НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

імені ІГОРЯ СІКОРСЬКОГО”

Кафедра прикладної математики

РОЗРАХУНКОВО-ГРАФІЧНА РОБОТА

з кредитного модуля

"Програмування 1. Основи програмування"

на тему:

«Програма обчислення мінорів заданої матриці»

Виконав Фірман Дмитро Богданович

група КМ-02 факультет прикладної математики

N залікової книжки КМ-0220

Керівник Олефір О.С. ( )

"12" грудня 2020р.

Захищена з оцінкою\_\_\_\_\_\_\_\_\_\_\_

Київ 2020

ЗМІСТ

[1. ПОСТАНОВКА ЗАДАЧІ 3](#_Toc58777457)

[2. ПРОЕКТУВАННЯ ПРОГРАМНИХ ЗАСОБІВ 5](#_Toc58777458)

[2.1 Методи вирішення задач 5](#_Toc58777459)

[2.2 Проектування алгоритмів 7](#_Toc58777460)

[2.3 Структура програмного забезпечення 9](#_Toc58777461)

[2.4 Опис розроблених алгоритмів 11](#_Toc58777462)

[2.5 Засоби керування програмами 12](#_Toc58777463)

[3. РЕАЛІЗАЦІЯ ПРОГРАМИ 15](#_Toc58777464)

[3.1 Опис вхідних даних 15](#_Toc58777465)

[3.2 Опис результатів 16](#_Toc58777466)

[3.3 Опис контрольних прикладів 16](#_Toc58777467)

[3.4 Експериментальні розрахунки 18](#_Toc58777468)

[ВИСНОВКИ 20](#_Toc58777469)

[ЛІТЕРАТУРА 21](#_Toc58777470)

[ДОДАТОК А 22](#_Toc58777471)

[ДОДАТОК Б 24](#_Toc58777472)

[ДОДАТОК В 32](#_Toc58777473)

1. ПОСТАНОВКА ЗАДАЧІ

Мінор матриці – поняття досить інтуїтивно зрозуміле. Вони застосовуються здебільшого як проміжні елементи на шляху. Для прикладу, без пошуку мінора не обійтись у випадку, коли потрібно отримати алгебраїчне доповнення елементу матриці. Іншим прикладом є використання мінорів для визначення рангу матриці, що є важливим кроком при розв’язуванні систем

Та якщо із пошуком мінорів невеликих порядків проблем не виникає, то для прикладу, уже мінор 5-го порядку знайти вручну стає складно, не кажучи вже про випадки, коли потрібно знайти не один, а усі можливі мінори. Написання програми, котра б могла впоратись із цим завданням, значно підвищує швидкість виконання задач схожого типу. Зі згаданих вище причин, вона може бути застосованою у програмах з іншим функціоналом як проміжна частина.

В загальному, мету роботи можна розділити на декілька підпунктів.

* Вивчення математичної складової задачі;
* Отримання базових знань мови програмування Python, необхідних для реалізації завдання;
* Створення схеми дії програми та її взаємодії з користувачем;
* Написання допоміжних функцій, що будуть потрібними для виконання програмою поставленої задачі;
* Забезпечення легкості та зрозумілості використання програми, щоб скористатись нею міг навіть користувач, не знайомий із програмуванням;
* Передбачення коректного вводу та виводу даних з метою отримання чітких та правильних результатів та недопуску помилок в ході виконання програми;
* Створення ескізів інтерфейсу програми (за можливості – його реалізація)

Для використання програми необхідною та достатньою умовою є встановлення на робочому девайсі користувача мови програмування Python (версії 3.8+) та бібліотек NumPy та Itertools, котрі використовуються у програмі для полегшення проміжних завдань та покращення візуального сприйняття виведених даних.

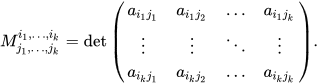
1. ПРОЕКТУВАННЯ ПРОГРАМНИХ ЗАСОБІВ
2. Методи вирішення задач

**Визна́чник** або **детерміна́нт** — це число; вираз складений за певним законом з n² елементів квадратної матриці. Одна з найважливіших характеристик квадратних матриць.

Для квадратної матриці розміру визна́чник є многочленом степеня *n* від елементів матриці, і є сумою добутків елементів матриці зі всіма можливими комбінаціями різних номерів рядків і стовпців (в кожному із добутків є рівно по одному елементу з кожного рядка і кожного стовпця). Кожному добутку приписується знак плюс чи мінус, в залежності від парності перестановки номерів.

Маючи означення визначника, можна перейти до розбору **мінорів**.

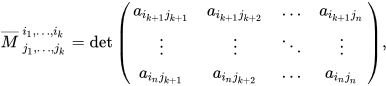
*Визначник* матриці, яка одержується з даної викреслюванням всіх рядків та стовпців, окрім вибраних, називається **мінором** *k*-го порядку, розташованим в рядках з номерами , ,..та стовпцях з номерами ,,… .



Якщо номери зазначених рядків збігаються з номерами зазначених стовпців, то мінор називається головним, а якщо відзначені перші *k* рядків і перші *k* стовпців - кутовим або провідним головним.

**Мінором** **елемента**  [квадратної матриці](https://uk.wikipedia.org/wiki/%D0%9A%D0%B2%D0%B0%D0%B4%D1%80%D0%B0%D1%82%D0%BD%D0%B0_%D0%BC%D0%B0%D1%82%D1%80%D0%B8%D1%86%D1%8F) *A* порядку *n* називається визначник (*n-1*) порядку, який одержуємо з визначника *n*-го порядку шляхом викреслювання *і*-го рядка та *j*-го стовпця, на перетині яких знаходиться елемент .

Визначник матриці, яка одержується викреслюванням тільки вибраних рядків та стовпців з матриці *A* у випадку, коли отримана матриця буде квадратною, називається **доповнювальним мінором** до мінору :



де <…< та <…< — номери не вибраних рядків і стовпців.

Нехай — деякий мінор порядку матриці *A*. Мінор порядку *k+1* матриці називається оточуючим для мінора , якщо його матриця містить в собі матрицю мінору . Таким чином, оточуючий мінор для мінора можна одержати дописуючи до нього один рядок і один стовпчик.

**Базисним мінором** ненульової матриці *A* (існує ненульовий елемент) називається мінор, який не дорівнює нулю, а всі його оточуючі мінори дорівнюють нулю, або їх не існує.

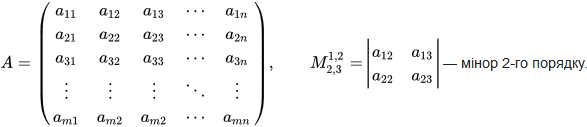
***Доведення існування базисного мінора***: утворимо мінор з єдиного ненульового елемента і будемо рекурсивно шукати ненульові оточуючі мінори аж до найбільшого.

**Зауваження.** В загальному випадку в матриці може існувати багато базисних мінорів.

Розмір базисного мінора матриці називається [**рангом матриці**](https://uk.wikipedia.org/wiki/%D0%A0%D0%B0%D0%BD%D0%B3_%D0%BC%D0%B0%D1%82%D1%80%D0%B8%D1%86%D1%96)**.**

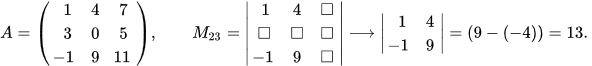
**Приклади:**

Розглянемо матрицю *A* розміру :



Таких мінорів можна скласти штук.

