

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

магістр

(ступінь вищої освіти)

на тему ДОСЛІДЖЕННЯ І ПРОГРАМНА РЕАЛІЗАЦІЯ ЗАСОБІВ
РОЗВ'ЯЗКУ ЗАДАЧ ОПТИМІЗАЦІЇ
RESEARCH AND SOFTWARE IMPLEMENTATION OF TOOLS FOR
SOLVING OPTIMIZATION PROBLEMS

Виконав(ла): студент(ка) 2 курсу, групи КНТ-113м
Спеціальності 121 Інженерія програмного
(код і найменування спеціальності)

забезпечення

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

ДЕЙНЕГА Л.Ю.

(ПРИЗВИЩЕ та ініціали)

Керівник ДУБРОВІН В.І.

(ПРИЗВИЩЕ та ініціали)

Рецензент ШИТІКОВА О.В.

(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет КНТ
Кафедра програмних засобів
Ступінь вищої освіти магістр
Спеціальність 121 Інженерія програмного забезпечення
(код і найменування)
Освітня програма (спеціалізація) Інженерія програмного забезпечення
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
Сергій СУББОТІН
“ ” 2024 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

ДЕЙНЕГИ Лариси Юріївни

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Дослідження і програмна реалізація засобів розв'язку задач оптимізації. Research and Software Implementation of Tools for Solving Optimization Problems

керівник проєкту (роботи) к.т.н., професор, ДУБРОВІН Валерій Іванович,
(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від “ 07 ” жовтня 2024 року № 398

2. Строк подання студентом проєкту (роботи) 15 грудня 2024 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2 Обґрунтування вибору програмних засобів розробки. 3 Основні рішення щодо реалізації компонентів системи. 4. Експлуатація, тестування та експериментальне дослідження програми.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, кількість слайдів, плакатів)

Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	ПРИЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	ДУБРОВІН В.І., професор		
Нормоконтроль	БЄЛОВА А.В., асистент		

7. Дата видачі завдання “9” вересня 2024 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання
2	Аналіз предметної області.	1 тиждень	Розділ 1
3	Вибір мови програмування та інших технологій розробки.	2 тиждень	Розділ 2
4	Розробка структури програми.	3 тиждень	Розділ 3
5	Розробка програми.	4-7 тижні	Розділи 3,4
6	Тестування та експериментальне дослідження програмного забезпечення.	8 тиждень	Розділ 4
7	Оформлення пояснювальної записки та документів до неї.	9-10 тижні	Додатки
8	Нормоконтроль та рецензування.	11 тиждень	
9	Захист роботи.	12 тиждень	

Студент(ка)

(підпис) Лариса ДЕЙНЕГА
(Ім'я ПРИЗВИЩЕ)

Керівник проєкту (роботи)

(підпис) Валерій ДУБРОВІН
(Ім'я ПРИЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи магістра:
87 с., 2 табл., 33 рис., 2 дод., 29 джерел.

ЗАДАЧІ ОПТИМІЗАЦІЇ, ЛІНІЙНЕ ПРОГРАМУВАННЯ, НЕЛІНІЙНА
ОПТИМІЗАЦІЯ, МОВА ПРОГРАМУВАННЯ, ПРОГРАМНИЙ
ЗАСТОСУНОК.

Об'єкт дослідження – процес розробки програмного забезпечення для
розв'язку задач оптимізації.

Предмет дослідження – програмні засоби для реалізації розв'язку задач
оптимізації.

Мета роботи – дослідження та розробка програмного забезпечення для
розв'язку задач оптимізації.

Матеріали, методи та технічні засоби: мова програмування Python,
об'єктно-орієнтоване програмування, персональний комп'ютер з процесором
AMD Athlon(tm) X4 під управлінням операційної системи Microsoft Windows
10.

Результати: розроблено програмне забезпечення для демонстрації
роботи методів розв'язку задач оптимізації.

Висновки: виконано проєктування та реалізація програмного
забезпечення для розв'язку задач оптимізації, здійснено тестування
розробленого програмного забезпечення.

Галузь використання – програмний застосунок може бути застосований
в багатьох галузях, наприклад, в навчальному процесі закладів освіти.

ABSTRACT

Explanatory note to the diploma qualifying work of the master: 87 pages, 2 tables, 33 figures, 2 appendixes, 29 sources.

OPTIMIZATION PROBLEMS, LINEAR PROGRAMMING, NONLINEAR OPTIMIZATION, PROGRAMMING LANGUAGE, SOFTWARE APPLICATION.

The object of the study is the process of developing software for solving optimization problems.

The subject of the study is software tools for implementing the solution of optimization problems.

The purpose of the work is the study and development of software for solving optimization problems.

Materials, methods and technical means: Python programming language, object-oriented programming, personal computer with an AMD Athlon(tm) X4 processor running the Microsoft Windows 10 operating system.

Results: software has been developed to demonstrate the operation of optimization problem solving methods.

Conclusions: software has been designed and implemented for solving optimization problems, and the developed software has been tested.

Field of use – the software application can be used in many areas, for example, in the educational process of educational institutions.

ЗМІСТ

	С.
Перелік скорочень та умовних познач	8
Вступ	9
1 Аналіз предметної області	11
1.1 Огляд основних задач оптимізації	11
1.1.1 Лінійне програмування	11
1.1.2 Квадратичне програмування	13
1.1.3 Нелінійна оптимізація	13
1.1.4 Стохастична оптимізація	15
1.1.5 Комбінаторна оптимізація	15
1.2 Галузі застосування оптимізаційних задач	16
1.3 Огляд електронних ресурсів для розв'язування оптимізаційних задач ..	17
1.3.1 Wolfram Alpha	17
1.3.2 MATLAB Optimization Toolbox	19
1.3.3 NEOS Server for Optimization	20
1.3.4 Gurobi Optimizer	21
1.3.5 Висновок	21
1.4 Висновок за розділом 1	22
2 Обґрунтування вибору програмних засобів розробки	23
2.1 Вибір мови програмування для розробки системи	23
2.1.1 Можливості C++	23
2.1.2 MATLAB - інструмент для наукових обчислень	24
2.1.3 Python – універсальна мова для широкого кола завдань	25
2.1.4 Google OR-Tools – ресурс для розв'язання задач оптимізації	27
2.1.5 Висновок	27
2.2 Вибір інтегрованого середовища розробки	29
2.2.1 Visual Studio Code	30
2.2.2 Jupyter Notebook	31

2.2.3 Spyder	32
2.2.4 PyCharm.....	32
2.2.5 Python IDLE	34
2.2.6 Висновок	35
2.3 Можливості вибраних пакетів мови Python.....	36
2.3.1 Пакет Scipy.....	36
2.3.2 Пакет Matplotlib	44
3 Основні рішення щодо реалізації компонентів системи.....	45
3.1 Структура проєкту.....	45
3.2 Структура модуля main_window.....	46
3.3 Структура і алгоритм роботи модуля lin_prog.....	47
3.4 Структура і алгоритм роботи модуля min_fx.....	48
3.5 Структура і алгоритм роботи модуля min_fxу.....	51
3.6 Висновок за розділом 3	53
4 Експлуатація, тестування та експериментальне дослідження програми	55
4.1 Інструкція роботи користувача з основним вікном проєкту.....	55
4.2 Інструкція роботи користувача з вікном "Задача лінійного програмування"	56
4.3 Інструкція роботи користувача з вікном "Нелінійна одновимірна оптимізація".....	60
4.4 Інструкція роботи користувача з вікном "Нелінійна двовимірна оптимізація".....	63
4.5 Висновок про експлуатацію і тестування програми.....	67
Висновки.....	69
Перелік джерел посилання	71
Додаток А Текст програми	74
Додаток Б Слайди презентації.....	80

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

IDE – integrated development environment;

GUI – graphical user interface;

ЛП – лінійне програмування;

ООП – об'єктно-орієнтоване програмування.

ВСТУП

Сучасні задачі оптимізації займають центральне місце у багатьох галузях, від економіки і до інженерії. Методи розв'язку цих задач дозволяють знаходити найкращі можливі рішення в умовах обмежених ресурсів, що має важливе значення у розподілі ресурсів, мінімізації витрат, максимізації прибутку та покращенні загальної продуктивності.

Дослідження ресурсів розв'язку задач оптимізації є надзвичайно актуальним у сучасному світі з кількох причин.

Швидкий розвиток технологій. Зростання обчислювальних потужностей комп'ютерів, поява нових алгоритмів та програмних засобів відкривають нові можливості для розв'язання складних оптимізаційних задач.

Глобалізація та конкуренція. В умовах глобалізації підприємства та організації стикаються з необхідністю приймати оптимальні рішення в умовах невизначеності та обмежених ресурсів. Оптимізація дозволяє підвищити ефективність виробництва, знизити витрати, покращити якість продукції та послуг.

Складність сучасних систем. Сучасні системи стають все більш складними та взаємопов'язаними. Для управління такими системами необхідні інструменти, які дозволяють знаходити оптимальні рішення в умовах великої кількості змінних та обмежень.

Міждисциплінарність. Задачі оптимізації виникають у різних галузях знань: економіці, фінансах, інженерії, медицині, логістиці тощо. Розвиток методів оптимізації сприяє вирішенню важливих соціальних та екологічних проблем.

Метою даної роботи є комплексне вивчення сучасних ресурсів для розв'язання задач оптимізації, їх класифікація, порівняльний аналіз та розробка програмного застосунку для вирішення конкретних типів задач.

Для досягнення мети необхідно вирішити наступні завдання:

- провести огляд літератури з питань теорії оптимізації та сучасних програмних засобів для розв’язання її задач;
- визначити основні класи задач оптимізації;
- провести аналіз програмних засобів для розв’язання задач оптимізації;
- розробити програмний застосунок для розв’язання конкретних типів задач оптимізації.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд основних задач оптимізації

Математичною оптимізацією або математичним програмуванням в математиці, інформатиці та дослідженні операцій називають відбір найкращого елементу (за певним критерієм) з множини доступних альтернатив [1].

У найпростішому випадку задача оптимізації полягає у знаходженні екстремуму (мінімуму або максимуму) дійсної функції шляхом систематичного вибору вхідних значень з дозволеного набору та обчислення значення функції. Подальші узагальнення теорії та методів оптимізації до інших формулювань становлять велику область прикладної математики. Взагалі, оптимізація охоплює знаходження «найкращих можливих» значень деякої цільової функції в межах області визначення, включаючи різні типи цільових функцій та різні типи областей значення [2].

Залежно від специфіки задачі, методів розв'язання та галузей застосування, задачі оптимізації поділяють на різні типи, які представлені нижче.

1.1.1 Лінійне програмування

Лінійне програмування (ЛП) передбачає максимізацію або мінімізацію лінійної цільової функції за наявності лінійних обмежень.

Загальною задачею лінійного програмування називається задача, яка полягає у визначенні максимального (мінімального) значення функції (1.1):

$$F = \sum_{j=1}^n c_j \cdot x_j, \quad (1.1)$$

за умов (1.2)-(1.4):

$$\sum_{j=1}^n a_{ij} \cdot x_j \leq b_i, i = 1..k, \quad (1.2)$$

$$\sum_{j=1}^n a_{ij} \cdot x_j = b_i, i = k + 1..m, \quad (1.3)$$

$$x_i \geq 0, i = 1..L, L \leq n, \quad (1.4)$$

де a_{ij}, b_i, c_j – задані постійні величини;

x_j – шукані змінні;

$k \leq m$ [3].

Задачу ЛП можна розв'язати такими методами:

- симплексний метод ефективний для розв'язання задач із невеликим числом обмежень;
- метод внутрішніх точок добре працює для великих задач;
- двоїстий симплексний метод використовується, коли є попереднє рішення [4].

Задачі лінійного програмування широко застосовуються в різних галузях діяльності. При плануванні виробництва для розрахунку оптимального виробничого плану з урахуванням обмежень на сировину, обладнання та трудові ресурси. В логістиці для оптимізації складських запасів: визначення оптимального рівня запасів для мінімізації витрат на зберігання. В фінансовій галузі для оптимізації портфеля інвестицій: складання портфеля цінних паперів з максимальною прибутковістю при заданому рівні ризику. В маркетингу для оптимізації рекламних кампаній: розподіл рекламних бюджетів між різними каналами для досягнення максимальної ефективності.

1.1.2 Квадратичне програмування

До задач квадратичного програмування відносять задачі, які мають лінійні обмеження, а цільова функція являє собою суму лінійної і квадратичної функції (1.5) - (1.7):

$$F = c_1x_1 + c_2x_2 + \dots c_nx_n + c_{11}x_1^2 + c_{22}x_2^2 \dots + 2c_{12}x_1x_2 + \dots + 2c_{n-1,n}x_{n-1}x_n, \quad (1.5)$$

$$\sum_{j=1}^n a_{ij} x_j = b_i \quad (i = 1..m), \quad (1.6)$$

$$x_j \geq 0 \quad (j = 1..n). \quad (1.7)$$

До методів розв'язання задач квадратичного програмування належать:

- метод Лагранжа для знаходження оптимуму з обмеженнями;
- метод Ньютона, що використовується для розв'язання задач із гладкими функціями [5].

Задачі квадратичного програмування мають широкий спектр застосувань. При оптимізації портфеля для складання портфеля цінних паперів з максимальною прибутковістю при заданому рівні ризику. Квадратична функція може відображати дисперсію прибутковості. При проектуванні інженерних систем для оптимізації розміщення елементів системи з мінімізацією ваги або витрат. В машинному навчанні задачі підбору параметрів моделей, такі як опорні вектори машини (SVM), часто зводяться до задач квадратичного програмування [6]. В робототехніці для планування траєкторій роботів з урахуванням обмежень на швидкість, прискорення та інші параметри.

1.1.3 Нелінійна оптимізація

Задача нелінійної оптимізації представляє собою задачу пошуку екстремуму деякої функції. Екстремум – це крайні значення мінімуму або

максимуму функції. Розрізняють екстремуми: локальні – екстремуми у досить малій околиці даної точки, глобальні – екстремуми у всій розглянутій області значень змінних.

Якщо глобальний екстремум функції шукають на всьому просторі незалежних змінних, то така задача оптимізації називається задачею на безумовний екстремум, а методи її вирішення – методами безумовної оптимізації. Якщо ж екстремум шукають при деяких обмеженнях на незалежні змінні, то така задача оптимізації називається задачею на умовний екстремум, а методи її вирішення – методами оптимізації з обмеженнями [3].

Для розв’язку задач нелінійної оптимізації застосовують такі методи:

- методи вилучення інтервалів, які використовують мінімаксну стратегію пошуку, застосовуються для унімодальних одновимірних функцій;
- метод Ньютона-Рафсона застосовується для одновимірних двічі диференційованих функцій;
- метод середньої точки та метод січних застосовується для одновимірних диференційованих функцій [7];
- метод найшвидшого спуску: на кожному кроці відбувається рух в напрямку, протилежному градієнту функції. Метод простий у реалізації, але може повільно сходитися для складних функцій;
- метод спряжених градієнтів зберігає інформацію про попередні напрямки пошуку, що дозволяє швидше досягати оптимального рішення;
- метод найкрутішого спуску з постійним кроком подібний до методу найшвидшого спуску, але довжина кроку є фіксованою. Метод може бути ефективним для простих задач;
- метод Ньютона використовує інформацію про гессіан функції (матрицю других похідних) для побудови квадратичної апроксимації цільової функції в околі поточної точки. Зазвичай сходиться швидше, ніж методи першого порядку, але вимагає обчислення гессіана;

– квазіньютонів методи апроксимують гессіан за допомогою інформації про градієнти. Ці методи більш ефективні, ніж метод Ньютона, коли обчислення гессіана є складним [5].

Нелінійної оптимізація застосовується в енергетиці для оптимізації розподілу ресурсів, на виробництві для оптимізації процесів та виробничих планів, в економіці для моделювання економічних систем, в фінансовій галузі для оптимізації портфеля інвестицій та цінової оптимізації, в машинному навчанні для підбору параметрів моделей, в наукових обчисленнях для розв'язання нелінійних рівнянь та оптимізації експериментів.

1.1.4 Стохастична оптимізація

Стохастична оптимізація виконується в умовах невизначеності, коли деякі параметри є випадковими.

Методом розв'язання таких задач є метод Монте-Карло, основна ідея якого полягає у використанні вибірки випадкових чисел для одержання імовірнісних або детермінованих оцінок будь-яких величин

Також в стохастичній оптимізації використовується динамічне програмування – пошук оптимального рішення в умовах змінних станів.

Застосовується стохастична оптимізація при управлінні ризиками, в економіці для прогнозування цін, в інженерії для проєктування систем у невизначених умовах [5].

1.1.5 Комбінаторна оптимізація

Комбінаторна оптимізація полягає в пошуку найкращого рішення з великої кількості дискретних варіантів. Прикладом задач цього типу є задача комівояжера, в якій потрібно знайти найкоротший шлях для відвідання всіх міст.

Методи розв'язання задач комбінаторної оптимізації:

- методи гілок і меж ефективні для розбиття задачі на менші частини;
- евристики, наприклад, алгоритм найближчого сусіда;
- методи рою частинок для задач з великою кількістю варіантів [8].

Галузі застосування таких задач:

- транспорт і логістика – для планування маршрутів;
- електроніка – для оптимізації схем;
- біоінформатика – для аналізу геномів.

1.2 Галузі застосування оптимізаційних задач

Після розгляду основних видів оптимізаційних задач можна узагальнити перелік галузей застосування оптимізаційних задач:

- логістика та транспорт: маршрутизація транспортних засобів, оптимізація вантажопотоків, розподіл запасів у ланцюгах постачання;
- енергетика: оптимізація розподілу енергоресурсів, управління енергосистемами, мінімізація втрат в енергетичних мережах;
- фінанси та економіка: портфельна оптимізація, управління ризиками, моделювання та прогнозування фінансових показників;
- виробництво та управління якістю: оптимізація технологічних процесів, зниження витрат, розподіл робочих ресурсів, мінімізація дефектів;
- медицина та біоінформатика: оптимізація лікувальних схем, моделювання та аналіз біологічних процесів, обробка генетичних даних;
- інформаційні технології та зв'язок: оптимізація мережевих ресурсів, управління трафіком, розподіл обчислювальних ресурсів у хмарних системах;
- екологія та управління ресурсами: моделювання стійкого використання природних ресурсів, мінімізація впливу на довкілля, розробка моделей сталого розвитку [9], [10].

Ці приклади демонструють, наскільки широким є спектр застосувань оптимізаційних методів. Завдяки їхньому використанню, компанії та

організації можуть приймати обґрунтовані рішення, підвищувати продуктивність і знижувати витрати.

