МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТІ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”

Кафедра систем штучного інтелекту

Лабораторна робота №3

з дисципліни

«Дискретна математика»

Виконав:

студент групи КН-113

Зварич Адріана

Викладач: Мельникова Н.І.

Львів – 2019 р.

Тема роботи:

Побудова матриці бінарного відношення.

Мета роботи:

Набуття практичних вмінь та навичок при побудові матриць бінарних відношень та визначені їх типів.

Теоретичні відомості:

Декартів добуток множин A і B (позначається A×B) – це множина всіх упорядкованих пар елементів (a, b), де a∈A, b∈B. При цьому вважається, що (a1,b1) = (a2,b2) тоді і тільки тоді, коли a1 = a2 , b1 = b2.

Потужність декартова добутку дорівнює |A×B|= |A| × |B|.

Бінарним відношенням R називається підмножина декартового добутку A×B (тобто R ⊂ A×B). Якщо пара (a,b) належить відношенню R , то пишуть (a, b)∈R , або aRb.

Областю визначення бінарного відношення R⊂X×Y називається множина δ_R = {x| ∃y (x, y)∈R} , а областю значень – множина ρ_R= {y| ∃x (x, y)∈R} (∃- існує).

Для скінчених множин бінарне відношення R ⊂ A×B зручно задавати за допомогою матриці відношення $R_{m\times n} = (r_{ij})$, де m = |A| , а n = |B| .

Види бінарних відношень.

Нехай задано бінарне відношення R на множині A^2 : R ⊆ A× A= {(a, b)| a∈ A, b∈ A} .

1. Бінарне відношення R на множині A називається **рефлексивним**, якщо для будь якого a ∈ A виконується aRa , тобто (a,a)∈R . Головна діагональ матриці рефлексивного відношення складається з одиниць. Граф рефлексивного відношення обов'язково має петлі у кожній вершині.

2. Бінарне відношення R на множині A називається **антирефлексивним**, якщо для будь якого a ∈ A не виконується aRa , тобто (a,a)∉ R . Головна діагональ матриці антирефлексивного відношення складається з нулів. Граф антирефлексивного відношення не має петель.

3. Бінарне відношення R на множині A називається **симетричним**, якщо для будь яких a,b∈ A з aRb слідує bRa , тобто якщо (a,b)∈R то і (b,a)∈ R . Матриця симетричного відношення симетрична відносно головної діагоналі. Граф симетричного відношення не є орієнтованим.

4. Бінарне відношення R на множині A називається **антисиметричним**, якщо для будь яких a,b∈ A з aRb та bRa слідує що a = b . Тобто якщо (a,b)∈R

і (b,a)∈ R , то a = b . Матриця антисиметричного відношення не має жодної пари одиниць, які знаходяться на симетричних місцях по відношенню до головної діагоналі. У графа антисиметричного відношення вершини з'єднуються тільки однією напрямною дугою.

5. Бінарне відношення R на множині А називається **транзитивним**, якщо для будь яких a, b, c∈A з aRb та bRc слідує, що aRc . Тобто якщо (a,b)∈R і (b,c)∈ R, то (a,c)∈ R . Матриця транзитивного відношення характеризується тим, що якщо елемент матриці σ_{ij} = 1 та σ_{jm} =1, то обов'язково σ_{im}=1. Граф транзитивного відношення такий, що якщо з'єднані дугами, наприклад, перша-друга та друга-третя вершини, то обов'язково є дуга з першої в третю вершину.

6. Бінарне відношення R на множині А називається **антитранзитивним**, якщо для будь яких a, b, c∈ A з aRb та bRc слідує що не виконується aRc . Тобто якщо (a, b)∈R і (b, c)∈ R, то (a, c)∉ R . Матриця антитранзитивного відношення характеризується тим, що якщо елемент матриці σ_{ij} = 1 та σ_{jm}=1, то обов'язково σ_{im} =0. Граф транзитивного відношення такий, що якщо з'єднані дугами, наприклад, перша-друга та друга-третя вершини, то обов'язково немає дуги з першої в третю вершину.

Види функціональних відношень

1. Функція називається **ін'єктивною** (ін'єкцією), якщо з умови f (x1) = f (x2) слідує, що x1 = x2 для будь-яких x1, x2 ∈ X . Функція ін'єктивна тоді і тільки тоді, коли для будь-яких x1, x2 ∈ X якщо x1 ≠ x2 , то f (x1) ≠ f (x2), тобто для різних аргументів функція f приймає різні значення.
2. Функція називається **сюр'єктивною** (сюр'єкцією), якщо для кожного y*∈Y знайдеться такий x*∈ X , що y* = f (x*) .
3. Функція називається **бієктивною** (бієкцією), якщо вона ін'єктивна та сюр'єктивна одночасно. Таку функцію ще називають взаємно-однозначним відображенням.

