Міністерство освіти і науки України

Нацональний університет «Львівська політехніка»

Кафедра систем штучного інтелекту

Лабораторна робота № 2

з дисципліни

**«Дискретна математика»**

Виконав:

Студент групи КН-114

**Кратко Денис**

Викладач:

**Мельникова Н.І.**

Львів – 2019

**Тема:** Моделювання основних операцій для числових множин

**Мета роботи:** Ознайомитись на практиці із основними поняттями теорії множин, навчитись будувати діаграми Ейлера-Венна операцій над множинами, використовувати закони алгебри множин, освоїти принцип включень-виключень для двох і трьох множин та комп’ютерне подання множин

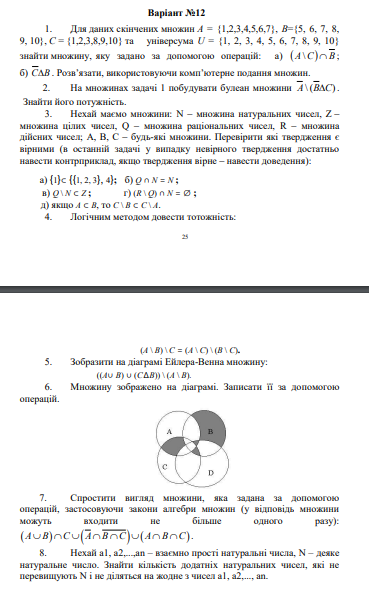
**Теоретичні відомості**

2.1. Основні поняття теорії множин. Операції над множинами Множина – це сукупність об’єктів, які називають елементами. Кажуть, що множина А є підмножиною множини S (цей факт позначають A ⊆ S , де ⊆ – знак нестрогого включення), якщо кожен її елемент автоматично є елементом множини S. Досить часто при цьому кажуть, що множина А міститься в множині S. Якщо A ⊆ S і S ≠ A , то A називають власною (строгою, істинною) підмножиною S (позначають A ⊂ S , де ⊂ – знак строгого включення). Дві множини А та S називаються рівними, якщо вони складаються з однакових елементів. У цьому випадку пишуть А=S. Якщо розглядувані множини є підмножинами деякої множини, то її називають універсумом або універсальною множиною і позначають літерою U (зауважимо, що універсальна множина існує не у всіх випадках). Множини як об’єкти можуть бути елементами інших множин, Множину, елементами якої є множини, інколи називають сімейством. Множину, елементами якої є всі підмножини множини А і тільки вони (включно з порожньою множиною та самою множиною А), називають булеаном або множиною-степенем множини А і позначають P(A). Потужністю скінченної множини А називають число її елементів, позначають |А|. Множина, яка не має жодного елемента, називається порожньою і позначається ∅. Вважається, що порожня множина є підмножиною будь-якої множини, а також A⊂A. 2 Множина всіх підмножин множини A називається булеаном і позначається P(A). Потужність скінченної множини дорівнює кількості її елементів, позначається A . Потужність порожньої множини дорівнює 0. Якщо A = n, то n P(A) = 2 . Приклад. {1, 4, 5}⊂ {−1, 0,1, 2, 3, 4, 5, 7}, але {1, 4, 5}∉{−1, 0,1, 2, 3, 4, 5, 7} Приклад. Знайти булеан множини A = {a, ,b c}. Розв’язання. Потужності множин A = 3 , P(A) = 8 . Булеан має вигляд P(A) = {∅,{a},{b},{c},{a,b},{a c, },{b c, },{a, b, c}}. Дві множини A і B рівні між собою, якщо A ⊂ B і B ⊂ A. Над множинами можна виконувати дії: об’єднання, переріз, доповнення, різницю, симетричну різницю, декартів добуток. Об’єднанням двох множин А і В (рис. 2.1, а) називають множину A∪ B ={x :(x∈ A) ∨ (x∈B)}. Перетином (перерізом) двох множин А і В (рис. 2.1, б) називають множину A∩ B ={x :(x∈ A) ∧ (x∈B)}. а) б) Рис. 2.1. Діаграми Ейлера-Венна об’єднання та перетину двох множин Різницею множин А та В (рис. 2.2, а) називають множину A \ B ={x :(x∈ A) ∧ (x∉ B)}. 