Міністерство освіти України

Національний технічний університет "ХПІ"

кафедра “ННІКНІТ"

**Звіт**

**Лабораторна робота 2**

з дисципліни "Програмна технологія .NET"

Виконав: студент групи КН-921Г

Верещак А. Г.

Перевірив:

Івашко А.В.

Харків 2023

**Зміст**

[Завдання 3](#__RefHeading___Toc2235_3286642050)

[Хід роботи 5](#__RefHeading___Toc2237_3286642050)

[Завдання 1: 5](#__RefHeading___Toc2239_3286642050)

[Завдання 2: 6](#__RefHeading___Toc2241_3286642050)

[Завдання 3: 7](#__RefHeading___Toc2243_3286642050)

[Завдання 4: 8](#__RefHeading___Toc2245_3286642050)

[Завдання 5: 9](#__RefHeading___Toc2247_3286642050)

[Завдання 6: 11](#__RefHeading___Toc2249_3286642050)

[Завдання 7: 12](#__RefHeading___Toc2251_3286642050)

[Завдання 8: 13](#__RefHeading___Toc2253_3286642050)

[Завдання 9: 14](#__RefHeading___Toc2255_3286642050)

[Висновок 15](#__RefHeading___Toc2257_3286642050)

Лабораторна робота №2. **Використання масивів і колекцій у мові C# .**

**Мета роботи**: набути навички в написанні програм з використання масивів і колекцій.

# Завдання

1. Використовуючи завдання власного варіанта до лабораторної роботи 8 з електронних методичних вказівок з основ програмування та алгоритмічних мов створити проект з реалізації цих завдань засобами мови C#.
2. Виконати минуле завдання, зробивши похідний масив динамічним, задавши його розмір з консолі.
3. Виконати минуле завдання, використовуючи кольорові можливості консолі для покращення інтерфейсу роботи програми. Наприклад, виділяти різними кольорами знайдені в масивах за завданням елементи чи їхні послідовності, розраховані параметри, тощо.
4. Виконати минуле завдання, використовуючи замість масивів одну з колекцій мови C# (List, LinkedList, ArrayList, чи ін.).
5. Використовуючи завдання власного варіанта до лабораторної роботи 9 з електронних методичних вказівок з основ програмування та алгоритмічних мов створити проект з реалізації цих завдань засобами мови C#.
6. Виконати минуле завдання, зробивши матрицю динамічною, задавши її розміри з консолі, використовуючи для цього прямокутний ([ , ]) двовимірний масив.
7. Виконати минуле завдання, зробивши матрицю динамічною, задавши її розміри з консолі, використовуючи для цього зубчастий ([][]) двовимірний масив.
8. Виконати минуле завдання, використовуючи кольорові можливості консолі для покращення інтерфейсу роботи програми. Наприклад, виділяти різними кольорами знайдені в матриці за завданням елементи чи їхні послідовності, розраховані параметри, тощо.
9. (Варіант 19, але завдань всього 12, тому було обрано завдання номер 7) Знайти суму елементів в тих рядках динамічної матриці, які містять хоча б один від’ємний елемент.

**Масиви в мові C#**:

Масиви - це упорядковані структури даних в мові C#, які можуть зберігати значення одного типу. Вони мають фіксований розмір, який встановлюється під час створення. Доступ до елементів масиву відбувається за допомогою індексів, починаючи з 0.

**Матриці в мові C#**:  
Матриця - це багатовимірний масив з двома або більше вимірами. У мові C# ви можете створити матрицю, вказавши розмір кожного виміру під час ініціалізації. Доступ до елементів матриці також здійснюється за допомогою індексів для кожного виміру.

**Колекції в мові C#**:

Колекції в мові C# дозволяють зберігати та керувати динамічними наборами даних. Вони більш гнучкі, ніж масиви і матриці, оскільки можуть змінюватися під час виконання програми. Основні типи колекцій включають List<T>, Dictionary<TKey, TValue>, HashSet<T>, Queue<T> та Stack<T>.

Колекції надають більше можливостей для додавання, видалення та пошуку елементів, що робить їх корисними для різних сценаріїв програмування.

# Хід роботи

# Завдання 1:

Створюємо новий проект Task\_1.