Мінор квадратної матриці *A* — визначник матриці, отриманий шляхом викреслювання рядка 2 та стовпчика 3:



1. Проектування алгоритмів

У ході розробки алгоритму роботи програми виникали нові завдання, вирішення яких варто було забезпечити у ході розробки програми. Далі наведені ці проблеми та рішення, прийняті для їх розв’язання:

1. Подача матриці у програмі у вигляді, зручному для використання та зрозумілому при виведенні. Цю проблему допомагає вирішити модуль NumPy, у котрому передбачені методи роботи з двохвимірними масивами, котрими і є матриці.
2. Забезпечення пошуку основної математичної основи для визначення мінору – визначника певної квадратної матриці довільної розмірності ( для більшої універсальності програми) без використання готових модулів. З цією метою була розроблена функція пошуку визначника з використанням циклів та методу перестановок із модуля Itertools.
3. Доступність пошуку як мінорів як прямокутної (утворених викреслюванням певної кількості рядків та стовпців), так і квадратної (утворених викреслюванням рядка та стовпця, що містять певни елемент) матриць. Для цього було створено дві підпрограми, котрі для зручності було подано через функції.
4. Перевірка введених даних. Для вирішення даної проблеми використовувались різні методи, такі як цикли while та допоміжні функції, котрих, через певні контекстуальні особливості, було створено 2. Про відмінність між ними буде вказано у подальшому описі програми.
5. Забезпечення легкості взаємодії. З цією метою для кожного вводу користувача забезпечено повідомлення, яким має бути ввід для того чи іншого кроку та/або додано опис помилковості кроку користувача таким чином, щоб користувач міг виправити помилку на тому ж кроці, не починаючи виконання програми заново.
6. Зручність використання. Для вирішення цього питання було застосовано цикли while, котрі дозволяють використовувати програму для різних елементів даної матриці без повторного введення її вмісту.
7. Структура програмного забезпечення

Виконання загальної програми починається із підпрограми «Початок виконання». У ній виводиться інформація про виконавця, період створення програми та її призначення.

Наступна підпрограма, «Обрання виду мінору», дає можливість користувачу обрати потрібний йому мінор (матричний чи мінор елемента) . Після неї можливі два шляхи подальшого ходу виконання програми.

Спільною для них обох є підпрограма «Введення розмірності матриці», з тією відмінністю, що на першому вводиться одне значення (оскільки це – випадок квадратної матриці), у другому – два (кількість рядків та стовпчиків відповідно).

Наступна підпрограма – «Наповнення матриці»- теж є необхідною для обох варіантів ходу виконання програми. У ній користувач вводить елементи матриці відповідно до їх індексу(при цьому забезпечується перевірка, щоб введені дані були числами).

Подальший хід підпрограм відрізняється.

У першому випадку, настає підпрограма «Створення матриці мінорів». На ній програма створює нульову матрицю того ж розміру, у якій елементи з певними індексами при виконанні наступної підпрограми «Наповнення матриці мінорів» замінюються на мінори елементів з тими ж індексами із початкової матриці. У пункті «Виведення результату в обраному форматі» юзер обирає вид виводу – як матрицю мінорів чи мінор елемента і програма подає відповідний результат.

У другому варіанті, користувач у підпрограмі «Обрання порядку мінору» вводить з клавіатури порядок шуканого мінору (програма при цьому перевіряє, щоб введене значення було доступним). На основі результату її виконання відбувається втілення пункту «Вибір потрібних рядків та стовпців», на якому користувач обирає рядки та стовпці, що входять до мінору (програма контролює, щоб не було повторів та кількість була рівною порядку мінора). На етапі «Отримання матриці на перетині потрібних рядків» програма заповнює нову матрицю елементами з індексами, котрі обираються із відсортованих списків, отриманих у попередньому пункті (оскільки для визначника положення елементів є важливим). Шуканий мінор отримується як визначник цієї матриці у підпрограмі «Пошук потрібного визначника». На етапі «Виведення результату» цей детермінант виводиться як наслідок дії програми.

Наступні підпрограми для обох варіантів мінорів однакові. Підпункт «Пропозиція про повторне виконання» дає змогу користувачу скористатись програмою для:

* тієї ж матриці;
* матриці тієї ж розмірності, але з іншими елементами;
* іншого типу мінору.

У разі відмови, користувач потрапляє на пункт «Завершення роботи», котрий закінчує виконання програми.

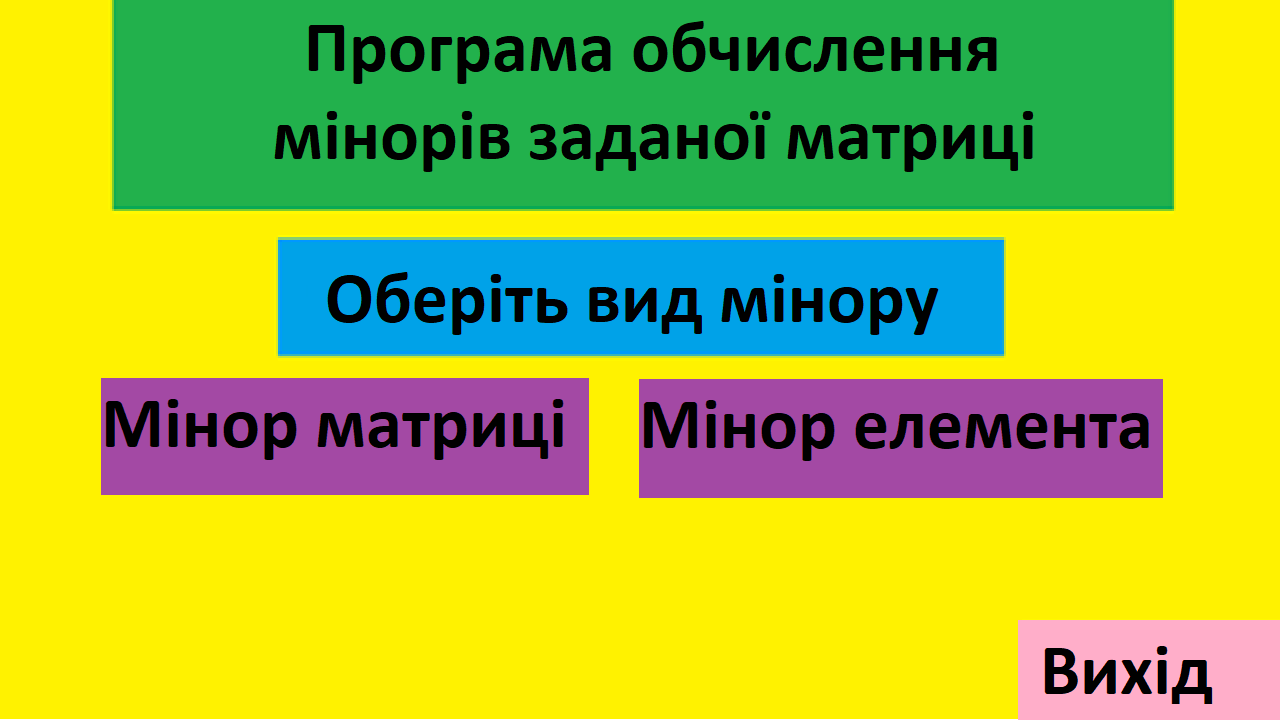
1. Опис розроблених алгоритмів

Загальний хід програми подано у блок-схемі в додатку А. Варто описати декілька функції, що застосовуються у програмі.