1.3 Огляд електронних ресурсів для розв'язування оптимізаційних задач

Кількість нових електронних інструментів і платформ, призначених для вирішення оптимізаційних задач, постійно зростає. Ці ресурси охоплюють широкий спектр методів та підходів: від класичних методів лінійного та нелінійного програмування до сучасних технік, таких як генетичні алгоритми, машинне навчання та штучний інтелект. Завдяки їх розвитку можна розв'язувати складні оптимізаційні задачі, які раніше були обчислювально заскладними.

Далі подано огляд сучасних електронних ресурсів, які забезпечують різні методи та інструменти для розв'язку задач оптимізації, а також їхні можливості, переваги та обмеження для практичного застосування.

1.3.1 Wolfram Alpha

Wolfram Alpha є універсальним інструментом для обчислень, включаючи можливості розв'язування задач оптимізації. Ресурс надає широкий вибір методів для оптимізації: лінійне програмування, нелінійна оптимізація, дискретна оптимізація та інші. За допомогою інструменту можна обчислювати максимуми, мінімуми функцій, вирішувати системи рівнянь та досліджувати різні математичні моделі [9].

Приклад використання ресурсу для мінімізації нелінійної одновимірної функції представлено на рис. 1.1.

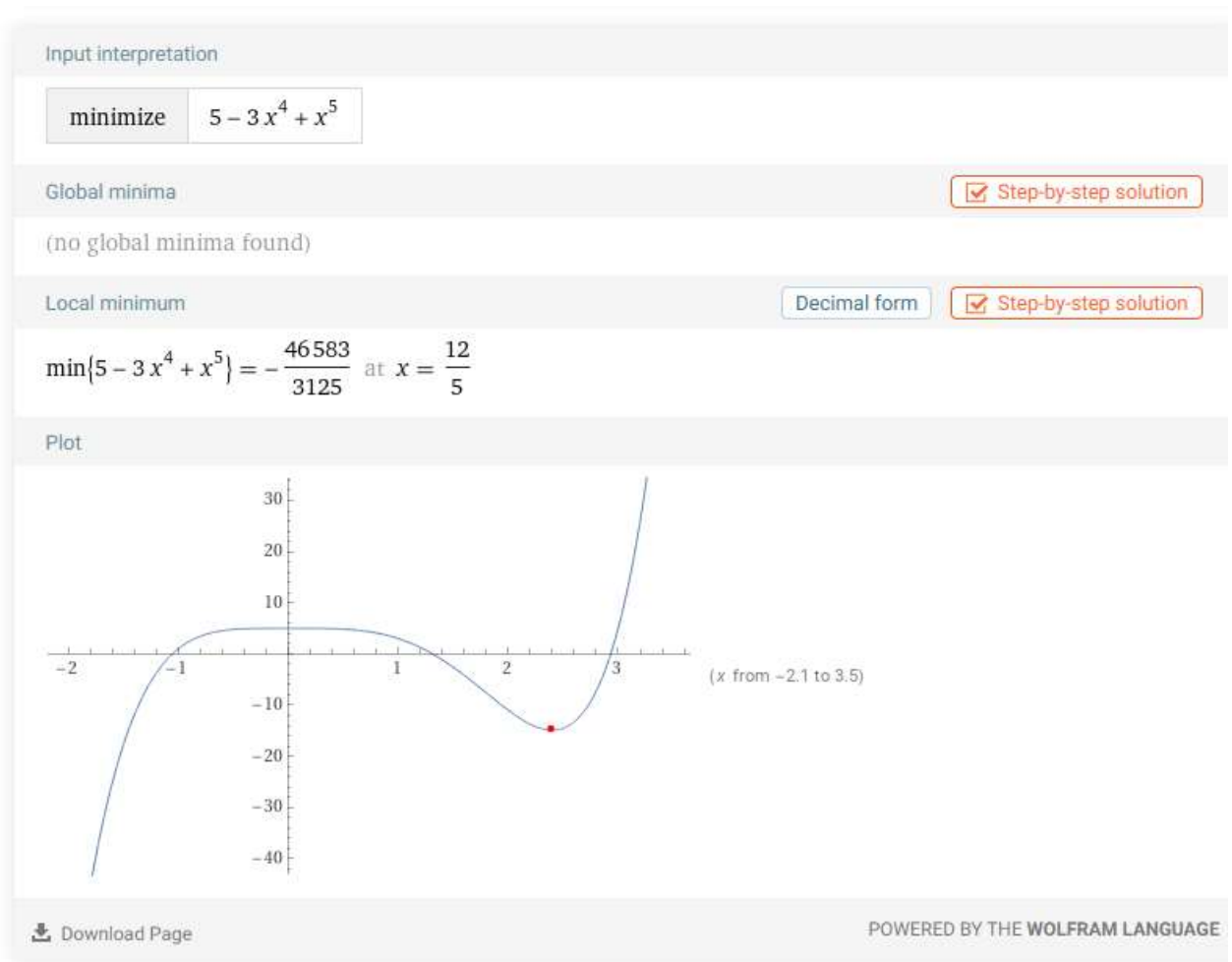


Рисунок 1.1 - Приклад використання Wolfram Alpha [9]

Переваги Wolfram Alpha:

- інтуїтивний інтерфейс;
- підтримка широкого спектру задач оптимізації;
- доступні вебзастосунок і мобільний застосунок.

Недоліки ресурсу:

- обмежена кількість функцій для безкоштовної версії;
- потребує доступу до інтернету.

1.3.2 MATLAB Optimization Toolbox

MATLAB є одним з найбільш популярних інструментів для наукових обчислень, і його додатковий модуль Optimization Toolbox пропонує набір алгоритмів для лінійної, нелінійної, глобальної оптимізації. MATLAB забезпечує широкі можливості для вирішення як класичних, так і специфічних задач оптимізації, використовуючи різні методи (градієнтні, генетичні алгоритми, симплексний метод тощо) [10].

Приклади використання MATLAB представлені на рис. 1.2, 1.3 [11].

```
f = @(x,y) x.*exp(-x.^2-y.^2)+(x.^2+y.^2)/20;  
fsurf(f,[-2,2], 'ShowContours','on')
```

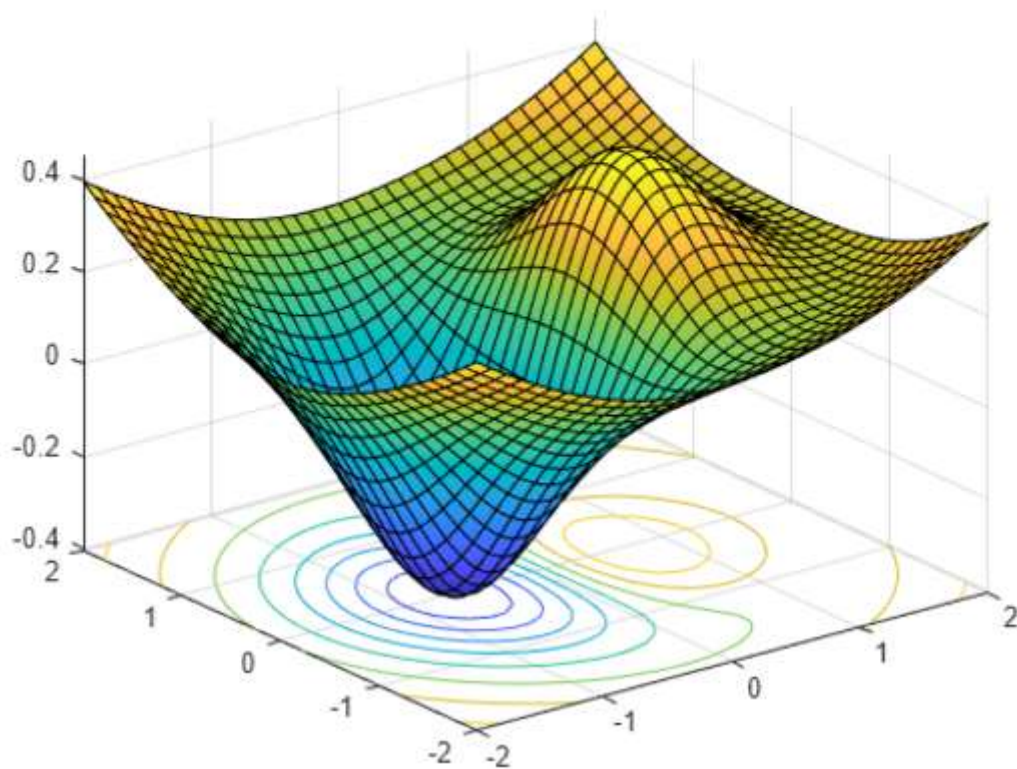


Рисунок 1.2 – Приклад побудови графіка нелінійної функції в MATLAB [11]

```

fun = @(x) f(x(1),x(2));
x0 = [-.5; 0];
options = optimoptions('fminunc','Algorithm','quasi-newton');
options.Display = 'iter';
[x, fval, exitflag, output] = fminunc(fun,x0,options);

```

Iteration	Func-count	f(x)	Step-size	First-order optimality
0	3	-0.3769		0.339
1	6	-0.379694	1	0.286
2	9	-0.405023	1	0.0284
3	12	-0.405233	1	0.00386
4	15	-0.405237	1	3.17e-05
5	18	-0.405237	1	3.35e-08

Рисунок 1.3 – Приклад пошуку мінімуму нелінійної функції в MATLAB [11]

Переваги MATLAB:

- можливість налаштовувати алгоритми під конкретну задачу;
- широкі можливості для візуалізації даних;
- висока швидкість обчислень.

Недоліки MATLAB:

- висока вартість ліцензії;
- потреба в навичках програмування.

1.3.3 NEOS Server for Optimization

NEOS Server – це онлайн платформа для розв’язання задач оптимізації, яка пропонує широкий вибір оптимізаційних алгоритмів, включаючи лінійне, нелінійне та цілочисельне програмування. Цей ресурс дозволяє користувачам надсилати задачі для обробки на високопродуктивних обчислювальних серверах та отримувати результати.

Це потужний онлайн інструмент. Він підтримує широке коло задач, включаючи лінійну, нелінійну, цілочисельну та інші види оптимізації. NEOS забезпечує користувачів засобами для розв’язання задач без потреби в локальних інсталяціях програмного забезпечення та доступний з різних

інтерфейсів, включаючи інтеграцію з Python через бібліотеки Puomo та інших інструментів [12].

Преваги NEOS:

- підтримка різних мов програмування та форматів для введення даних;
- обчислення виконуються на потужних серверах.

Недоліки NEOS:

- потребує інтернету;
- можливі затримки через завантаженість сервера.

1.3.4 Gurobi Optimizer

Gurobi – це комерційне програмне забезпечення для оптимізації, яке пропонує інструменти для лінійного, нелінійного та змішаного програмування. Воно вважається одним з найшвидших і надійних інструментів для вирішення великих задач оптимізації і часто використовується у промислових та наукових проєктах [13].

Переваги Gurobi:

- висока продуктивність;
- підтримка паралельних обчислень;
- оптимізація великих задач.

Недоліки Gurobi:

- висока вартість;
- доступ до безкоштовної версії обмежений для академічних користувачів.

1.3.5 Висновок

Сучасні електронні ресурси для розв'язку задач оптимізації надають широкий спектр інструментів, які дозволяють користувачам досліджувати і

вирішувати складні проблеми в різних галузях. Кожен ресурс має свої переваги та недоліки, тому вибір оптимального інструменту залежить від специфіки задачі, наявності ресурсів та рівня підготовки користувача.

1.4 Висновок за розділом 1

Після аналізу доступних електронних ресурсів для розв'язання задач оптимізації стає очевидною потреба у створенні нового безкоштовного програмного забезпечення, яке зможе задовольнити широке коло користувачів і уникнути недоліків існуючих рішень.

Створений програмний продукт має бути інтуїтивно зрозумілим у використанні, здатним розв'язувати найпоширеніші задачі оптимізації та працювати в офлайн-режимі, що дозволить уникнути залежності від підключення до Інтернету. Це програмне забезпечення стане вагомим інструментом для користувачів, надаючи їм зручний і доступний спосіб вирішення задач оптимізації без необхідності вкладати значні кошти або витрачати час на освоєння складних систем.

2 ОБҐРУНТУВАННЯ ВИБОРУ ПРОГРАМНИХ ЗАСОБІВ РОЗРОБКИ

2.1 Вибір мови програмування для розробки системи

2.1.1 Можливості C++

C++ є однією з найбільш затребуваних та потужних програмних мов, особливо в галузях, що потребують високої продуктивності та прямого доступу до апаратних ресурсів.

Однією з ключових особливостей C++ є її висока продуктивність. Мова дозволяє програмістам керувати пам'яттю і ресурсами на низькому рівні, що робить її ідеальною для завдань, які потребують максимальної ефективності. Можливість прямої взаємодії з апаратурою і керування пам'яттю дає C++ перевагу в сценаріях, де швидкість є критично важливою.

C++ пропонує багатий набір функцій, включаючи об'єктно-орієнтоване програмування (ООП), багатозадачність, керування пам'яттю та шаблони. Ці можливості дозволяють створювати складні та високо ефективні програмні системи. Завдяки підтримці як низькорівневих, так і високорівневих операцій, C++ може бути використаний для широкого спектру завдань [16].

C++ – мова з декількома парадигмами, які можуть вирішити проблему багатьма різними способами. C++ підтримує процедурну, функціональну і об'єктно-орієнтовану парадигми. Багатопарадигмова мова програмування дозволяє програмістам вибирати конкретний підхід або змішувати частини різних парадигм програмування, надаючи гнучкість для розв'язання різних проблем [17].

Переваги використання багатомовності включають скорочення часу, витраченого на вивчення нових носіїв для різних мов, і полегшення переходу до різних парадигм.

C++ має велику колекцію бібліотек, які полегшують розробку програмного забезпечення. Ці бібліотеки охоплюють різні області – від математичних обчислень до роботи з мережею і графікою. Стандартна бібліотека шаблонів (STL) є однією з найпотужніших і найпопулярніших

бібліотек, що надає контейнери, алгоритми та ітератори для ефективної роботи з даними.

C++ також знаходить застосування у наукових та інженерних обчисленнях. Мова використовується для розробки симуляцій, моделювання фізичних процесів та аналізу даних. Висока продуктивність і можливість роботи з великими обсягами даних роблять її доречною для завдань, які потребують інтенсивних обчислень [16].

Однією з головних проблем мови C++ є її складний синтаксис. Це може ускладнити процес навчання для новачків, а також збільшити ризик помилок під час написання коду.

Хоча управління пам'яттю є перевагою, воно також може стати недоліком. Неправильне використання пам'яті може призвести до витоку пам'яті та інших проблем, таких як сегментаційні помилки, що ускладнює відлагодження та підтримку коду.

C++ не забезпечує вбудованих механізмів захисту, таких як управління буфером або перевірка меж масивів. Це може призвести до вразливостей у коді, які можуть бути використані зловмисниками.

Великі проекти на C++ можуть вимагати значного часу на компіляцію. Це може уповільнити процес розробки, особливо при частих змінах у коді [18].

2.1.2 MATLAB - інструмент для наукових обчислень

MATLAB - це високоефективна мова програмування та інтерактивне середовище для виконання складних математичних обчислень, моделювання систем, аналізу даних, візуалізації та розробки алгоритмів. Він широко використовується у наукових дослідженнях, інженерії, фінансах та інших галузях, де потрібна обробка великих обсягів даних і виконання складних математичних операцій.

MATLAB спеціалізується на роботі з матрицями і векторами, що робить його ідеальним інструментом для лінійної алгебри, статистичного аналізу та інших чисельних методів.

Ресурс має велику бібліотеку вбудованих функцій для математичних операцій, статистичного аналізу, обробки сигналів, оптимізації та ін. MATLAB дозволяє створювати різноманітні графіки, діаграми та анімації для візуалізації даних та результатів обчислень. Також він має широкий спектр додаткових інструментів для вирішення конкретних задач, таких як Simulink для моделювання систем, Statistics and Machine Learning Toolbox для статистичного аналізу та машинного навчання, і багато інших.

MATLAB дозволяє розробляти власні алгоритми і функції, а також інтегрувати їх з іншими мовами програмування, має інтерактивне середовище для розробки, тестування та налагодження програм.

Недоліки MATLAB:

- це комерційний продукт, що може бути дорогим для індивідуальних користувачів та невеликих компаній;
- для дуже великих обчислень MATLAB може бути повільнішим за інші мови програмування;
- деякі вбудовані функції MATLAB можуть бути складними для розуміння і модифікації [19].

2.1.3 Python – універсальна мова для широкого кола завдань

Python - це одна з найпопулярніших мов програмування в світі, відома своєю простотою, читабельністю та універсальністю. Завдяки великій кількості бібліотек і фреймворків, Python використовується в різних галузях, від веб-розробки до наукових обчислень і штучного інтелекту.

Python є потужним інструментом для створення веб-додатків. Фреймворки Django і Flask дозволяють швидко розробляти максимально ефективні вебзастосунки [20].

У наукових дослідженнях та аналізі даних Python став однією з найпопулярніших мов програмування. Її простота, гнучкість і багата екосистема інструментів ідеально підходять для роботи з великими обсягами даних і виконання складних аналізів. Бібліотека NumPy має потужний функціонал для роботи з багатовимірними масивами і виконання математичних операцій, що підходить для наукових обчислень. Бібліотека Pandas надає високорівневі структури даних та інструменти для аналізу даних, спрощує роботу з табличними даними, дає змогу фільтрувати, сортувати, агрегувати та об'єднувати дані. Pandas також має потужні можливості для обробки пропущених даних і часових рядів. Бібліотека Matplotlib надає інструменти для візуалізації даних, дозволяючи створювати графіки, діаграми та сюжети. Вона чудово підходить для створення якісних візуалізацій у наукових публікаціях і презентаціях. SciPy – це набір бібліотек для виконання наукових та інженерних обчислень. SciPy надає інструменти, необхідні для виконання складних алгоритмів та аналізу даних, функції для чисельної оптимізації, статистичного моделювання, опрацювання сигналів, опрацювання зображень і багато чого іншого [21].

Крім того, Python дозволяє автоматизувати рутинні завдання за допомогою скриптів: автоматизація роботи з файлами, збір даних з веб-сайтів, управління базами даних та багато іншого. Бібліотеки TensorFlow, PyTorch і scikit-learn надають потужні інструменти для створення моделей машинного навчання і нейронних мереж [22].

Код Python добре структурований і легко зрозумілий, що полегшує його підтримку і модифікацію. Бібліотека Tkinter дозволяє створити зручний графічний інтерфейс користувача (GUI – graphical user interface).

Недоліки Python:

- Python, як інтерпретована мова, може бути повільнішим за компільовані мови, такі як C++ або Java, особливо для обчислювально інтенсивних задач;

– динамічна типізація може призводити до помилок під час виконання, якщо типи даних не визначені правильно.