3 Зазначимо, що A \ B = A ∩ B. Симетричною різницею множин А та В (рис. 2.2, а) називають множину AΔB ={x :((x∈ A) ∧ (x∉ B)) ∨ ((x∈B) ∧ (x∉ A))} . а) б) Рис. 2.2. Діаграма Ейлера-Венна різниці та симетричної різниці двох множин В означенні різниці не розглядають випадок B ⊂ A. Якщо B ⊂ A , то різницю A B\ називають доповненням множини В до множини А і позначають BA . Для підмножини А універсальної множини U можна розглядати доповнення А до U, тобто U \ A , її позначають A ={x : ¬(x∈ A)} ⇔ A ={x : x∉ A)} і називають доповненням множини А (рис. 2.3). Рис. 2.3. Діаграма Ейлера-Венна доповнення множини Пріоритет виконання операцій у спадному порядку – доповнення, переріз, об’єднання, різниця, симетрична різниця. 4 2.2. Закони алгебри множин Закони асоціативності A∪(B ∪C) = (A∪ B) ∪C A∩(B ∩C) = (A∩ B) ∩C Закони комутативності A∪B = B∪ A A∩B = B∩ A Закони тотожності A∪∅ = A A∩U = A Закони домінування A∪U =U A∩∅ = ∅ Закони ідемпотентності A∪ A= A A∩ A= A Закони дистрибутивності A∩(B ∪C) = (A∩ B) ∪(A∩C) A∪(B ∩C) = (A∪ B) ∩(A∪C) Закони поглинання (A∪ B)∩ A = A (A∩ B)∪ A = A Закони доповнення A∪ A =U A∩ A = ∅ U = ∅ ∅ =U A= A A= A Закони де Моргана (A∪ B) = A∩ B (A∩ B) = A∪ B Вивчення законів алгебри множин дозволяє зауважити, що кожна з тотожностей правої колонки може бути одержана з відповідної тотожності лівої шляхом заміни ∪ на ∩ , ∅ на U і навпаки. Таку відповідність тотожностей називають законом двоїстості, а відповідні тотожності – двоїстими одна одній. Використовуючи цей закон, можна обгрунтувати двоїсту тотожність, довівши пряму і обернувши операції. 2.3. Формули включень та виключень для двох і трьох множин. Комп’ютерне подання множин | A∪ B|=| A| +| B| −| A∩ B| , A∪ B ∪C = A + B +C − ( A∩ B + B ∩C + A∩C )+ A∩ B ∩C. Одним із найпоширеніших та найпростіших способів є подання множин за допомогою бітових рядків. Нехай універсальна множина U 5 містить п елементів. Упорядкуємо довільним способом елементи універсальної множини. Тоді U a a a a a ={ 1 2 3 1 , , ,... , . n n − } Множину A ⊂U зображають у комп’ютері рядком із 0 та 1 довжини п так: якщо ai ∈ A, то і-й біт дорівнює 1, якщо ai ∉ A, то і-й біт дорівнює 0. Такий рядок бітів називають характеристичним вектором підмножини А.

**Завдання (12 варіант)**

**Додаток 1**

**Умова**



Розв’язок

**Завдання 1**

A = {1, 2, 3, 4, 5, 6, 7}

B = {5, 6, 7, 8, 9, 10}

C = {1, 2, 3, 8, 9, 10}

U = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

A = (1111111000)  
C = (1110000111)  
 = (0001111000)  
B = (0000111111)  
 = (1111000000)  
 = (1111000000)

Отже,

C = (1110000111)  
 = (0001111000)  
B = (0000111111)  
 = (0001000111)

Отже,

**Завдання 2**

A = {1, 2, 3, 4, 5, 6, 7}

B = {5, 6, 7, 8, 9, 10}

C = {1, 2, 3, 8, 9, 10}

U = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

A = (1111111000)  
B = (0000111111)  
C = (1110000111)  
 = (1111000000)  
 = (0001000111)  
 = (0000000111)  
 = (0000000000) =

= {}

= 1

**Завдання 3**

а)

Строге включення () означає, що кожен елемент з A є елементом множини B і .

Однак множина B містить два елементи: множину і число 4, а число 1 не містить.