1. isnumb(x) – функція для перевірки аргумента х на те, чи є він числом. Повертає введене значення лише тоді, коли воно є числовим. Використовується для перевірки вводу при заповненні матриці.
2. input\_check(x, y) – рекурсивна функція для перевірки на те, чи є аргумент х числом, котре не менше 1 і не більше значення аргумента у, котра повертає х лише у випадку виконання обох згаданих вище умов або викликає саму себе, дозволяючи змінити значення х. Використовується для випадків, коли ввід користувача має бути обмеженим (для прикладу, коли потрібно ввести номер рядка матриці – обмежувачем зверху є загальна кількість рядків у матриці).
3. permut\_list(dim) – функція, що приймає один аргумент – розмірність квадратної матриці, і повертає список, до якого входять кортежі, склад яких – всі можливі перестановки чисел від 0 до dim-1. Вони надалі використовуються як ітератори для пошуку визначника
4. determ(matrix, perm\_list, dimension) – функція для пошуку визначника. Вона приймає три параметри: matrix – матриця, для якої потрібно обчислити детермінант; perm\_list – набір відповідних перестановок для розмірності матриці; dimension – сама розмірність. За допомогою циклу ця функція проходить по певній перестановці із perm\_list, знаходить її парність та відповідний їй добуток елементів матриці. Усі такі добутки збираються в окремий список, далі елементи цього списку сумуються, результатом чого стає потрібне число – значення визначника.
5. Засоби керування програмами

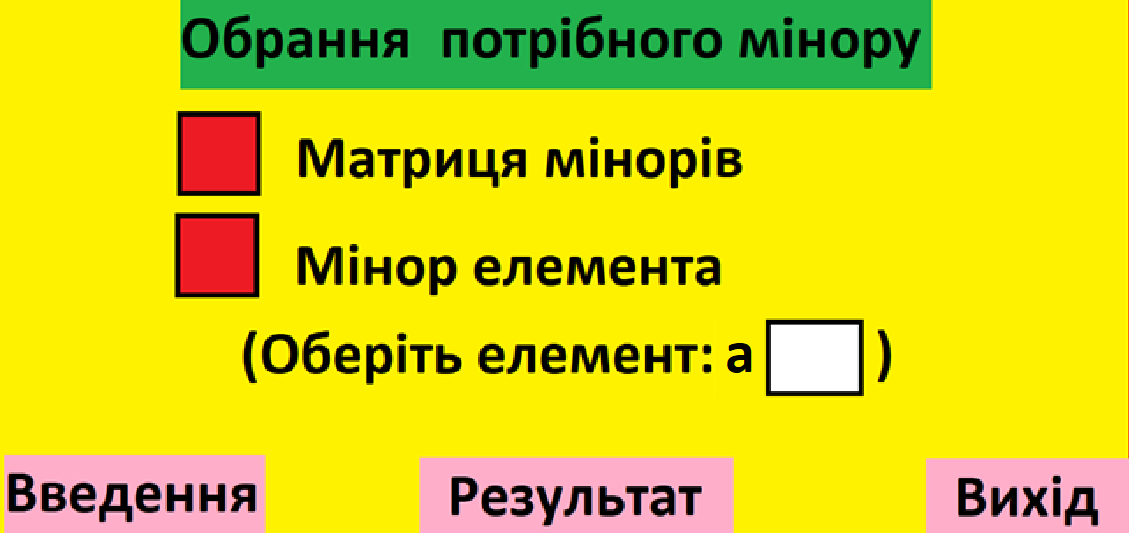
**Ескізи інтерфейсу програми обчислення мінорів заданої матриці**

Інтерфейс програми розрахований на введення лише числових даних (з використанням особливих символів « - » для введення від’ємних чисел і « . » для введення дробових чисел у підпрограмі «заповнення матриці») на етапах заповнення матриці елементами, обрання елемента, мінор якого потрібно знайти, а також обрання рядків та стовпців, що входять до мінору. Решта елементів з метою полегшення роботи з програмою мають бути реалізовані через інтерактивні кнопки. Через невисоку варіативність можливостей додаткові налаштування у програмі не передбачені. Для керування ходом виконання кожна підпрограма передбачає можливість завершення роботи та повернення до попередньої підпрограми.

**Рис. 1 Ескіз головного меню**



**Рис. 2 Ескіз меню заповнення матриці**



**Рис. 3 Ескіз меню вибору варіанту отримання результату**



**Рис. 4 Ескіз екрану результатів**

Завдяки тому, що у спеціалізованому інтерфейсі елементи, окрім введення даних, використовуються для вибору подальшого ходу виконання програми, її реалізація може бути доступною через командний рядок (події вибору в цьому випадку реалізуються через введення відповідей на запитання щодо подальшого ходу виконання (з указанням потрібних відповідей для кожного варіанту розвитку подій) із недопуском введення некоректних даних). Приклад такої взаємодії наведено у додатку В.

1. РЕАЛІЗАЦІЯ ПРОГРАМИ
2. Опис вхідних даних

У програмі передбачено 3 типи введених даних:

1. Розмірність матриці – ціле число, не менше 1. Обмежень зверху немає, проте вищою є продуктивність для невеликих значень.
2. Потрібні рядки та стовпці – цілі числа, не менші 1 і не більші кількості рядків/стовпців у матриці. Можуть бути застосовані як для опису елемента (що лежить на перетині вказаного рядка і стовпця), так і для опису мінору (на перетині вказаних рядків і стовпців) – у цьому випадку їх кількість обмежена розмірністю мінору.
3. Елементи матриці – дійсні числа, дробові вводяться у відповідності до типу float мови програмування Python – із крапкою в якості знаку між цілою та дробовою частинами. Для кращої роботи програми рекомендується використання невеликих чисел (одно- та двоцифрових) та чисел з невеликою кількістю (1-2) знаків після коми.

Для взаємодії у командному рядку застосовується введення у діалоговому форматі. Приклад таких діалогів:

* «Оберіть тип мінора ("Матриця" або "Елемент"):» - у відповіді потрібно ввести слово «Матриця» для отримання мінору прямокутної матриці, або слово «Елемент» для отримання мінору елемента. При цьому варто враховувати регістр (для прикладу, відповідь «матриця» буде хибною) та застосовувати кирилицю (для прикладу, введення літери М із латиниці буде прийнято як помилка).
* «Введіть кількість рядків матриці:» та аналогічні варіанти очікуватимуть у відповідь цілі числа, не менші 1
* «Введіть елемент а11:» та аналогічні варіанти потребують введення дійсного числа, що відповідає вказаному елементу.
* «Знайти інший мінор цієї матриці(для підтвердження введіть "Так")?:» та схожі – запитання для забезпечення повторного виконання програми для певних значень без необхідності повного рестарту програми, приймають лише варіанти «Так» як вказівку до повторного виконання того чи іншого етапу та будь-який інший ввід як негативну відповідь.

1. Опис результатів

В залежності від обраного користувачем виду мінору отримані результати можуть бути такими:

* Введена матриця та матриця мінорів, котра виводяться на екран як двовимірний масив із модуля NumPy;
* Мінор вказаного елемента / мінор для заданих рядків і стовпців – дійсне число, виведене у форматі float
* Помилка – вказівка на некоректність вводу користувача із конкретним зазначенням, яку хибу було допущено (вказання значення не із доступного інтервалу, хибний тип введених даних і т.д.)