2.1.4 Google OR-Tools – ресурс для розв'язання задач оптимізації

Google OR-Tools — це відкрита бібліотека, розроблена компанією Google, яка надає широкий спектр алгоритмів і інструментів для вирішення різноманітних задач оптимізації.

OR-Tools підтримує різні мови програмування, включаючи C++, Python, Java і C#. Бібліотека надає інтуїтивний інтерфейс для формулювання задач оптимізації, що дозволяє зосередитися на самій проблемі, а не на деталях реалізації алгоритмів. OR-Tools використовує сучасні алгоритми оптимізації, що дозволяють розв'язувати складні задачі в розумний час. Бібліотека підтримує різні типи моделей оптимізації, включаючи лінійні, цілочисельні, потокові і змішані моделі. OR-Tools можна легко інтегрувати з іншими інструментами і бібліотеками. Але для деяких дуже специфічних задач можуть знадобитися більш спеціалізовані інструменти.

Бібліотека має відкритий код, що дозволяє вивчати її внутрішню роботу і вносити власні зміни, активно розвивається, що гарантує додавання нових функціональних можливостей і покращення продуктивності. OR-Tools є безкоштовною для використання в будь-яких проектах.

2.1.5 Висновок

Всі описані вище характеристики різних засобів розробки структуровані в табл. 2.1.

Таблиця 2.1 – Характеристики програмних засобів розробки

Критерій	C++	Python	Google OR-Tools	MATLAB
Призначення	Високопродуктивні програми, що потребують точного контролю над ресурсами	Швидка розробка, аналіз даних, наукові та освітні проекти	Оптимізація, комбінаторні та лінійні задачі	Аналіз даних, обчислення, моделювання, створення прототипів
Синтаксис	Складний, потужний, вимагає точного управління пам'яттю	Простий, зручний, орієнтований на швидку розробку	Інтеграція з Python і C++ для оптимізації	Інтуїтивно зрозумілий, спеціалізований для математичних розрахунків
Швидкість виконання	Дуже висока, підходить для задач реального часу	Помірна, залежить від зовнішніх бібліотек	Висока, якщо використовується з C++	Висока для чисельних розрахунків, повільніше для графічних задач
Наявні бібліотеки для оптимізації	Eigen, CGAL, Gurobi, Boost	SciPy, NumPy, PuLP, TensorFlow	Вбудовані функції для оптимізації, легко інтегрується з іншими бібліотеками	Вбудовані пакети для лінійного, нелінійного програмування та задач динамічного програмування
Можливість створення GUI	Можлива через Qt, GTK, але складна в реалізації	Легко реалізується через Tkinter, PyQt	GUI не підтримується, потребує інтеграції з іншими інструментами	Інтегровані засоби розробки GUI (GUIDE)
Вартість використання	Безкоштовний (переважно), можливі платні бібліотеки	Безкоштовний (з відкритим кодом), інколи потрібні платні бібліотеки	Безкоштовний, доступний з відкритим кодом	Дорога ліцензія, доступні академічні знижки

За результатами огляду мов програмування можна зробити такий висновок. Python — оптимальний вибір для розробки застосунків з графічним інтерфейсом для задач оптимізації завдяки поєднанню простоти, доступності,

потужного інструментарію для математичних обчислень і гнучкості у створенні графічного інтерфейсу користувача. Він дозволяє зосередитися на розв'язанні задачі, мінімізуючи витрати часу на технічну реалізацію.

2.2 Вибір інтегрованого середовища розробки

Інтегроване середовище розробки (IDE – integrated development environment) — це платформа, яка об'єднує у собі всі необхідні інструменти для ефективної розробки програмного забезпечення, такі як текстовий редактор, компілятор/інтерпретатор, відладчик, а також різноманітні корисні засоби, у зручному та єдиному інтерфейсі.

Існує декілька ключових причин, чому IDE є невід'ємною частиною процесу розробки програмного забезпечення.

Зручний інтерфейс. IDE зазвичай має зручний та інтуїтивно зрозумілий інтерфейс, який дозволяє легко орієнтуватися та швидко знаходити необхідні функції

Ефективність. IDE допомагає заощадити час та зусилля завдяки автоматизації, шаблонам коду, інтелектуальним підказкам та іншим інструментам, які полегшують процес розробки

Підтримка різних мов програмування. Багато IDE підтримують кілька мов програмування, що дозволяє розробникам працювати з різними технологіями в одному середовищі.

Налагодження та тестування. Інтегровані засоби надають можливість ефективної налагодження коду, тестування програм та виявлення помилок [25].

Для обґрунтованого вибору середовища розробки далі представлені найпопулярніші IDE та редактори коду Python з переліком їхніх особливостей.

2.2.1 Visual Studio Code

Visual Studio Code (часто аббревіатура VS Code) – це безкоштовне відкрите середовище розробки від Microsoft. Хоча це не є IDE, виключно призначеним для Python, VS Code здатний використовуватися для розробки Python завдяки своїм розширенням і гнучкості.

Особливості VS Code, які роблять його зручним для розробки на Python:

- розширення Python для VS Code, розроблене Microsoft, надає множинну функцій для розробки Python, включаючи інтелектуальне автодоповнення, форматування коду, налагодження та підтримку Jupyter Notebooks;

- VS Code має вбудовані інструменти для налагодження коду, що дозволяють встановлювати точки зупинки, переглядати змінні та контролювати виконання вашого програмного коду;

- VS Code включає підтримку Git, що дозволяє зручно переглядати різницю між версіями без використання командного рядка;

- підтримка розширень, яка дозволяє налаштовувати середовище розробки під свої потреби. Є велика кількість розширень для підтримки різних мов, бібліотек, фреймворків, інструментів форматування коду та ін.;

- налаштування і гнучкість: можна налаштовувати все, від теми і шрифтів до поведінки редактора коду і інтеграції з іншими інструментами [26].

Недоліком середовища є те, що деякі розширення можуть призвести до зниження швидкодії редактора, особливо при роботі з великими проєктами.

2.2.2 Jupyter Notebook

Jupyter Notebook — це відкрите середовище розробки, що популярне серед науковців даних, викладачів, студентів та програмістів, що працюють з даними. Він особливо популярний серед користувачів Python, але підтримує багато інших мов програмування.

Ключові особливості Jupyter Notebook:

- Jupyter Notebook дозволяє виконувати код в інтерактивних “блокнотах”, які змішують код, текст, математичні формули, візуалізації та інше. Це дозволяє вам працювати з кодом і даними в більш візуальний і експериментальний спосіб;
- Jupyter Notebook підтримує Markdown для форматування тексту і LaTeX для математичних формул;
- інтеграція з Python та іншими мовами;
- Jupyter може виконувати розподілені обчислення для обробки великих наборів даних;
- Jupyter інтегрується з багатьма популярними бібліотеками Python, такими як NumPy, Pandas, Matplotlib, Scikit-Learn і багатьма іншими, що дозволяє працювати з даними, виконувати складні обчислення і створювати візуалізації прямо в браузері;
- спільна робота та віддалена робота [26].

Але використання Jupyter Notebook для великих та складних проєктів може бути неефективним через обмежену можливість організації та управління кодом. Також недоліком є те, що робота в Jupyter Notebook залежить від браузера, тому можуть виникати проблеми з сумісністю або продуктивністю на деяких платформах [25].

2.2.3 Spyder

Spyder — це відкрита інтегрована середовище розробки (IDE), що спеціально розроблена для мови програмування Python. Spyder була створена для науковців, інженерів і аналітиків даних, що працюють з науковими бібліотеками Python, такими як NumPy, SciPy, Matplotlib, Pandas, IPython, SymPy та Cython.

Spyder має багатофункціональний редактор коду з підсвічуванням синтаксису, автозавершенням коду, навігацією по коду та іншими функціями, що полегшують написання коду Python.

Spyder інтегрується з IPython, що дозволяє вам використовувати інтерактивні блокноти IPython та інші інструменти IPython безпосередньо в IDE.

Spyder має вбудований налагоджувач для пошуку та виправлення помилок в коді Python.

Підтримка додатків: Spyder можна розширювати за допомогою додатків, що дозволяє вам додавати нові функції та інструменти в IDE [26].

2.2.4 PyCharm

PyCharm вже кілька років вважається одним із найпотужніших і зручних інтегрованих середовищ розробки (IDE) для мови Python, і це звання заслужено завдяки її багатофункціональності та зручності використання. Однією з головних переваг PyCharm є її регулярне оновлення та підтримка всіх сучасних версій Python.

Однією з ключових функцій PyCharm є її інтелектуальний редактор коду. Він не тільки підсвічує синтаксис, а й пропонує автодоповнення, аналізує код у реальному часі та вказує на помилки, попереджаючи про можливі проблеми ще до того, як програма буде запущена. Це значно прискорює процес написання коду, а також підвищує його якість.

PyCharm пропонує потужний вбудований налагоджувач, який дозволяє запускати код по кроках, відстежувати значення змінних та бачити стек викликів. Це допомагає швидко знаходити помилки та розуміти, як працює програма. Крім того, PyCharm підтримує різні інструменти для тестування, як-от юніт-тести, що дозволяє розробникам переконатися у правильній роботі їхнього коду ще до його розгортання.

Для розробників, що працюють з даними, PyCharm пропонує вбудовані засоби для взаємодії з базами даних. За допомогою цієї IDE можна підключатися до різних СУБД (як-от MySQL, PostgreSQL, SQLite та інших), виконувати SQL-запити та керувати даними, не залишаючи робочого середовища. Це особливо зручно при створенні програм, що працюють з більшими обсягами даних.

PyCharm підтримує роботу з різними системами контролю версій, як-от Git, Mercurial та SVN. IDE надає зручний візуальний інтерфейс для керування репозиторіями, дозволяючи відстежувати зміни в коді, створювати гілки, зливати їх та переглядати історію змін. Це допомагає ефективно управляти проєктами, особливо, коли над ними працює команда розробників.

PyCharm вигідно виділяється на тлі інших IDE своєю глибокою інтеграцією з широким спектром інструментів та сервісів, що робить її потужним та гнучким інструментом для розробників. Ця інтеграція допомагає автоматизувати процеси, покращити продуктивність та легко взаємодіяти з сучасними технологіями, необхідними для повноцінної розробки та розгортання проєктів.

Як і будь-яке інше інтегроване середовище розробки, PyCharm має свої сильні сторони, які роблять його популярним серед Python-розробників. А також має деякі недоліки, які можуть виявитися важливими залежно від вимог проєкту та уподобань користувача.

Одним із суттєвих недоліків PyCharm є її вимогливість до ресурсів комп'ютера. На менш потужних пристроях IDE може працювати повільно, особливо якщо в проєкті залучені великі обсяги даних або багато файлів.

Хоча PyCharm доступна у безкоштовній версії Community, її професійна версія (PyCharm Professional) пропонує набагато більше можливостей, як-от робота з базами даних, розширена підтримка фреймворків та інтеграція з Docker. Однак, професійна версія є платною, що може бути мінусом для індивідуальних розробників або невеликих команд з обмеженим бюджетом.

PyCharm надає безліч функцій, які можуть бути корисні для досвідчених розробників, однак це робить інтерфейс складнішим для розуміння новачками. Спочатку новачки можуть бути приголомшені кількістю інструментів та налаштувань, що потребує часу на освоєння [27].

2.2.5 Python IDLE

Це стандартне інтегроване середовище розробки, яке поставляється разом з Python і може бути першим кроком для багатьох початківців у програмуванні. Хоча середовище IDLE може здатися менш потужним порівняно з іншими, воно має свої унікальні переваги та приваблює користувачів своєю простотою та доступністю.

Python IDLE має такі переваги:

- має простий та зрозумілий інтерфейс, що робить його ідеальним вибором для початківців;
- наявність вбудованого інтерпретатора Python, що дозволяє виконувати код безпосередньо в середовищі розробки;
- доступний на різних операційних системах, що робить його доступним для широкого кола користувачів.

Недолік Python IDLE – в порівнянні з більш розширеними інтегрованими середовищами розробки, IDLE може мати обмежені можливості для складних та великих проєктів [25].

2.2.6 Висновок

Результати огляду найпопулярніших IDE для розробки на Python в контексті теми роботи представлені в табл. 2.2.

Таблиця 2.2 – Огляд популярних IDE

Критерій	PyCharm	Visual Studio Code	Spyder	Jupyter Notebook	Python IDLE
Призначення	Повнофункціональний IDE для Python, підтримка для веб-розробки (Django, Flask), розробка для наукових обчислень	Легке, гнучке середовище з багатьма розширеннями, загальний редактор коду	Орієнтоване на наукові обчислення та аналіз даних	Інтерактивне середовище для роботи з кодом, ідеально підходить для аналізу даних	Легке вбудоване середовище для основного кодування
Вартість	Платне (з можливістю безкоштовної версії Community)	Безкоштовне	Безкоштовне	Безкоштовне	Безкоштовне
Простота використання	Середня, багато функцій, але може бути складним для початківців	Дуже проста, але потребує налаштувань для повної функціональності	Середня, зручне для користувачів, що працюють з наукою і даними	Просте для інтерактивного програмування	Дуже проста, для початківців
Швидкість	Помірна, зважаючи на наявність багатьох функцій та підтримку великого коду	Швидка, оскільки є легким редактором	Повільніше, зважаючи на зручність роботи з великими обсягами даних	Повільна при роботі з великими наборами даних	Дуже швидка, оскільки це базовий редактор

Кінець таблиці 2.2

Критерій	PyCharm	Visual Studio Code	Spyder	Jupyter Notebook	Python IDLE
Наявність налагоджувача	Так, з розширеними можливостями налагодження	Так, через плагіни (наприклад, Python Debugger)	Так, з підтримкою для налагодження коду наукових обчислень	Так, базовий налагоджувач, зручно для інтерактивного кодування	Так, базовий налагоджувач

Враховуючи проведений аналіз найкращим вибором для написання проєкту з графічним інтерфейсом для розв'язку задач оптимізації буде PyCharm. PyCharm — це потужне середовище для розробки на Python з багатим набором інструментів, яке підтримує розробку як для наукових обчислень, так і для створення повноцінних програм з графічними інтерфейсами. Хоча PyCharm має платну версію, безкоштовна версія (PyCharm Community) також є потужним інструментом, що цілком підходить для більшості завдань. IDE має вбудовану документацію та підказки, що значно полегшує процес розробки. PyCharm має потужний вбудований налагоджувач, який дозволяє детально вивчати виконання коду, що є важливим при розробці складних алгоритмів оптимізації та графічних інтерфейсів.

2.3 Можливості вибраних пакетів мови Python

2.3.1 Пакет Scipy

Scipy побудований на базі NumPy (пакет для збереження даних масивів та операції з ними) та дозволяє керувати даними, а також візуалізувати їх за допомогою різних високорівневих команд.

Можливості пакета розподілені за кількома модулями або підпакетами, які об'єднані призначенням. Це потрібно, щоб фахівець міг підключити замість

повної бібліотеки потрібну частину для економії ресурсів. Основні модулі SciPy:

- кластерний модуль (`scipy.cluster`). Кластеризація — популярний метод категоризації даних шляхом об'єднання їх у групи;
- модуль констант (`scipy.constants`). Модуль містить у собі безліч фізико-математичних констант та одиниць, дуже зручно, адже не треба вручну прописувати їх значення, а використовувати вже готові. Константи використовуються при фізичних або математичних розрахунках;
- набори даних (`scipy.datasets`). Модуль використовує та залежить від `Roosh`, пакета Python, створеного для спрощення отримання файлів даних. `Roosh` використовує ці сховища для отримання відповідних файлів набору даних під час виклику функції набору даних;
- оптимізація та пошук кореня (`scipy.optimize`). Модуль надає функції для мінімізації (або максимізації) цільових функцій, можливо з обмеженнями. Він містить розв'язування нелінійних задач (з підтримкою як локальних, так і глобальних алгоритмів оптимізації), лінійне програмування, обмежені та нелінійні найменші квадрати, пошук кореня та підгонку кривої. Цей процес може бути корисним в різних областях, таких як: машинне навчання, оптимізація виробничих процесів, фінансовий аналіз.

`Scipy.optimize` містить ряд корисних методів для оптимізації різних типів функцій:

- `minimize_scalar()` і `minimize()` – мінімізація функції однієї змінної та багатьох змінних відповідно;
- `curve_fit()` – метод оптимізації, який знаходить параметри моделі, що найкращим чином апроксимує задані дані;
- `root_scalar()` і `root()` – пошук нулів функції однієї і багатьох змінних відповідно;
- `linprog()` – мінімізація лінійної цільової функції з лінійною системою обмежень [28].

2.3.1.1 Метод `linprog()`

Метод `linprog()` в бібліотеці SciPy є інструментом для розв'язування задач лінійного програмування (linear programming). Метод знаходить мінімальне значення лінійної функції лінійного програмування з лінійними обмеженнями на змінні.

Метод може бути застосований в форматі

`scipy.optimize.linprog(c, A_ub=None, b_ub=None, A_eq=None, b_eq=None, bounds=None, method='highs', callback=None, options=None, x0=None, integrality=None)` з такими параметрами:

- `c` - одновимірний масив коефіцієнтів лінійної цільової функції, обов'язковий параметр;
- `A_ub` - двовимірний масив, необов'язковий - матриця обмежень-нерівностей;
- `b_ub` - одновимірний масив, необов'язковий - вектор обмежень-нерівностей. Кожен елемент є верхню межею відповідного значення $A_ub * x$;
- `A_eq` - двовимірний масив, необов'язковий - матриця обмежень-рівностей;
- `b_eq` - одновимірний масив, необов'язковий - вектор обмежень-рівностей. Кожен елемент має дорівнювати відповідному елементу $A_eq * x$;
- `bounds`, необов'язковий - послідовність пар для кожного елемента вектору x , що визначає мінімальне та максимальне значення цієї змінної рішення x_i . Якщо надається єдиний кортеж, то i буде служити межами для всіх змінних рішення. Використовуйте `(-inf, inf)`, щоб вказати, що обмеження немає. Пара меж за замовчуванням `(0, None)` означає, що всі змінні рішення є невід'ємними;
- `method`, необов'язковий - алгоритм розв'язання задачі стандартної форми. Підтримуються 'highs', 'highs-ds', 'highs-ipm';

- `callback`, необов'язковий - функція зворотного виклику, якщо вона надається, буде викликана принаймні один раз за ітерацію алгоритму;
- `options`, необов'язковий - словник опцій;
- `x0` - одновимірний масив, необов'язковий – початкові значення змінних рішення, які будуть уточнені алгоритмом оптимізації. Цей аргумент наразі використовується лише «переглянутим симплексним» методом і може бути використаний, лише якщо `x0` представляє базове можливе рішення;
- `integrality` - одновимірний масив або `int`, необов'язковий, вказує тип обмеження цілісності для кожної змінної рішення, використовується лише методом 'highs' і ігнорується в інших випадках.

