**Міністерство освіти і науки України**

**Національний університет**

**“Львівська політехніка”**

**Кафедра систем штучного інтелекту**

**Лабораторна робота № 3**

**з дисципліни**

**«Дискретна математика»**

**Виконав:**

**студент групи КН-115**

**Лукавий Мар’ян**

**Викладач:**

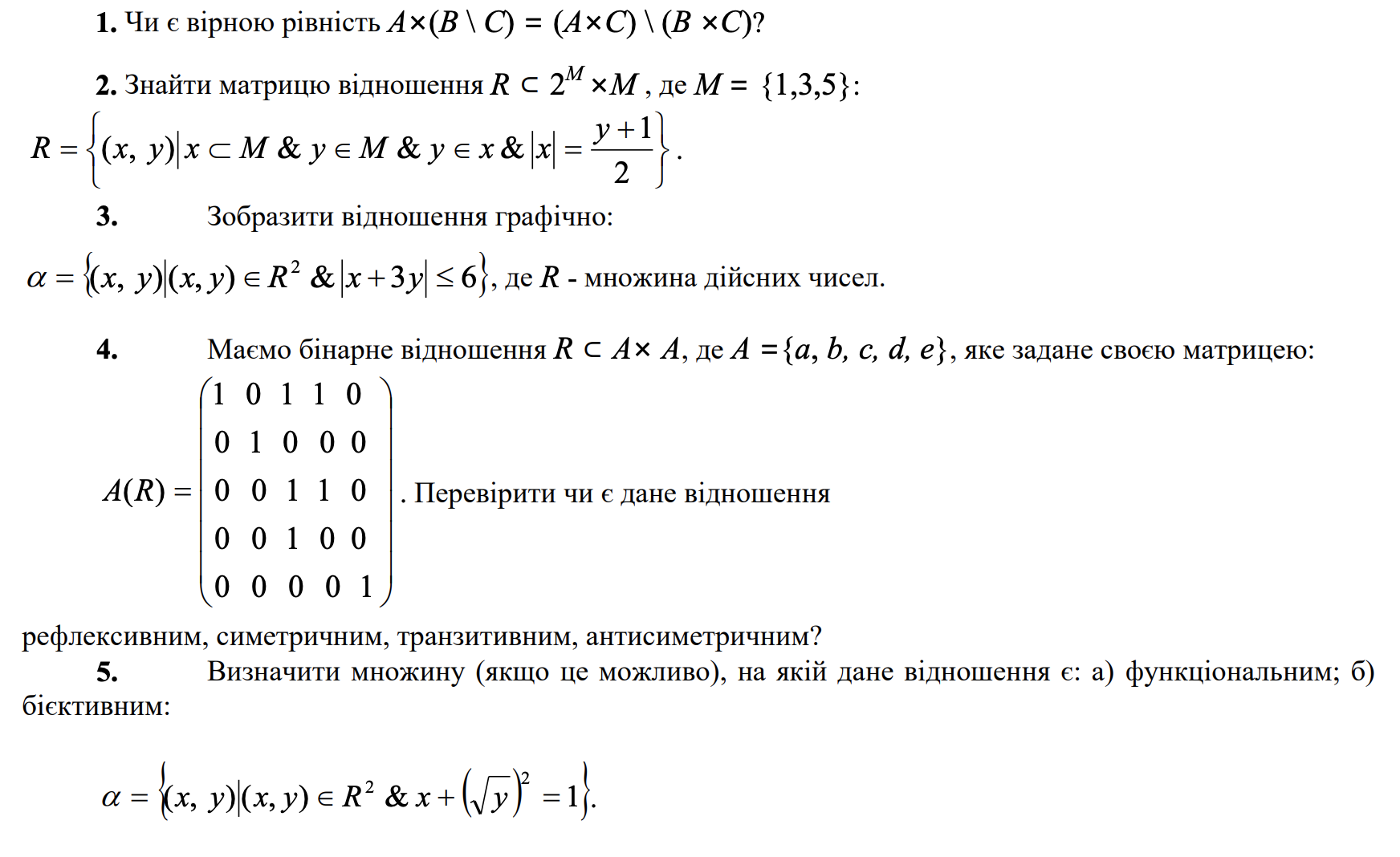
**Мельникова Н. І.**

**Львів – 2019р.**

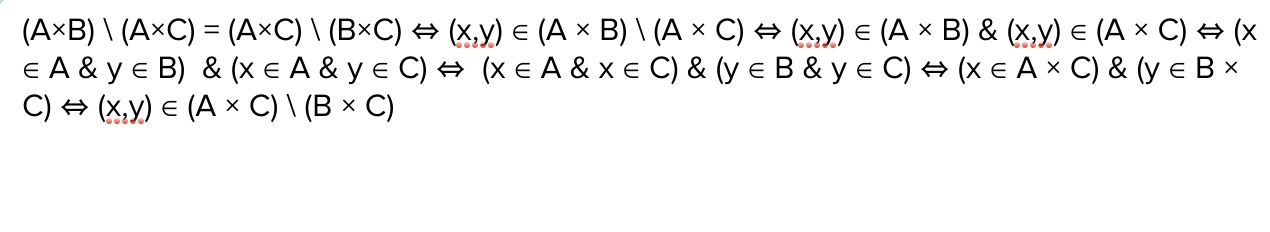
**Тема: Побудова матриці бінарного відношення**