1. Опис контрольних прикладів

**Приклад 1.** Знайти мінор   M 23 {\displaystyle \ M\_{23}} квадратної матриці   A {\displaystyle \ A} А:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| А = | |  |  |  |  |  | | --- | --- | --- | --- | --- | | ( | 1 | 2 | 3 | ( | | 4 | 5 | 6 | | 7 | 8 | 9 | |

**Розв’зання:** Мінор   M 23 {\displaystyle \ M\_{23}} квадратної матриці   A {\displaystyle \ A} А— визначник матриці, отриманий шляхом викреслювання рядка 2 та стовпчика 3: =1\*8-7\*2=8-14=-6

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | 1 | 2 | □ | | □ | □ | □ | | 7 | 8 | □ | | = |

|  |  |
| --- | --- |
| 1 | 2 |
| 7 | 8 |

**Приклад 2.** Знайти мінори матриці В

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| В = | |  |  |  |  |  | | --- | --- | --- | --- | --- | | ( | 5 | 7 | 1 | ( | | -4 | 1 | 0 | | 2 | 0 | 3 | |

**Розв’язання:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| M11 = | |  |  | | --- | --- | | 1 | 0 | | 0 | 3 | | = 1·3 - 0·0 = 3 - 0 = 3 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| M12 = | |  |  | | --- | --- | | -4 | 0 | | 2 | 3 | | = -4·3 - 0·2 = -12 -0 = -12 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| M13 = | |  |  | | --- | --- | | -4 | 1 | | 2 | 0 | | = -4·0 - 1·2 = 0 - 2 = -2 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| M21 = | |  |  | | --- | --- | | 7 | 1 | | 0 | 3 | | = 7·3 - 1·0 = 21 - 0 = 21 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| M22 = | |  |  | | --- | --- | | 5 | 1 | | 2 | 3 | | = 5·3 - 1·2 = 15 - 2 = 13 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| M23 = | |  |  | | --- | --- | | 5 | 7 | | 2 | 0 | | = 5·0 - 7·2 = 0 - 14 = -14 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| M31 = | |  |  | | --- | --- | | 7 | 1 | | 1 | 0 | | = 7·0 - 1·1 = 0 - 1 = -1 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| M32 = | |  |  | | --- | --- | | 5 | 1 | | -4 | 0 | | = 5·0 - 1·(-4) = 0 + 4 = 4 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| M33 = | |  |  | | --- | --- | | 5 | 7 | | -4 | 1 | | = 5·1 - 7·(-4) = 5 + 28 = 33 |

**Приклад 3.** Знайти мінори ,, матриці А:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| А = | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | ( | 2 | 0 | 1 | 0 | ( | | -2 | 3 | 1 | 4 | | -1 | 0 | 0 | 1 | | 2 | 0 | 1 | 2 | |

**Розв’язання:**

****

=3\*0\*2+1\*1\*0+0\*1\*4-0\*0\*4-0\*1\*2-3\*1\*1=-3



=2\*1\*2+1\*4\*2+(-2)\*1\*0-0\*1\*2-(-2)\*1\*2-1\*2\*4=8



=2\*3\*1+0\*4\*(-1)+0\*(-2)\*0-0\*3\*(-1)-(-2)\*0\*1-0\*4\*2=6

1. Експериментальні розрахунки

У ході тестування програми програми було перевірено працездатність програми, засвідчено недопуск переривання виконання програми в ході дій користувача (вводу хибних даних і т.д.), перевірено влучність підказок для юзерів, додано можливість повторного виконання майже на кожному важливому етапі програми. Було перевірено виконання програми на контрольних прикладах і отримано результати, котрі співпали із отриманими вручну, що засвідчує правильність виконання програми. Детальніше із прикладами виконання програми можна ознайомитись у додатку В.

ВИСНОВКИ

Отже, у результаті виконання розрахунково-графічної роботи було вивчено мінори матриці як явище та методи їх пошуку на основі математичних знань. Було отримано навички розробки алгоритму програми, написання схем взаємодії та блок-схем, вивчено можливості мови програмування Python для реалізації поставлених цілей. Було отримано навчики застосування мови програмування Python для втілення потрінбних логічних та математичних операцій, досліджено застосування циклів та створення функцій для виконання тих чи інших підзавдань. На основі отриманих знань було розроблено програму обчислення мінорів заданої матриці, неодноразово перевірено її працездатність та звірено результати із тими, що були отримані в ході ручних розрахунків.

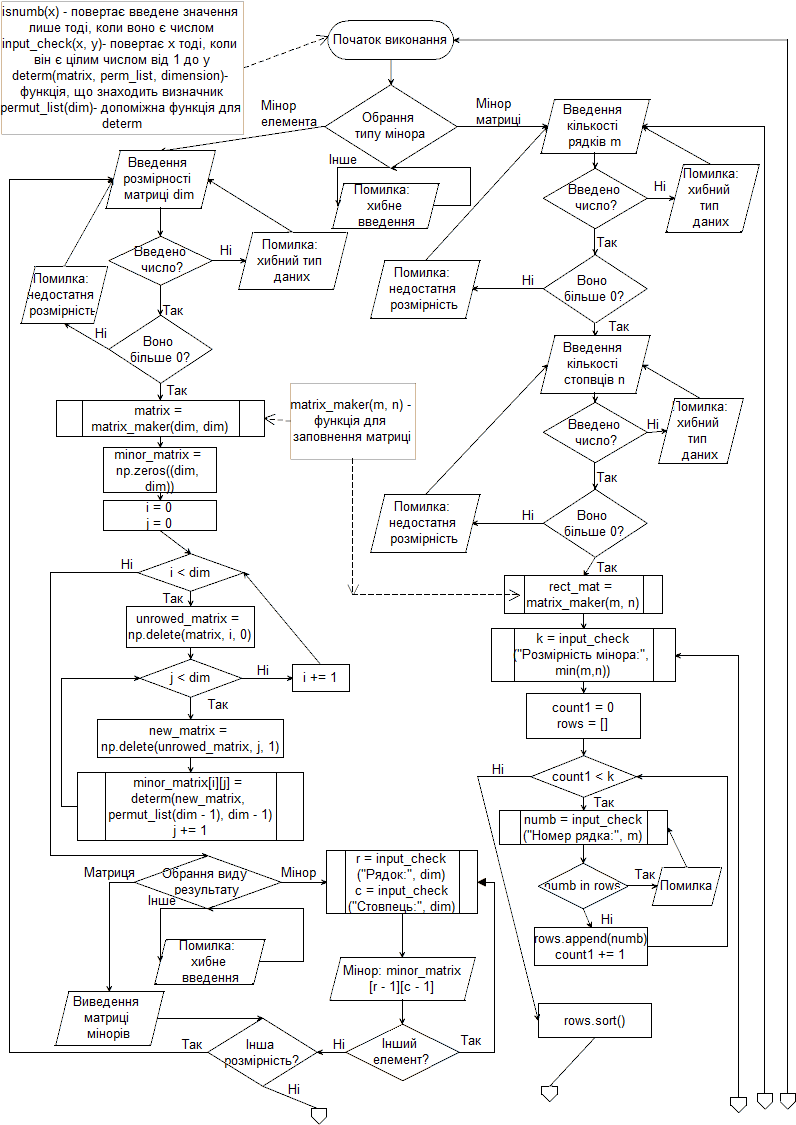
На основі тестування було розроблено та покращену схему зручної взаємодії між користувачем та програмою, створено ескізи інтерфейсу майбутньої програми та передбачено можливість застосування програми без додаткового інтерфейсу (зокрема через командний рядок).