Для вирішення поставленого завдання пишемо наступний код:

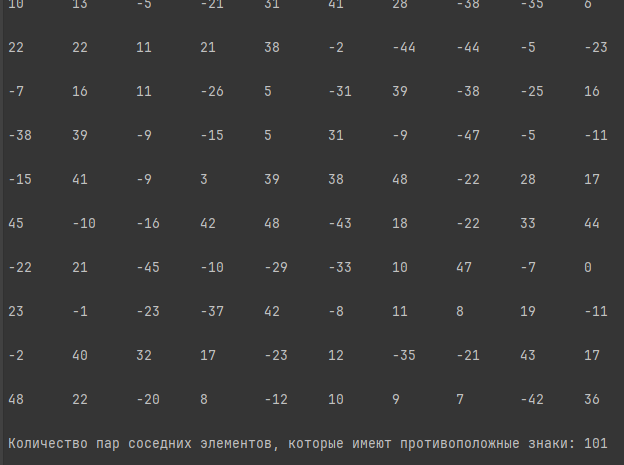
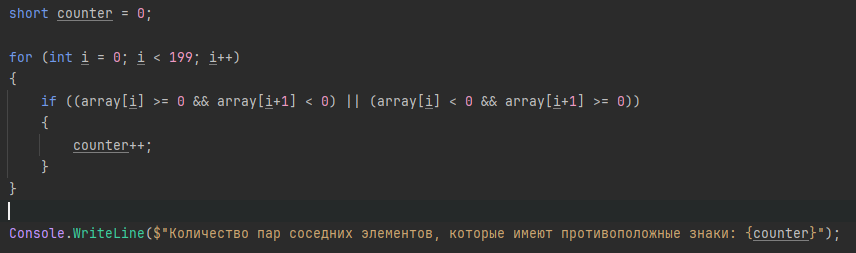
Рисунок 2.1 — Фрагмент коду Task\_1

Рисунок 2.2 — Робота програми Task\_1

# Завдання 2:

Створюємо новий проект Task\_2.

Для вирішення поставленого завдання до минулого варіанту коду пишемо додатковий:

# 

Рисунок 2.3 — Фрагмент коду Task\_2

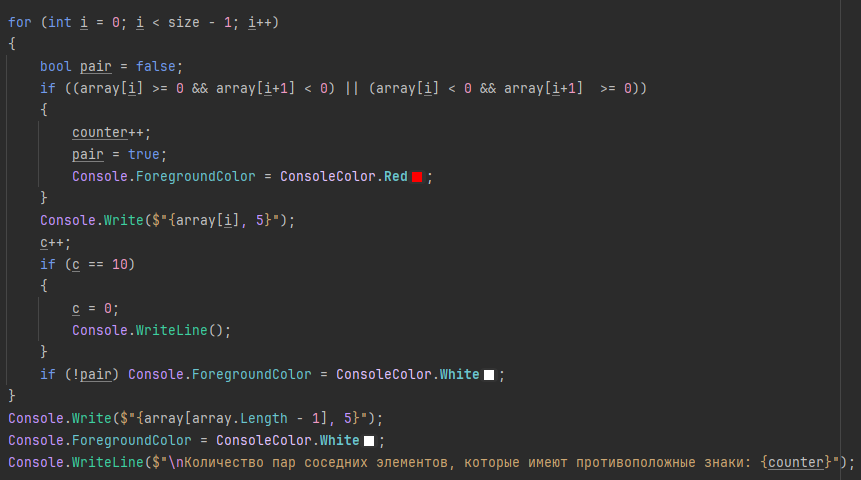
# 

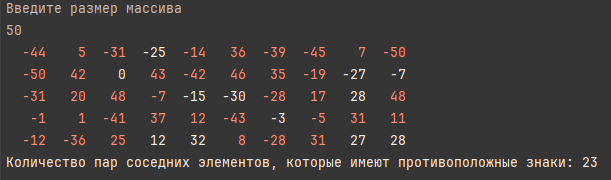
Рисунок 2.4 — Робота коду Task\_2

# Завдання 3:

Створюємо новий проект Task\_3.

