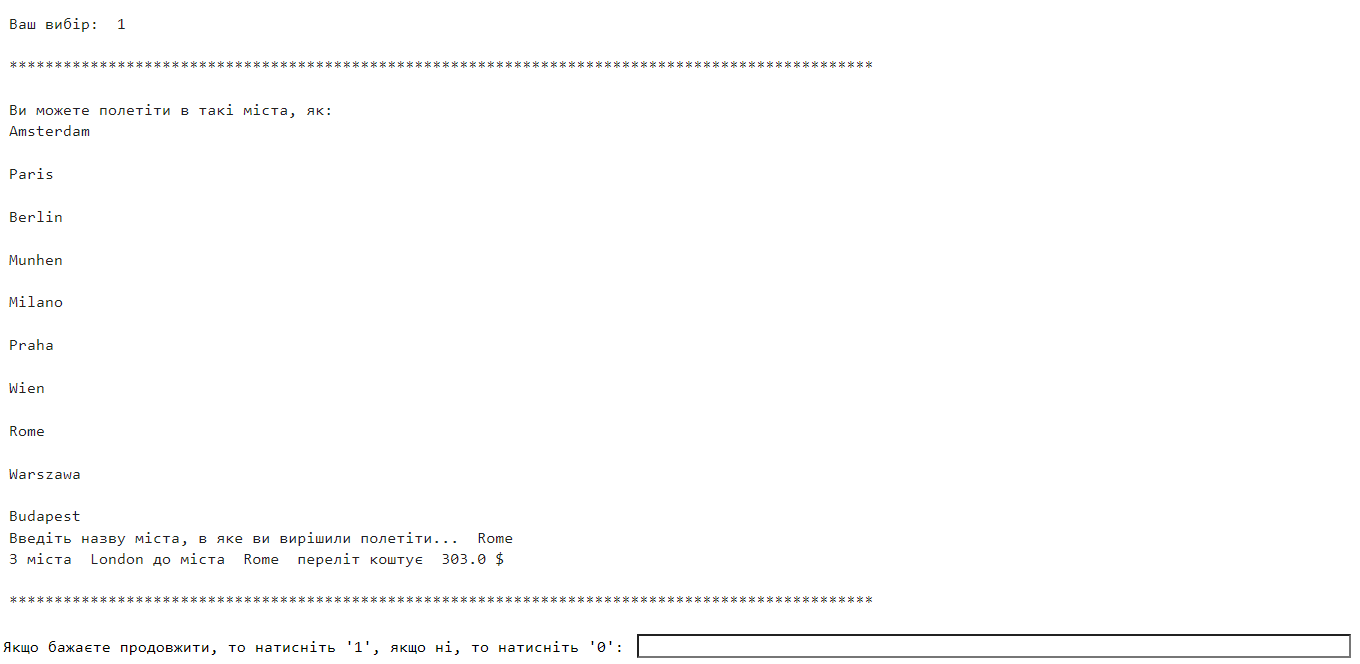


Рисунок 3.2 – Пункт меню №1

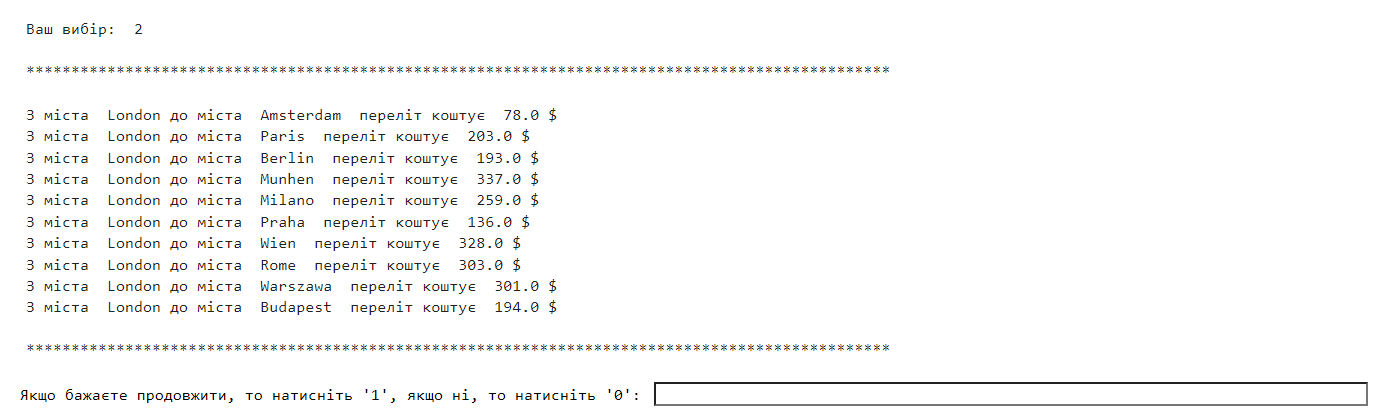


Рисунок 3.3 – Пункт меню №2

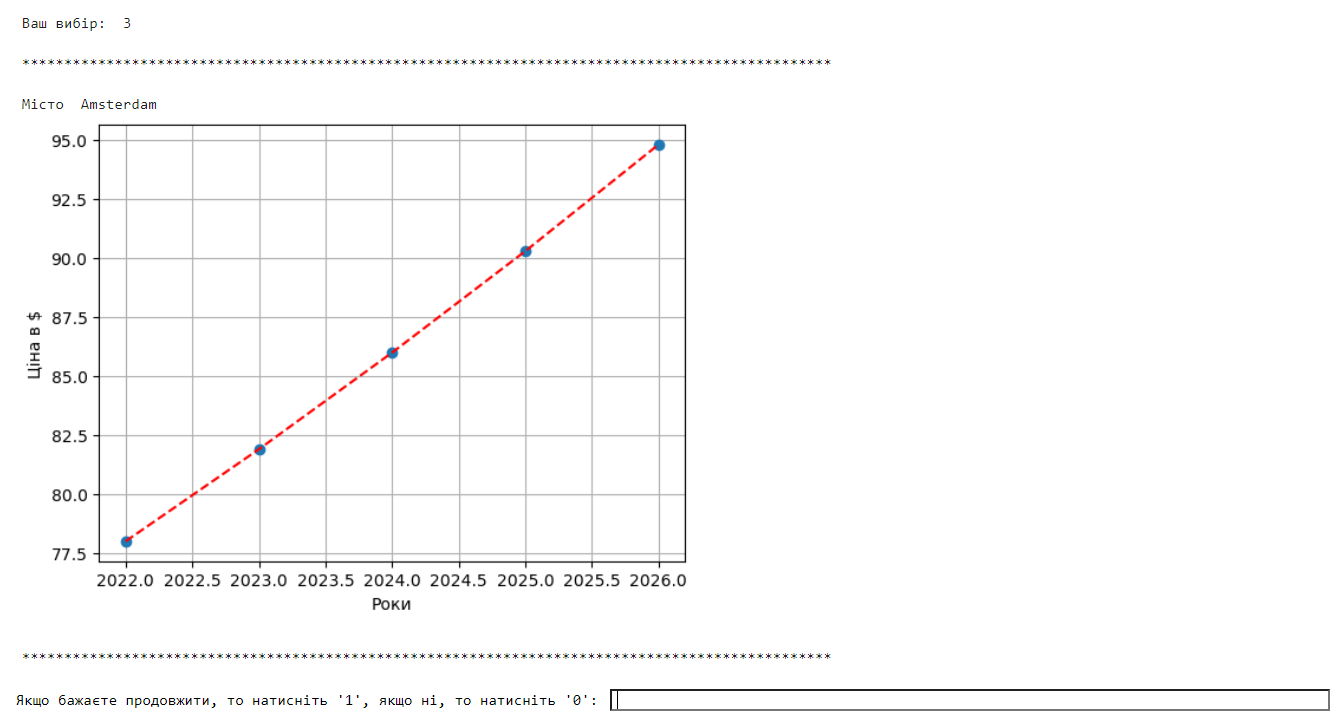


Рисунок 3.4 – Пункт меню №3

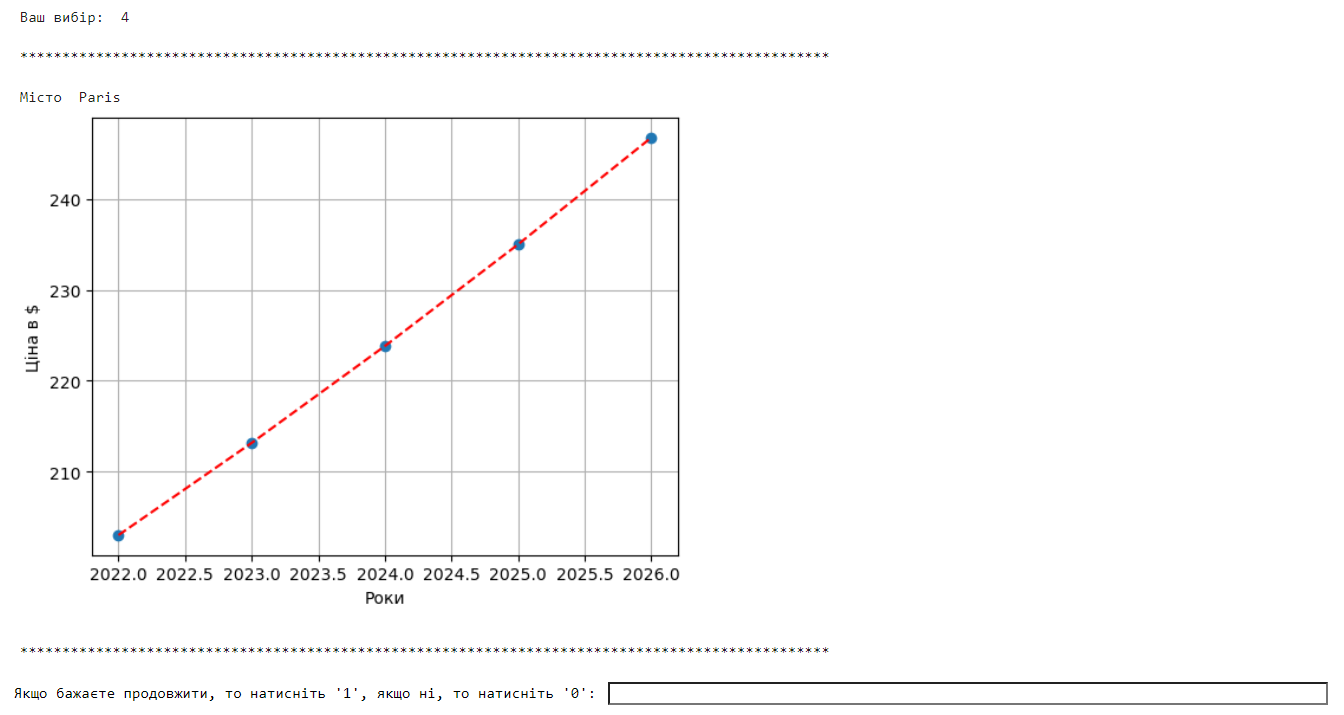


Рисунок 3.5 – Пункт меню №4

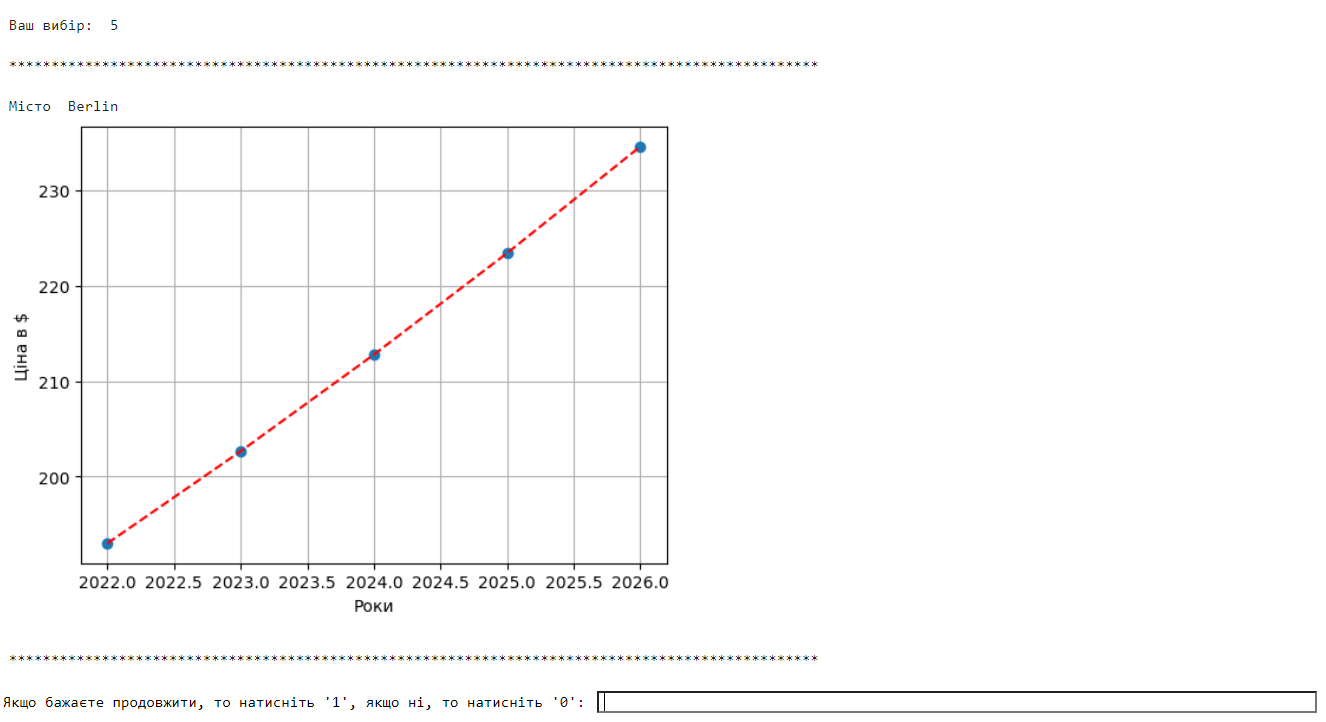


Рисунок 3.6 – Пункт меню №5

