Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

**Дискретна математика**

**Лабораторна робота №1**

«Множини: основні властивості та операції над ними, діаграми Венна»

Виконав:

студент групи ІО-92

*Михайлюк Євгеній Сергійович*

Залікова книжка № ІО-9219

Перевірив:

Новотарський М.А.

Київ – 2019

**1. Тема:**

Множини: основні властивості та операції над ними, діаграми Венна.

**2. Мета:**

Вивчити основні аксіоми, закони і теореми теорії множин, навчитися застосовувати їх на практиці. Обчислити логічний вираз шляхом послідовного застосування операцій над множинами.

**3. Загальне завдання:**

1. Виконати лабораторну роботу з застосуванням графічного інтерфейсу.

2. Спростити логічний вираз з застосуванням тотожностей алгебри множин.

3. В окремому модулі написати функцію обчислення початкового логічного виразу (1), вибраного відповідно до індивідуального варіанта.

4. В окремому модулі написати функцію обчислення спрощеного логічного виразу.

5. В окремому модулі написати функцію виконання логічної операції (2), вибраної відповідно до індивідуального варіанта.

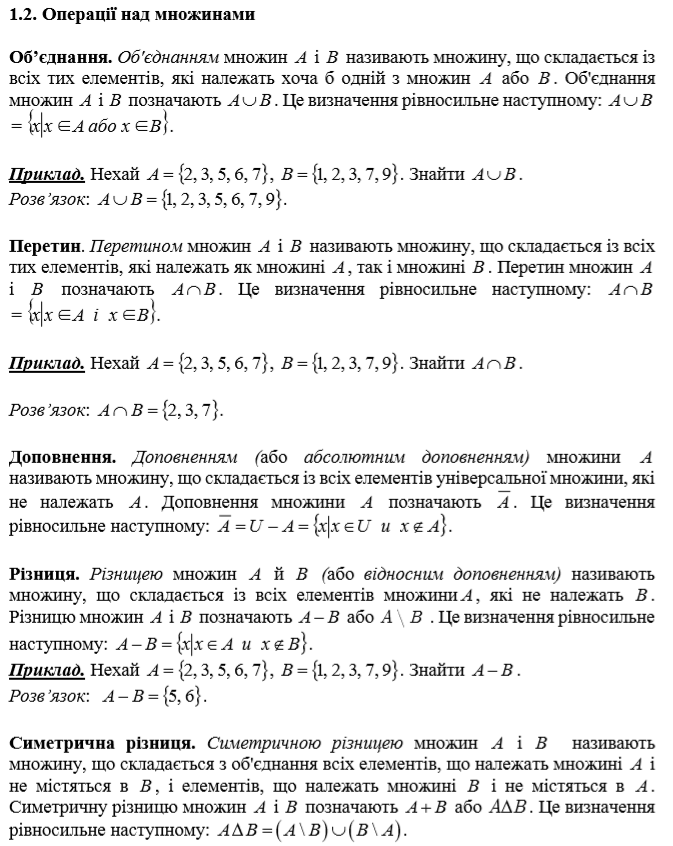
6. В окремому модулі виконати порівняння результатів:

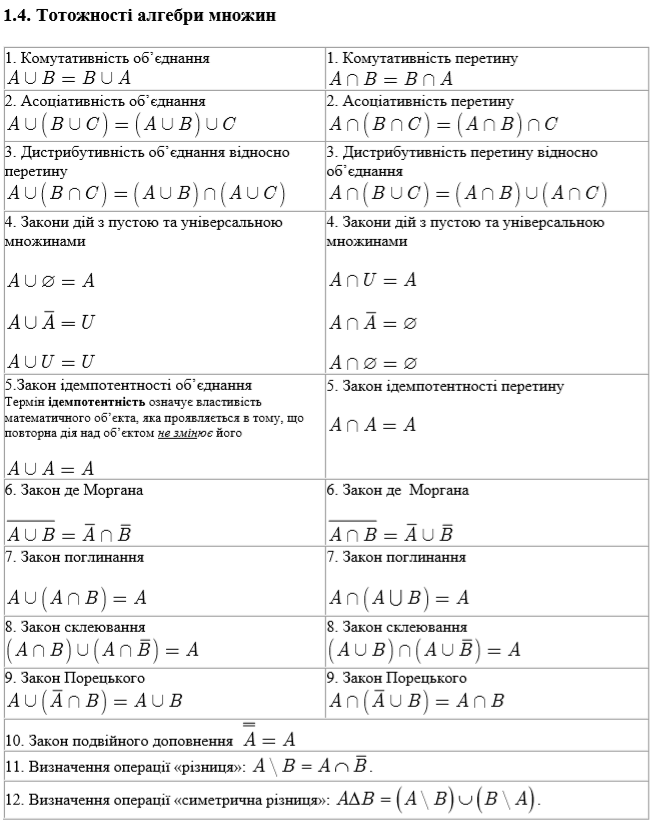
А) обчислення початкового та спрощеного виразу

