**Лабораторна робота 4. Елементи лінійної алгебри. Розв’язання систем лінійних алгебраїчних рівнянь**

1. Розв’язати 1 завдання з кожного підрозділу.

1. Знайти матрицю :

1) ;

2) , ;

3), .

2. Піднести матриці до степеня:

1) ; 2) ; 3) .

3. Знайти добуток матриць:

1) ;

2)

3);

4) ;

5) .

1. Обчислити визначники:

1) ; 2) ; 3) ;

4) ; 5) .

1. Обчислити визначники.
2. ; 2) ; 3) .
3. Знайти обернену матрицю до матриць:

1) ; 2) ;

3) ; 4); 5) .

7. Визначити ранг матриці:

1) ; 2) ; 3) .

1. Розв’язати систему лінійних рівнянь методом Крамера, матричним методом:

4)

9)

11)

13) 14)

15) 16

18)

20)

22)

28)

29)

**2. Розв’язати 1 завдання з наведених нижче.**

1.Створіть прямокутну матрицю A, яка має N рядків і M стовпців з випадковими елементами. Знайдіть найменший стовпчастий елемент матриці A, для якого сума абсолютних значень елементів максимальна.

2. Створіть прямокутну матрицю A з N рядками та стовпцями M з випадкових елементів. Знайдіть найнижче значення серед середніх значень для кожного рядка матриці.

3. Створіть прямокутну матрицю A, яка має N рядків і M стовпців з випадковими елементами. Визначте середні значення для всіх рядків і стовпців матриці.

4. Створіть прямокутну матрицю A, яка має N рядків і M стовпців з випадковими елементами. Знайдіть суму елементів всієї матриці. Визначте, яку долю в цій сумі складає сума елементів кожного стовпця.

5. Створіть прямокутну матрицю A, яка має N рядків і M стовпців з випадковими елементами. Знайдіть суму елементів всієї матриці. Визначте, яку долю в цій сумі складає сума елементів кожного рядка.

6. Створіть прямокутну матрицю A, яка має N рядків і стовпці M з випадковими елементами. Визначте, скільки негативних елементів міститься в кожному стовпці та в кожному рядку матриці.

**Приклад коду**

import numpy as np

m\_sqr\_arr = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]]) **#Задаємо матрицю 3\*3**

print(m\_sqr\_arr)

m\_sqr\_mx = np.matrix([[1, 2, 3], [4, 5, 6], [7, 8, 9]]) **#Ще один варіант як задати матрицю**

print(m\_sqr\_mx)

**#Побудуємо діагональ**

m\_sqr\_mx = np.matrix('1 2 3; 4 5 6; 7 8 9')

diag = np.diag(m\_sqr\_mx)

print(diag)

**#Побудуємо діагональну матрицю на базі полученої діагоналі**

m\_diag\_np = np.diag(np.diag(m\_sqr\_mx))

print(m\_diag\_np)

**#Транспонована матриця**

= np.matrix('1 2 3; 4 5 6')

print('A=',a)

a\_t = a.transpose()

print(a\_t)

print(a.T) **# ще один способ транспонування матриці**

a = np.matrix('1 2; 3 4')

a\_det = np.linalg.det(a) **#Визначник матриці**

print(format(a\_det, '.9g'))

**#Множення матриці на число**

a = np.matrix('1 2 3; 4 5 6')

c = 3 \* a

print(c)

**#Додавання матриць**

a = np.matrix('1 6 3; 8 2 7')

b = np.matrix('8 1 5; 6 9 12')

c = a + b

print(c)

**#Добуток матриць**

a = np.matrix('1 2 3; 4 5 6')

b = np.matrix('7 8; 9 1; 2 3')

c = a.dot(b)

print(c)

**# Обернена матриця**

A = np.matrix('1 -3; 2 5')

A\_inv = np.linalg.inv(A)

print(A\_inv)

**#Ранг матриці**

rank = np.linalg.matrix\_rank(A)

print(rank)

**#Матричний метод**

A = np.matrix('3 -1 2; 1 4 -1; 2 3 1')

B = np.matrix('-4; 10; 8')

print('A=', A)

print('B=',B)

A\_inv = np.linalg.inv(A)

print(A\_inv)

X = A\_inv.dot(B)

print('X=',X)

**#Метод Крамера**

a = np.matrix('3 -1 2; 1 4 -1; 2 3 1')

print('A=',a)

b = np.matrix('-4; 10; 8')

print('B=',b)

def kramer (a, b):

a\_det = np.linalg.det(A)

print('Детермінант матриці = ', a\_det)

#Зробити реревірку,що детермінант не дорівнює нулю

if (a\_det != 0):

print ('Розв\*язуємо систему')

x\_m = np.matrix(a)

x\_m[:, 0] = b # формування допоміжної матриці (1 ст. замінюємо на ст. b)

print('x\_m=', x\_m)

y\_m = np.matrix(a) #2 ст. замінюємо на ст. b

y\_m[:, 1] = b #2 c

print('y\_m=',y\_m)

z\_m = np.matrix(a) #3 ст. замінюємо на ст. b

z\_m[:, 2] = b

print('z\_m=',z\_m)

x = np.linalg.det(x\_m) / a\_det

y = np.linalg.det(y\_m) / a\_det

z = np.linalg.det(z\_m) / a\_det

print('X = ', round(x,5))

print('Y=', round(y,5))

print('Z=', round(z,5))

else:

print('Розв\*язків немає')

kramer(a,b)

**#Перевірка за допомогою методом solve() пакету linalg:**

X = np.linalg.solve(a, b)

print('Перевірка X=',X)

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

**Метод Крамера**

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

**Звіт має містити**

1. ПІП, номер варіанта, група
2. Розв’язання задач з лінійної алгебри (по 1 з кожного розділу). Код+ скрін
3. Розв’язання системи лінійних алгебраїчних рівнянь методом Крамера, Матричний метод. Код+скрін.
4. Перевірка отриманого розв’язку.
5. Завдання на вибір. Код+скрін