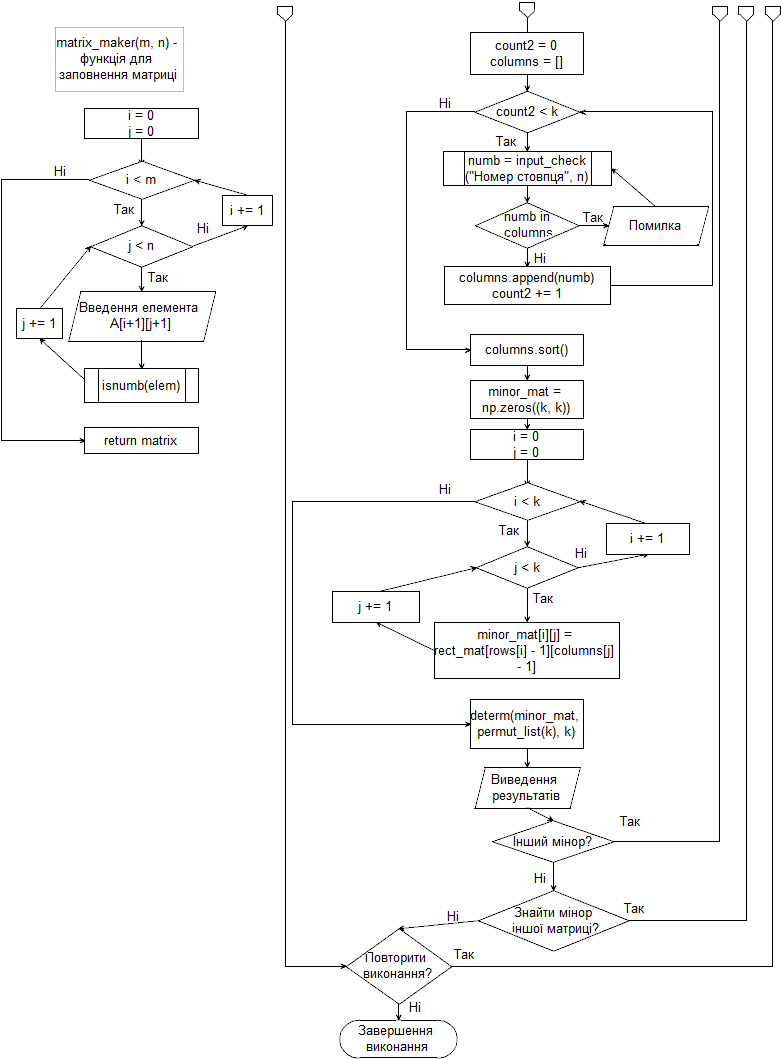
ЛІТЕРАТУРА

1. Визначники // Вища математика в прикладах і задачах / Клепко В.Ю., Голець В.Л.. — 2-ге видання. — К. : Центр учбової літератури, 2009. — С. 13-15. — 594 с.
2. *Гантмахер Ф. Р.* Теория матриц. — 2 изд. — Москва : [Наука](https://uk.wikipedia.org/wiki/%D0%9D%D0%B0%D1%83%D0%BA%D0%B0_(%D0%B2%D0%B8%D0%B4%D0%B0%D0%B2%D0%BD%D0%B8%D1%86%D1%82%D0%B2%D0%BE)), 1967. — 576 с. — ISBN 5-9221-0524-8.
3. Мінор та алгебраїчне доповнення матриці. / OnlineMSchool / [Електронний ресурс] - http://ua.onlinemschool.com/math/library/matrix/minors/

ДОДАТОК А

Блок-схема програми





ДОДАТОК Б

Текст програми

import numpy as np

import itertools

def isnumb(x):

while type(x) != float:

try:

x = float(x)

except ValueError:

print("Помилка. Введене значення не є числом")

x = input("Введіть числове значення елемента (дробові числа вводяться з крапкою):")

return float(x)

def input\_check(x, y):

input\_data = input(x)

if not input\_data.isdigit():

return input\_check("Ви ввели не додатнє число. Введіть існуючий номер:", y)

if 1 <= int(input\_data) <= y:

return int(input\_data)

return input\_check("Помилка. Введіть доступне значення:", y)

def permut\_list(dim):

"""

Функція створює список перестановок чисел від 0 до розмірність-1, котрі потім використовуються для обчислення

визначника матриці довільної розмірності.

"""

ind\_list = []

for i in range(0, dim):

ind\_list.append(i)

perm = list(itertools.permutations(ind\_list, dim))

return perm

def determ(matrix, perm\_list, dimension):

"""

Ця функція обчислює елементи для майбутнього додавання, обчислюючи добутки потрібних елементів:

їх перші індекси рівні 1, 2, ... n, а другі іденксами є перестановками чисел від 1 до n.

Функція одразу визначає знак потрібного елементу: якщо відповідна йому перестановка є парною,

елемент береться із знаком +, у протилежному випадку-зі знаком -. На останньому кроці функція додає

отримані елементи. Результат виконання функції-визначник матриці, число.

"""

res = []

a = []

for i in range(0, len(perm\_list)):

a.append(0)

for j in range(0, dimension):

for k in range(0, dimension):

if j < k and perm\_list[i][j] > perm\_list[i][k]:

a[i] += 1

if a[i] % 2 == 1:

a[i] = -1

else:

a[i] = 1

res.append(1)

for j in range(0, dimension):

res[i] \*= matrix[j][perm\_list[i][j]]

res[i] = res[i] \* a[i]

s = 0

for i in range(0, len(res)):

s += res[i]

return s

def matrix\_maker(m, n):

matrix = np.zeros((m, n))

# Створення матриці нулів даного розміру для майбутнього заповнення елементами

for i in range(0, m):

for j in range(0, n):

matrix[i][j] = isnumb(input("Введіть елемент а" + str(i + 1) + str(j + 1) + ":"))

return matrix

def rectang():

while True:

m = input("Введіть кількість рядків матриці:")

while type(m) != int:

try:

m = int(m)

if m <= 0:

m = input("Помилка:вказане вам число не більше 0, тому не може бути розмірністю матриці."

"Введіть кількість рядків матриці:")

except ValueError:

m = input("Помилка: введене вами значення не є цілим числом. Введіть розмірність матриці:")

n = input("Введіть кількість стовпців матриці:")

while type(n) != int:

try:

n = int(n)

if n <= 0:

n = input("Помилка:вказане вам число не більше 0, тому не може бути розмірністю матриці."

"Введіть кількість стовпців матриці:")

except ValueError:

n = input("Помилка: введене вами значення не є цілим числом. Введіть розмірність матриці:")

rect\_mat = matrix\_maker(m, n)

print("Ваша матриця:")

print(rect\_mat)

while True:

k = input\_check("Введіть кількість рядків та стовпців, для яких потрібно знайти мінор:", min(m, n))

count1 = 0

rows = []

while count1 < k:

numb = input\_check("Введіть номер рядка, що входить до мінору:", m)

while numb in rows:

print("Цей рядок уже входить до мінору. Будь ласка, оберіть інший.")

numb = input\_check("Введіть номер рядка, що входить до мінору:", m)

rows.append(numb)

count1 += 1

rows.sort()

count2 = 0

columns = []

while count2 < k:

numb = input\_check("Введіть номер стовця, що входить до мінору:", n)

while numb in columns:

print("Цей стовпчик уже входить до мінору. Будь ласка, оберіть інший стовпчик.")

numb = input\_check("Введіть номер стовпця, що входить до мінору:", n)

columns.append(numb)

count2 += 1

columns.sort()

minor\_mat = np.zeros((k, k))

for i in range(0, k):

for j in range(0, k):

minor\_mat[i][j] = rect\_mat[rows[i] - 1][columns[j] - 1]

print("Отримана матриця:")

print(minor\_mat)

print("Ваш мінор:" + str(determ(minor\_mat, permut\_list(k), k)))

if input('Знайти інший мінор цієї матриці(для підтвердження введіть "Так")?:') != "Так":

break

if input('Знайти мінор іншої матриці(для підтвердження введіть "Так")?:') != "Так":

break

def square():

while True:

dim = input("Введіть розмірність матриці:")

while type(dim) != int:

try:

dim = int(dim)

if dim <= 0:

dim = input("Помилка:вказане вам число не більше 0, тому не може бути розмірністю матриці."