Результат роботи методу `res` - об'єкт типу `OptimizeResult` складається з наведених нижче полів (типи результатів полів можуть залежати від успішності оптимізації, тому рекомендується спершу перевірити `OptimizeResult.status`):

- `x` – одновимірний масив - значення змінних рішення, які мінімізують цільову функцію при дотриманні обмежень;
- `fun` - оптимальне значення цільової функції;
- `stack` - одновимірний масив - значення змінних $b_{ub} - A_{ub} * x$;
- `con` - одновимірний масив – значення обмежень-рівностей $b_{eq} - A_{eq} * x$;
- `success` має значення `True`, коли алгоритму вдається знайти оптимальне рішення;
- `status` - ціле число, що представляє статус виходу з алгоритму, 0, якщо оптимізацію успішно завершено;
- `nit` - загальна кількість ітерацій, виконаних на всіх етапах\$
- `message` - повідомлення статусу виходу з алгоритму [28].

2.3.1.2 Метод мінімізації функції однієї змінної `minimize_scalar()`

Математична функція, яка приймає одне число та дає один результат, називається скалярною функцією.

Мета даної функції пакету - знайти мінімальне значення заданої одновимірної функції.

Результатом `minimize_scalar()` є екземпляр `OptimizeResult` з атрибутами:

- `success` - логічне значення, яке вказує, чи успішно завершилася оптимізація;
- `x` – точка мінімуму;
- `fun` - це значення цільової функції в точці `x`;
- `nfev` – кількість обчислень досліджуваної функції;
- `nit` – кількість ітерацій.

Якщо функція не має мінімуму (наприклад, x^3), то робота `minimize_scalar()` закінчиться помилкою `OverflowError` (занадто велике число) оскільки оптимізатор зрештою намагається отримати число, яке є занадто великим, щоб його обчислив комп'ютер.

З іншого боку, для функцій, що мають більше одного мінімуму, робота `minimize_scalar()` не гарантує знаходження глобального мінімуму функції.

Функція `minimize_scalar()` може використовувати не один метод знаходження мінімуму. Щоб вказати яким саме методом треба провести оптимізацію, необхідно при виклику функції вказати параметр `method='назва методу'`.

Всього існує три вбудованих у функцію методи:

- метод `Brent` використовує алгоритм Брента для знаходження локального мінімуму. Алгоритм використовує зворотну параболічну інтерполяцію, коли це можливо, щоб прискорити збіжність методу золотого перетину;

- метод Golden використовує метод пошуку золотого перетину. Він використовує аналог методу поділу навпіл для зменшення інтервалу в дужках;
- метод Bounded може виконувати обмежену мінімізацію. Він використовує метод Брента для знаходження локального мінімуму в інтервалі $x1 < xopt < x2$.

За замовчанням застосовується метод Bounded, якщо задані межі пошуку, або метод Brent, якщо межі не вказані [28].

2.3.1.3 Метод мінімізації функції багатьох змінних (minimize())

Ця функція може обробляти багатоваріантні вхідні та вихідні дані та має більш складні алгоритми оптимізації. Крім того, minimize() може обробляти обмеження щодо вирішення поставленої задачі. Можна вказати три типи обмежень:

- LinearConstraint. Розв'язок обмежується шляхом взяття внутрішнього добутку значень x рішення з масивом, введеним користувачем, і порівняння результату з нижньою та верхньою межами;
- nonlinearConstraint. Розв'язок обмежений шляхом застосування наданої користувачем функції до значень x рішення та порівняння поверненого значення з нижньою та верхньою межею;
- Bounds. Значення x розв'язку обмежені нижньою та верхньою межею.

Використання цих обмежень може звузити конкретний вибір методу оптимізації, який можна застосувати, оскільки не всі доступні методи підтримують обмеження таким чином.

Далі представлені методи, які можуть бути задані як параметр функції minimize().

Метод Nelder-Mead використовує симплексний алгоритм Нелдера-Міда. Цей алгоритм надійний у багатьох програмах. Однак, якщо можна довіряти числовому обчисленню похідної, інші алгоритми, що

використовують інформацію про першу та/або другу похідну, можуть бути кращими для кращої продуктивності в цілому.

Метод Powell є модифікацією методу спряжених напрямків Пауелла. Він виконує послідовну одновимірну мінімізацію вздовж кожного вектора набору напрямків, який оновлюється на кожній ітерації основного циклу мінімізації. Функція не повинна бути диференційованою.

Метод CG шукає мінімум скалярної функції однієї чи декількох змінних за допомогою алгоритму спряженого градієнта.

Метод BFGS використовує алгоритм Бroyдена-Флетчера-Голдфарба-Шенно. Щоб швидше наблизитися до розв'язку, ця процедура використовує градієнт цільової функції. Якщо градієнт не задано користувачем, він оцінюється за допомогою перших різниць. Метод Бroyдена-Флетчера-Голдфарба-Шанно (BFGS) зазвичай вимагає меншої кількості викликів функцій, ніж симплексний алгоритм, навіть якщо потрібно оцінити градієнт.

Метод Newton-CG - модифікований метод Ньютона, який використовує алгоритм спряженого градієнта для інвертування локального гессіана.

Метод TNC мінімізує скалярну функцію однієї чи декількох змінних за допомогою усіченого алгоритму Ньютона.

Метод COBYLA використовує метод обмеженої оптимізації за допомогою лінійної апроксимації. Алгоритм заснований на лінійних наближеннях цільової функції та кожного обмеження.

Метод SLSQP використовує послідовне програмування найменших квадратів для мінімізації функції однієї чи декількох змінних із будь-якою комбінацією обмежень, рівності та нерівності.

Метод trust-constr – це алгоритм мінімізації скалярної функції з урахуванням обмежень, які задані системою нерівностей.

Методи Newton-CG, trust-ncgi trust-krylov придатні для вирішення масштабних задач із тисячами змінних. Оскільки для отримання розв'язку потрібен лише добуток матриці Гессе з довільним вектором, алгоритм має

низькі вимоги до пам'яті та дозволяє значну економію часу для цих розріджених задач.

Якщо метод не вказано, вибирається один із BFGS, L-BFGS-B, SLSQP, залежно від того, чи має задача обмеження чи межі.

Результати роботи функції `minimize()` із застосуванням різних методів відрізняються несуттєво. Відрізняється кількість ітерацій, обчислень функції та додаткових матриць [28].

2.3.1.4 Метод `curve_fit()`

Метод `scipy.optimize.curve_fit()` є одним з методів оптимізації, який знаходить параметри моделі, яка найкращим чином апроксимує задані дані.

Конкретно, метод `curve_fit()` знаходить найкращі значення параметрів моделі, яка найбільш точно відтворює задані дані. Цей метод можна використовувати для побудови кривих регресії, функцій апроксимації та інших математичних моделей.

Метод `curve_fit()` вимагає задати функцію, яку треба апроксимувати, а також вхідні дані, які складаються з x та y координат. Також можна вказати початкові значення параметрів, які метод буде використовувати під час пошуку найкращих значень параметрів моделі.

Після запуску методу `curve_fit()`, він повертає кортеж з оптимальними значеннями параметрів моделі та коваріаційною матрицею. Коваріаційна матриця вказує на точність оцінки параметрів моделі.

Отже, метод `scipy.optimize.curve_fit()` допомагає виконати апроксимацію даних та знайти параметри моделі, які найбільш точно відтворюють задані дані [28].

2.3.2 Пакет Matplotlib

Matplotlib – програмний пакет Python для візуалізації даних з використанням двовимірної 2D і тривимірної 3D графіки. Пакет Matplotlib за будовою подібний до NumPy, SciPy і IPython.

Пакет підтримує наступні види графіків та діаграм:

- графіки (line plot);
- діаграми розсіювання (scatter plot);
- стовпчасті діаграми (bar chart) і гістограми (histogram);
- секторні діаграми (pie chart);
- деревоподібні діаграми (stem plot);
- контурні графіки (contour plot);
- поля градієнтів (quiver);
- спектральні діаграми (spectrogram).

Користувач може вказати вісі координат, сітку, додати підписи і пояснення, використовувати логарифмічну шкалу або полярні координати.

Нескладні тривимірні графіки можна будувати з допомогою набору інструментів (toolkit) mplot3d. Існують і інші набори інструментів: для картографії, для роботи з Excel, утиліти для GTK та інші.

З допомогою Matplotlib можна створювати як звичайні, так і анімовані зображення.

3 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ КОМПОНЕНТІВ СИСТЕМИ

Програма «Задачі оптимізації» є навчальним або прикладним інструментом для демонстрації та розв'язання задач оптимізації. Вона забезпечує:

- зручний графічний інтерфейс;
- використання сучасних методів з бібліотек Python `scipy.optimize`, `matplotlib` для розв'язання задач та побудови графіків;
- інтерактивний вибір задач і методів їх розв'язання.

Ця програма може бути використана як демонстраційний проєкт у навчальних цілях або для тестування задач оптимізації.

3.1 Структура проєкту

Проєкт побудований за принципами модульної структури, яка має ряд переваг:

- кожен модуль відповідає за розв'язання конкретної задачі, що спрощує внесення змін;
- модулі можна використовувати в інших проєктах, які потребують подібного функціоналу;
- легко розширити функціонал проєкту додаванням нових задач оптимізації, створивши окремий модуль і підключивши його до головного вікна;
- функціонал чітко розподілений за різними модулями. Головне вікно відповідає за інтерфейс і навігацію, а функції окремих частин проєкту реалізовані у спеціалізованих модулях.

Структура проєкту представлена на рис. 3.1.

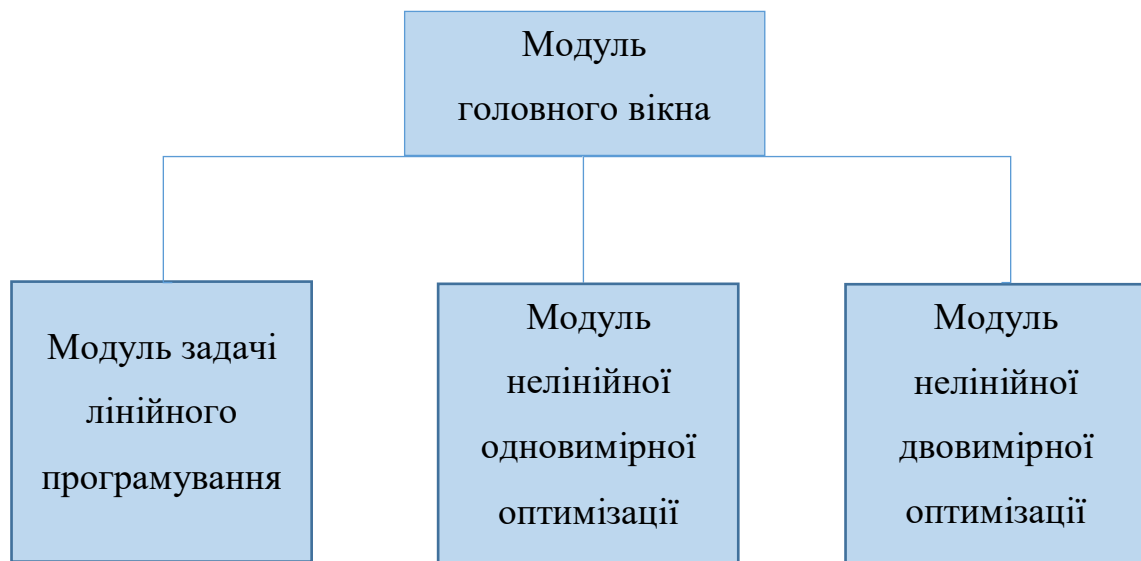


Рисунок 3.1 – Структура проєкту

3.2 Структура модуля `main_window`

Модуль призначений для імпортування бібліотеки `customtkinter` і додаткових модулів `min_fx`, `min_fxy`, `lin_prog`, які містять реалізації функцій для відповідних задач, а також для створення головного вікна графічного інтерфейсу. Головне вікно містить текстовий елемент, який пояснює, що програма демонструє розв'язування задач оптимізації, та кнопки для вибору задачі:

- кнопка "Задача лінійного програмування" відкриває функцію `linear_programming_window` з модуля `lin_prog`;
- кнопка "Нелінійна одновимірна оптимізація" викликає функцію `optimization_window1` з модуля `min_fx`;
- кнопка "Нелінійна двовимірна оптимізація" викликає функцію `optimization_window2` з модуля `min_fxy`.

Головне вікно відображається в нескінченному циклі за допомогою `main.mainloop()`.

3.3 Структура і алгоритм роботи модуля `lin_prog`

Модуль `lin_prog.py` реалізує графічний інтерфейс для розв'язання задачі лінійного програмування за допомогою бібліотеки `customtkinter` та функції `scipy.optimize.linprog`. Модуль дозволяє вводити дані для цільової функції, системи нерівностей та рівнянь, виконувати оптимізацію і відображати результати.

Модуль містить:

- імпорт бібліотек;
- функцію `linear_programming_window`;
- функцію `find_minimum`;
- функцію `result_window`.

Імпортовані бібліотеки:

- `customtkinter` – для створення сучасного графічного інтерфейсу;
- `numpy` – для роботи з багатовимірними масивами;
- `scipy.optimize.linprog` – для розв'язання задачі лінійного програмування;
- `tkinter.messagebox` – для відображення повідомлень про помилки.

Функція `linear_programming_window` створює головне вікно для вводу даних задачі, запуску оптимізації і виведення результату. Створене вікно містить поля для вводу цільової функції, нерівностей і рівнянь, кнопку "Знайти мінімум" для запуску обчислення мінімуму.

Вбудована функція `find_minimum` призначена для виклику `linprog` і обробки результатів.

Функція `find_minimum` виконує обчислення мінімуму за заданими даними. Вона зчитує вхідні дані з полів вводу, перевіряє відповідність розмірностей матриць і векторів, викликає `scipy.optimize.linprog` з переданими обмеженнями та параметрами і обробляє результат і виводить його в новому вікні.

Функція `result_window` створює нове вікно для відображення результатів: точки мінімуму, значення функції або повідомлення про помилку. Вікно також містить кнопку для закриття вікна.

Далі подано алгоритм роботи модуля `lin_prog`.

Крок 1. Ініціалізація вікна і введення даних.

Функція `linear_programming_window` створює головне вікно з текстовими полями для вводу даних задачі:

- цільова функція $f(x)$ – список коефіцієнтів;
- матриця обмежень $A1$ і вектор правих частин $b1$;
- матриця рівнянь $A2$ і вектор правих частин $b2$.

Крок 2. Обробка натискання кнопки "Знайти мінімум".

Викликається функція `find_minimum`, яка:

- зчитує введені дані і перетворює їх у масиви NumPy;
- перевіряє коректність розмірностей $A1-b1$ і $A2-b2$;
- виконує розв'язання задачі за допомогою `scipy.optimize.linprog`.

Крок 3. Аналіз результату.

Якщо розв'язок знайдено, виводиться точка мінімуму і значення функції в цій точці.

Якщо функція необмежена або задача нерозв'язна, виводиться відповідне повідомлення.

Крок 4. Відображення результату.

Результати виводяться в новому вікні, створеному функцією `result_window`. Точка мінімуму вводиться як список дійсних чисел, значення функції – дійсне число. Користувач може закрити вікно результату за допомогою кнопки "Закрити".

3.4 Структура і алгоритм роботи модуля `min_fx`

Модуль `min_fx.py` реалізує графічний інтерфейс для розв'язання задачі нелінійної одновимірної оптимізації. За допомогою функції `minimize_scalar`

модуля `scipy.optimize` знаходиться мінімум нелінійної функції на заданому користувачем інтервалі з можливістю візуалізації графіка функції.

Модуль містить:

- імпорт бібліотек;
- функцію `optimization_window1`;
- функцію `find_minimum1`;
- функцію `plot_function1`;
- функцію `result_window1`.

Імпортовані бібліотеки:

- `customtkinter` – для створення сучасного графічного інтерфейсу;
- `numpy` – для роботи з числовими масивами;
- `matplotlib.pyplot` – для побудови графіків;
- `scipy.optimize.minimize_scalar` – для оптимізації функції;
- `tkinter.messagebox` – для відображення повідомлень про помилки.

Функція `optimization_window1` створює головне вікно для введення даних задачі, вибору методу оптимізації і запуску обчислень. Створене вікно містить: поле для введення функції $f(x)$, поля для введення інтервалу пошуку $[a, b]$, чекбокс для вибору опції побудови графіка, комбобокс для вибору методу оптимізації та кнопку "Знайти мінімум" для запуску обчислень.

Функція `find_minimum1` виконує обчислення мінімуму функції за допомогою `minimize_scalar`. Вона зчитує введені дані з полів інтерфейсу, перевіряє коректність інтервалу $[a, b]$, перетворює введену функцію з рядка в Python-об'єкт. Викликає функцію `minimize_scalar` з вибраним методом і інтервалом та виводить результат у новому вікні, яке створює функція `result_window1`. Якщо вибрано опцію побудови графіка, викликає функцію `plot_function1`.

Функція `plot_function1` будує графік функції на заданому інтервалі з виділенням точки мінімуму. Вона генерує значення функції для великої кількості точок у заданому інтервалі, будує графік функції та виділяє точку мінімуму на графіку.

Функція `result_window1` відображає результати оптимізації у новому вікні. Вона виводить метод оптимізації, точку мінімуму та значення функції. Містить також кнопку "Закрити" для закриття вікна.

Далі подано алгоритм роботи модуля `min_fx`.

Крок 1. Ініціалізація вікна і введення даних.

Функція `optimization_window1` створює графічне вікно для введення даних:

- а) функція $f(x)$ в символьному вигляді;
- б) інтервалу пошуку $[a, b]$ – дійсні числа;
- в) чекбокс для вибору опції побудови графіка;
- г) комбобокс для вибору методу оптимізації:
 - 1) метод Брента (`brent`);
 - 2) метод золотого перетину (`golden`);
 - 3) метод обмеженої оптимізації (`bounded`).

Крок 2. Запуск оптимізації.

Після натискання кнопки "Знайти мінімум" зчитуються введені дані, перевіряється коректність введеного інтервалу та методу та викликається функція `minimize_scalar` з відповідними параметрами.

Крок 3. Обробка результату.

Якщо мінімум знайдено, у новому вікні виводяться результати (крок 4).

Якщо активовано чекбокс графіка, будується графік функції з позначенням мінімуму (крок 5).

Якщо виникла помилка (наприклад, некоректний інтервал), користувач отримує повідомлення про помилку.

Крок 4. Виведення результатів.

У новому вікні відображаються: метод оптимізації (рядок), точка мінімуму (дійсне число), значення функції у точці мінімуму (дійсне число).