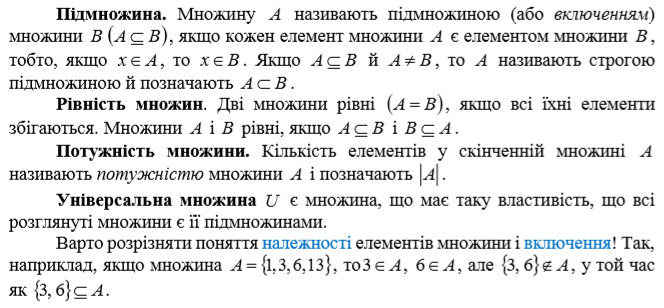
Б) виконання логічної операції Вашою функцією та відповідною стандартною логічною операцією або функцією С#.

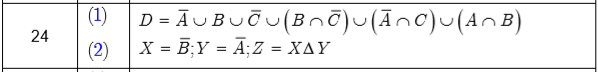
**4. Теоретичні дані:**

**Множина** – є сукупність визначених об'єктів, різних між собою, об’єднаних за певною ознакою чи властивістю. Множини позначають великими латинськими буквами. Об'єкти, що складають множини, називають елементами і позначають малими буквами латинського алфавіту. Якщо множина не містить жодного елемента, її називають порожньою і позначають Ø.





**4. Завдання**

****

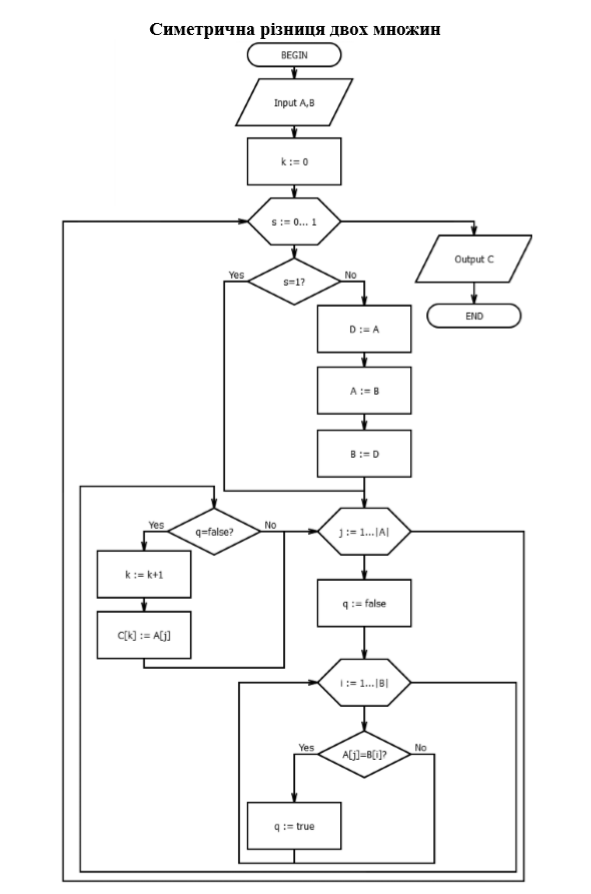
**5. Спрощення виразу**

1. Для наочності змінимо послідовність

1. Виділимо необхідні частини виразу

1. Використаємо властивість поглинання(табл. 1.4 п.7)

**6. Блок-схема алгоритму для обрахування множини Z**

****

**7. Код програми для обрахування множини D по початковій формулі**

private void nextStep\_Click(object sender, EventArgs e)

{

switch (ind)