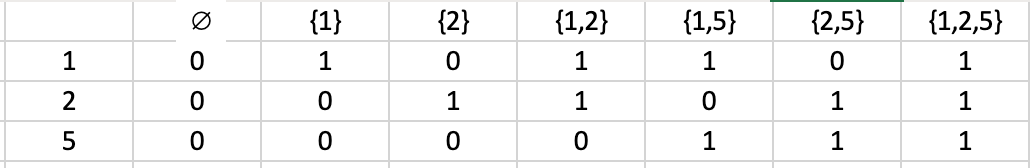
**Мета роботи: набуття практичних вмінь та навичок при побудові матриць бінарних відношень та визначені їх типів.**

**Варіант #16**

****

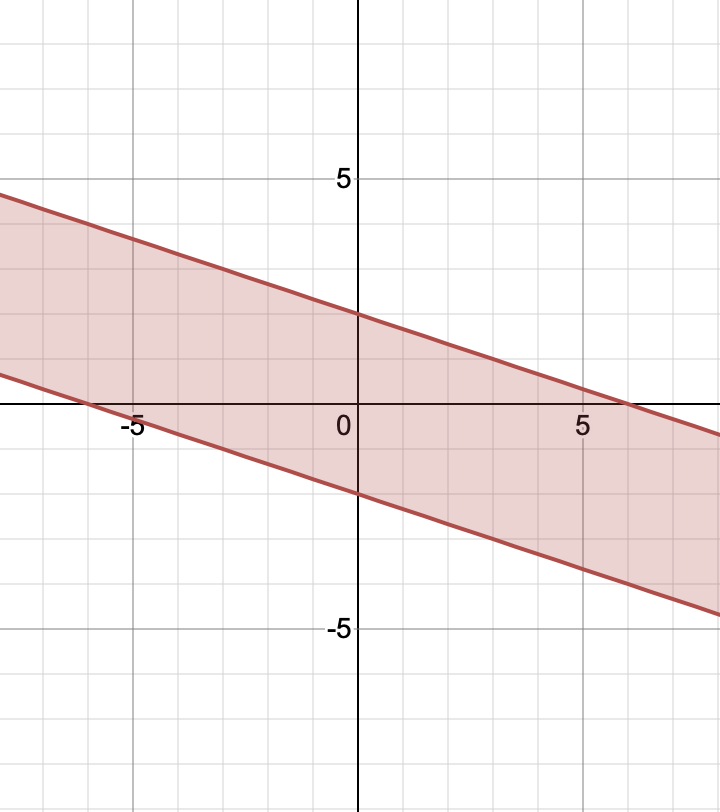
**1**

****

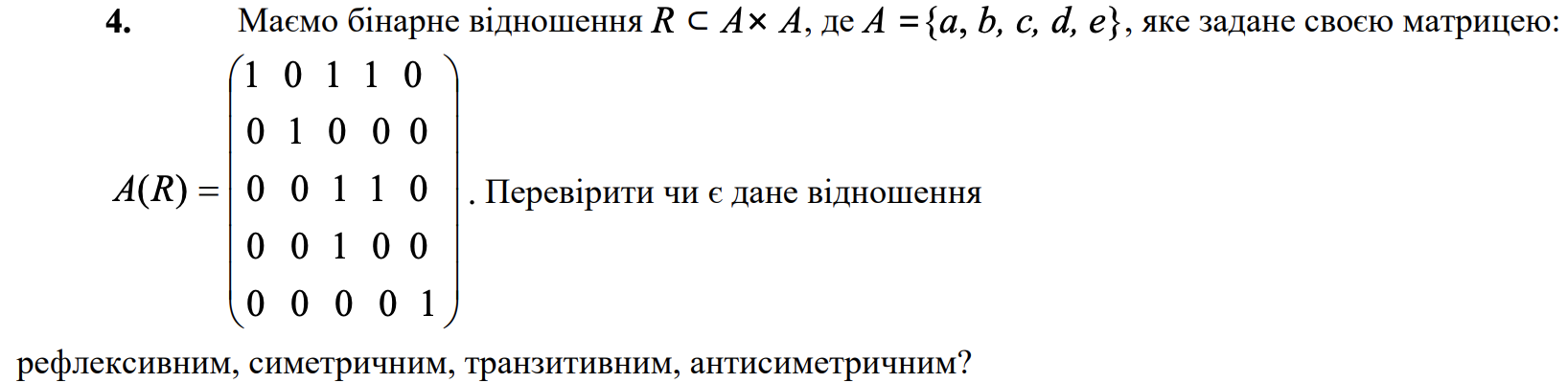
**2**

**3**

| x + 3x | <= 6

****

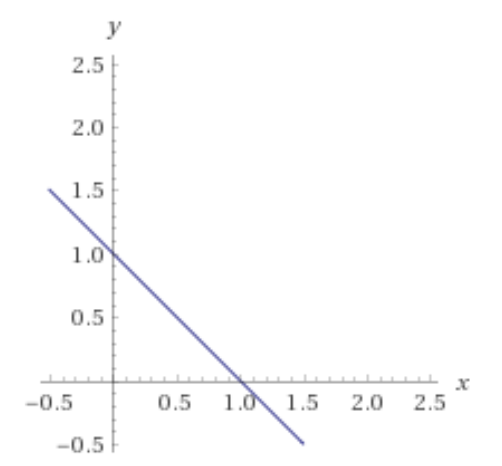
**4**

****

не є рефлективним, симетричним, антисиметричним

є транзитивним

**5**

****

Функціональне і бієктивне

x є  R

y є [0, +)

Додаток #2

#include <iostream>

using namespace std;

#define SIZE 100

int FuncReflexsive(int Matrix[][SIZE],int NumberOfElementsForA, bool\* Reflexsive)

{

int i = 0;

int k = 0;

while (i < NumberOfElementsForA-1 && k < NumberOfElementsForA-1)

{

if (Matrix[i][k]==1) {

if (Matrix[i][k] == Matrix[i + 1][k + 1])

{

\*Reflexsive = true;

}

else

{

\*Reflexsive = false;

break;

}

}

else{

\*Reflexsive = false;

break;

}

if (\*Reflexsive == false) {

break;

}

i++;

k++;

}

return \*Reflexsive;

}

int FuncSymmetrical(int Matrix[][SIZE], int NumberOfElementsForA, bool\* Symmetrical)

{

for (int i = 0; i < NumberOfElementsForA; i++)

{

for (int k = 0; k < NumberOfElementsForA; k++)

{

if (Matrix[i][k]==1) {

if (Matrix[i][k] == Matrix[k][i])

{

\*Symmetrical = true;

}

else

{

\*Symmetrical = false;

break;

}

} else {

\*Symmetrical = false;

break;

}

if( \*Symmetrical == false) {

break;

}

}

}

return \*Symmetrical;

}

int main()

{

int A[SIZE];