Крок 5. Побудова графіка.

Створюється графік функції на інтервалі $[a, b]$. Виділяється точка мінімуму з текстовою анотацією.

3.5 Структура і алгоритм роботи модуля `min_fxy`

Модуль `min_fxy.py` реалізує графічний інтерфейс для розв'язання задачі нелінійної двовимірної оптимізації. Він дозволяє вводити функцію двох змінних, вибирати початкову точку та метод оптимізації, знаходити мінімум і, за бажанням, будувати тривимірний графік функції з позначенням точки мінімуму.

Модуль містить:

- імпорт бібліотек;
- функцію `optimization_window2`;
- функція `find_minimum2`;
- функція `plot_function2`;
- функція `result_window2`.

Імпортовані бібліотеки:

- `customtkinter` – для створення сучасного графічного інтерфейсу;
- `numpy` – для роботи з масивами;
- з бібліотеки `math` імпортовані функції `sin`, `cos`, `tan`, `pow`, `log`, `exp`;
- `matplotlib.pyplot` та `mpl_toolkits.mplot3d` – для побудови графіків;
- `scipy.optimize.minimize` – для оптимізації функцій;
- `tkinter.messagebox` – для виводу повідомлень про помилки.

Функція `optimization_window2` створює графічне вікно для введення даних задачі, вибору методу оптимізації і запуску розрахунків. Створене вікно містить елементи:

- поле для введення функції $f(x,y)$;
- поля для введення початкової точки (x_0,y_0) ;
- чекбокс для вибору опції побудови графіка;
- комбобокс для вибору методу оптимізації;
- кнопку "Знайти мінімум" для запуску обчислень.

Функція `find_minimum2` виконує обчислення мінімуму функції за допомогою функції `minimize` бібліотеки `scipy.optimize`. Функція зчитує введені дані з полів інтерфейсу, перетворює функцію з рядка в об'єкт Python, викликає `minimize` з відповідними параметрами та виводить результат у новому вікні через функцію `result_window2`. Якщо вибрано чекбокс побудови графіка, викликає функцію `plot_function2`.

Функція `plot_function2` будує тривимірний графік функції $f(x,y)$ на діапазоні, що залежить від початкової точки, з позначенням точки мінімуму.

Функція генерує сітку значень для функції $Z=f(x,y)$, створює тривимірний графік поверхні функції та виділяє точку мінімуму червоним маркером з текстовою анотацією.

Функція `result_window2` відображає результати оптимізації у новому вікні. Вона виводить метод оптимізації, точку мінімуму та значення функції у цій точці. Вікно також має кнопку "Закрити" для закриття вікна.

Далі подано алгоритм роботи модуля `min_fxu`.

Крок 1. Ініціалізація вікна та введення даних користувачем.

Функція `optimization_window2` створює інтерфейс із полями для введення даних:

- а) функції $f(x,y)$ в символьному вигляді;
- б) початкової точки (x_0,y_0) – двох дійсних чисел;
- в) чекбокс для вибору опції побудови графіка;
- г) комбобокс для вибору методу оптимізації:
 - 1) метод Нелдера-Міда (Nelder-Mead);
 - 2) метод Пауелла (Powell);
 - 3) метод спряжених градієнтів (CG).

Крок 2. Запуск оптимізації.

Після натискання кнопки "Знайти мінімум" викликається функція `find_minimum2`, яка спочатку перевіряє коректність введених даних. Застосовуються базові перевірки: чи введено функцію, чи обраний метод оптимізації, чи правильно вказані числові значення початкової точки. Якщо

дані введено некоректно, з'являється спливаюче повідомлення з вказанням проблеми. За відсутності проблем введених даних виконується оптимізація за допомогою `scipy.optimize.minimize`.

Крок 3. Обробка результатів.

Якщо пошук мінімуму виявився успішним, виводяться його результати (крок 4).

Якщо оптимізація виявилась неуспішною, з'являється спливаюче повідомлення «Оптимізація не вдалася».

Якщо активовано побудову графіка, викликається `plot_function2`, яка будує тривимірний графік (крок 5).

Крок 4. Виведення результатів

В новому вікні відображаються: обраний метод (рядок), точка мінімуму (два дійсних числа), значення функції (дійсне число).

Крок 5.

Створюється графік тривимірної поверхні функції $f(x,y)$ в діапазоні навколо точки мінімуму, що допомагає оцінити форму функції. Точка мінімуму чітко виділена червоним кольором із підписом, що включає координати (два дійсних числа) та значення функції (дійсне число).

3.6 Висновок за розділом 3

Проект має модульну структуру, що забезпечує гнучкість, зручність у підтримці та розширенні функціоналу. Кожна задача оптимізації реалізована в окремому модулі, а головний модуль відповідає за інтеграцію і запуск програми.

Застосунок має зручний графічний інтерфейс із чіткими підказками для введення даних, інтерактивний вибір методу оптимізації, візуалізація результату через побудову графіків.

Також реалізована обробка помилок. Виводяться зрозумілі повідомлення у разі некоректного введення чи неможливості отримання результату.

4 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ

Програма «Задачі оптимізації» призначена для демонстрації та розв'язання задач оптимізації. Вона має інтуїтивно зрозумілий інтерфейс, що дозволяє легко обирати задачі. Кожна задача оптимізації реалізована у своєму модулі, що забезпечує масштабованість. Користувач може взаємодіяти з кожним вікном окремо та налаштовувати параметри задач.

Проект може працювати під будь-якою з операційних систем: Windows, macOS, Linux. Перед запуском проекту має бути встановлений Python 3.6 або новіший та підключені стандартні бібліотеки: customtkinter, numpy, matplotlib, scipy.

Для початку роботи з проектом користувач має запустити головний файл `main_window.py`.

4.1 Інструкція роботи користувача з основним вікном проекту

Після запуску проекту відкривається головне вікно з назвою "Задачі оптимізації". Воно представлено на рис. 4.1.

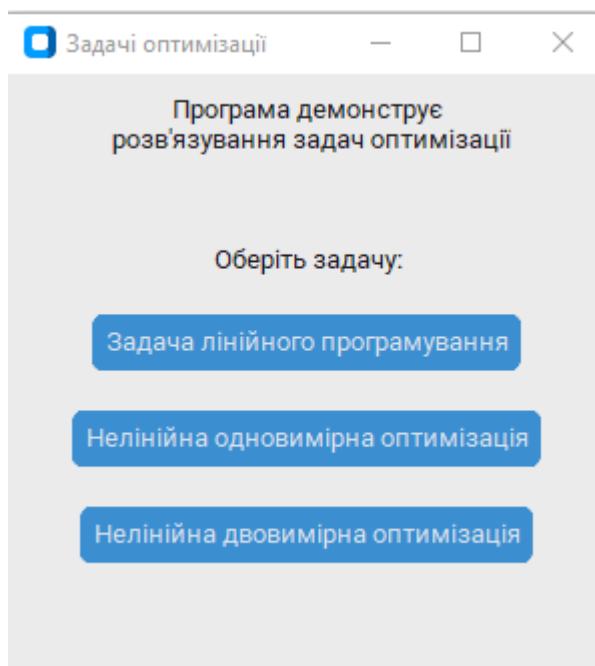


Рисунок 4.1 – Головне вікно програми

Для продовження роботи користувач має обрати задачу оптимізації натисканням на одну з кнопок:

- "Задача лінійного програмування";
- "Нелінійна одновимірна оптимізація";
- "Нелінійна двовимірна оптимізація".

Для закінчення роботи з програмою потрібно закрити головне вікно.

4.2 Інструкція роботи користувача з вікном "Задача лінійного програмування"

Після натискання в головному вікні програми кнопки "Задача лінійного програмування" відкривається відповідне вікно. Воно представлене на рис. 4.2.

Задача лінійного програмування

Введіть вхідні дані задачі

Знайти мінімум функції $f(x)$:

Наприклад, [1, 2, 3]

в області, заданій системою нерівностей $A1 * x \leq b1$:

$A1 =$

Наприклад, [[1, -1], [2, 1]]

$b1 =$

Наприклад, [1, 4]

та рівнянь $A2 * x = b2$ (не обов'язково):

$A2 =$

Наприклад, [[1, 1]]

$b2 =$

Наприклад, [2]

Знайти мінімум

Рисунок 4.2 – Вікно "Задача лінійного програмування"

Для отримання результату цієї задачі оптимізації користувачу потрібно ввести вхідні дані.

У текстовому полі під текстом "Знайти мінімум функції $f(x)$:" користувач має ввести коефіцієнти лінійної функції, яку потрібно мінімізувати. Наприклад: [1, 2, 3].

Під текстом "в області, заданій системою нерівностей $A1 * x \leq b1$:" користувач має заповнити два поля:

- $A1$ – матрицю коефіцієнтів для системи нерівностей. Наприклад: $[[1, -1], [2, 1]]$;
- $b1$ – вектор правих частин нерівностей. Наприклад: $[1, 4]$.
- Під текстом "та рівнянь $A2 * x = b2$ (не обов'язково):" користувач може ввести додаткові умови:
 - $A2$ – матрицю коефіцієнтів для рівнянь. Наприклад: $[[1, 1]]$;
 - $b2$ – вектор правих частин рівнянь. Наприклад: $[2]$.

Якщо обмежень-рівнянь немає, ці поля можна залишити порожніми.

Для запуску обчислень користувач має натиснути кнопку "Знайти мінімум".

Якщо розмірності некоректні або дані введено неправильно, користувач отримає спливаюче повідомлення про помилку.

Після коректного введення даних програма викликає функцію `scipy.optimize.linprog`, яка обчислює мінімум функції з заданими обмеженнями.

Якщо розв'язок знайдено, в новому вікні виводиться точка мінімуму і значення функції. Приклад такого результату наведено на рис. 4.3.

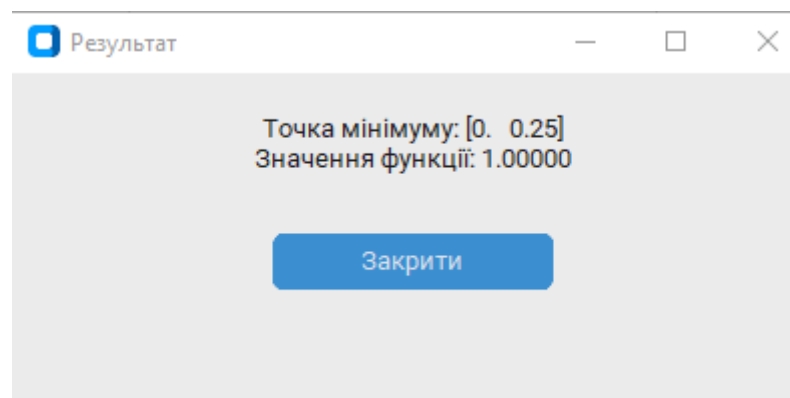


Рисунок 4.3 – Результат задачі ЛП

Закрити вікно можна стандартним способом або натисканням кнопки "Закрити".

Якщо функція необмежена в заданій області, відповідне повідомлення буде виведено в новому вікні. Приклад такого результату продемонстровано на рис. 4.4.

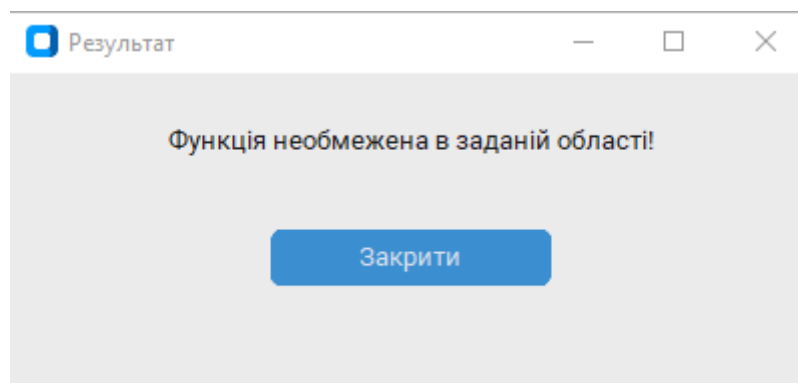


Рисунок 4.4 – Вікно з повідомленням про необмеженість функції

Якщо система обмежень не має розв'язку, відповідне повідомлення буде виведено в новому вікні. Приклад такого результату продемонстровано на рис. 4.5.

Закрити вікно можна стандартним способом або натисканням кнопки "Закрити".

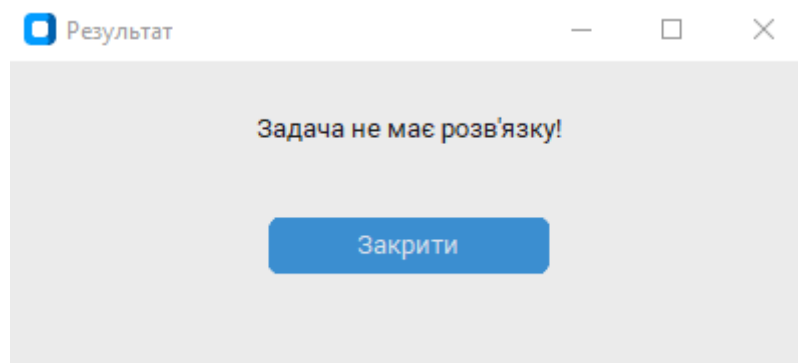


Рисунок 4.5 – Вікно з повідомленням про відсутність розв'язку

Закрити вікно можна стандартним способом або натисканням кнопки "Закрити".

Після завершення роботи користувач може закрити головне вікно програми або натиснути кнопку для вибору задачі оптимізації.

4.3 Інструкція роботи користувача з вікном "Нелінійна одновимірна оптимізація"

Після натискання в головному вікні програми кнопки "Нелінійна одновимірна оптимізація" відкривається відповідне вікно. Воно представлено на рис. 4.6.

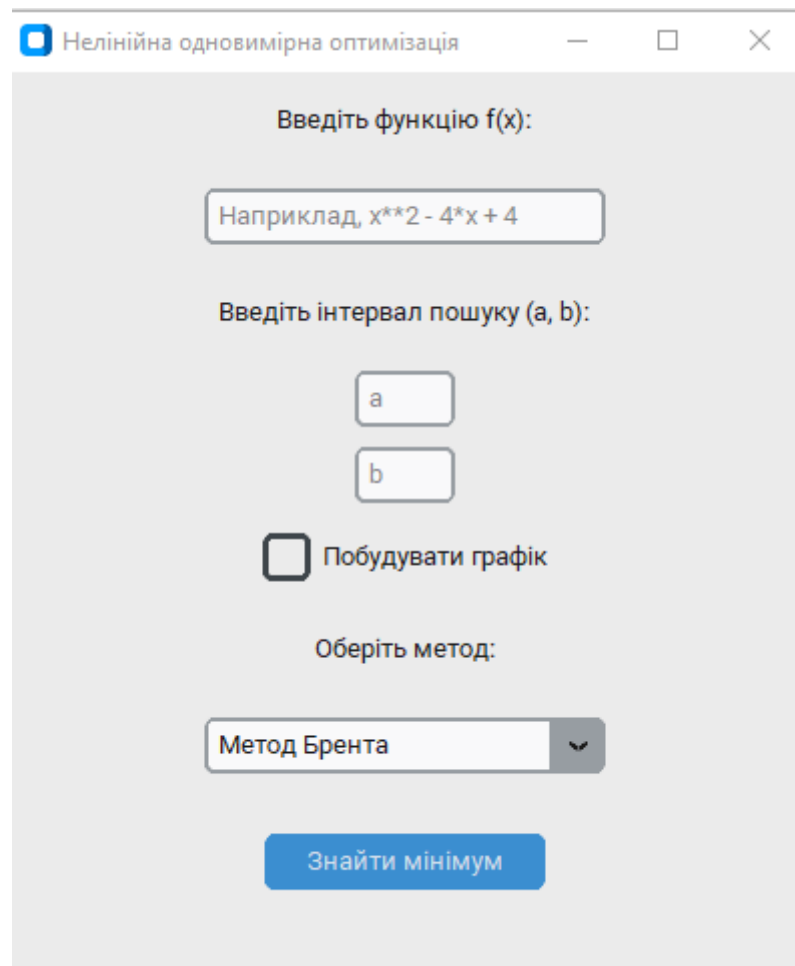


Рисунок 4.6 – Вікно "Нелінійна одновимірна оптимізація"

Для отримання результату цієї задачі оптимізації користувачу потрібно ввести вхідні дані.

В полі під текстом "Введіть функцію $f(x)$:" користувач має ввести математичну функцію однієї змінної x у вигляді рядка. Наприклад: $x**2 - 4*x + 4$.

Під текстом "Введіть інтервал пошуку (a, b) :" користувач має заповнити два поля:

- a – ліва межа інтервалу, наприклад: 0;
- b – права межа інтервалу, наприклад: 5.

Під текстом "Оберіть метод:" користувач має вибрати один із доступних методів оптимізації у спадному списку:

- метод Брента – для стандартної оптимізації;
- метод золотого перетину – для пошуку мінімуму з використанням золотого перетину;
- метод обмеженої оптимізації – для пошуку мінімуму в чітко заданому інтервалі.

Користувач може активувати чекбокс "Побудувати графік", якщо хоче побачити графік функції з позначеною точкою мінімуму.

Для запуску обчислення користувач має натиснути кнопку "Знайти мінімум".

Відбувається перевірка правильного введення даних: чи коректно введено інтервал (ліва межа a повинна бути меншою за праву b), чи вибрано метод оптимізації.

У разі некоректного введення даних в спливаючому вікні виводиться повідомлення про помилку. Приклад представлено на рис. 4.7.

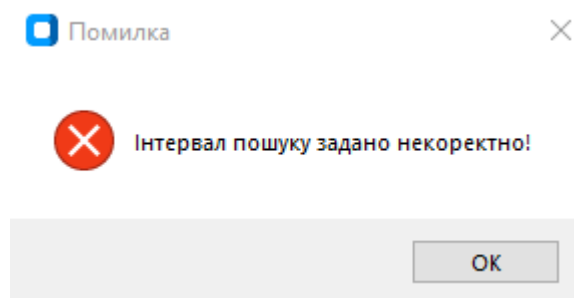


Рисунок 4.7 – Вікно з повідомленням про некоректні дані

При коректному введенні даних результат розраховується залежно від вибраного методу на основі вхідних даних.

Якщо мінімум знайдено, відкривається нове вікно з результатами. Приклад вікна результату наведено на рис. 4.8.