Варіант 9

Завдання 1. Чи є вірною рівність A×(B∪C) = (A×B)∪(A×C)

Нехай (x,y)∈A×(B∪C), тоді: x∈A, y∈(B∪C)=x∈A, y∈B або y∈C=(x,y)∈
∈(A×B) або (x,y)∈(A×C)=(x,y)∈(A×B)∪(A×C)

Отже, рівність A×(B∪C) = (A×B)∪(A×C) є вірною.

Завдання 2. Знайти матрицю відношень R⊂M×2^M:

R = {(x,y)|x∈M&y⊂M&|y|-1=x}, де M={x|x∈Z |x-1|<2}, Z-множина цілих чисел.

M={0,1,2};

2^M={ ,{0},{1},{2},{0,1},{0,2},{1,2},{0,1,2}};

M/2^M	∅	{0}	{1}	{2}	{0,1}	{0,2}	{1,2}	{0,1,2}
0	0	1	1	1	0	0	0	0
1	0	0	0	0	1	1	1	0
2	0	0	0	0	0	0	0	1

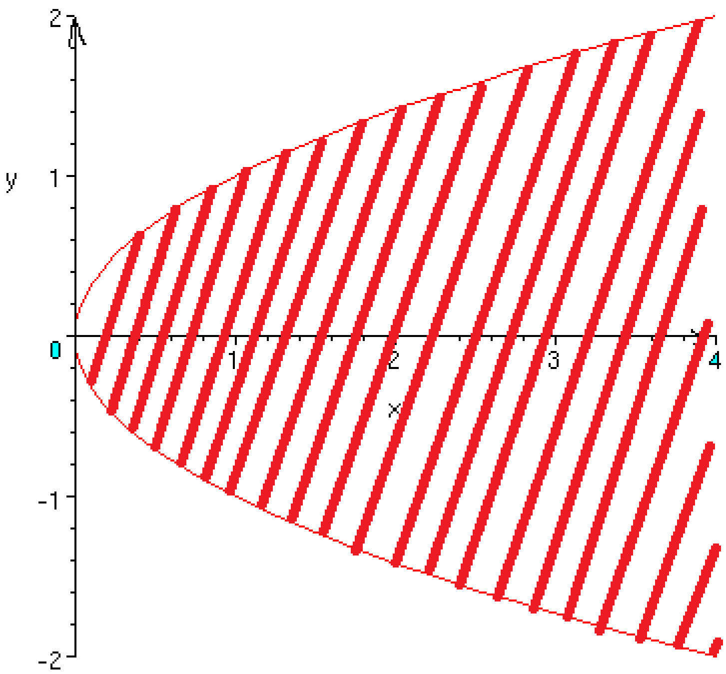
Завдання 3. Зобразити відношення графічно: α={(x,y)|(x,y)∈R^2&x-y^2>0}, де R-множина дійсних чисел.

x-y^2>0

x>y^2

y<√x   y>-√x

Графік виглядатиме так:



Завдання 4. Навести приклад бінарного відношення R⊂A×A, де A={a, b, c, d, e}, яке є рефлексивне, антисиметричне, нетранзитивне та побудувати його матрицю.

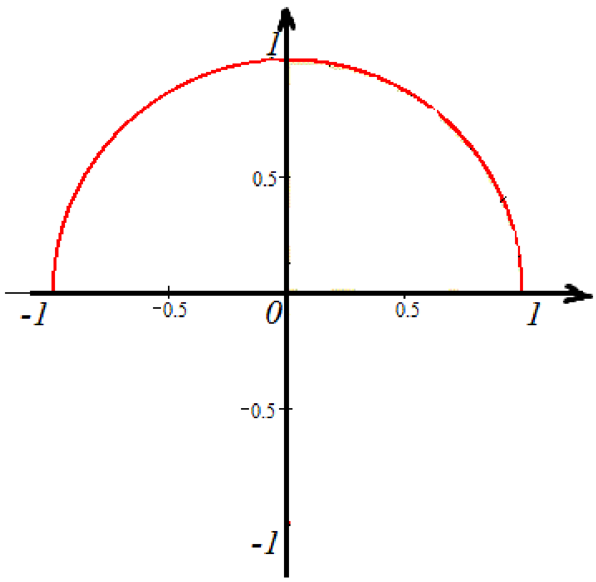
Дана матриця буде виглядати так:

$$R(A)=\begin{pmatrix} 1\,0\,0\,0\,0 \\ 1\,1\,0\,0\,0 \\ 1\,1\,1\,0\,0 \\ 1\,1\,1\,1\,0 \\ 1\,1\,1\,1\,1 \end{pmatrix}$$

Завдання 5. Визначити множину (якщо це можливо), на якій дане відношення є: а) функціональним; б) бієктивним:

$\alpha=\{(x,y)|(x,y)\in R^2 \& y=\sqrt{1-x^2}\}$

Зобразимо рівняння графічно:



Отже, відношення є функціональним(кожному х відповідає лише одне значення у) на множині [-1;1], а бієктивним(певному значенню у відповідає певне значення х, і певному значенню х відповідає певне значення у) на множині $[-1;0)_{\vee}(0;1]$;

Завдання №2: Написати програму, яка знаходить матриць бінарного відношення ρ⊂A×B , заданого на двох числових множинах. Реалізувати введення цих множин, та виведення на екран матриці відношення. Перевірити програмно якого типу є задане відношення. Навести різні варіанти тестових прикладів.