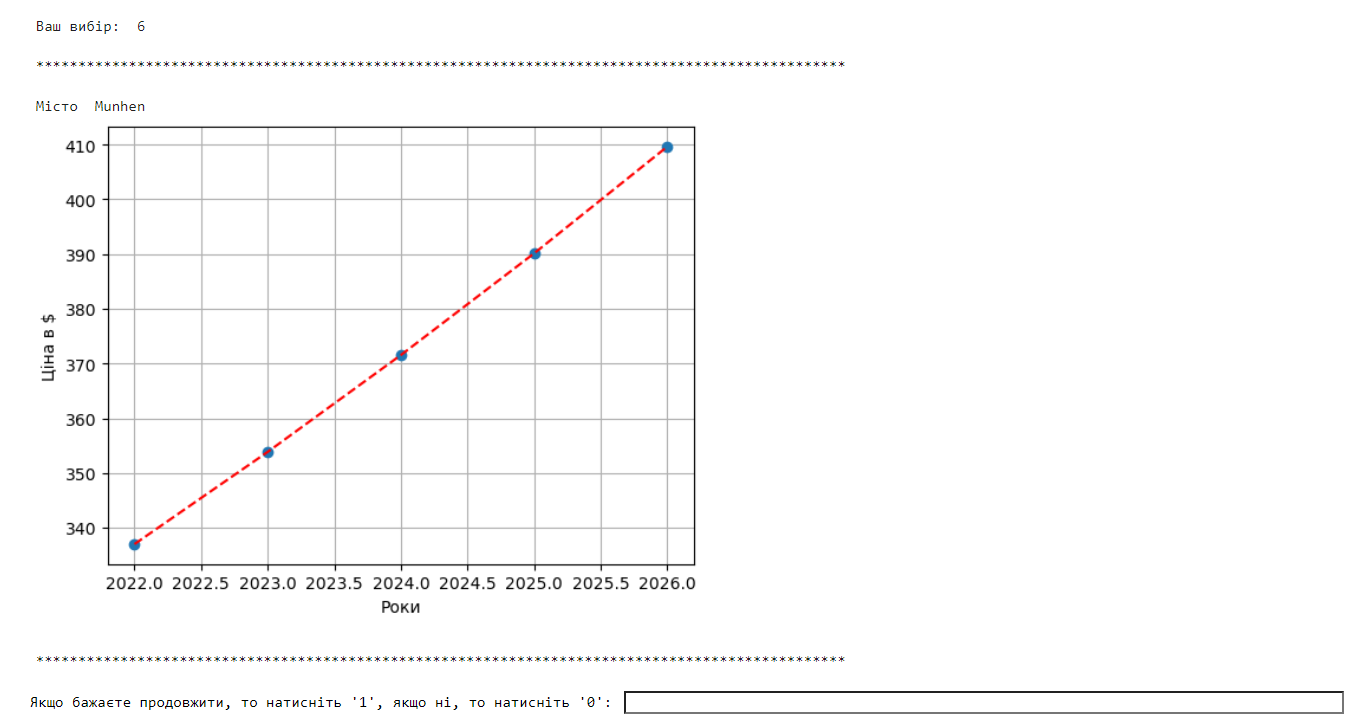


Рисунок 3.7 – Пункт меню №6

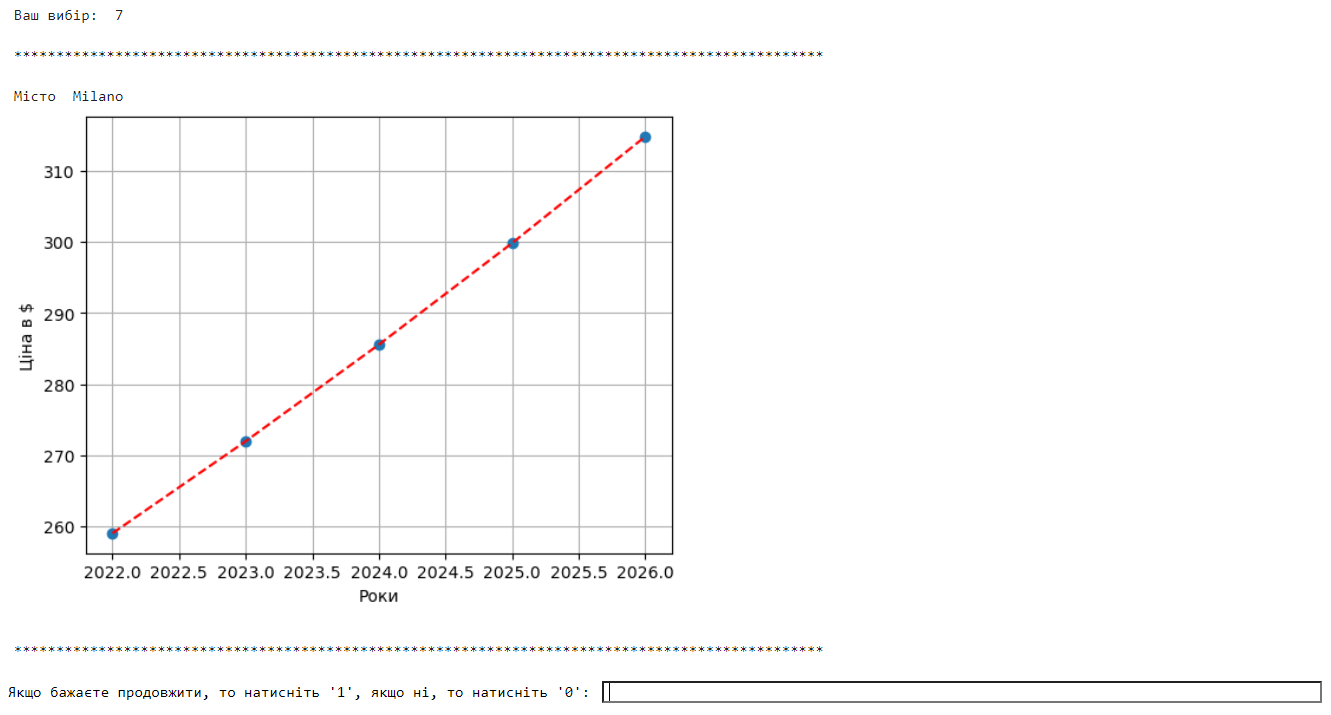


Рисунок 3.8 – Пункт меню №7

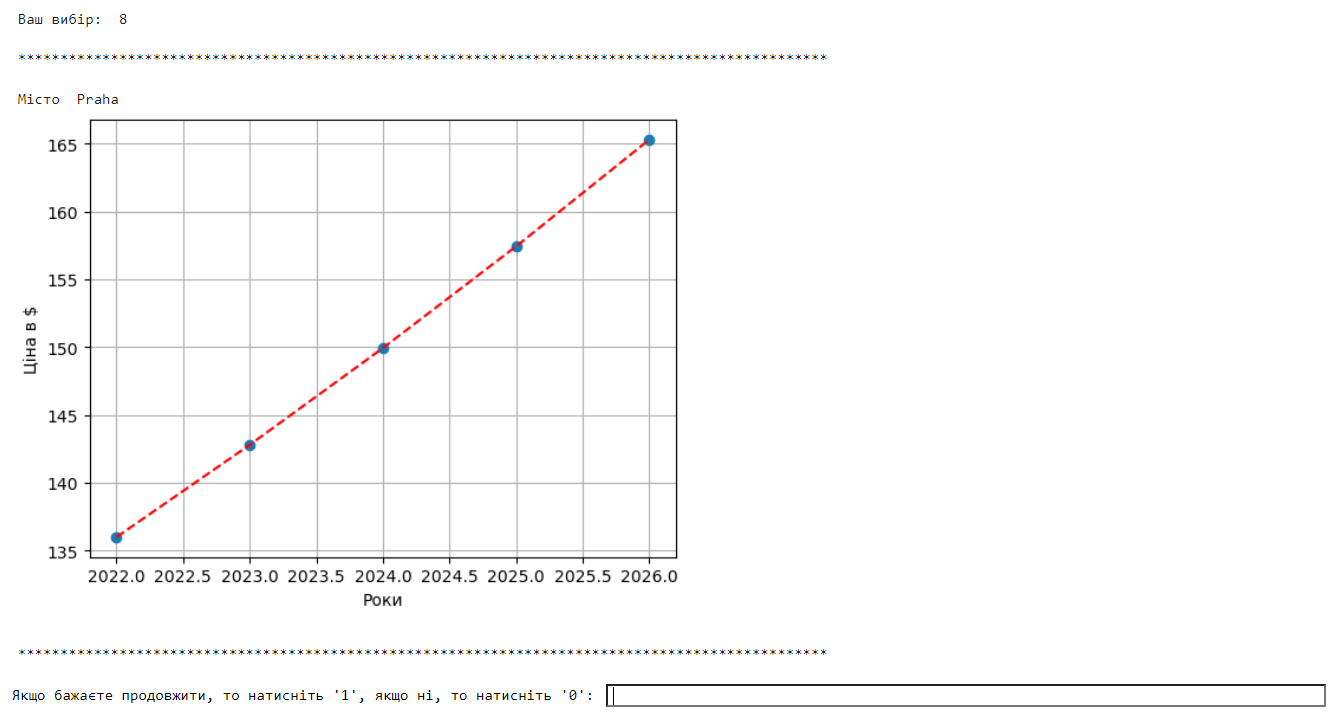


Рисунок 3.9 – Пункт меню №8

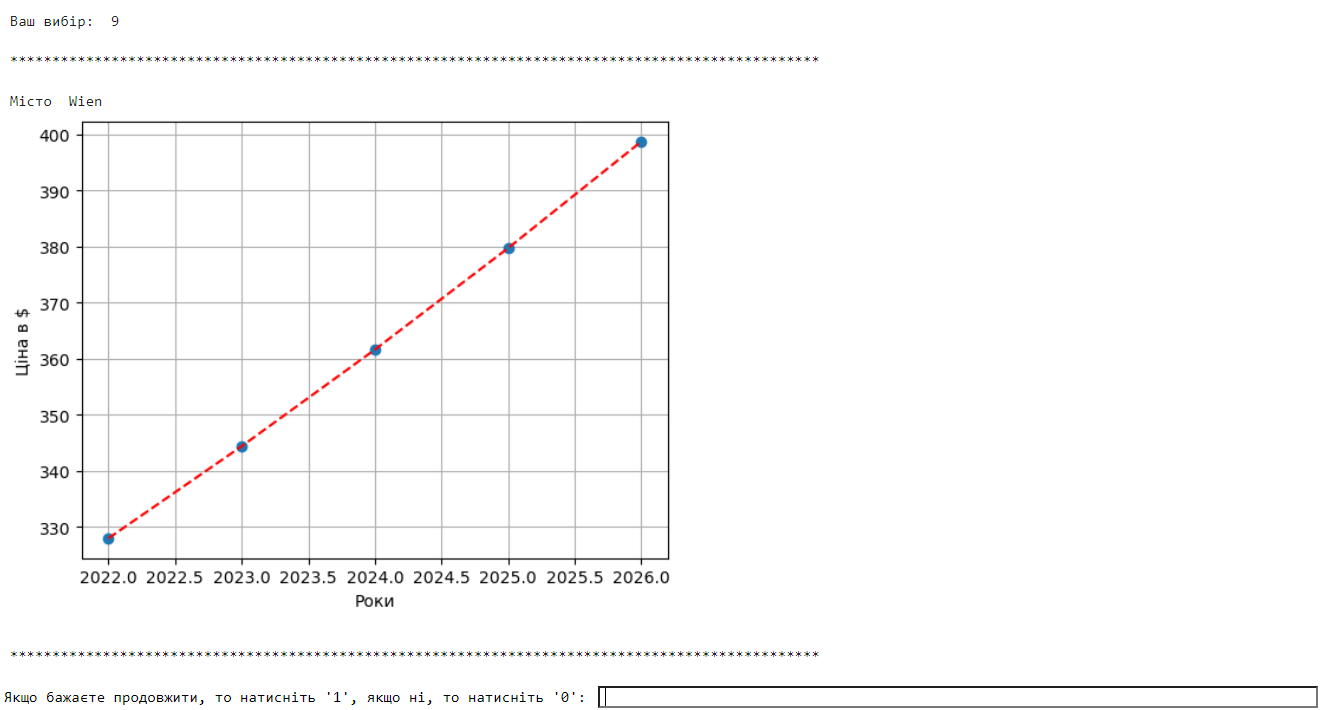


Рисунок 3.10 – Пункт меню №9

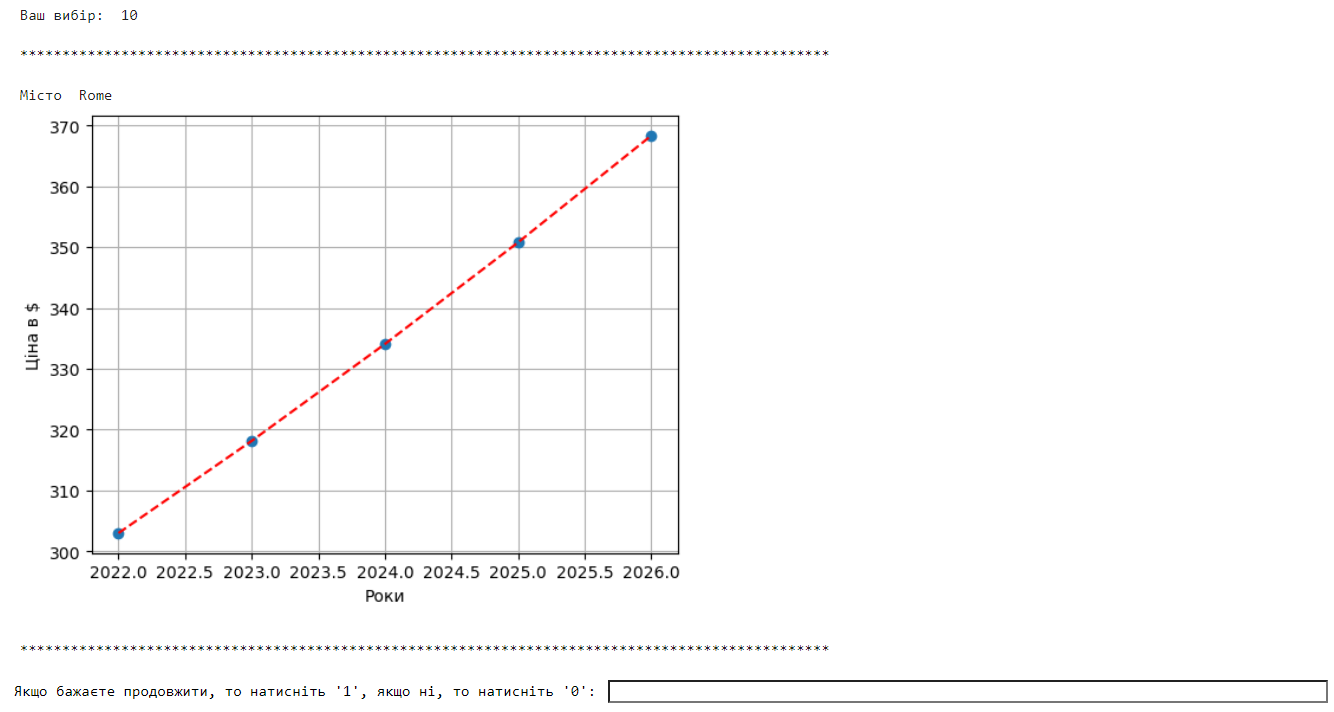


Рисунок 3.11 – Пункт меню №10