{

case 0:

setNotA.ExceptWith(Main.settA);

foreach (int i in setNotA) { notABox.Text += i.ToString() + ", "; }

ind++;

break;

case 1:

setNotC.ExceptWith(Main.settC);

foreach (int i in setNotC) { notCBox.Text += i.ToString() + ", "; }

ind++;

break;

case 2:

set1.IntersectWith(setNotC);

foreach (int i in set1) { textBox1.Text += i.ToString() + ", "; }

ind++;

break;

case 3:

set2.IntersectWith(setNotA);

foreach (int i in set2) { textBox2.Text += i.ToString() + ", "; }

ind++;

break;

case 4:

set3.IntersectWith(Main.settB);

foreach (int i in set3) { textBox3.Text += i.ToString() + ", "; }

ind++;

break;

case 5:

set4.UnionWith(setNotA);

foreach (int i in set4) { textBox4.Text += i.ToString() + ", "; }

ind++;

break;

case 6:

set4.UnionWith(setNotC);

foreach (int i in set4) { textBox5.Text += i.ToString() + ", "; }

ind++;

break;

case 7:

set1.UnionWith(set2);

foreach (int i in set1) { textBox6.Text += i.ToString() + ", "; }

ind++;

break;

case 8:

set4.UnionWith(set1);

foreach (int i in set4) { textBox7.Text += i.ToString() + ", "; }

ind++;

break;

case 9:

set4.UnionWith(set3);

foreach (int i in set4) { setDBox.Text += i.ToString() + ", "; }

nextStep.Enabled = false;

saveButton.Visible = true;

saveButton.Focus();

break;

}

}

**8. Код програми для обрахування множини D по скороченій формулі**

private void nextStep\_Click(object sender, EventArgs e)

{

switch (ind)

{

case 0:

setNotA.ExceptWith(Main.settA);

foreach (int i in setNotA) { notABox.Text += i.ToString() + ", "; }

ind++;

break;

case 1:

setNotC.ExceptWith(Main.settC);

foreach (int i in setNotC) { notCBox.Text += i.ToString() + ", "; }

ind++;

break;

case 2:

setNotA.UnionWith(setNotC);

foreach (int i in setNotA) { textBox1.Text += i.ToString() + ", "; }

ind++;

break;

case 3:

setNotA.UnionWith(Main.settB);

foreach (int i in setNotA) { setDBox.Text += i.ToString() + ", "; }

nextStep.Enabled = false;

saveButton.Visible = true;

saveButton.Focus();

break;

}

}

**9. Код програми для обрахування множини Z**

for(int s = 0; s <=1; s++)

{

if(s == 1)

{

HashSet<int> setTemp = new HashSet<int>(setX);

setX.Clear();

setX.UnionWith(setY);

setY.Clear();

setY.UnionWith(setTemp);

}

foreach(int i in setX)

{

bool q = true;

foreach(int j in setY)

{

if(i == j)

{

q = false;

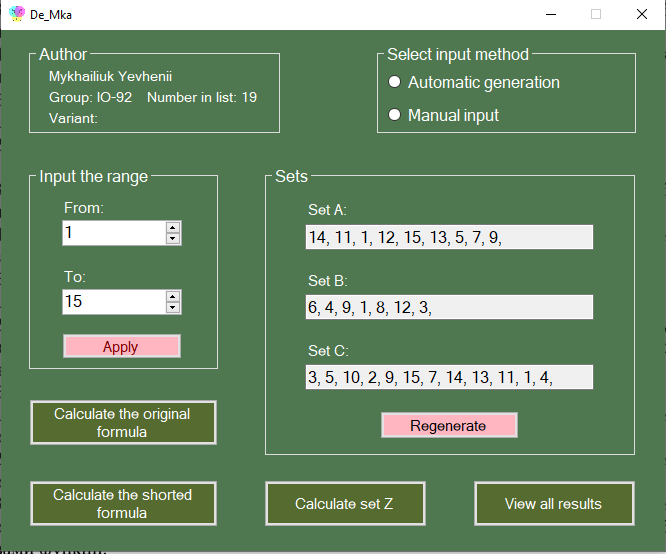
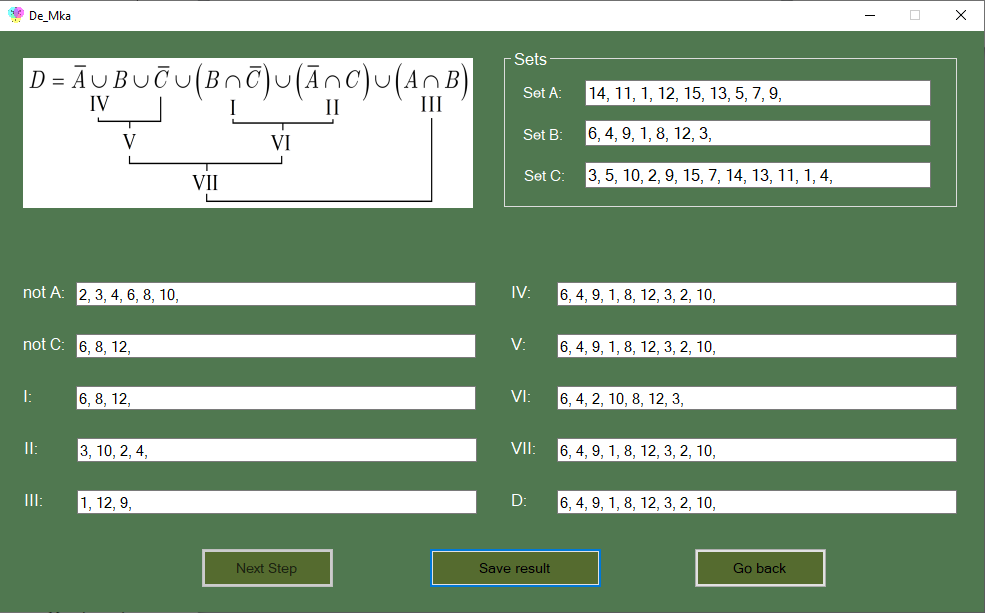
}