"Введіть розмірність матриці:")

except ValueError:

dim = input("Помилка: введене вами значення не є цілим числом. Введіть розмірність матриці:")

while True:

matrix = matrix\_maker(dim, dim)

# Цикл заповнює матрицю елементами, котрі вказує користувач

print("Ваша матриця:")

print(matrix)

minor\_matrix = np.zeros((dim, dim))

# Створення матриці нулів даного розміру для заповнення мінорами

for i in range(0, dim):

unrowed\_matrix = np.delete(matrix, i, 0)

for j in range(0, dim):

new\_matrix = np.delete(unrowed\_matrix, j, 1)

minor\_matrix[i][j] = determ(new\_matrix, permut\_list(dim - 1), dim - 1)

# Спершу цикли видаляють рядок і стовпець матриці, у якому стояв елемент, для якого обчислюється мінор.

# Наступним кроком за допомогою функції обчислюється визначник отрмианої матриці.

ask = input("Отримати матрицю мінорів (введіть 'Матриця') чи мінор елемента(введіть 'Елемент')?:")

# Цикл дає змогу визначити, в якому виді подати результати

while True:

if ask == "Матриця":

print("Матриця мінорів:")

print(minor\_matrix)

break

elif ask == "Елемент":

while True:

r = input\_check("Введіть номер рядка:", dim)

c = input\_check("Введіть номер стовпця:", dim)

print("Мінор елемента а" + str(r) + str(c) + " = " + str(minor\_matrix[r - 1][c - 1]))

if input("Обрати інший елемент (введіть 'Так')? ") != "Так":

break

break

else:

print("Помилка: хибне введення.")

ask = input("Отримати матрицю мінорів (введіть 'Матриця') чи мінор елемента(введіть 'Елемент')?:")

if input("Повторити виконання для того ж розміру матриці (для підтвердження введіть 'Так')? ") != 'Так':

break

if input("Знайти мінор елемента іншої матриці (для підтвердження введіть 'Так')?:") != 'Так':

break

def prog():

decide = input('Оберіть тип мінора ("Матриця" або "Елемент"):')

while decide not in ['Матриця', 'Елемент']:

print("Хибне введення. Будь ласка, оберіть один із варіантів")

decide = input('Оберіть тип мінора ("Матриця" або "Елемент"):')

if decide == 'Матриця':

rectang()

elif decide == 'Елемент':

square()

print("\n\n\n", "\t\t\t\t\t\t", "Розрахунково-графічна робота", sep="")

print("\n", "\t\t\t\t\t", " З дисципліни: «Програмування» 1-й семестр", sep="")

print("\n\n", "\t\t\t\t", "На тему: «Програма обчислення мінорів заданої матриці»", sep="")

print("\n\n\n\n\n", "\t\t\t", "Виконав", "\t\t\t\t\t\t\t\t", "Студент групи КМ-02", sep="")

print("\n", "\t\t\t", "\t\t\t\t\t\t\t\t\t", "Фірман Д.Б.", sep="")

print("\n\n\n", "\t\t\t\t\t\t\t\t\t\t\t\t\t", "2020", "\n\n\n", sep="")

while True:

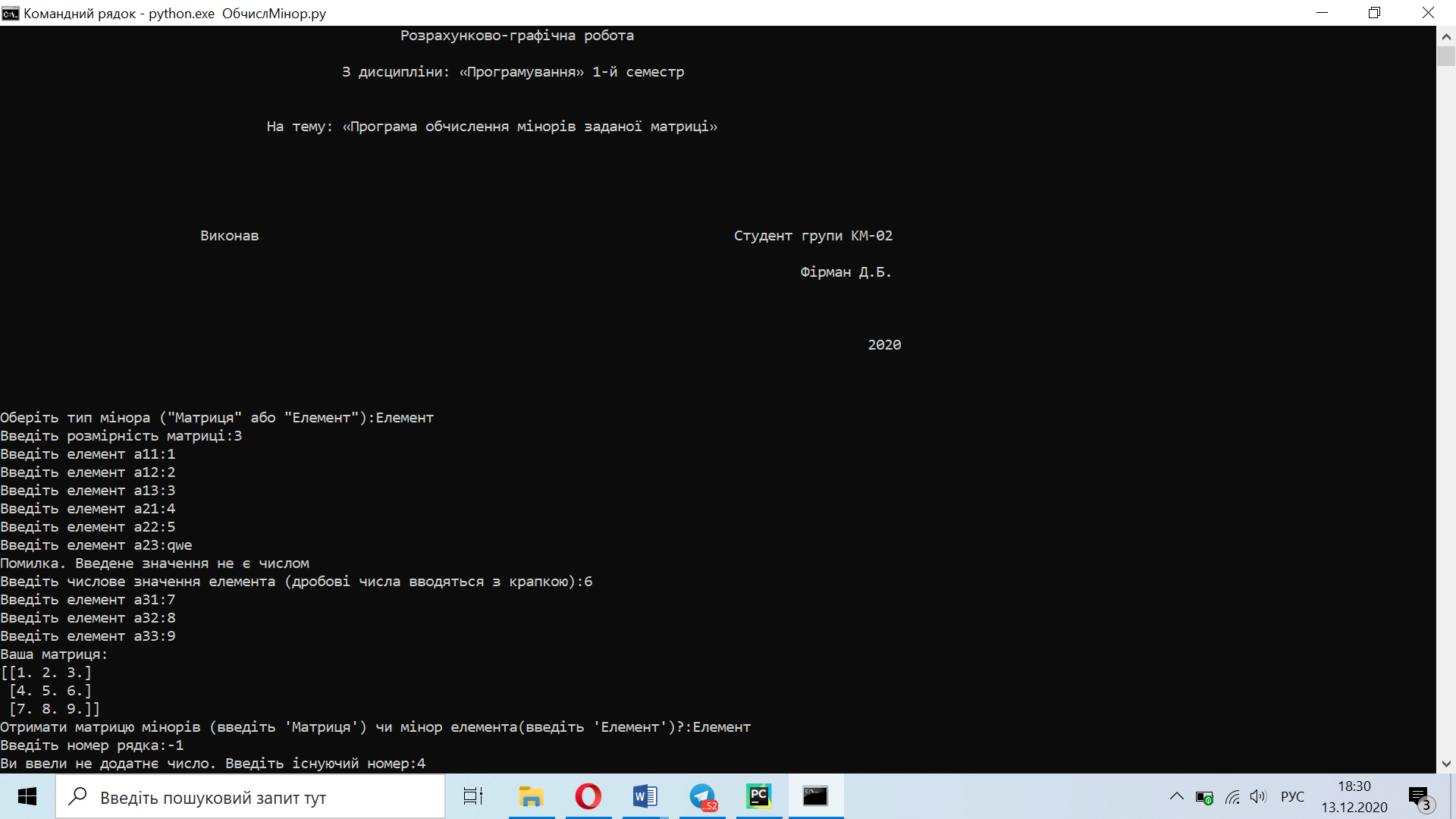
prog()