int B[SIZE];

int NumberOfElementsForA;

int NumberOfElementsForB;

cout << "A: " << endl;

cin >> NumberOfElementsForA;

cout << "B: " << endl;

cin >> NumberOfElementsForB;

cout << "Elements for A: " << endl;

for (int i = 0; i < NumberOfElementsForA; i++)

{

cin >> A[i];

}

cout << "Elements of A" << endl;

for (int i = 0; i < NumberOfElementsForA; i++)

{

cout << A[i] << endl;

}

cout << "Elements for B: " << endl;

for (int i = 0; i < NumberOfElementsForB; i++)

{

cin >> B[i];

}

cout << "Elements of B: " << endl;

for (int i = 0; i < NumberOfElementsForB; i++)

{

cout << B[i] << endl;

}

int Matrix[SIZE][SIZE];

int TrueOrFalse;

for (int i = 0; i < NumberOfElementsForA; i++)

{

for (int k = 0; k < NumberOfElementsForB; k++)

{

if (abs(B[k]-A[i]) < 2)

{

TrueOrFalse = 1;

}

else

{

TrueOrFalse = 0;

}

Matrix[i][k] = TrueOrFalse;

}

}

cout << "Matrix of relations" << endl;

for (int i = 0; i < NumberOfElementsForA; i++)

{

for (int k = 0; k < NumberOfElementsForB; k++)

{

cout << Matrix[i][k] << " ";

}

cout << endl<<endl;

}

if (NumberOfElementsForA != NumberOfElementsForB)

{

cout << "The matrix must be square in order to indicate the type of relation"<< endl;

return 0;

}

bool Reflexsive = false;

FuncReflexsive(Matrix, NumberOfElementsForA, &Reflexsive);

cout <<"Reflexsive "<< Reflexsive<< endl;

bool Symmetrical = false;

FuncSymmetrical(Matrix, NumberOfElementsForA, &Symmetrical);

cout<< "Symmetrical " << Symmetrical << endl;

}