### Кам’янець-Подільський національний університет імені Івана Огієнка

### КАФЕДРА КОМП’ЮТЕРНИХ НАУК

### Навчальна дисципліна «Об’єктно-орієнтоване програмування»

### ЛАБОРАТОРНА РОБОТА #0106

## Тема:

Класи колекцій і протоколи ітерації

Варіант №2

**Виконав**:  
студент 1-го курсу  
групи KNms1-B23  
Веретко С. І.

**Прийняла**:  
старший викладач,  
Мястковська М.О.

Кам’янець-Подільський – 2023

1. Теоретичні відомості

Будь-який набір даних, об'єктів можна назвати колекцією.

Масив з простору імен System.Array - це теж свого роду колекція, але статична, його не можна ні розширити, ні стиснути при необхідності.

.NET Являє собою безліч класів колекцій, що дозволяють працювати з наборами даних більш гнучко ніж масив.

Кожен об'єкт, що міститься в колекції, називається її елементом. Деякі колекції зберігають дані як прямий список елементів, інші містять списки пар ключів і значень.

**Популярні класи колекцій**:

**System.Collections.ArrayList class** – подібний до масиву, але на відміну від масиву може розширюватися.

**System.Collections.Stack class** – для тимчасового розміщення об'єктів, які після використання будуть видалені. Ця колекція працює за принципом *LIFO* (last-in-first-out) - послідовність вилучення об'єктів протилежна послідовності їх додаванню до колекції.

**System.Collections.Queue class** – головною відмінністю даної колекції від Stack полягає в принципі FIFO (first-in-first-out), за яким працює черга. Елементи вилучаються з колекції в тій же послідовності, в якій вони були в її розміщені. Індексація до елементів даної колекції не застосовується.

**System.Collections.SortedList class** – колекція пар об'єктів ключ-значення, які сортуються по ключу і за індексом. За замовчуванням ємність колекції дорівнює 0. При додаванні першого елемента ємність встановлюється в 16 елементів.

**System.Collections.Hashtable class** – являє собою набори даних, що містять пари ключ-значення, причому обидва вони відносяться до типу Object, а це означає, що тип і ключа і його парного значення може бути зовсім будь-який. Ця колекція відноситься до так званої групи словників.

Всі класи колекцій знаходяться в просторі імен System.Collections.

У доповненні до цього простору існує ще один простір імен System.Collections.Specialized, який містить не настільки широко відомих строго типізованих класів колекцій.

**System.Collections.Specialized.ListDictionary class** – клас поводиться як Hashtable, але перевершує його за швидкістю при роботі з 10 і менш елементами. При роботі з великою кількістю елементів його використовувати не рекомендується.

**System.Collections.Specialized.HybridDictionary class** – складається з двох вбудованих колекцій, List Dictionary і Hashtable. Одночасно використовується тільки один з цих класів. ListDictionary використовується поки колекція містить 10 або менше елементів, а потім відбувається перемикання на використання Hashtable. Якщо колекція вже переключалася на Hashtable, переключитися назад вже не можливо.

**System.Collections.Specialized.CollectionsUtil class** – створює колекції, що ігнорують регістр в рядках. Містить 2 статичних методи: один для створення регістро незалежного Hashtable, а інший для створення регістро незалежного SortedList.

**System.Collections.Specialized.NameValueCollection class** – це колекція складається з пар ключ-значення, обидва з яких відносяться до типу string. Особливість: може зберігати безліч значень при єдиному ключі.

**System.Collections.Specialized.StringCollection class** – це звичайний список елементів типу string. У ньому можна поміщати null елементи і дубльовані елементи. Цей список чутливий до регістру.

**System.Collections.Specialized.StringDictionary class** – це Hashtable, містять ключі і значення типу string. Перш ніж бути доданими до колекції, ключі приводяться до нижнього регістру.