Для вирішення поставленого завдання до минулого варіанту коду пишемо додатковий:

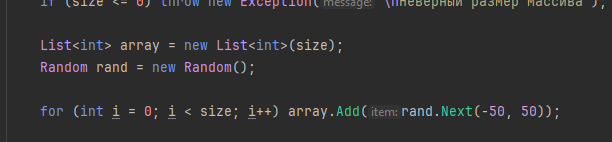
Рисунок 2.5 — Фрагмент коду Task\_3

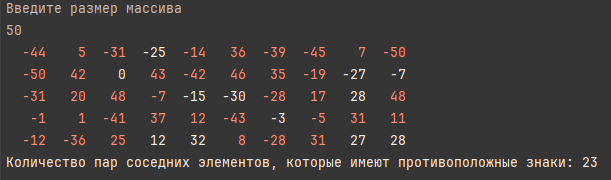
Рисунок 2.6 — Робота коду Task\_3

# Завдання 4:

Створюємо новий проект Task\_4.

Для вирішення поставленого завдання до минулого варіанту коду змінюємо його:

Рисунок 2.7 — Фрагмент коду Task\_4

Рисунок 2.8 — Робота коду Task\_4

# Завдання 5:

Створюємо новий проект Task\_5.

Для вирішення поставленого завдання пишемо наступний код:

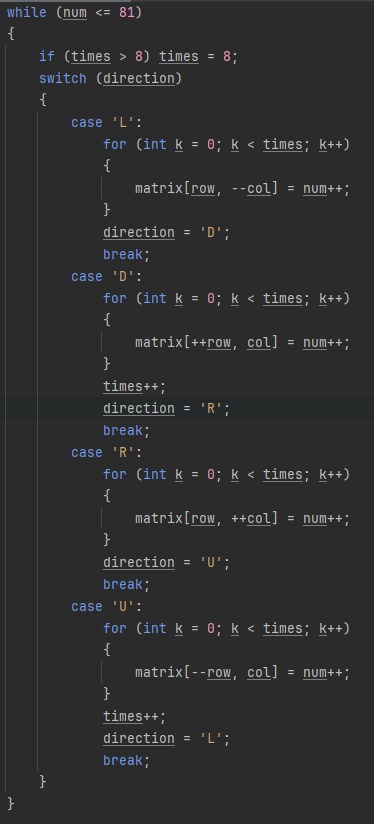


Рисунок 2.9 — Код програми Task\_5

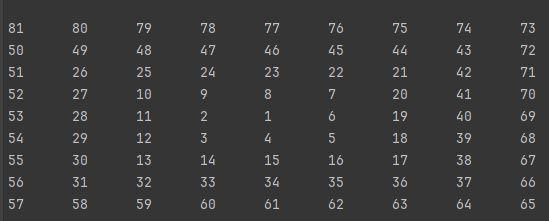
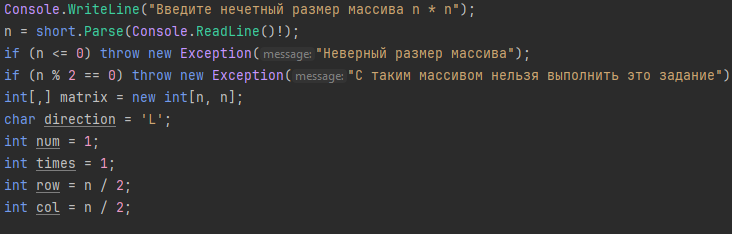


Рисунок 2.10 — Робота коду Task\_5

# Завдання 6:

Створюємо новий проект Task\_6.

Для вирішення поставленого завдання до минулого варіанту коду змінюємо його:

Рисунок 2.11 — Фрагмент коду Task\_6

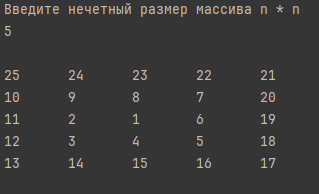
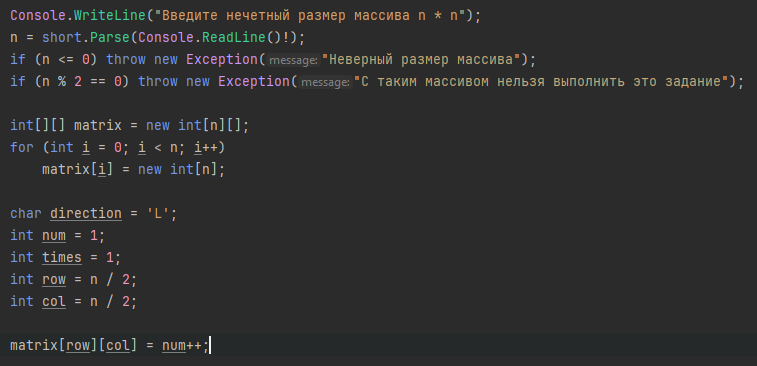


Рисунок 2.12 — Робота коду Task\_6

# Завдання 7:

Створюємо новий проект Task\_7.

