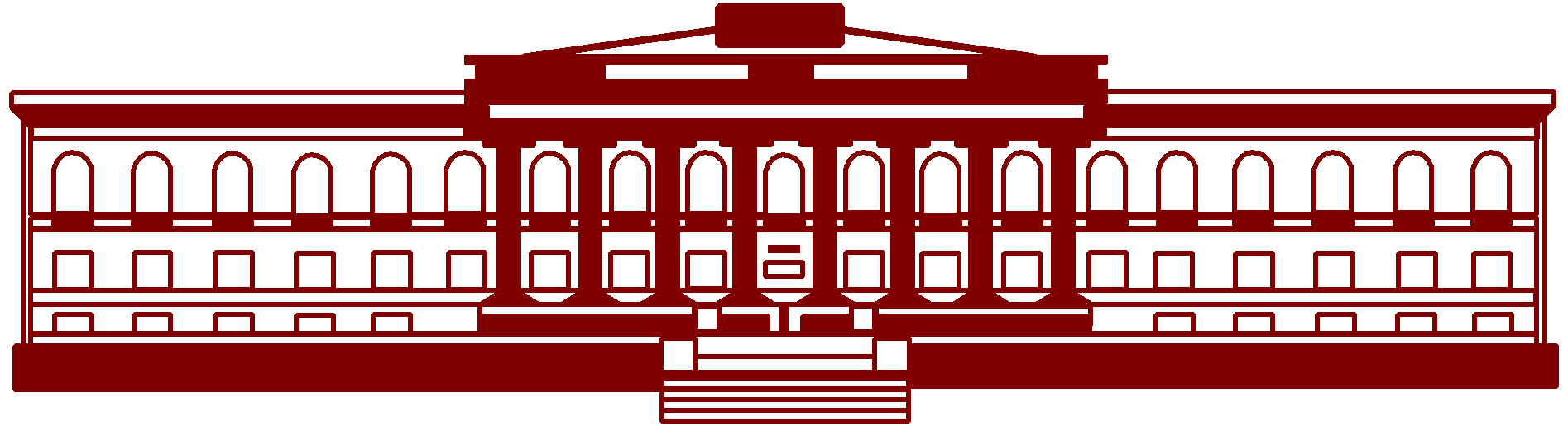
**КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ**

**імені ТАРАСА ШЕВЧЕНКА**

****

**ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ**

**Кафедра прикладних інформаційних систем**

**Звіт до лабораторної роботи №5**

**з курсу**

**«Системний аналіз та теорія прийняття рішень»**

*Студентa 3 курсу*

*групи ПП-32*

*спеціальності 122 «Комп'ютерні науки»*

*ОП «Прикладне програмування»*

*Кондратова Івана Андрійовича*

*Викладач:*

Плескач В.Л.

**Київ – 2023**

**Тема:** Теорія ігор

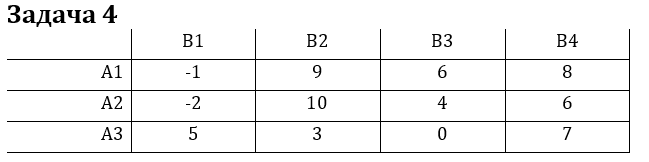
**Мета:** Набутинавички пошуку раціональних рішень в умовах конфліктів.

**Завдання 1:** розв’язання гри (в чистих стратегіях) з заданою матрицею платежів

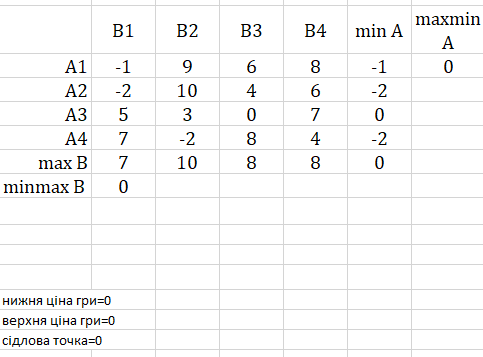
1. Вивчення теорії.

2. Визначення за заданою матрицею платежів нижньої та верхньої ціни гри. Встановити чи існує в грі рівновага в чистих стратегіях?

3. Розв’язання задачі програмно або з допомогою пакета MS Excel.



**Розв’язання в Excel**

****

**Розв’язаня на Python**

import numpy as np

def get\_minmax(matrix):

    transpose\_matrix = matrix.T

    cols\_min\_values = []

    for col in transpose\_matrix:

        cols\_min\_values.append(np.max(col))

    cols\_min\_values.append(get\_maxmin(matrix))

    return np.min(cols\_min\_values)

def get\_maxmin(matrix):

    rows\_min\_values = []

    for row in matrix:

        rows\_min\_values.append(np.min(row))

    return np.max(rows\_min\_values)

def main():

    matrix = np.array([

        [-1, 9, 6, 8],

        [-2, 10, 4, 6],

        [5, 3, 0, 7],

        [7, -2, 8, 4]

    ])

    print(get\_minmax(matrix))

    print(get\_maxmin(matrix))

if \_\_name\_\_ == '\_\_main\_\_':

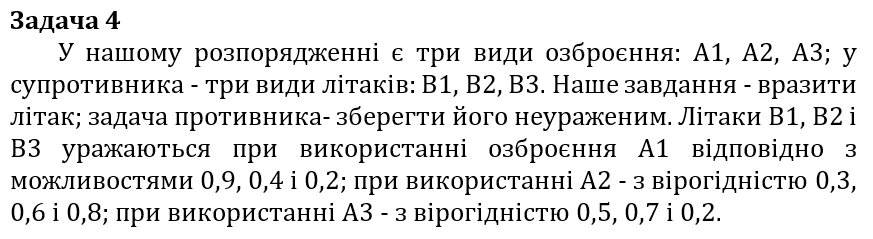
    main()

**Завдання 2**: розв’язання гри

1. Вивчення прикладів.

2. Побудова матриці платежів.

3. Розв’язання задачі програмно і з допомогою пакета MS Excel.



**Розв’язання в Excel**



**Розв’язаня на Python**

import numpy as np

def get\_minmax(matrix):

    transpose\_matrix = matrix.T

    cols\_min\_values = []

    for col in transpose\_matrix:

        cols\_min\_values.append(np.max(col))

    cols\_min\_values.append(get\_maxmin(matrix))

    return np.min(cols\_min\_values)

def get\_maxmin(matrix):

    rows\_min\_values = []

    for row in matrix:

        rows\_min\_values.append(np.min(row))

    return np.max(rows\_min\_values)

def main():

    matrix = np.array([

        [0.9, 0.4, 0.2],

        [0.3, 0.6, 0.8],

        [0.5, 0.7, 0.2]

    ])

    print(get\_minmax(matrix))

    print(get\_maxmin(matrix))

if \_\_name\_\_ == '\_\_main\_\_':

    main()

**Висновок:** виконуючи цю лабораторну роботу, я набувнавичок пошуку раціональних рішень в умовах конфліктів.