Програма:

```
#include <iostream>

using namespace std;

void addElem(int *arr, int n) {
    cout<<" Write nums "<< endl;
    for (int i=0; i<n; i++) {
    cin >> arr[i];

}
    for (int i=0; i<n; i++){
        cout<< arr[i] <<" | ";
    }
}

void coutElems(int **arr, int x, int y) {
    for (int i=0; i<x; i++) {
        for(int j=0; j<y; j++) {
            cout<< arr[i][j] <<" ";
        }
        cout<<endl;
    }
}

void reflex(int** p, int x) {
    bool reflex,antireflex;

         for (int i=0; i<x; i++) {
                for(int j=0; j<x; j++){
            if(i==j&&p[i][j]==1) {
                reflex = true;
            }
            else{
                reflex = false;
                break;
            }
         }
    }
    if(reflex==false){
        for(int i=0; i<x; i++) {
                for(int j=0; j<x; j++){
            if(i==j && p[i][j]==0) {
                antireflex = true;
            }
            else{
                antireflex = false;
                break;
            }
        }
    }
    cout<<"Reflex = "<< reflex <<endl;
    cout<<"Antireflex = "<< antireflex <<endl;
}
}
```

```
void symetr(int** p, int x) {
    bool symetr, asymetr, antisymetr;

    for(int i=0; i<x; i++) {
        for(int j=0; j<x;j++) {
            if(p[i][j]==p[j][i]) {
                symetr=true;
            }
            else{
                symetr=false;
                goto point;
            }
        }
    }
    point:;

    for(int i=0; i<x;i++) {
        for(int j=0; j<x; j++) {
            if(i != j &&(p[i][j]==1&&p[j][i]==0)||(p[i][j]==0&&p[j][i]==1)||(p[i][j]==0&& p[j][i]==0)) {
                asymetr=true;
            }
            else if(i==j){
                continue;
            }
            else{
                asymetr=false;
                goto point1;
            }
        }
    point1:;

    for (int i=0; i<x; i++) {
        for (int j=0; j<x; j++) {
            if(i!=j && p[i][j]!=p[j][i]) {
                antisymetr=true;
            }
            else if(i==j){
                continue;
            }
            else {
                antisymetr=false;
                goto point2;
            }
        }
    }
    point2:;
    cout<<"Symetric ="<<symetr<<endl;
    cout<<"Asymetric ="<<asymetr<<endl;
    cout<<"Antisymetric ="<<antisymetr<<endl;
}
void trans(int**p, int x) {
    bool trans, antitrans;

      for (int i=0; i<x; i++) {
        for (int j=0; j<x; j++) {
            for(int k=0; k<x; k++){
                if(p[i][j]==1 && p[i][k]==1 && p[j][k]==1) {
                    trans=true;
                }
```

```
                else if(p[i][j]==0 || p[j][k]==0){
                    continue;
                }
                else{
                    trans=false;
                    goto point;
                }
            }
        }
    }
    point:;
    for(int i=0; i<x; i++){
        for(int j=0; j<x; j++){
            for(int k=0; k<x; k++){
                if(p[i][j]==1 && p[j][k]==1 && p[i][k]==0){
                    antitrans=true;
                }
                else{
                    antitrans=false;
                    goto point1;
                }
            }
        }
    }
    point1:;

    cout<<"Trans = "<< trans <<endl;
    cout<<"Antitrans = "<< antitrans <<endl;
}
int main()
{

    int x;
    cout<<" Insert length plural " << endl;

    cin>>x;


    int* M = new int[x];
    addElem(M, x);
    int* N = new int[x];
    addElem(N, x);

    int** p;
    p = new int* [x];
    for (int i=0; i<x; i++) {
        p[i] = new int[i];
    }

    for (int i=0; i<x; i++) {
        for(int j=0; j<x; j++) {
            if((M[i]*M[i])<N[j]) {
                p[i][j] = 1;
            }
            else {
                p[i][j] = 0;
            }
        }
    }
```

```cpp

    cout<<endl<<endl<<"matrix"<<endl;

    coutElems(p, x, x);

    reflex(p, x);
    symetr(p, x);
    trans(p, x);
}

```

Результат програми:

```
 Insert length plural
4
 Write nums
1 2 3 4
1 ¦ 2 ¦ 3 ¦ 4 ¦  Write nums
3 5 7 2
3 ¦ 5 ¦ 7 ¦ 2 ¦

matrix
1 1 1 1
0 1 1 0
0 0 0 0
0 0 0 0
Reflex = 0
Antireflex = 0
Symetric =0
Asymetric =1
Antisymetric =0
Trans = 1
Antitrans = 0

Process returned 0 (0x0)   execution time : 8.577 s
Press any key to continue.
```

Висновки: на цій лабораторній роботі ми набули практичних вмінь та навичок при побудові матриць бінарних відношень та визначенні їх типів.