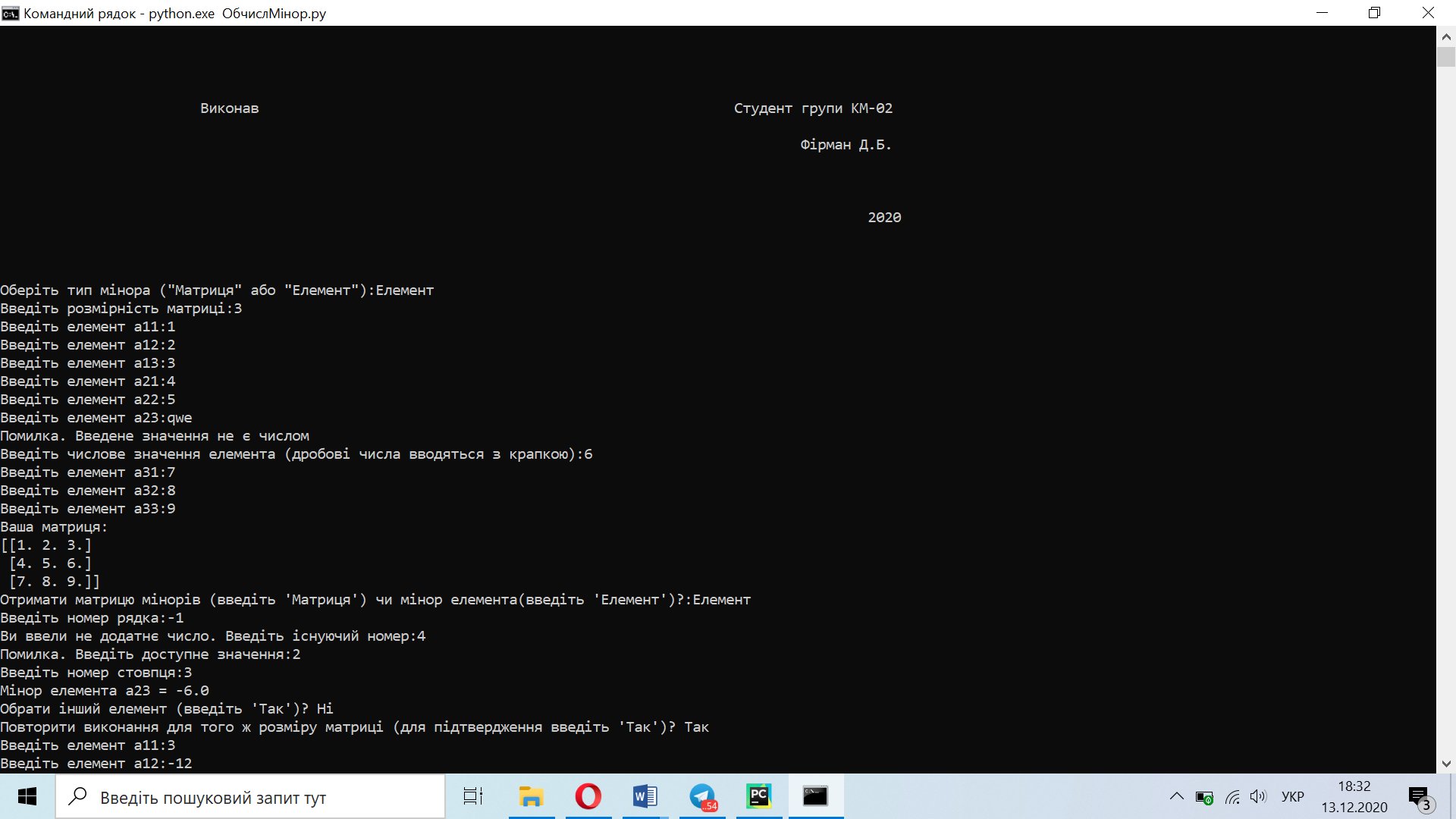
if input("Бажаєте повторно скористатись програмою (для підтвердження введіть 'Так')?:") != 'Так':