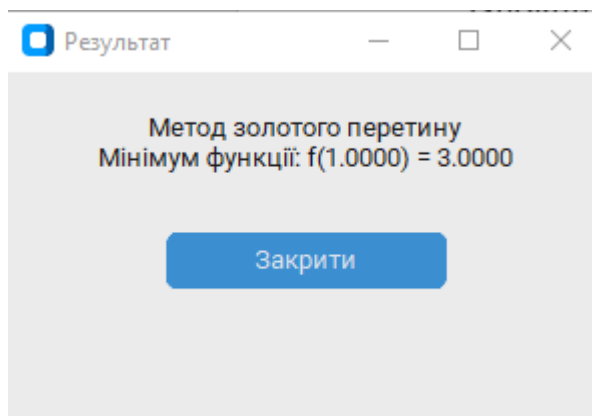


Рисунок 4.8 – Результат нелінійної одновимірної оптимізації

Якщо активовано чекбокс побудови графіка, будується графік функції на заданому інтервалі з позначенням точки мінімуму (червона точка). Приклад побудови графіка представлено на рис. 4.9.

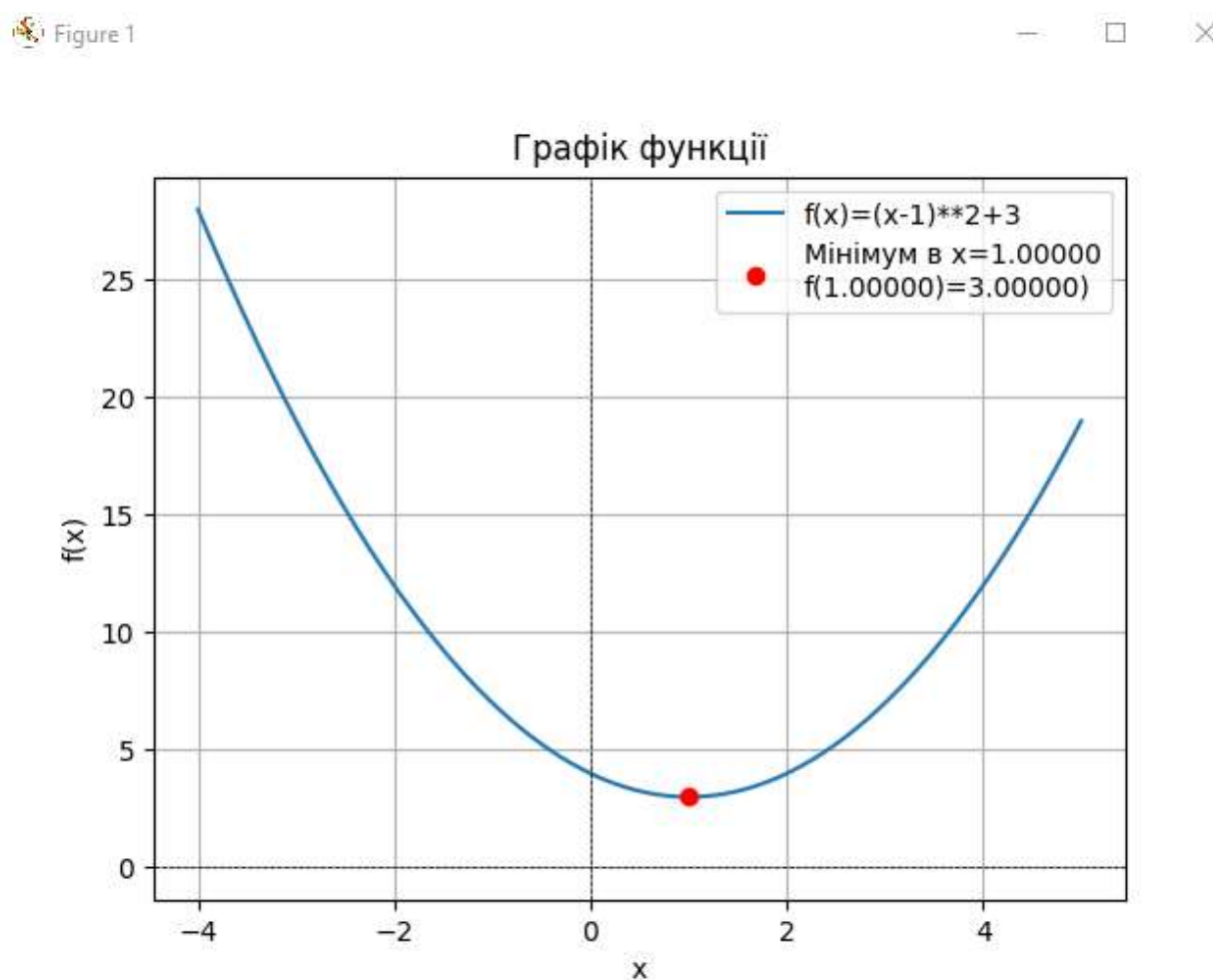


Рисунок 4.9 – Графік одновимірної функції з вказаним мінімумом

Якщо оптимізація не вдалася, відображається відповідне спливаюче повідомлення, приклад якого подано на рис. 4.10.

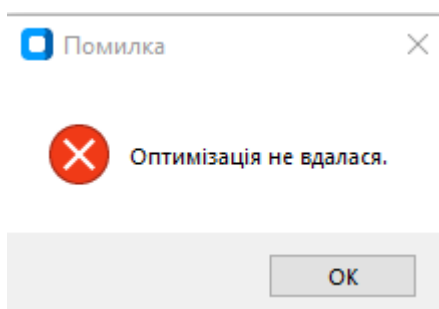


Рисунок 4.10 – Повідомлення при неможливості проведення одновимірної оптимізації

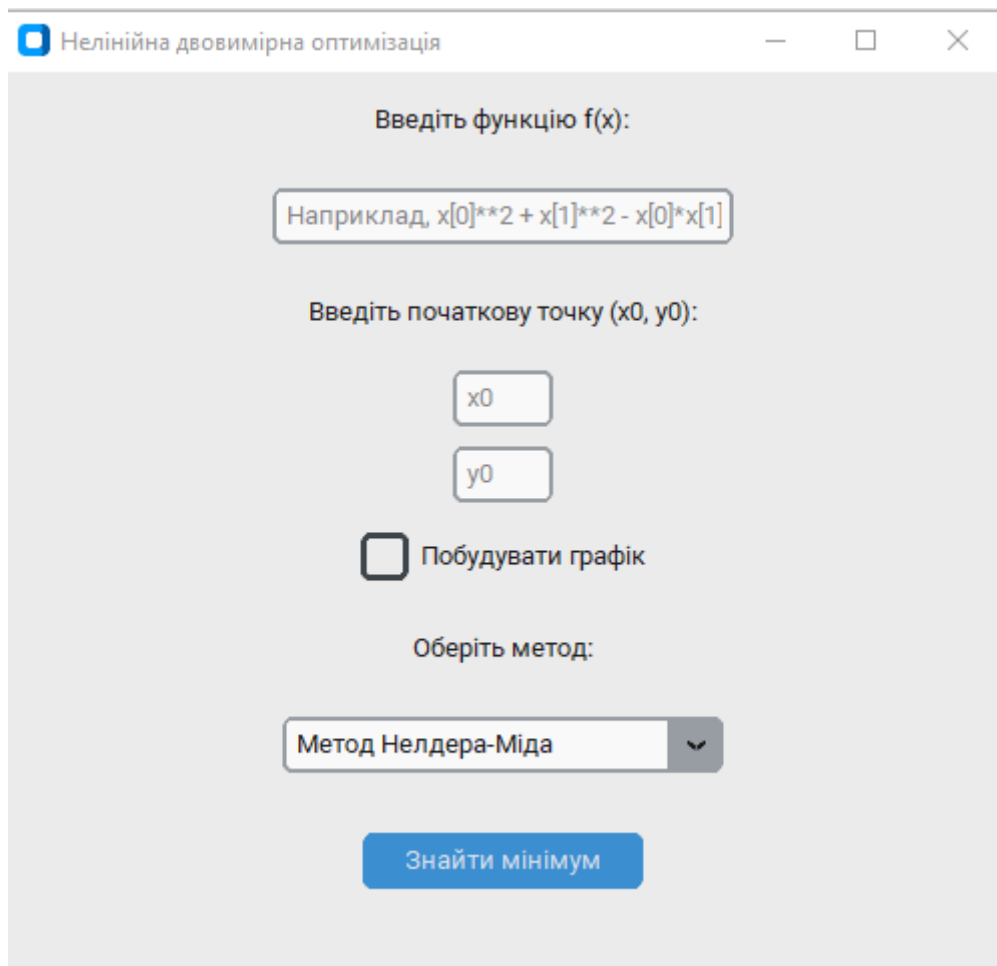
Після завершення роботи користувач може закрити результуюче вікно стандартним способом або натиснувши кнопку "Закрити".

Головне вікно залишається відкритим для повторного використання, якщо потрібно провести інші розрахунки.

4.4 Інструкція роботи користувача з вікном "Нелінійна двовимірна оптимізація"

Після натискання в головному вікні програми кнопки "Нелінійна двовимірна оптимізація" відкривається відповідне вікно. Воно представлене на рис. 4.11.

Для отримання результату цієї задачі оптимізації користувачу потрібно ввести вхідні дані.



Нелінійна двовимірна оптимізація

Введіть функцію $f(x)$:

Наприклад, $x[0]**2 + x[1]**2 - x[0]*x[1]$

Введіть початкову точку (x_0, y_0) :

x_0

y_0

☐ Побудувати графік

Оберіть метод:

Метод Нелдера-Міда

Знайти мінімум

Рисунок 4.11 – Вікно "Нелінійна двовимірна оптимізація"

У полі під текстом "Введіть функцію $f(x)$:" користувач має ввести математичну функцію двох змінних у вигляді Python-рядка. Наприклад: $x[0]**2 + x[1]**2 - x[0]*x[1]$.

Під текстом "Введіть початкову точку (x_0, y_0) :" користувач має заповнити два поля:

- x_0 : Початкове значення першої змінної, наприклад: 1;
- y_0 : Початкове значення другої змінної, наприклад: 1.

Під текстом "Оберіть метод:" користувач має вибрати один із доступних методів у комбобоксі:

- метод Нелдера-Міда;
- метод Пауелла;
- метод спряжених градієнтів;

– метод Ньютона.

Користувач може активувати чекбокс "Побудувати графік", щоб побачити тривимірний графік функції з позначеною точкою мінімуму.

Для запуску обчислення користувач має натиснути кнопку "Знайти мінімум".

Програма перевіряє правильність введення функції, початкової точки і вибраного методу, перетворює функцію з текстового формату у функцію Python.

У разі помилок (наприклад, некоректна функція) користувач отримує спливаюче повідомлення, наприклад як на рис. 4.12.

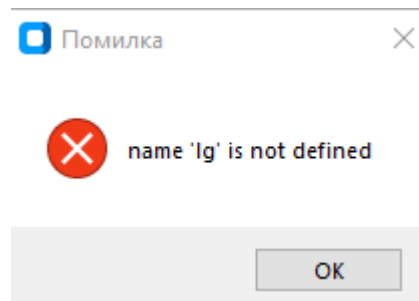


Рисунок 4.12 – Повідомлення про некоректне введення даних

Після правильного введення даних викликається функцію `scipy.optimize.minimize`, яка обчислює мінімум функції обраним методом.

Якщо оптимізація успішна, то відкривається нове вікно з результатами. Приклад вікна результату наведений на рис. 4.13.

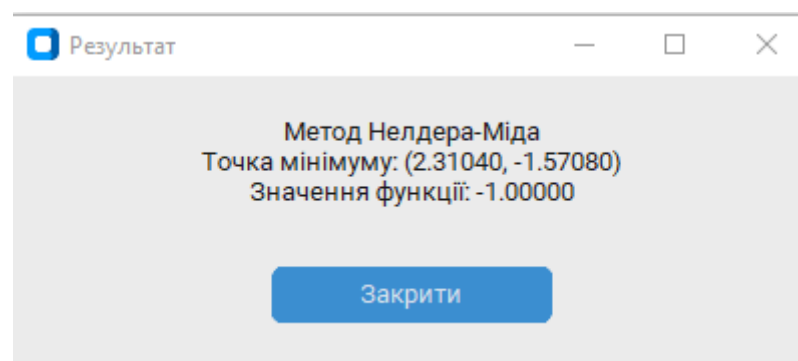


Рисунок 4.13 – Приклад вікна результату

Якщо активовано побудову графіка, то будується тривимірний графік функції, де поверхня функції відображається в 3D, а точка мінімуму виділена червоною точкою із текстовою анотацією. Приклад графіка подано на рис. 4.14.

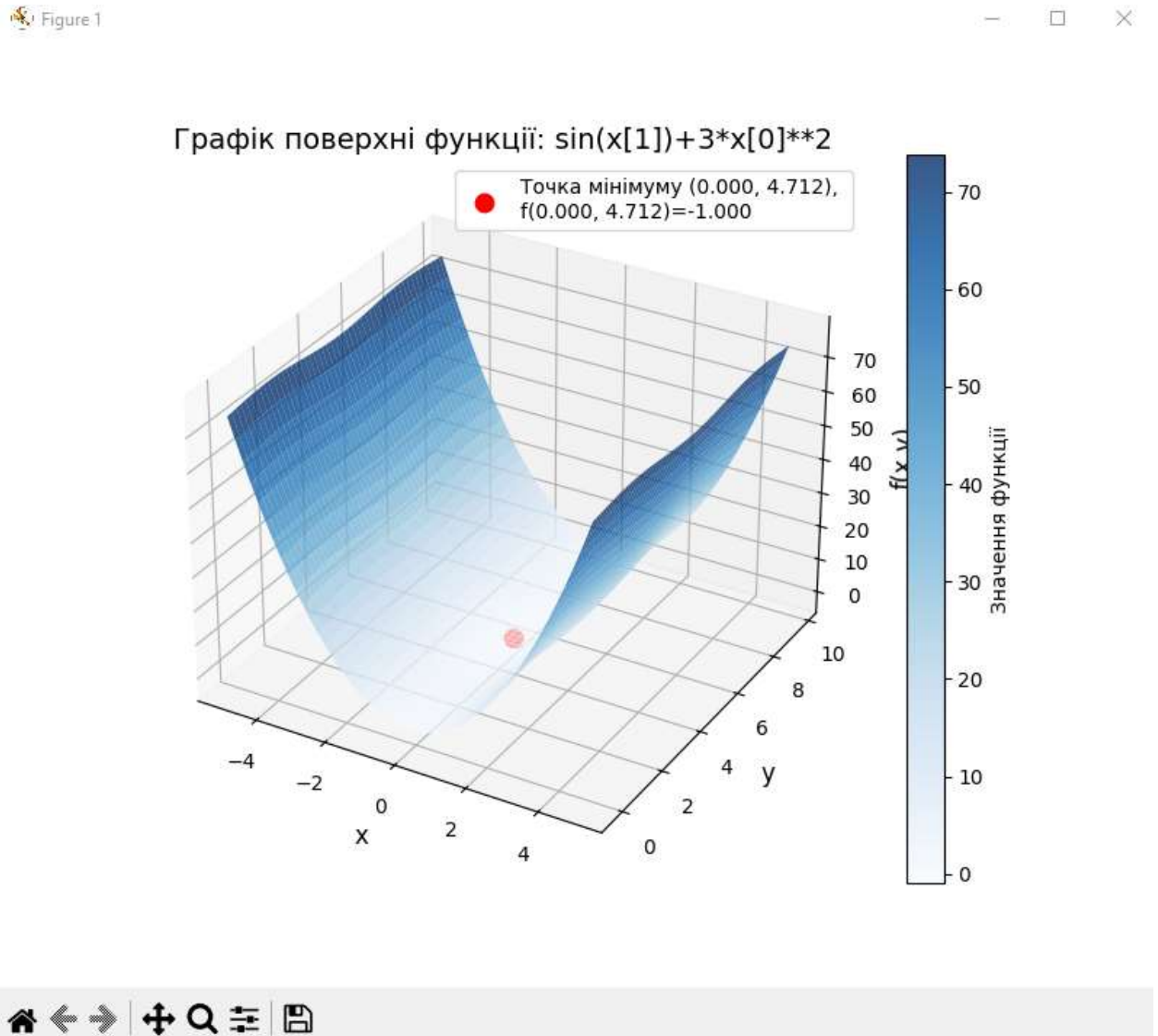


Рисунок 4.14 – Графік нелінійної функції з позначеною точкою мінімуму

Якщо оптимізація не вдалася (наприклад, через некоректну функцію або дані), користувач отримує спливаюче повідомлення. Приклад вікна з повідомленням представлено на рис. 4.15.

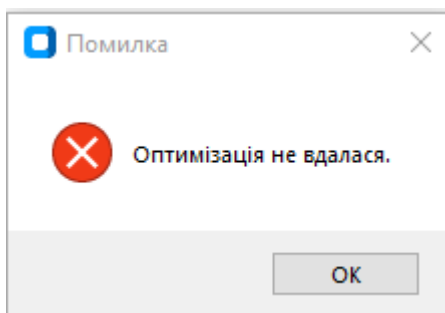


Рисунок 4.15 – Повідомлення про невдачу оптимізацію

Користувач може закрити результуюче вікно стандартним способом або натиснувши кнопку "Закрити".

Головне вікно залишається відкритим для повторного використання.

4.5 Висновок про експлуатацію і тестування програми

Програма має інтуїтивно зрозумілий графічний інтерфейс. Головне меню має чітку структуру. Привітання пояснює основне призначення програми. Три кнопки в головному меню ведуть до окремих вікон для вирішення різних задач оптимізації.

Поля введення мають приклади форматів даних.

Вибір методів оптимізації реалізовано через комбобокси, що виключає можливість неправильного введення. Можливість зручного вибору методу оптимізації дозволяє легко порівняти результати роботи кожного з них без повторного введення інших даних.

Зручний вибір додаткової опції побудови графіка реалізовано через чекбокс.

У кожному модулі реалізовано перевірку коректності введених даних (наприклад, розмірностей матриць, типів значень).

У разі помилки виводиться повідомлення з поясненням проблеми.

Оптимізація виконується за допомогою бібліотеки `scipy.optimize`, що гарантує надійність і точність математичних обчислень.

Побудова графіків реалізована через `matplotlib`, що забезпечує якісну візуалізацію.

Проект є надійним в використанні і повністю готовим до експлуатації.

ВИСНОВКИ

Під час виконання магістерської роботи були досягнуті всі поставлені цілі. Етапи і їх результати представлено далі.

Проведено дослідження теоретичних основ оптимізації. Виокремлено три основні типи задач: методи лінійного програмування, одновимірної та багатовимірної нелінійної оптимізації.

За результатами огляду мов програмування засобом розробки обрано Python завдяки поєднанню простоти, доступності, потужного інструментарію для математичних обчислень і гнучкості у створенні графічного інтерфейсу користувача. Він дозволяє зосередитися на розв'язанні задачі, мінімізуючи витрати часу на технічну реалізацію.