2. Завдання для виконання

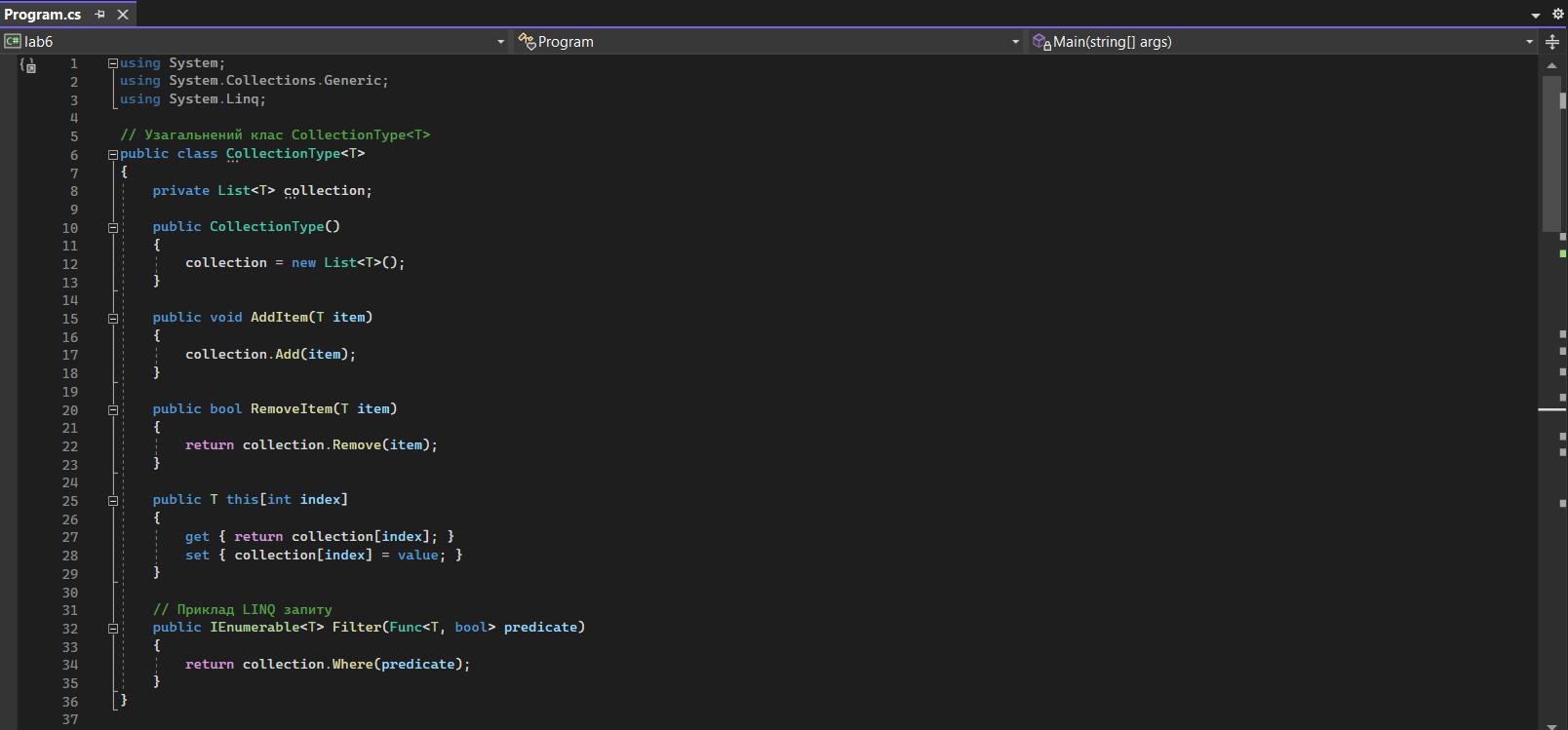
1. Створіть узагальнений клас CollectionType<T>. Визначити в класі конструктори, методи додавання та видалення елементів, інші необхідні методи та, якщо потрібно, перевантажені операції. Визначити індексатори та властивості. CollectionType можна реалізувати на основі стандартних колекцій (List, Stack, Array тощо).

2. Візьміть, створений тип (клас) з лабораторної №1, та реалізувати в ньому інтерфейс IComparable<T>. Використовуйте цей клас як параметр вашого узагальненого класу. Створіть кілька колекцій. Виконайте сортування, LINQ-запити відповідно до варіанта.