Для вирішення поставленого завдання до минулого варіанту коду змінюємо його:

Рисунок 2.13 — Фрагмент коду Task\_7

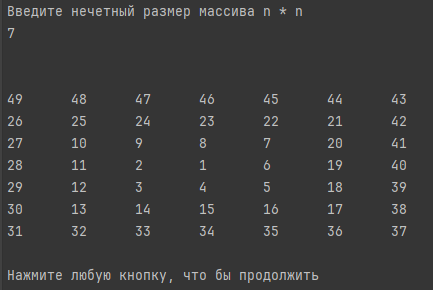
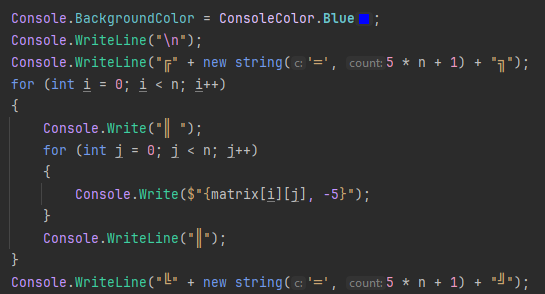


Рисунок 2.14 — Робота коду Task\_7

# Завдання 8:

Створюємо новий проект Task\_8.

Для вирішення поставленого завдання до минулого варіанту коду змінюємо його:

Рисунок 2.15 — Фрагмент коду Task\_8

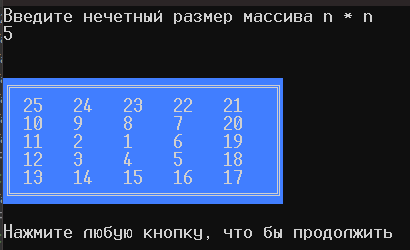


Рисунок 2.16 — Робота коду Task\_8

# Завдання 9:

Створюємо новий проект Task\_9.

Для вирішення поставленого завдання пишемо код:

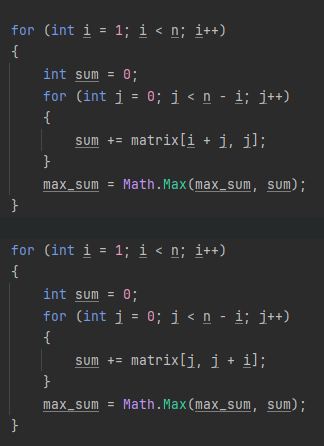


Рисунок 2.17 — Фрагмент коду Task\_9

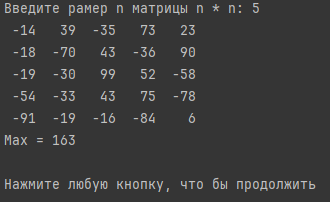


Рисунок 2.18 — Робота коду Task\_9

# Висновок

Під час вивчення та використання масивів і колекцій у мові програмування C#, я зрозумів, що обирання між ними залежить від конкретних вимог та потреб проекту.

Масиви становлять ідеальний вибір, коли ми маємо справу із статичними наборами даних, тобто випадки, коли розмір масиву залишається сталим і не змінюється протягом життя програми. Вони дозволяють легко і ефективно доступатися до елементів за їхніми індексами та швидко виконувати операції з даними.

З іншого боку, колекції надають більше гнучкості, коли маємо справу з динамічними даними або потребою у розширенні функціональності програми. Вони дозволяють динамічно додавати, видаляти та модифікувати елементи колекції, не зв'язуючись із фіксованим розміром. Бібліотека колекцій .NET містить різноманітні типи колекцій, такі як List, Dictionary, Queue, і багато інших, що розширюють можливості програміста для роботи з даними та спрощують розробку.

Таким чином, вибір між масивами і колекціями у C# повинен ґрунтуватися на конкретних вимогах та природі даних, з якими ви працюєте в своєму проекті, з метою досягнення максимальної ефективності та гнучкості у вашому програмному рішенні.