Вивчено можливості сучасних бібліотек Python `scipy.optimize`, `numpy`, `matplotlib`, які широко використовуються у прикладних задачах оптимізації завдяки своїй надійності та точності.

Зроблено порівняльний аналіз методів оптимізації, доступних у бібліотеці `scipy.optimize`. Виявлено, що ці методи мають різну ефективність залежно від специфіки задачі, що враховано у програмному застосунку.

Практичним результатом виконання даної роботи є якісний і оригінальний програмний продукт, який забезпечує вирішення різних типів задач оптимізації. Його модульна структура дозволяє легко розширювати функціонал, додаючи нові типи задач або методи оптимізації. Надійність роботи програмного продукту забезпечена ефективним механізмом обробки виняткових ситуацій. Застосунок має зручний графічний інтерфейс з інтерактивним введенням параметрів задачі (функцій, обмежень, початкових точок), можливістю вибору методу оптимізації через інтуїтивний комбобокс, з візуалізацією результатів через графіки.

Завдяки описаним перевагам розроблений застосунок може використовуватися у навчальних курсах для демонстрації теоретичних

методів оптимізації, а також для проведення експериментів із різними методами оптимізації в наукових дослідженнях та інших галузях.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Піддубний, О. М. Варіаційне числення та методи оптимізації : підручник / О. М. Піддубний, Ю. І. Харкевич. – Луцьк : Гадяк Ж. В., 2015. – 331 с.
2. Оптимізація [Електронний ресурс]. – Режим доступу: [https://uk.wikipedia.org/wiki/%D0%9E%D0%BF%D1%82%D0%B8%D0%BC%D1%96%D0%B7%D0%B0%D1%86%D1%96%D1%8F_\(%D0%BC%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0\)](https://uk.wikipedia.org/wiki/%D0%9E%D0%BF%D1%82%D0%B8%D0%BC%D1%96%D0%B7%D0%B0%D1%86%D1%96%D1%8F_(%D0%BC%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)).
3. Дубровін, В. І. Методи оптимізації та їх застосування в задачах навчання нейронних мереж : навчальний посібник / В. І. Дубровін, С. О. Субботін. – Запоріжжя : ЗНТУ, 2003. – 136 с.
4. Лінійне програмування : навч. посіб. для студентів вузів, що вивчають дисципліну "Математичне програмування" / Г. Г. Цегелик. - Львів: Світ, 1995. - 215 с.
5. Гончаренко Я.В. Математичне програмування / Я.В. Гончаренко. – К.: НПУ імені М.П.Драгоманова, 2010. — 184 с.
6. Яка основна мета машини опорних векторів (SVM) у задачах класифікації? [Електронний ресурс]. – Режим доступу: <https://uk.eitca.org/artificial-intelligence/eitc-ai-mlp-machine-learning-with-python/support-vector-machine/support-vector-machine-introduction-and-application/examination-review-support-vector-machine-introduction-and-application/what-is-the-main-objective-of-a-support-vector-machine-svm-in-classification-tasks/>
7. Математичні основи оптимізації телекомунікаційних систем : підручник / [М. В. Захарченко, С. М. Горохов, М. М. Балан та ін.]; під загальною ред. М. В. Захарченко – Одеса: ОНАЗ ім. О.С. Попова, 2010. – 240 с.

8. Оптимізаційні методи та моделі в підприємницькій діяльності: Навчальний посібник. / Волонтир Л.О, Потапова Н.А., Ушкаленко І.М., І.А.Чіков. – Вінниця: ВНАУ, 2020. – 404 с.
9. Оптимізаційні методи та моделі : навчальний посібник / Н. В. Буреннікова, О. В. Зелінська, І. М. Ушкаленко, Ю. Ю. Буренніков. – Вінниця : ВНТУ, 2019. – 121 с.
10. Воронков О. О. Конспект лекцій з курсу «Оптимізаційні методи і моделі» (для студентів заочної форми навчання освітньо-кваліфікаційного рівня «бакалавр» напряму підготовки 6.030504 - Економіка підприємства) / О. О. Воронков. – Харків : ХНУМГ ім. О. М. Бекетова, 2016. – 110 с.
11. Wolfram Alpha [Electronic source]. – Access mode: <https://www.wolframalpha.com>.
12. MATLAB Optimization Toolbox [Electronic source]. – Access mode: <https://www.mathworks.com/products/optimization.html>.
13. Tutorial for Optimization Toolbox [Electronic source]. – Access mode: <https://www.mathworks.com/help/optim/ug/optimization-toolbox-tutorial.html>.
14. NEOS Server: State-of-the-Art Solvers for Numerical Optimization [Electronic source]. – Access mode: <https://neos-server.org/neos/>.
15. Gurobi 12.0 Documentation [Electronic source]. – Access mode: <https://docs.gurobi.com/current/>.
16. C++ у високопродуктивному та системному програмуванні [Електронний ресурс]. – Режим доступу: <https://itproger.com/ua/news/c-v-visokoproizvoditelnom-i-sistemnom-programmirovanii>.
17. 10 переваг мови C++, яка і досі залишається актуальною [Електронний ресурс]. – Режим доступу: <https://dou.ua/forums/topic/42888/>.
18. Переваги та недоліки мови програмування C/C++ [Електронний ресурс]. – Режим доступу: <https://rozumni-ideyi.com/posts/view/perevahy-ta-nedoliky-movy-prohramuvannya-cc-vybir-efektyvnykh-instrumentiv-dlya-rozrobky/>.

19. MATLAB [Electronic source]. – Access mode: <https://www.mathworks.com/help/matlab/>.
20. Що таке мова Python та які сфери її застосування [Електронний ресурс]. – Режим доступу: <https://university.sigma.software/what-is-python/>.
21. Мова програмування Python та її застосування в різних галузях [Електронний ресурс]. – Режим доступу: <https://foxminded.ua/devykorystovuietsia-python/>.
22. Популярність мови програмування Python: причини та переваги [Електронний ресурс]. – Режим доступу: <https://buki.com.ua/blogs/populiarnist-movi-programuvannia-python-pricini-ta-perevagi/>.
23. Порівняння C++ з іншими мовами програмування [Електронний ресурс]. – Режим доступу: <https://nikvesti.com/news/business/porivnannia-c-plus-plus-z-inshymi-movami-programuvannia-ua>.
24. OR-Tools [Electronic source]. – Access mode: <https://developers.google.com/optimization>.
25. Що таке IDE та навіщо воно потрібно? [Електронний ресурс]. – Режим доступу: <https://university.sigma.software/best-python-ide-and-code-editors/>.
26. Кращі IDE для Python в 2023 році [Електронний ресурс]. – Режим доступу: <https://mate.academy/blog/python/ide-for-python-2023/>.
27. PyCharm: Ідеальне середовище для Python-розробки та її можливості [Електронний ресурс]. – Режим доступу: <https://pingvin.pro/blogy/software-blogy/pycharm-idealne-seredovyshhe-dlya-python-rozrobky-ta-yiyi-mozhlyvosti.html>.
28. Optimization and root finding [Electronic source]. – Access mode: <https://docs.scipy.org/doc/scipy/reference/optimize.html>.
29. Using Matplotlib [Electronic source]. – Access mode: <https://matplotlib.org/stable/users/index.html>.

Додаток А
Текст програми

A.1 Модуль main_window

```

import customtkinter as ctk
import min_fx
import min_fxy
import lin_prog

# Головне вікно
def main_window():
    main = ctk.CTk()
    main.title("Задачі оптимізації")
    main.geometry("300x300+300+200")
    # Привітання
    greeting_label = ctk.CTkLabel(main, text="Програма демонструє\n
розв'язування задач оптимізації\n\n\n\nОберіть задачу:")
    greeting_label.pack(pady=10)
    # Кнопка "Функція ЛП"
    function1_button = ctk.CTkButton(main, text="Задача лінійного
програмування", command=lin_prog.linear_programming_window)
    function1_button.pack(pady=10)
    # Кнопка "Функція 1"
    function1_button = ctk.CTkButton(main, text="Нелінійна одновимірна
оптимізація", command=min_fx.optimization_window1)
    function1_button.pack(pady=10)
    # Кнопка "Функція 2"
    function2_button = ctk.CTkButton(main, text="Нелінійна двовимірна
оптимізація", command=min_fxy.optimization_window2)
    function2_button.pack(pady=10)
    main.mainloop()
# Запуск програми

```

```

if __name__ == "__main__":
    ctk.set_appearance_mode("System") # Налаштування теми
    ctk.set_default_color_theme("blue") # Налаштування кольору
    main_window()

```

A.2 Модуль lin_prog

```

import customtkinter as ctk
import numpy as np
from scipy.optimize import linprog
from tkinter import messagebox

def linear_programming_window():
    # Головне вікно
    window = ctk.CTk()
    window.title("Задача лінійного програмування")
    window.geometry("600x600")
    # Поле для введення функції
    label_func = ctk.CTkLabel(window, text="Введіть вхідні дані  
задачі\n\nЗнайти мінімум функції f(x):")
    label_func.pack(pady=10)
    function_entry = ctk.CTkEntry(window, placeholder_text="Наприклад,  
[1, 2, 3]", width=160)
    function_entry.pack(pady=10)

    # Область нерівностей
    label_constraints_ineq = ctk.CTkLabel(window, text="в області, заданій  
системою нерівностей  $A1 * x \leq b1$ :")
    label_constraints_ineq.pack(pady=10)

```

```

label_A1 = ctk.CTkLabel(window, text="A1 =")
label_A1.pack(pady=5)
A1_entry = ctk.CTkEntry(window, placeholder_text="Наприклад, [[1, -
1], [2, 1]]", width=160)
A1_entry.pack(pady=5)
label_b1 = ctk.CTkLabel(window, text="b1 =")
label_b1.pack(pady=5)
b1_entry = ctk.CTkEntry(window, placeholder_text="Наприклад, [1,
4]")
b1_entry.pack(pady=5)
# Область рівностей
label_constraints_eq = ctk.CTkLabel(window, text="та рівнянь  $A2 * x =$ 
b2 (не обов'язково):")
label_constraints_eq.pack(pady=10)

label_A2 = ctk.CTkLabel(window, text="A2 =")
label_A2.pack(pady=5)
A2_entry = ctk.CTkEntry(window, placeholder_text="Наприклад, [[1,
1]])
A2_entry.pack(pady=5)

label_b2 = ctk.CTkLabel(window, text="b2 =")
label_b2.pack(pady=5)
b2_entry = ctk.CTkEntry(window, placeholder_text="Наприклад, [2]")
b2_entry.pack(pady=5)
# Функція для обчислення мінімуму
def find_minimum():
    try:
        # Зчитування вхідних даних
        func = np.array(eval(function_entry.get()))

```

```

A1 = np.array(eval(A1_entry.get())) if A1_entry.get() else None
b1 = np.array(eval(b1_entry.get())) if b1_entry.get() else None
A2 = np.array(eval(A2_entry.get())) if A2_entry.get() else None
b2 = np.array(eval(b2_entry.get())) if b2_entry.get() else None
# Перевірка розмірності
if A1 is not None and b1 is not None and A1.shape[0] != len(b1):
    raise ValueError("Розмірності A1 та b1 не збігаються!")
if A2 is not None and b2 is not None and A2.shape[0] != len(b2):
    raise ValueError("Розмірності A2 та b2 не збігаються!")

# Виклик scipy.optimize.linprog
result = linprog(c=func, A_ub=A1, b_ub=b1, A_eq=A2, b_eq=b2,
method="highs")
# Обробка результату
if result.success:
    result_message = (f"Точка мінімуму: {result.x}\n"
                      f"Значення функції: {result.fun:.5f}")
elif result.status == 3: # Unbounded
    result_message = "Функція необмежена в заданій області!"
else: # Infeasible
    result_message = "Задача не має розв'язку!"
# Виведення результату
result_window("Результат", result_message)
except Exception as e:
    messagebox.showerror("Помилка", str(e))
# Кнопка для запуску обчислення
calculate_button = ctk.CTkButton(window, text="Знайти мінімум",
command=find_minimum)
calculate_button.pack(pady=20)
window.mainloop()

```

```
# Вікно для відображення результату
def result_window(title, message):
    result = ctk.CTkToplevel()
    result.title(title)
    result.geometry("400x200")
    result_label = ctk.CTkLabel(result, text=message, wraplength=380)
    result_label.pack(pady=20)
    close_button = ctk.CTkButton(result, text="Закрити",
command=result.destroy)
    close_button.pack(pady=10)
```

Додаток Б
Слайди презентації



Рисунок Б.1 – Слайд 1

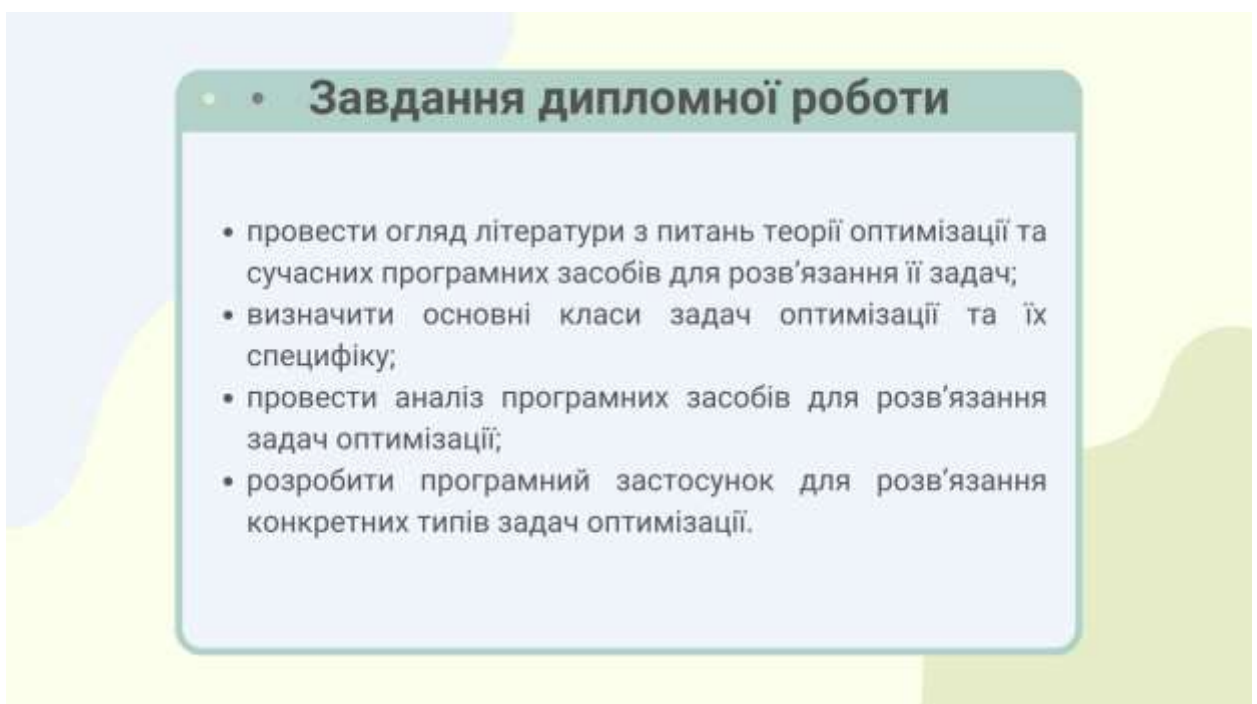


Рисунок Б.2 – Слайд 2

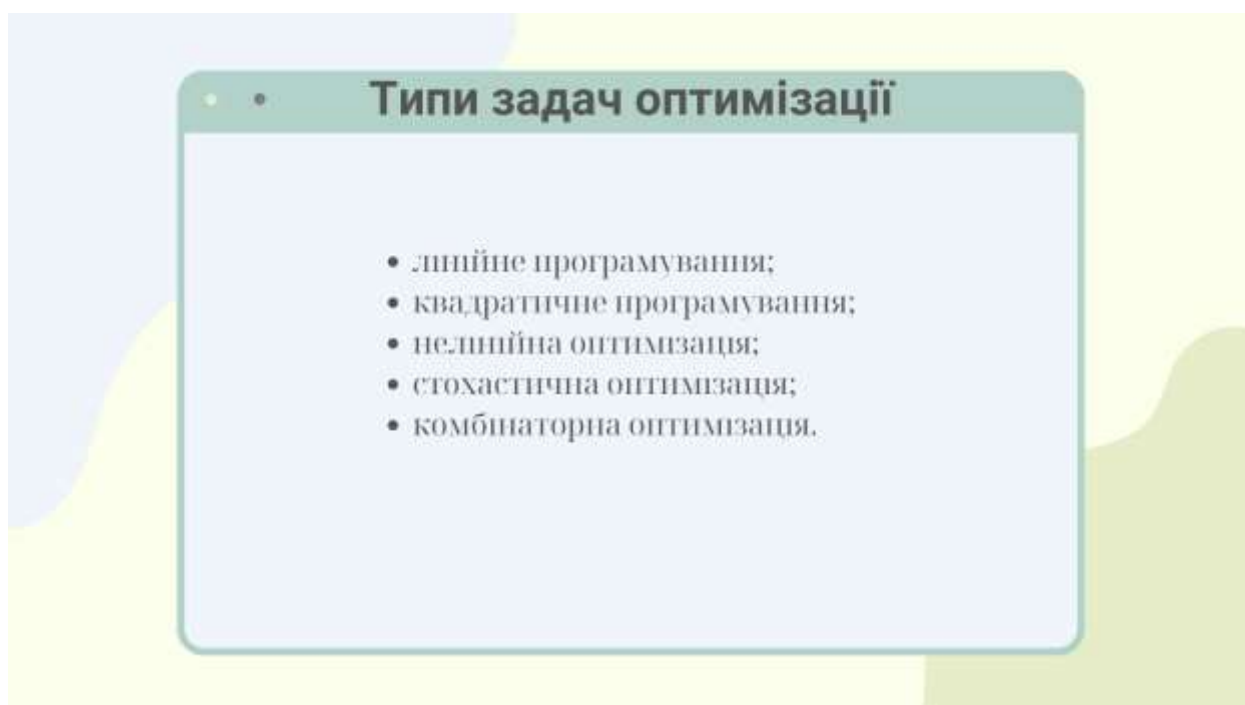


Рисунок Б.3 – Слайд 3



Рисунок Б.4 – Слайд 4

Вибір мови програмування				
Критерій	C++	Python	Google OR-Tools	MATLAB
Призначення	Високопродуктивні програми, що потребують точного контролю над ресурсами	Швидка розробка, аналіз даних, наукові та освітні проекти	Оптимізація, комбінаторні та лнійні задачі	Аналіз даних, обчислення, моделювання, створення прототипів
Синтаксис	Складний, потребує аналізу точного управління пам'яттю	Простий, зручний, орієнтований на швидку розробку	Інтеграція з Python і C++ для оптимізації	Інтуїтивно зрозумілий, спеціалізований для математичних розрахунків
Швидкість виконання	Дуже висока, підходить для задач реального часу	Помірна, залежить від завантажених бібліотек	Висока, якщо використовується з C++	Висока для чисельних розрахунків, повільніша для графічних задач
Назви бібліотек для оптимізації	Eigen, CGAL, Gurobi, Boost	SciPy, NumPy, PuLP, TensorFlow	Вбудовані функції для оптимізації, легко інтегрується з іншими бібліотекими	Вбудовані пакети для лінійного, нелінійного програмування та задач динамічного програмування
Можливість створення GUI	Можливо через Qt, GTK, або складна в реалізації	Легко реалізується через Tkinter, PyQt	GUI не підтримується, потребує інтеграції з іншими інструментами	Інтегровані засоби розробки GUI (GUIDE)
Вартість використання	Безкоштовний (переважно), можливі платні бібліотеки	Безкоштовний, іноді потрібні платні бібліотеки	Безкоштовний, доступний з відкритим кодом	Дорого ліцензії, доступні академічні ліцензії