*Твердження неправильне.*

U

R

Q

Z

**N**

б)

Перетин заштриховано.

*Твердження правильне.*

U

R

Q

Z

**N**

Q\N

в)

*Твердження неправильне.*

г)

U

R

Q

Z

**N**

R\Q

не має перетину з N (з рисунка)

*Твердження правильне.*

д) Якщо , то

B

A

- C \ A

- C \ B

C

*Твердження правильне.*

**Завдання 4**

Перетворимо праву частину:

Тотожність доведено.

**Завдання 5**

-

-

-

-

- D

B

A

C

**Завдання 6**

**Завдання 7**

A

B

C

**Завдання 8**

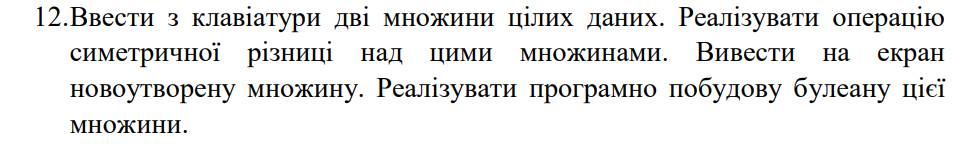
– взаємно прості натуральні числа

Не діляться ті числа, які не кратні будь-якому числу з B.

|A| чисел

Додаток 2

Умова



Розв’язок

// #include <bits/valarray\_before.h>  
#include <bits/stdc++.h>  
#define BLOCK\_SIZE 16  
  
using namespace std;  
  
typedef unsigned short int block;  
  
unsigned char get\_set\_bit(block\* st, unsigned long int bit\_number) {  
 return (st[bit\_number / BLOCK\_SIZE] >> (bit\_number % BLOCK\_SIZE)) % 2;  
};  
  
void add\_set\_bit(block\* st, unsigned long int bit\_number) {  
 st[bit\_number / BLOCK\_SIZE] |= 1 << (bit\_number % BLOCK\_SIZE);  
};  
  
unsigned char increment\_set(block\* st, unsigned long int more\_int\_number) {  
 (\*st)++;  
 if (!\*st) {  
 if (more\_int\_number) {  
 increment\_set(st + 1, more\_int\_number - 1);  
 };  
 return more\_int\_number;  
 };  
 return 1;  
};  
  
void print\_set(block\* st, long int start, long int stop) {  
 unsigned long int u\_length = stop - start;  
 unsigned long int counter = 0;  
 printf("{");  
 for (unsigned long int i = 0; i < u\_length; i++) {  
 if (get\_set\_bit(st, i)) {  
 counter++;  
 printf("%4d,", start + i);  
 };  
 };  
 printf("} (power: %d)\n", counter);  
};  
  
block\* prepare\_set(long int start, long int stop) {  
 return (block\*)calloc((stop - start) / BLOCK\_SIZE, sizeof(block));  
};  
  
void reset\_set(block\* st, long int start, long int stop) {  
 unsigned long int\_number = (stop - start) / BLOCK\_SIZE + 1;  
 for (unsigned long i = 0; i < int\_number; i++) {  
 st[i] = 0;  
 };  
};  
  
void add\_to\_set(block\* st, long int elem, long int start, long int stop) {  
 if (elem < stop) {  
 add\_set\_bit(st, elem - start);  
 };  
};  
  
block\* extension(block\* st, long int start, long int stop) {  
 unsigned long int int\_number = (stop - start) / BLOCK\_SIZE + 1;  
 block\* new\_set = prepare\_set(start, stop);  
 for (unsigned long int i = 0; i < int\_number; i++) {  
 new\_set[i] = ~st[i];  
 };  
 return new\_set;  
};  
  
block\* intersection(block\* a, block\* b, long int start, long int stop) {  
 unsigned long int int\_number = (stop - start) / BLOCK\_SIZE + 1;  
 block\* new\_set = prepare\_set(start, stop);  
 for (unsigned long int i = 0; i < int\_number; i++) {  
 new\_set[i] = a[i] & b[i];  
 };  
 return new\_set;  
};  
  