break

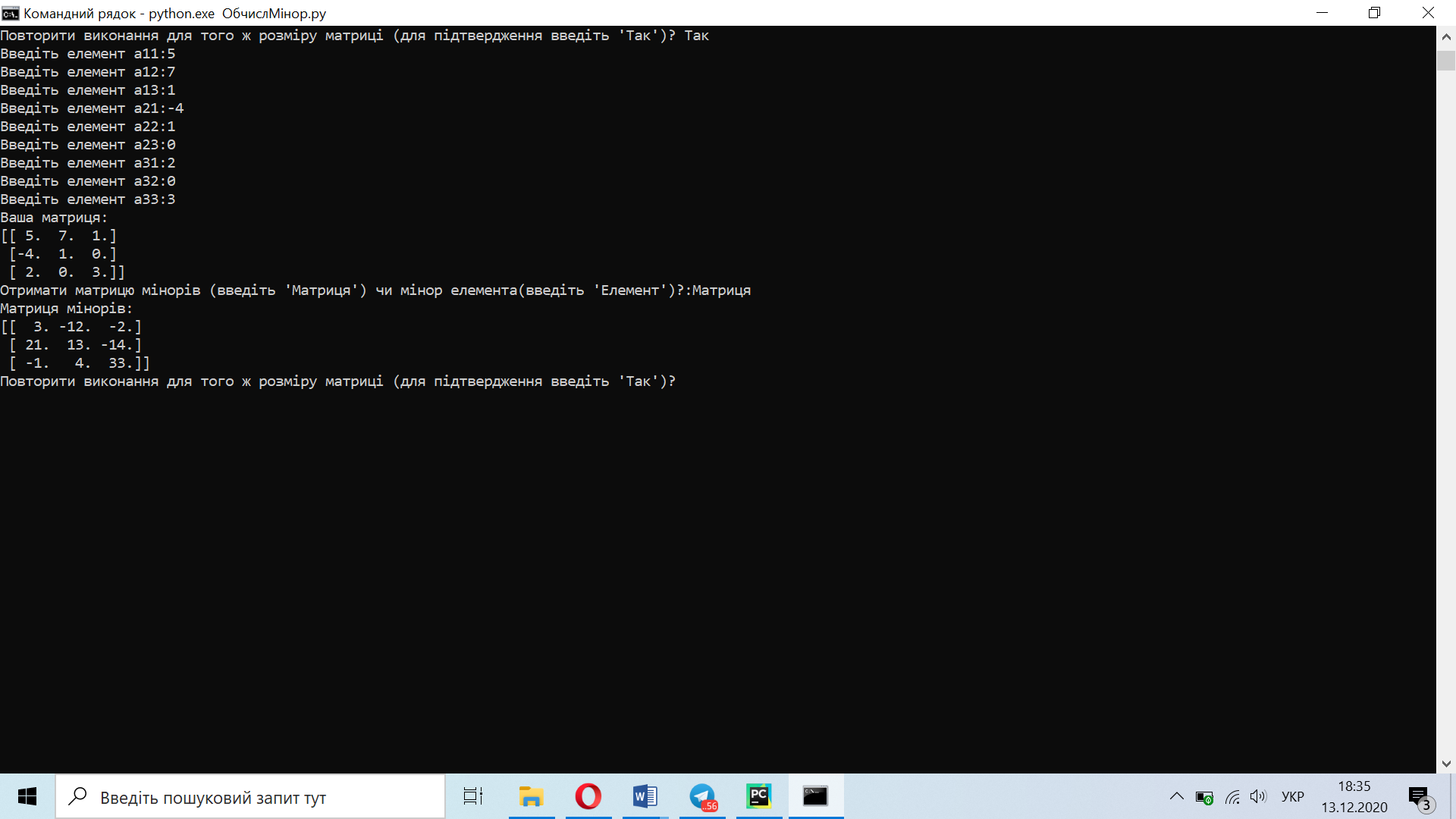
ДОДАТОК В

Експериментальні розрахунки

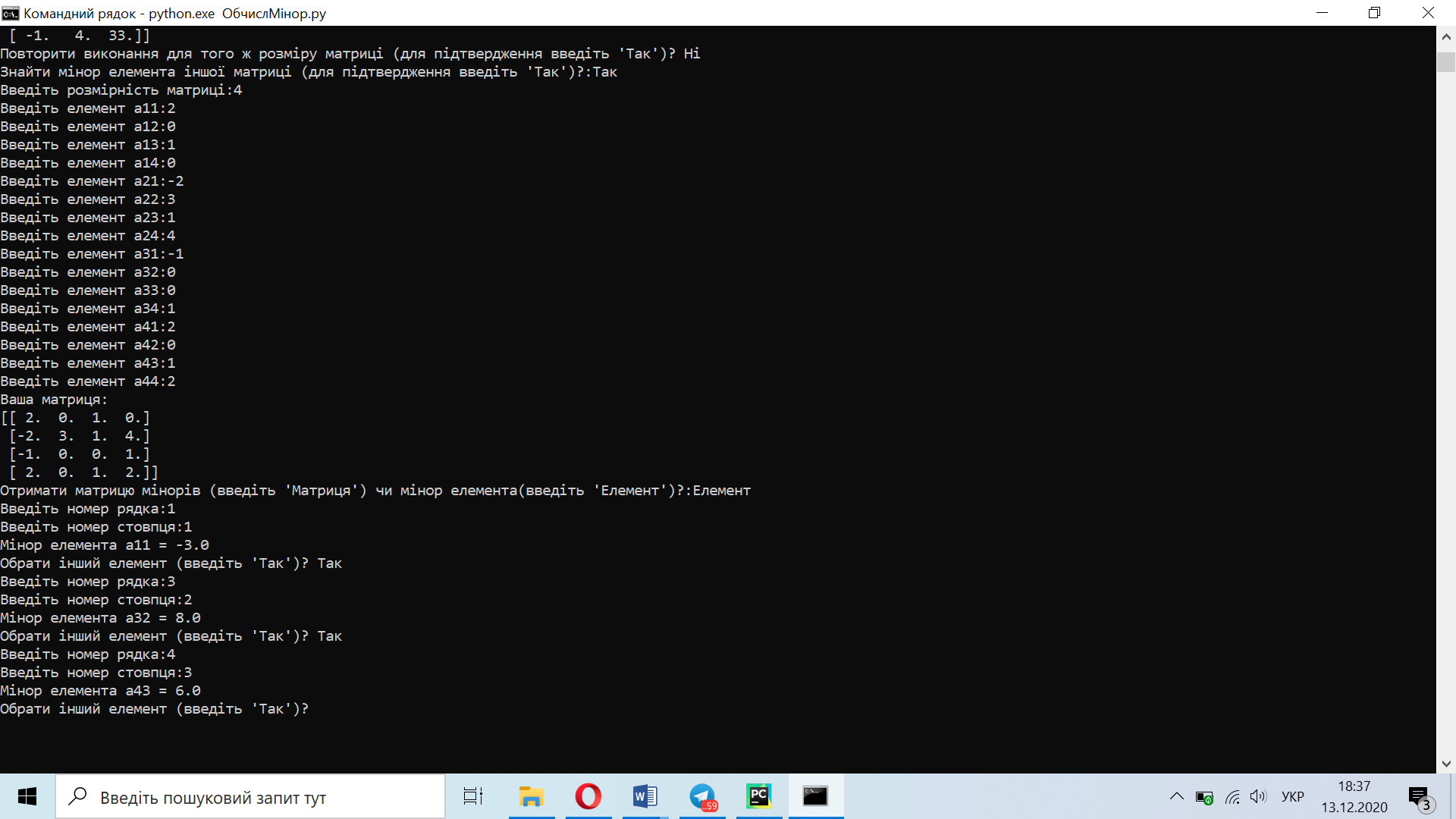




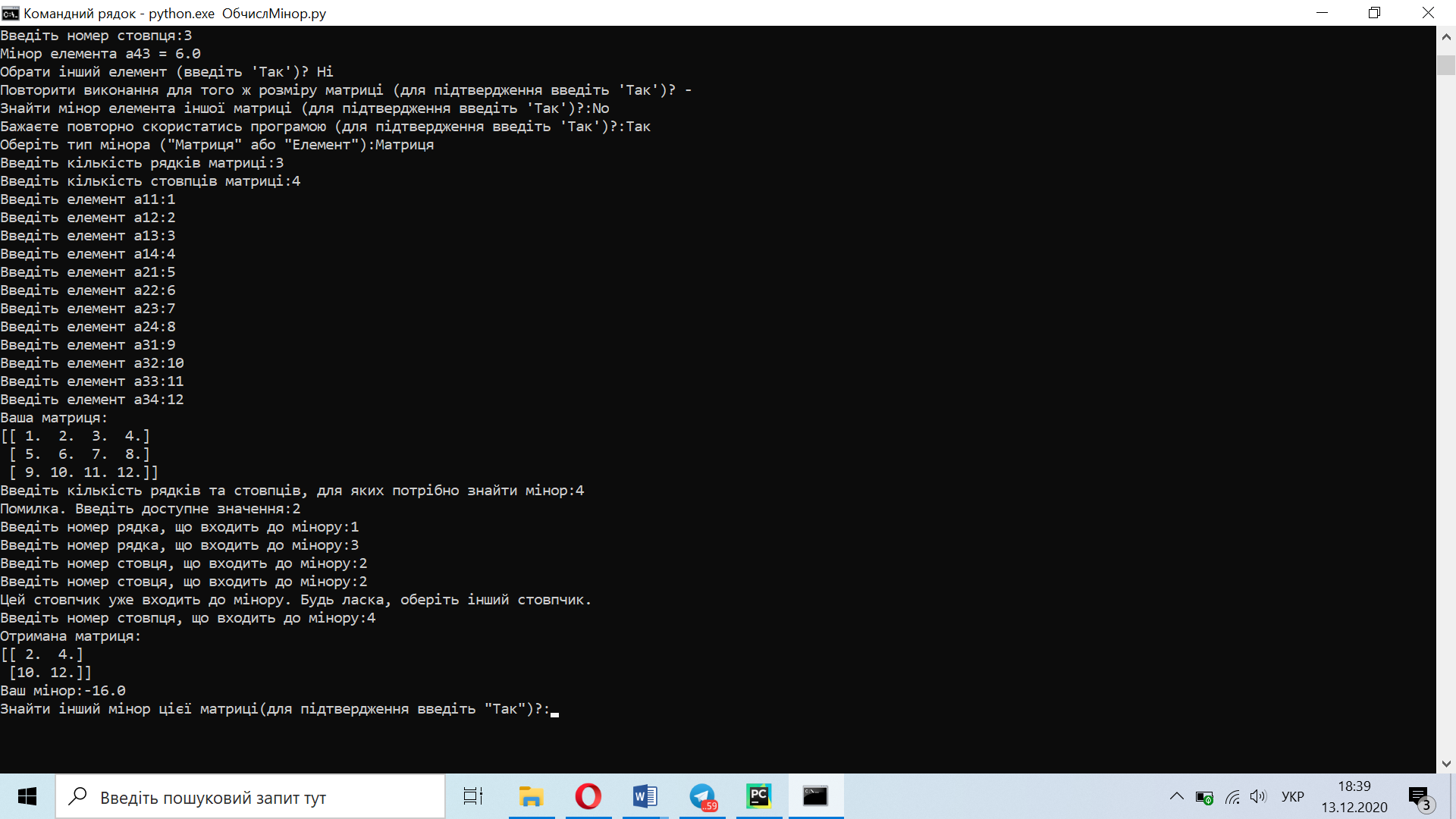
**Рис. 1 Тестування на прикладі 1**



**Рис. 2 Тестування на прикладі 2**



**Рис. 3 Тестування на прикладі 3**



**Рис. 4 Тестування для мінору прямокутної матриці**

На останньому прикладі водночас продемонстровані деякі особливості програми, зокрема:

* Прийняття будь-якої відповіді, окрім «Так», як хибної;
* Перевірка доступності введеного значення;
* Недопуск повторного використання одного й того ж рядка у мінорі;
* Візуалізація введених даних та отриманих результатів для кращого сприйняття.