МІНІСТЕРСТВО ОСВІТИ І НАУКИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Інститут комп'ютерних наук та інформаційних технологій

Кафедра систем штучного інтелекту



Лабораторна робота №3

з курсу “Дискретна математика ”

Виконав:  
 Збриський Костянтин

ст. гр. КН-110

Викладач:

Мельникова Н.І.

Львів – 2018

**Мета:** Набуття практичних вмінь та навичок при побудові матриць бінарних відношень та визначенні їхніх типів

1.1 Чи є вірною рівність?

(

Для перевірки, предикат множини

*}*

*Тому дане відношення не є вірним.*

1.2 Знайти матрицю відношення

*:*

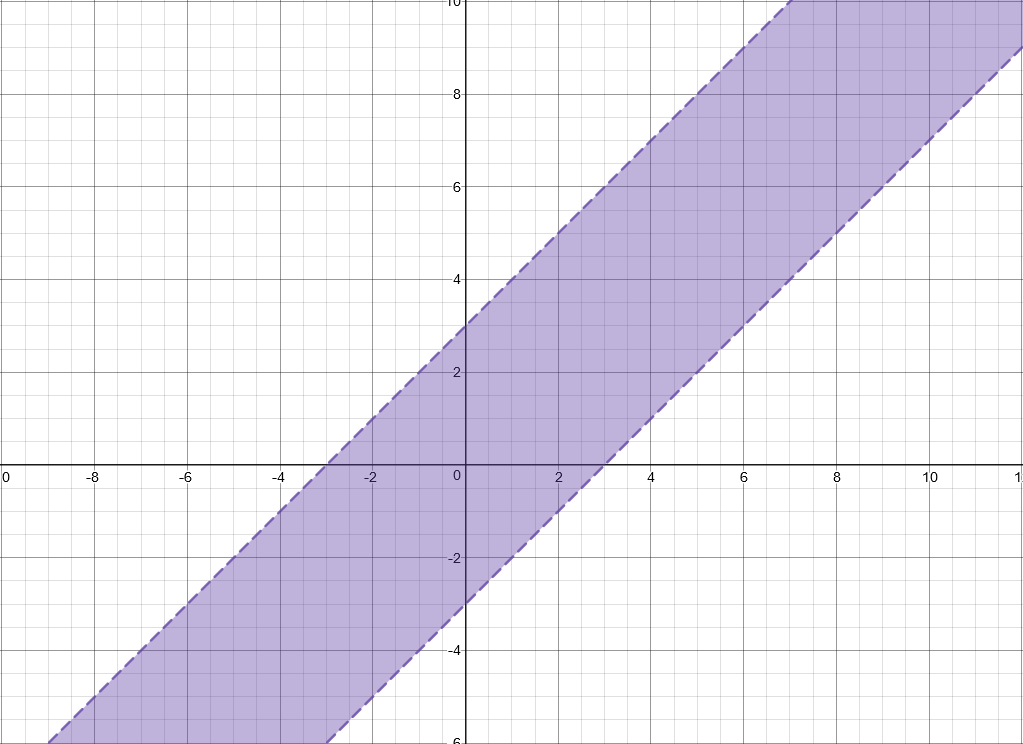
*, Z – множина цілих чисел*

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *2M*  *M* |  | *{-1}* | *{0}* | *{1}* | *{-1,0}* | *{0,1}* | *{-1,0}* | *{-1,0,1}* |
| *-1* | *0* | *1* | *1* | *1* | *1* | *1* | *1* | *1* |
| *0* | *0* | *1* | *0* | *1* | *1* | *1* | *1* | *1* |
| *1* | *0* | *0* | *0* | *0* | *0* | *0* | *0* | *0* |

1.3 Зобразити графічно відношення

, де R – множина дійсних чисел.

Рівняння



*1.4 Навести приклад бінарного відношення*

, де яке є нерефлексивне, симетричне, транзитивне, та побудувати його матрицю.

Данна матриця задовільняє умови завдання

1.5 Визначити множину на, якій дане відношення є:

а) функціональним; б) бієктивним:

.

