**Министерство образования и науки Российской Федерации**

Федеральное государственное бюджетное образовательное учреждение

Высшего образования

**«Владимирский государственный университет**

**Имени Александра Григорьевича и Николая Григорьевича Столетовых»**

(ВлГУ)

Кафедра «Функциональный анализ и его приложения»

**ЛАБОРАТОРНАЯ РАБОТА №1**

По дисциплине   
«Дискретная математика»

На тему:

**«Моделирование операции “пересечение” для двух числовых множеств»**

Выполнил:

Студент гр.ПМИ-123

Космачев Д.А.

Принял:

к.т.н., доцент каф. ФиПМ

Горлов В.Н.

Владимир, 2025

**ЦЕЛЬ РАБОТЫ:** изучить способы численного моделирования операции пересечения для множеств и составить компьютерную программу для выполнения пересечения двух конечных множеств.

**ЗАДАНИЕ:** составить компьютерную программу для выполнения проверки пересечения двух конечных множеств.

**ТЕОРЕТИЧЕСКАЯ ЧАСТЬ**

**1. Понятие множества и операции пересечения**

Множество — это совокупность уникальных элементов, объединенных по определенному признаку. В математике и информатике множества используются для хранения и обработки данных. Операция пересечения двух множеств — это операция, результатом которой является новое множество, содержащее только те элементы, которые присутствуют в обоих исходных множествах. Формально пересечение множеств A и B обозначается как A∩B.

Пример:

* Пусть A={1,2,3,4}, A={3,4,5,6}.
* Тогда A∩B={3,4}.

**2. Математическая основа операции пересечения**

Операция пересечения множеств основана на теории множеств, которая является фундаментом современной математики. Основные свойства пересечения:

* **Коммутативность**: A∩B=B∩.
* **Ассоциативность**: (A∩B)∩C=A∩(B∩C.
* **Идемпотентность**: A∩A=A.
* **Пересечение с пустым множеством**: A∩∅=∅.

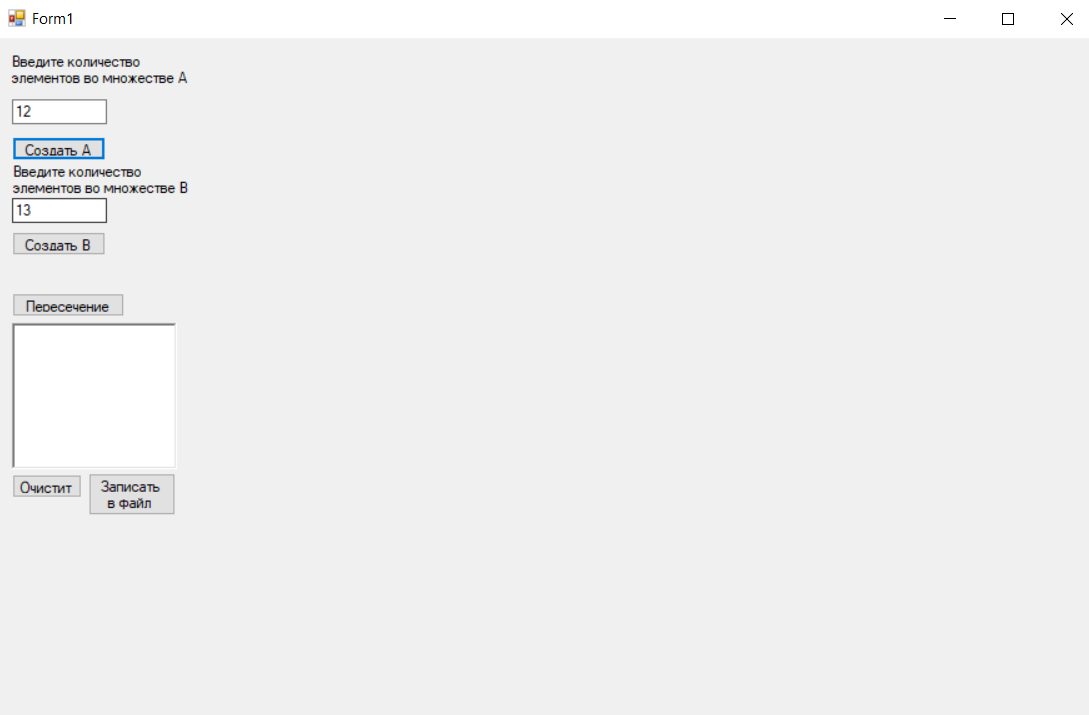
**3. Алгоритм выполнения операции пересечения**