Рисунок Б.5 – Слайд 5



Рисунок Б.6 – Слайд 6

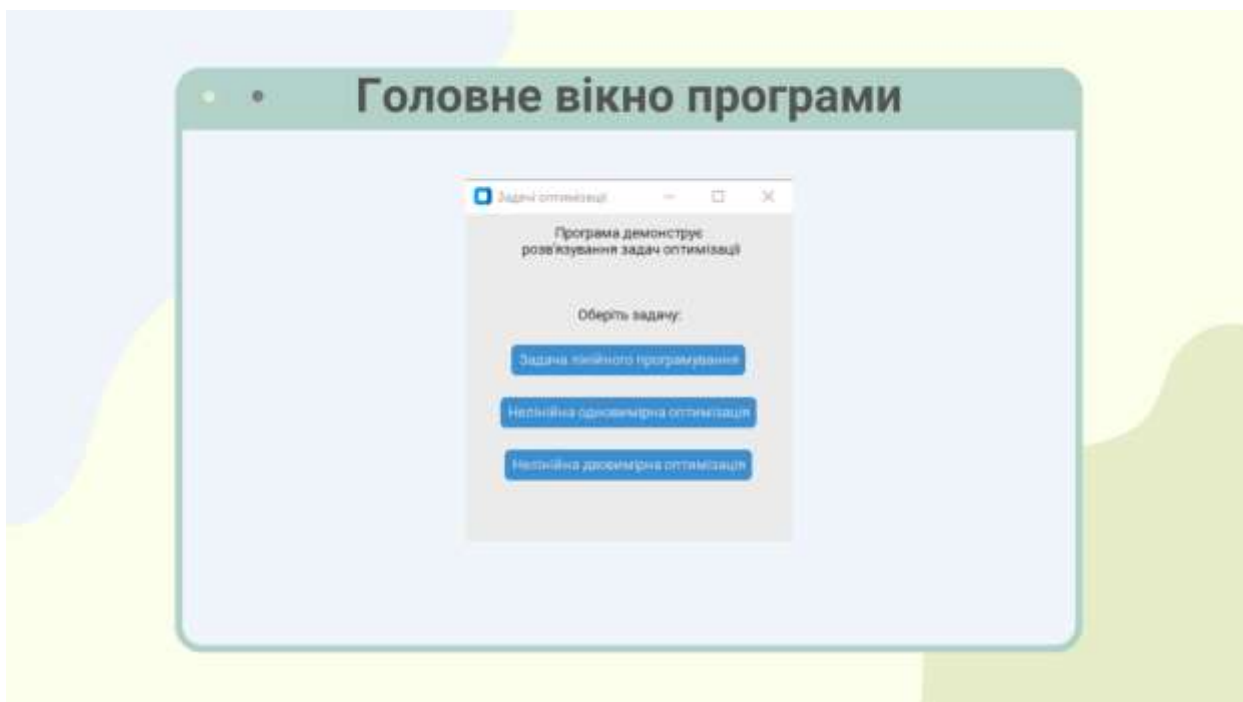


Рисунок Б.7 – Слайд 7

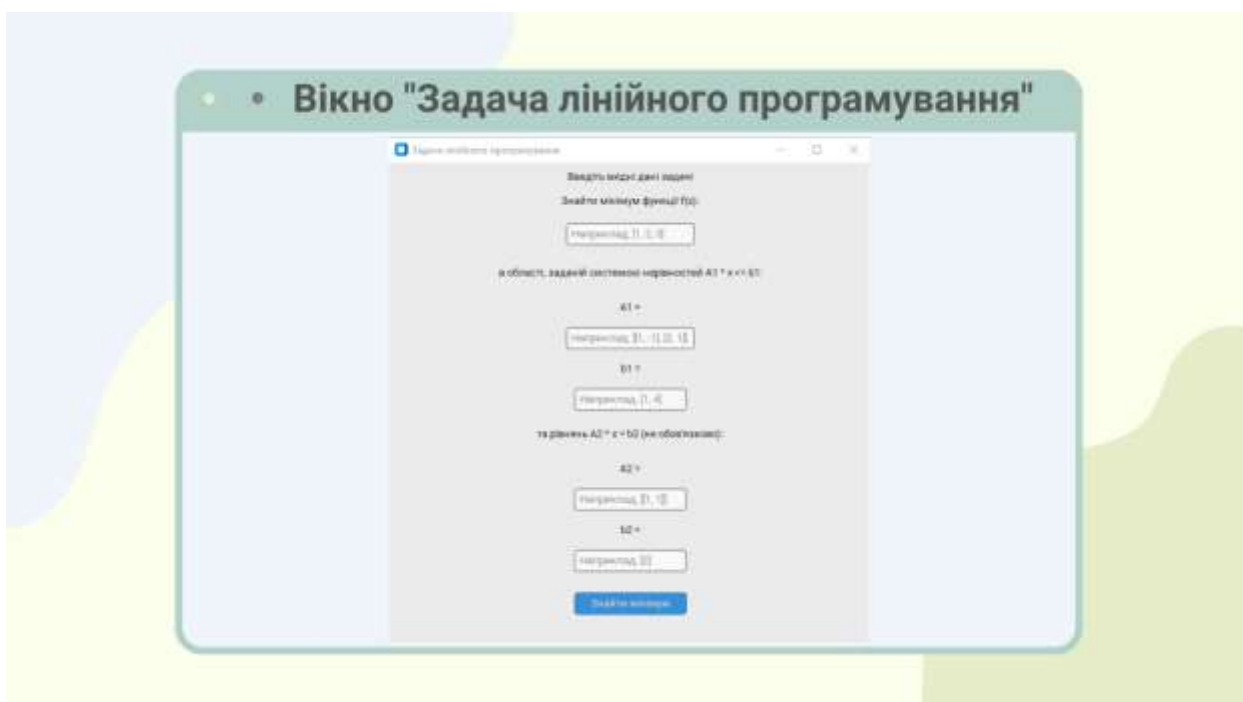


Рисунок Б.8 – Слайд 8

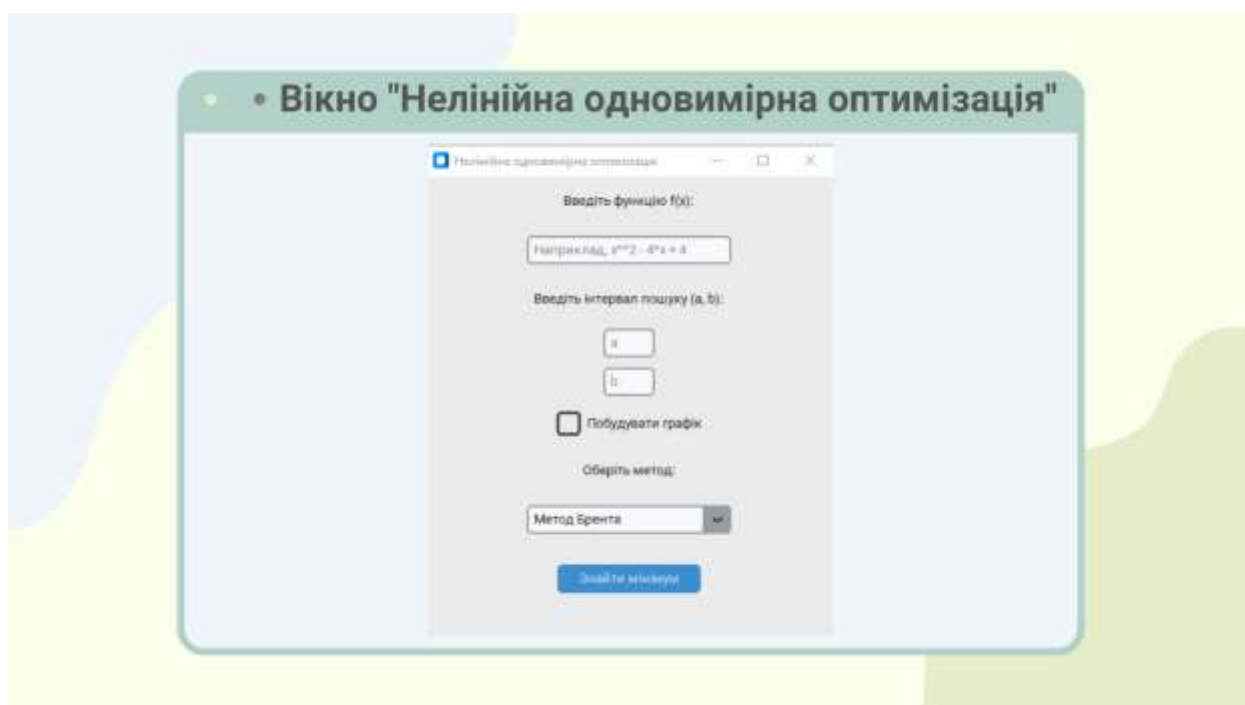


Рисунок Б.9 – Слайд 9



Рисунок Б.10 – Слайд 10

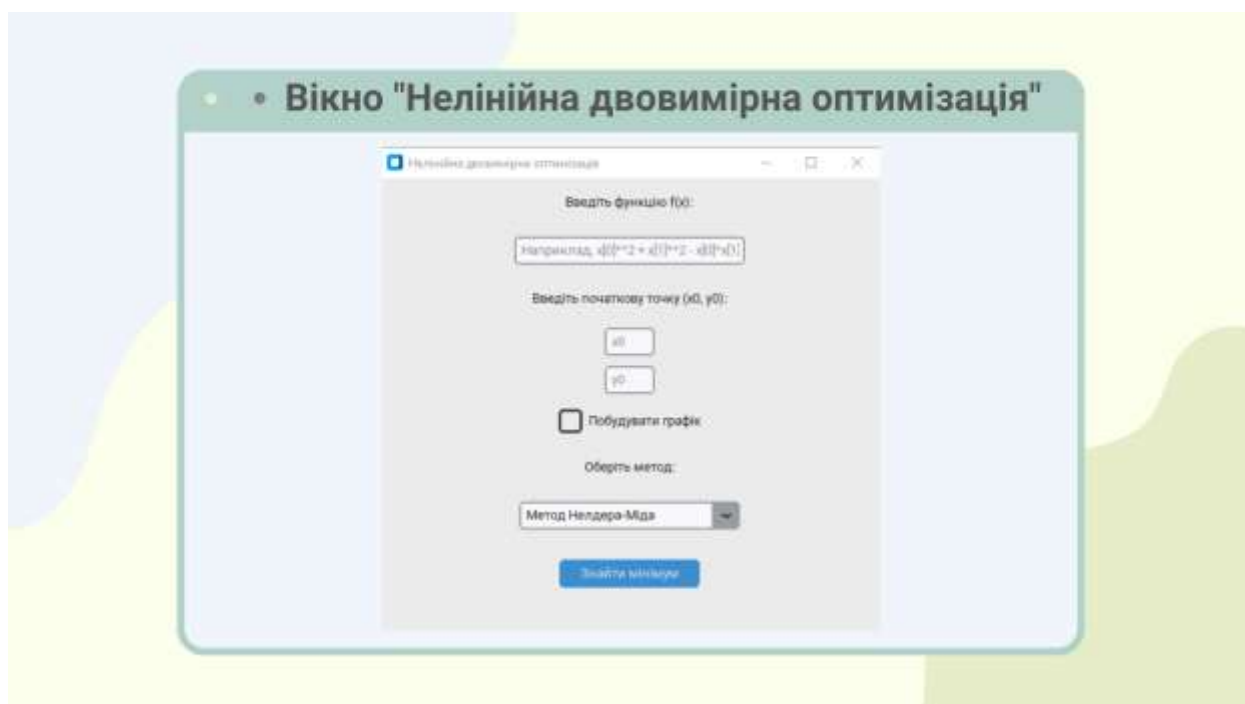


Рисунок Б.11 – Слайд 11



Рисунок Б.12 – Слайд 12

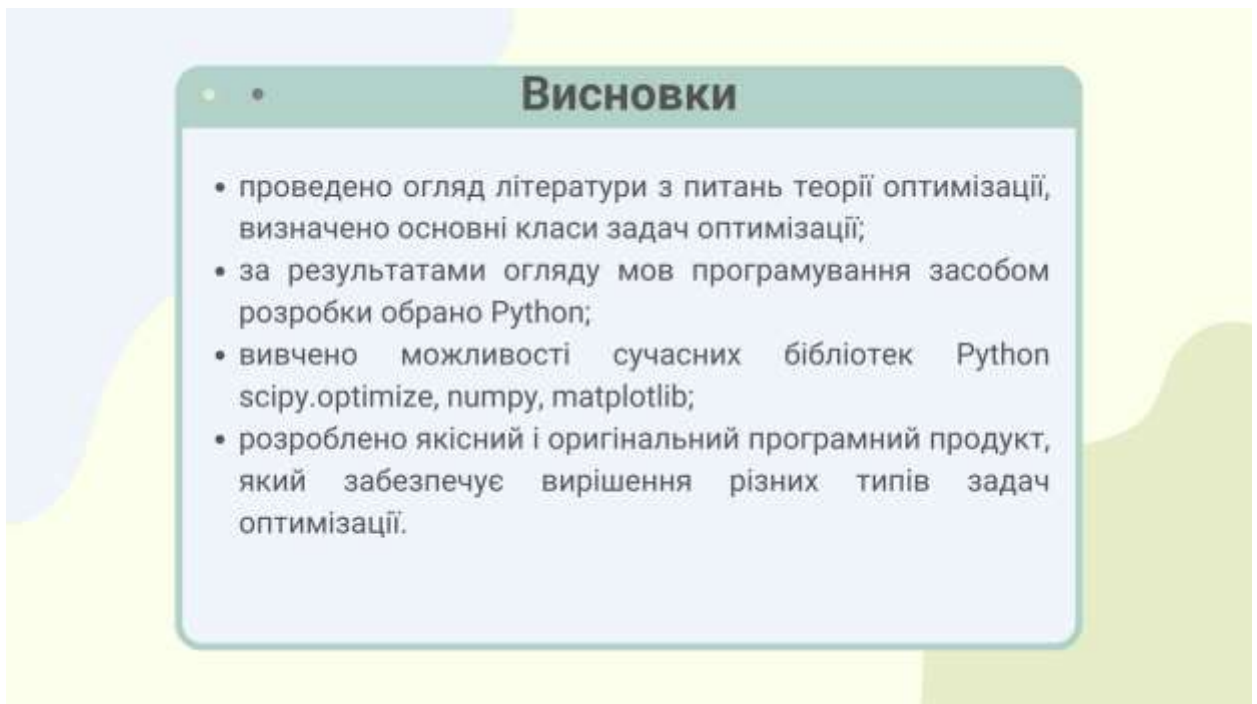


Рисунок Б.13 – Слайд 13

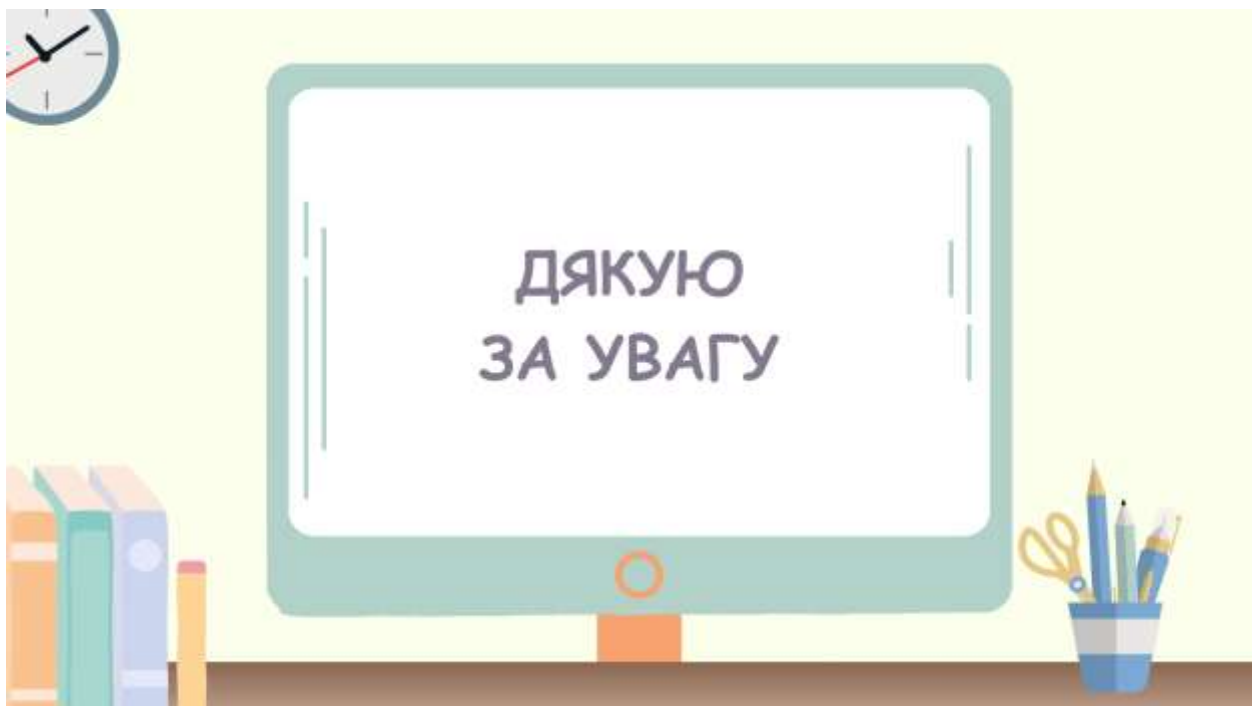


Рисунок Б.14 – Слайд 14