Б) , тому множина на якій відношення буде функціональним є (&;

А) , тому множина на якій відношення буде бієктивним є

Код программи:

#include <iostream>

// initialise array

void initialise(int \*set, int set\_size) {

for (int i = 0; i < set\_size; i++) {

set[i] = 322;

}

}

//check if element from set is unic

bool is\_unic(int \*set, int element) {

int i = 0;

while (true) {

if (set[i] == 322 ) {

return true;

}

else if (set[i] == element) {

return false;

}

i++;

}

}

// setting and filling with elements sets

void set\_set(int \* set) {

int size = 16;

initialise(set, size);

std::cout << " starting adding elemnts" << std::endl;

std::cout << "your elemnt should be unic, press esc to stop" << std::endl;

for (int i = 0; i < size;) {

int elm;

/\*printe(set);\*/

system("cls");

std::cout << "print element of set here(-7 to stop adding new numbers): ";

std::cin >> elm;

if (elm == -7) {

break;

}

else if (is\_unic(set, elm)) {

set[i] = elm;

i++;

}

else {

std::cout << "type another elemnt " << std::endl;

}

}

}

int length(int \*set) {

int i = 1;

if (set[0] == 322)

{

return 0;

}

while (true) {

if (set[i] == 322) {

return i;

}

i++;

}

}

// reflex check

void reflex\_check(int \*\* matrix, int len, int len2) {

if (len != len2) {

std::cout << "matrix is not reflexive" << std::endl;

return;

}

else {

int to\_compare = matrix[1][1];

for (int i = 1; i < len; i++) {

if (matrix[i][i] != to\_compare)

{

std::cout << "matrix isn't reflexive or antireflexive" << std::endl;

return;

}

}

if (to\_compare) {

std::cout << "matrix is reflexive" << std::endl;

}

else {

std::cout << "matrix is anti-reflexive";

}

}

}

void sym\_check(int \*\* matrix, int len, int len2) {

// sym\_check

if (len != len2) {

std::cout << "imposible to define symetricality" << std::endl;

return;

}

for (int i = 1; i < len; i++) {

for (int k = 1; k < len2; k++) {

if (matrix[i][k] != matrix[k][i]) {

std::cout << "matrix is not symetrical" << std::endl;

return;

}

}

}

std::cout << "matrix is symetrical" << std::endl;

return;

}

int tran\_check(int \*\*matrix, int len, int len2)

{

int k = len;

int transitive = 0;

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

{

for (int j = 0; j < len; j++)

{

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

{

if (matrix[i][j] == 1 && matrix[j][k] == 1 && matrix[i][k] == 1)

{

transitive = 1;

}

else

{

std::cout << "this matrix is antitransitive" << std::endl;

return 0;

}

}

}

}

if (transitive)

std::cout << "this matrix is transitive" << std::endl;

else

std::cout << "this natrix is not transitive" << std::endl;

return 1;

}

int main() {

// creating set objects

int arr[16];

int arr2[16];

// initialising objects

set\_set(arr);

int len = length(arr) + 1;

set\_set(arr2);

int len2 = length(arr2) + 1;

// creating dynamic matrix

int \*\* matrix = new int\*[len];

for (int i = 0; i < len; i++) {

matrix[i] = new int[len2];

}

// creating zeros

for (int i = 0; i < len; i++) {

for (int k = 0; k < len2; k++) {

int x = matrix[i][0];

int y = matrix[0][k];

matrix[i][k] = 0;

}

}

// filling with set numbers

for (int i = 1; i < len; i++) {

matrix[i][0] = arr[i - 1];

}

for (int i = 1; i < len2; i++) {

matrix[0][i] = arr2[i - 1];

}

// operation

for (int i = 1; i < len; i++) {

for (int k = 1; k < len2; k++) {

int x = matrix[i][0];

int y = matrix[0][k];

if ((2 \* x - y) < 3) {

matrix[i][k] = 1;

}

}

}

std::cout << std::endl << std::endl;

// out

for (int i = 0; i < len; i++) {

for (int k = 0; k < len2; k++) {

}

}

for (int i = 1; i < len; i++) {

if (matrix[0][i] == matrix[i][0]) {

matrix[i][i] = 1;

}

}

std::cout << std::endl << std::endl;

std::cout << "condition is 2a - b < 3" << std::endl;

// output

for (int i = 0; i < len; i++) {

for (int k = 0; k < len2; k++) {

std::cout << ' ' << matrix[i][k] << ' ';

}

std::cout << std::endl;

}

reflex\_check(matrix, len, len2);

sym\_check(matrix, len, len2);

tran\_check(matrix, len, len2);

system("pause");

}