Для выполнения операции пересечения двух множеств можно использовать следующий алгоритм:

1. Взять первое множество A и второе множество B.
2. Пройти по каждому элементу множества A и проверить, содержится ли он в множестве B.
3. Если элемент присутствует в обоих множествах, добавить его в результирующее множество C.
4. Вернуть множество C как результат пересечения.

**Решение:**

Конструктор формы.



Листинг программного кода:

using System;

using System.Diagnostics;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace dis\_mat\_laba1

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

List<int> A = new List<int>();

List<int> B = new List<int>();

List<int> C = new List<int>();

private int[] array\_filling(int size, TextBox tb)

{

int[] arr=new int[size];

int i = 0;

Random rand = new Random();

HashSet<int> unique\_number = new HashSet<int>();

while (unique\_number.Count < size)

{

int rand\_number=rand.Next(1, 25);

unique\_number.Add(rand\_number);

}

unique\_number.CopyTo(arr);

i = 0;

while (i < size)

{

rtb\_info.Text += $"{arr[i++]} ";

}

rtb\_info.Text += "\n";

return arr;

}

private void but\_create\_a\_Click(object sender, EventArgs e)

{

try

{

rtb\_info.Text += "Множество А: ";

int size = Convert.ToInt32(tb\_a.Text);

int[] arr = new int[size];

int i = 0;

Random rand = new Random();

HashSet<int> unique\_number = new HashSet<int>();

while (unique\_number.Count < size)

{

int rand\_number = rand.Next(1, 25);

unique\_number.Add(rand\_number);

}

unique\_number.CopyTo(arr);

A = unique\_number.ToList();

i = 0;

while (i < size)

{

rtb\_info.Text += $"{arr[i++]} ";

}

rtb\_info.Text += "\n";

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка при создании множества А: {ex.Message}");

pb\_info.SizeMode = PictureBoxSizeMode.Zoom;

pb\_info.Image = Image.FromFile("oshibka.png");

}

}

private void but\_create\_b\_Click(object sender, EventArgs e)

{

try

{

rtb\_info.Text += "Множество B: ";

int size = Convert.ToInt32(tb\_a.Text);

int[] arr = new int[size];

int i = 0;

Random rand = new Random();

HashSet<int> unique\_number = new HashSet<int>();

while (unique\_number.Count < size)

{

int rand\_number = rand.Next(1, 25);

unique\_number.Add(rand\_number);

}

unique\_number.CopyTo(arr);

B = unique\_number.ToList();

i = 0;

while (i < size)

{

rtb\_info.Text += $"{arr[i++]} ";

}

rtb\_info.Text += "\n";

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка при создании множества B: {ex.Message}");

pb\_info.SizeMode = PictureBoxSizeMode.Zoom;

pb\_info.Image = Image.FromFile("oshibka.png");

}

}

private void but\_intersection\_Click(object sender, EventArgs e)

{

C.Clear();

C = A.Intersect(B).ToList();

C.Sort();

pb\_info.SizeMode = PictureBoxSizeMode.Zoom;

if (C.Count == 0)

{

rtb\_info.Text += "Нет перечечения.\n";

pb\_info.Image = Image.FromFile("net.png");

}

else

{

pb\_info.Image = Image.FromFile("da.png");

rtb\_info.Text += "Пересечение: ";

int i = 0;

while (i<C.Count)

{

rtb\_info.Text += $"{C[i++]} ";

}

rtb\_info.Text += "\n";

}

}

private void button1\_Click(object sender, EventArgs e)

{

rtb\_info.Text = "";

pb\_info.Image = null;

A.Clear();

B.Clear();

C.Clear();

}

private void button2\_Click(object sender, EventArgs e)