block\* symmetric\_difference(block\* a, block\* b, long int start, long int stop) {  
 block \*new\_set = prepare\_set(start, stop);  
 unsigned long int int\_number = (stop - start) / BLOCK\_SIZE + 1;  
 for (unsigned long int i = 0; i < int\_number; i++) {  
 new\_set[i] = a[i] ^ b[i];  
 };  
 return new\_set;  
};  
  
unsigned char is\_nonzero(block\* st, long int start, long int stop) {  
 unsigned long int int\_number = (stop - start) / BLOCK\_SIZE + 1;  
 for (unsigned long int i = 0; i < int\_number; i++) {  
 if (st[i]) {  
 return 1;  
 };  
 };  
 return 0;  
};  
  
long int\* prepare\_arr() {  
 return (long int\*)calloc(0, sizeof(long int));  
};  
  
unsigned long int input\_arr(long int\* arr, long int\* min\_value\_pointer, long int\* max\_value\_pointer) {  
 // повертає нову довжину  
 unsigned long int n = 0;  
 char space;  
 long int\* min\_p = arr;  
 long int\* max\_p = arr;  
  
 do {  
 n++;  
 realloc(arr, n);  
 scanf("%d%c", arr + (n - 1), &space);  
 printf("");  
 if (arr[n - 1] < \*min\_p) {  
 min\_p = arr + (n - 1);  
 };  
 if (arr[n - 1] > \*max\_p) {  
 max\_p = arr + (n - 1);  
 };  
 } while (space != '\n');  
  
 \*min\_value\_pointer = \*min\_p;  
 \*max\_value\_pointer = \*max\_p;  
  
 return n;  
};  
  
int main() {  
 // Вводимо через пробіл першу множину цілих чисел  
  
 printf("Enter first set:\n");  
 long int\* arr\_a = prepare\_arr();  
 long int min\_in\_a = 0;  
 long int max\_in\_a = 0;  
 unsigned long int arr\_a\_len = input\_arr(arr\_a, &min\_in\_a, &max\_in\_a);  
  
 printf("Enter second set:\n");  
 long int\* arr\_b = prepare\_arr();  
 long int min\_in\_b = 0;  
 long int max\_in\_b = 0;  
 unsigned long int arr\_b\_len = input\_arr(arr\_b, &min\_in\_b, &max\_in\_b);  
  
 long int global\_start = min(min\_in\_a, min\_in\_b);  
 long int global\_stop = max(max\_in\_a, max\_in\_b) + 1;  
  
 // set 1  
 block\* set1 = prepare\_set(global\_start, global\_stop);  
 reset\_set(set1, global\_start, global\_stop);  
 for (unsigned long int i = 0; i < arr\_a\_len; i++) {  
 add\_to\_set(set1, arr\_a[i], global\_start, global\_stop);  
 };  
  
 // set 2  
 block\* set2 = prepare\_set(global\_start, global\_stop);  
 reset\_set(set2, global\_start, global\_stop);  
 for (unsigned long int i = 0; i < arr\_b\_len; i++) {  
 add\_to\_set(set2, arr\_b[i], global\_start, global\_stop);  
 };  
  
 block\* result = symmetric\_difference(set1, set2, global\_start, global\_stop);  
 printf("Symmetric difference is:\n");  
 print\_set(result, global\_start, global\_stop);  
  
 printf("\n");  
 printf("Bulean is:\n");  
  
 // виводимо булеан  
 unsigned long int u\_length = global\_stop - global\_start;  
 unsigned long int more\_int\_number = u\_length / BLOCK\_SIZE;  
 unsigned char t = 1;  
 block\* result\_extension = extension(result, global\_start, global\_stop);  
 block\* st = prepare\_set(global\_start, global\_stop);  
 unsigned long int counter = 0;  
 for (reset\_set(st, global\_start, global\_stop); t; t = increment\_set(st, more\_int\_number)) {  
 if (!is\_nonzero(intersection(result\_extension, st, global\_start, global\_stop), global\_start, global\_stop)) {  
 // belongs to bulean  
 counter++;  
 printf("%4d: ", counter);  
 print\_set(st, global\_start, global\_stop);  
 };  
 };  
  
 return 0;  
};

Результат виконання програми

Enter first set:

56 43 -6 1

Enter second set:

4 0 90 6 -3

Symmetric difference is:

{ -6, -3, 0, 1, 4, 6, 43, 56, 90,} (power: 9)

Bulean is:

1: {} (power: 0)

2: { -6,} (power: 1)

3: { -3,} (power: 1)

4: { -6, -3,} (power: 2)

5: { 0,} (power: 1)

6: { -6, 0,} (power: 2)

7: { -3, 0,} (power: 2)

8: { -6, -3, 0,} (power: 3)

9: { 1,} (power: 1)

10: { -6, 1,} (power: 2)

11: { -3, 1,} (power: 2)

12: { -6, -3, 1,} (power: 3)

13: { 0, 1,} (power: 2)

14: { -6, 0, 1,} (power: 3)

15: { -3, 0, 1,} (power: 3)

16: { -6, -3, 0, 1,} (power: 4)

17: { 4,} (power: 1)

18: { -6, 4,} (power: 2)

19: { -3, 4,} (power: 2)

20: { -6, -3, 4,} (power: 3)

21: { 0, 4,} (power: 2)

22: { -6, 0, 4,} (power: 3)

23: { -3, 0, 4,} (power: 3)

24: { -6, -3, 0, 4,} (power: 4)

25: { 1, 4,} (power: 2)

26: { -6, 1, 4,} (power: 3)

27: { -3, 1, 4,} (power: 3)

28: { -6, -3, 1, 4,} (power: 4)

29: { 0, 1, 4,} (power: 3)

30: { -6, 0, 1, 4,} (power: 4)

31: { -3, 0, 1, 4,} (power: 4)

32: { -6, -3, 0, 1, 4,} (power: 5)

33: { 6,} (power: 1)

34: { -6, 6,} (power: 2)

35: { -3, 6,} (power: 2)

36: { -6, -3, 6,} (power: 3)

37: { 0, 6,} (power: 2)

38: { -6, 0, 6,} (power: 3)

39: { -3, 0, 6,} (power: 3)

40: { -6, -3, 0, 6,} (power: 4)

41: { 1, 6,} (power: 2)

42: { -6, 1, 6,} (power: 3)

43: { -3, 1, 6,} (power: 3)

44: { -6, -3, 1, 6,} (power: 4)

45: { 0, 1, 6,} (power: 3)

46: { -6, 0, 1, 6,} (power: 4)

47: { -3, 0, 1, 6,} (power: 4)

48: { -6, -3, 0, 1, 6,} (power: 5)

49: { 4, 6,} (power: 2)

50: { -6, 4, 6,} (power: 3)

51: { -3, 4, 6,} (power: 3)

52: { -6, -3, 4, 6,} (power: 4)

53: { 0, 4, 6,} (power: 3)

54: { -6, 0, 4, 6,} (power: 4)

55: { -3, 0, 4, 6,} (power: 4)

56: { -6, -3, 0, 4, 6,} (power: 5)

57: { 1, 4, 6,} (power: 3)

58: { -6, 1, 4, 6,} (power: 4)

59: { -3, 1, 4, 6,} (power: 4)

60: { -6, -3, 1, 4, 6,} (power: 5)

61: { 0, 1, 4, 6,} (power: 4)

62: { -6, 0, 1, 4, 6,} (power: 5)

63: { -3, 0, 1, 4, 6,} (power: 5)

64: { -6, -3, 0, 1, 4, 6,} (power: 6)

Enter first set:

5 4 3 9

Enter second set:

3 1 6 3 9

Symmetric difference is:

{ 1, 4, 5, 6,} (power: 4)

Bulean is:

1: {} (power: 0)

2: { 1,} (power: 1)

3: { 4,} (power: 1)

4: { 1, 4,} (power: 2)

5: { 5,} (power: 1)

6: { 1, 5,} (power: 2)

7: { 4, 5,} (power: 2)

8: { 1, 4, 5,} (power: 3)

9: { 6,} (power: 1)

10: { 1, 6,} (power: 2)

11: { 4, 6,} (power: 2)

12: { 1, 4, 6,} (power: 3)

13: { 5, 6,} (power: 2)

14: { 1, 5, 6,} (power: 3)

15: { 4, 5, 6,} (power: 3)

16: { 1, 4, 5, 6,} (power: 4)

Process finished with exit code 0