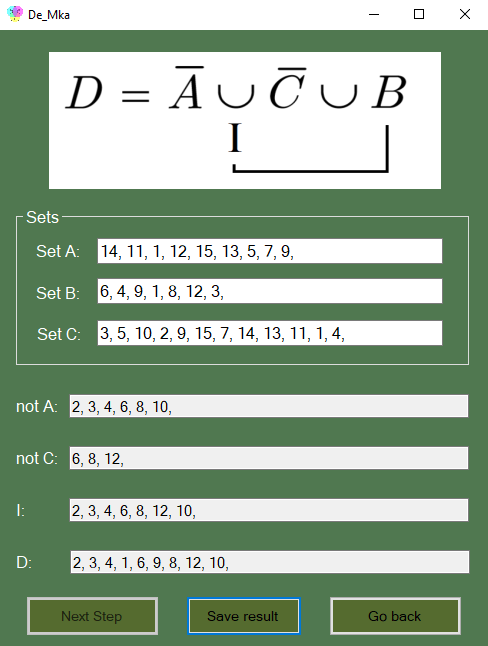
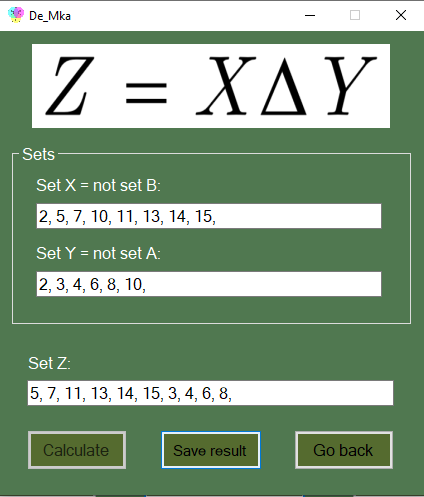
}

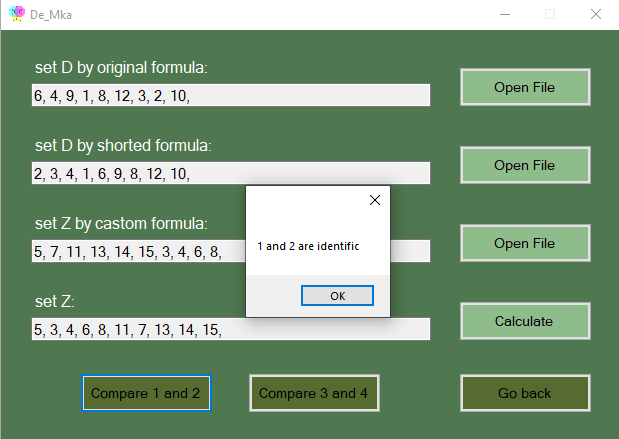
if (q) setZ.Add(i);

}

}

**10. Скріншоти виконання програми**

****

****

**Висновок:** на створення цієї програми я витратив дуже багато часу та сил. Але в решті-решт вона вийшла досить зручною та з повним функціоналом, заданим методичкою (навіть більше). Результат роботи 2 та 3 вікон (початкова та скорочена формули) повністю співпадають, в чому можна переконатися в 5 вікні. Алгоритм для обрахування множини Z було написано власноруч та результати звірено з результатами стандартної функції С#. Саме ця мова та структура данних HashSet були використані для досягнення мети лабораторної.