{

using (StreamWriter writer = new StreamWriter("txt.txt"))

{

writer.WriteLine("Множество А: ");

int i = 0;

foreach(int el in A)

{

if (i % 10 == 0 && i != 0)

{

writer.WriteLine("");

}

writer.Write($"{el} ");

i++;

}

writer.WriteLine("\nМножество B: ");

i = 0;

foreach (int el in B)

{

if (i % 10 == 0 && i != 0)

{

writer.WriteLine("");

}

writer.Write($"{el} ");

i++;

}

writer.WriteLine("\nМножество C: ");

i = 0;

foreach (int el in C)

{

if (i % 10 == 0 && i != 0)

{

writer.WriteLine("");

}

writer.Write($"{el} ");

i++;

}

}

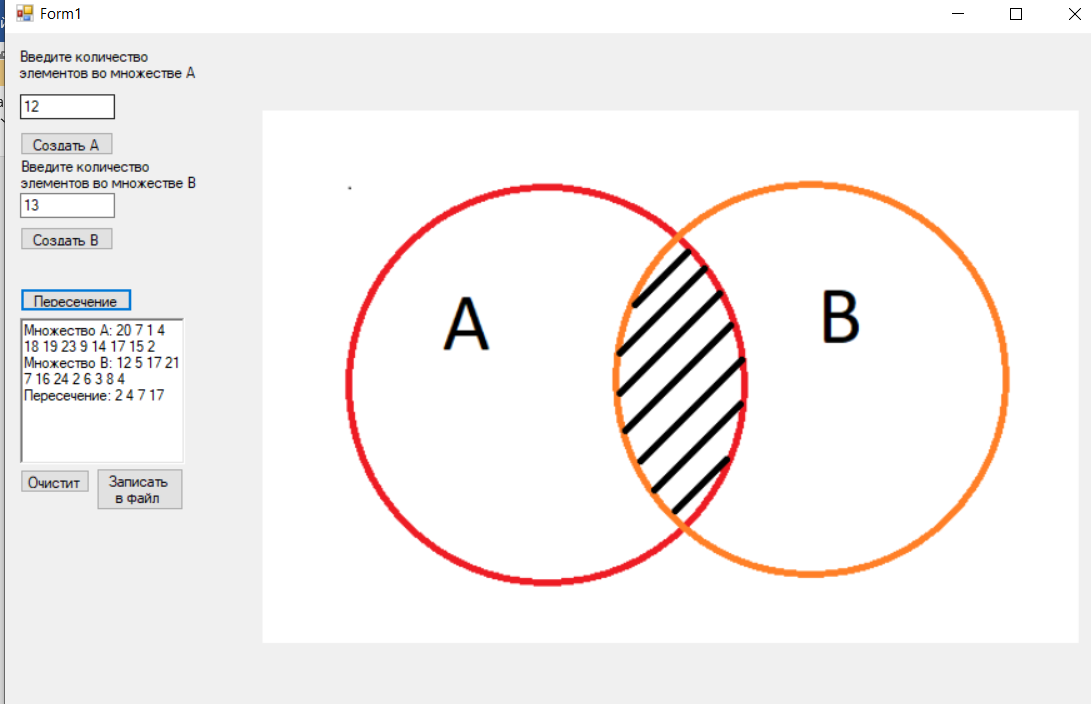
Process.Start("txt.txt");

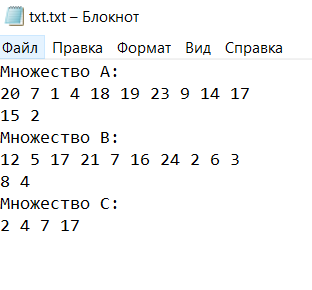
}

}

}

Результат работы программы:





**Вывод:** в ходе лабораторной работы было разработано оконное приложение на C# для моделирования операции пересечения двух числовых множеств. Пересечение множеств — это важная операция, позволяющая находить общие элементы между коллекциями данных. Приложение предоставляет удобный интерфейс для ввода данных и отображения результата, что делает процесс наглядным и простым для пользователя. Работа способствовала изучению основ работы с множествами, а также развитию навыков создания графических приложений на C#.