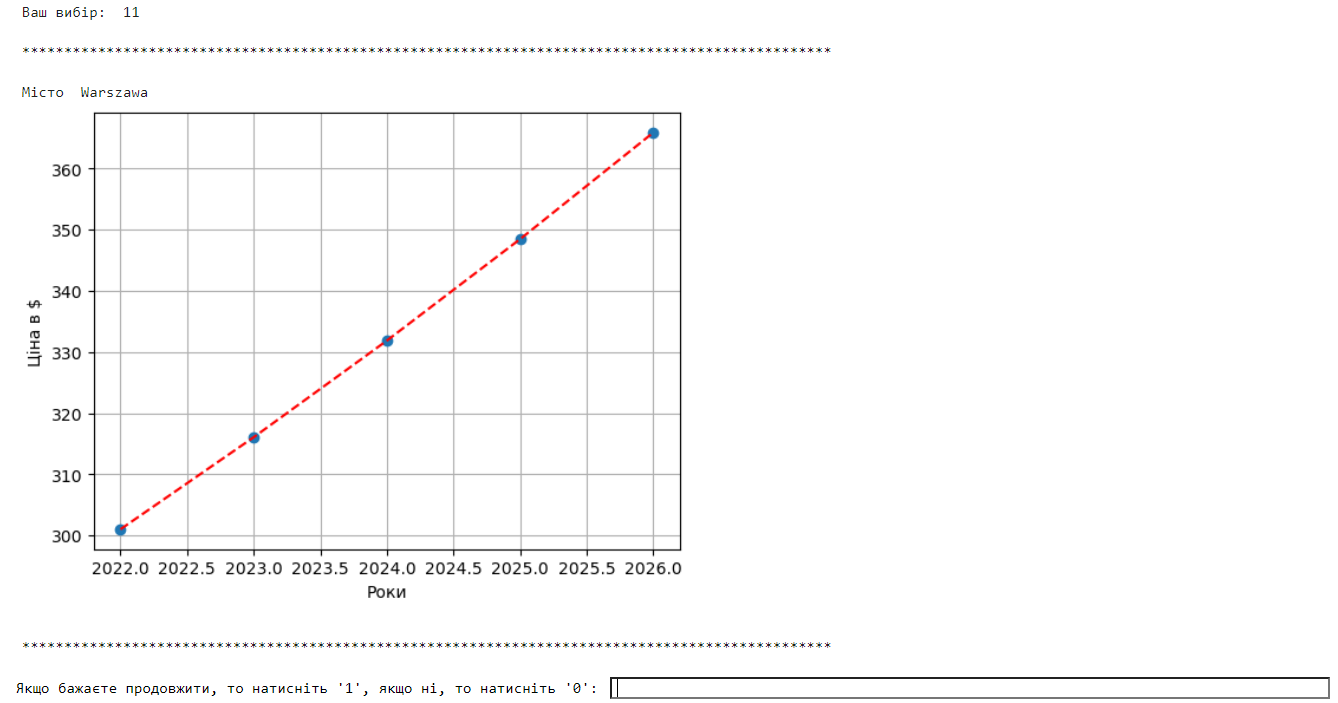


Рисунок 3.12 – Пункт меню №11

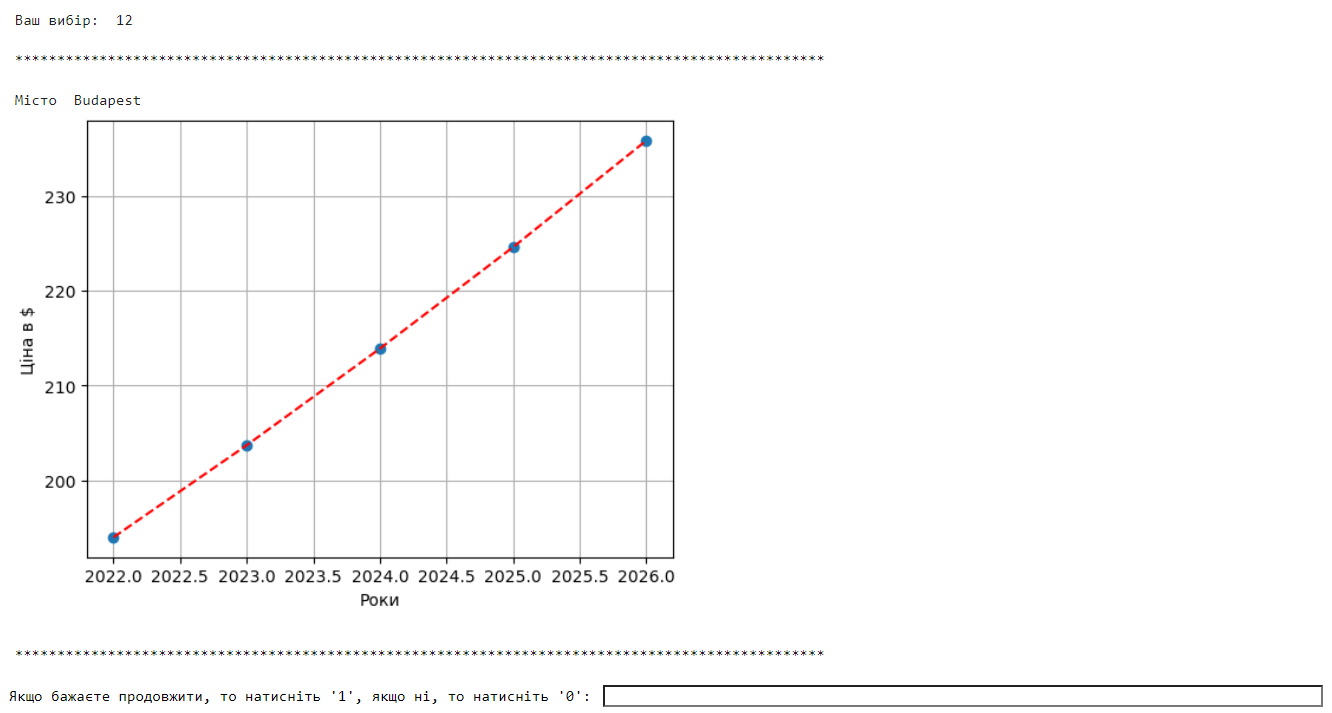


Рисунок 3.13 – Пункт меню №12

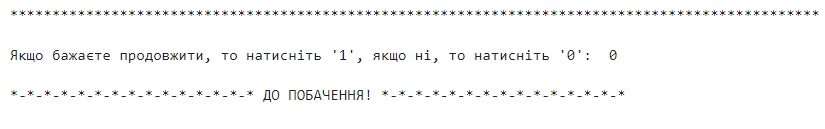


Рисунок 3.14 – Вихід з програми

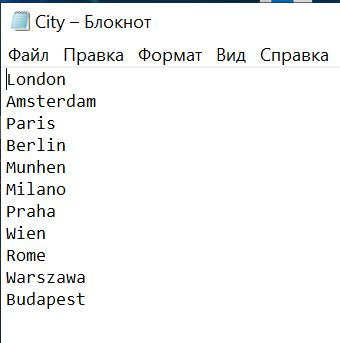


Рисунок 3.15 – Файл з назвами міст

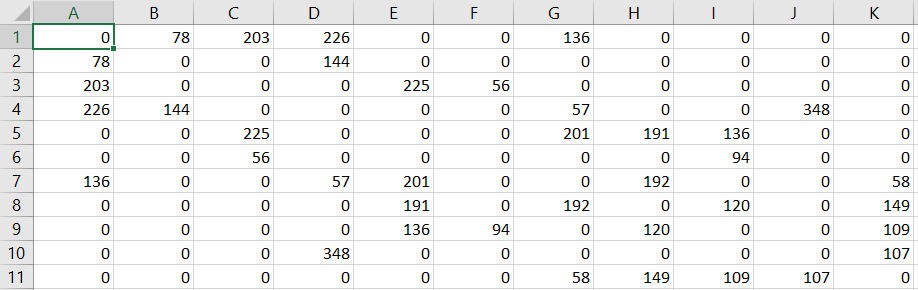