3. Виконайте кілька складних LINQToObject запитів (мінімум 5) до колекції об'єктів, використовуючи одночасно більше трьох операцій (приклад: where + select + orderBy, first + any + min).

4. Створіть узагальнену стандартну колекцію з простору імен System.Collections вказану у варіанті з рядками та виконайте введення-виведення, пошук рядків, що містять певне значення, підрахунок кількості рядків довжини n, сортування у зростаючому та спадному порядку.

5.створити масив об'єктів CollectionType, запити – знайти колекції з негативними елементами (вибрати будь-яке поле об'єкта), знайти максимальну та мінімальну колекцію в масиві, що містить зазначений елемент. Узагальнена колекція - Dictionary<T>.

3. Послідовність виконання завданнь  


4. Лістинг програми

using System;

using System.Collections.Generic;

using System.Linq;

// Узагальнений клас CollectionType<T>

public class CollectionType<T>

{

private List<T> collection;

public CollectionType()

{

collection = new List<T>();

}

public void AddItem(T item)

{

collection.Add(item);

}

public bool RemoveItem(T item)

{

return collection.Remove(item);

}

public T this[int index]

{

get { return collection[index]; }

set { collection[index] = value; }

}

// Приклад LINQ запиту

public IEnumerable<T> Filter(Func<T, bool> predicate)

{

return collection.Where(predicate);

}

}

// Приклад класу з лабораторної №1, що реалізує IComparable<T>

public class LabClass : IComparable<LabClass>

{

// ваші поля та методи

public int CompareTo(LabClass other)

{

if (other == null)

{

return 1; // Якщо переданий об'єкт other є null, поточний об'єкт більший

}

// Порівняння за критеріями вашого класу LabClass

// Повернення значення, яке показує, чи цей об'єкт менший (-1), рівний (0) чи більший (1) за other

// Наприклад, якщо порівнювати за числовим полем SomeNumber:

// return this.SomeNumber.CompareTo(other.SomeNumber);

// Реалізуйте логіку порівняння за власними критеріями вашого класу

return 0; // Повертаємо 0 як приклад, оскільки логіка порівняння не визначена

}

}

class Program

{

static void Main(string[] args)

{

// Створення колекцій та робота з ними

CollectionType<int> intCollection = new CollectionType<int>();

intCollection.AddItem(1);

intCollection.AddItem(2);

intCollection.AddItem(3);

CollectionType<string> stringCollection = new CollectionType<string>();

stringCollection.AddItem("apple");

stringCollection.AddItem("banana");

stringCollection.AddItem("orange");

// Приклад складного LINQ-запиту

var filteredStrings = stringCollection.Filter(s => s.Contains("a"));

foreach (var item in filteredStrings)

{

Console.WriteLine(item);

}

// Створення стандартної колекції та операції з нею

List<string> strings = new List<string>();

strings.Add("Hello");

strings.Add("World");

var filtered = strings.Where(s => s.Contains("o"));

int count = strings.Count(s => s.Length == 5);

var sortedAscending = strings.OrderBy(s => s);

var sortedDescending = strings.OrderByDescending(s => s);

// Робота з масивом об'єктів CollectionType та Dictionary

CollectionType<int>[] arrayOfCollections = new CollectionType<int>[5];

// Шукати колекції з негативними елементами, знаходження максимуму та мінімуму тощо.

// Можливо, знадобиться переглянути логіку класу CollectionType та зробити відповідні операції.

Dictionary<int, string> dictionary = new Dictionary<int, string>();

dictionary.Add(1, "one");

dictionary.Add(2, "two");

dictionary.Add(3, "three");

foreach (var pair in dictionary)

{

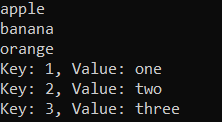
Console.WriteLine($"Key: {pair.Key}, Value: {pair.Value}");

}

}

}

5. Результат роботи програми



6. [xanax4rl/lab (github.com)](https://github.com/xanax4rl/lab)