Рисунок 3.16 – CSV файл з таблицею сміжності

# [**ВИСНОВКИ**](#_ВИСНОВКИ)

У результаті виконання індивідуального домашнього завдання була реалізована програма, яка чисельними методами проаналізувала зібрану інформацію для подальшого його застосування.

Програма була реалізована за допомогою бібліотеки Python – SciPy. Для цього були зібрані та опрацьовані дані з сайту купівлі квитків. Було проведено порівняння бібліотек SciPy та NumPy, після чого було вирішено застосувати першу бібліотеку для розв’язання поставлених задач.

# **[СПИСОК ЛІТЕРАТУРИ](#_СПИСОК_ЛІТЕРАТУРИ)**

1. Matplotlib 3.6.2 documentation [Електронний ресурс] – URL: [https://matplotlib.org/stable/api/index.html#](https://matplotlib.org/stable/api/index.html)
2. SciPy Tutorial: What is Python SciPy and How to use it? [Електронний ресурс] – URL: <https://www.edureka.co/blog/scipy-tutorial/>
3. SciPy [Електронний ресурс] – URL: <https://blog.skillfactory.ru/glossary/scipy/>
4. NumPy [Електронний ресурс] – URL: <https://blog.skillfactory.ru/glossary/numpy/>
5. Вивчення NumPy з візуальними прикладами для початківців [Електронний ресурс] – URL: <https://python-scripts.com/numpy>
6. NumPy, частина 1: початок роботи [Електронний ресурс] – URL: <https://pythonworld.ru/numpy/1.html>
7. Посібник з SciPy: що це, і як її використовувати [Електронний ресурс] – URL: <https://pythonru.com/biblioteki/scipy-python>
8. What is the difference between NumPy and SciPy in Python? [Електронний ресурс] – URL: <https://www.educative.io/answers/what-is-the-difference-between-numpy-and-scipy-in-python>
9. Чисельні методи в комп‘ютерних науках. Том 2 [Електронний ресурс] – URL: <http://ism.lpnu.ua/uk/content/chyselni-metody-v-kompyuternyh-naukah-tom-2>
10. Путівник мовою програмування Python [Електронний ресурс] – URL: <https://pythonguide.rozh2sch.org.ua/>