|  |  |
| --- | --- |
| Міністерство освіти і науки України  НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ  “ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ” | |
| Кафедра управління проєктами в інформаційних технологіях | |
| Звіт з лабораторної № 2  «Колекції C#, робота з даними з використанням технології LINQ to Objects використання методів розширень та узагальнень у C#.»  з дисципліни «Стек технологій»  Варіант 4 | |
|  | *Виконали*:  студенти  групи КН-322а  Данилова Ангеліна Вадимівна  Непоменко Дмитро Григорович  Лобач Даніїл Артемович |
| Харків, НТУ “ХПІ”  2024 | |

**Мета:**

Навчитися використовувати методи розширення та узагальнення у мові програмування C#, вивчити можливості колекцій, отримати практичні навички роботи з колекціями .NET, освоїти методи обробки даних за допомогою технології LINQ to Objects. Ознайомитись підходом до розробки Test Driven Development та розробкою unit-тестів.

**Теоретичні відомості**

1. Колекції в C# - це структури даних, які дозволяють зберігати та керувати наборами елементів одного або різних типів. Вони використовуються для організації даних, забезпечення ефективного доступу до елементів та виконання різних операцій над ними.

2. Переваги узагальнених колекцій: Безпека типів: компілятор перевіряє типи елементів під час компіляції, що зменшує кількість помилок. Гнучкість: можливість використання колекцій з різними типами даних без необхідності явного приведення типів. Підтримка LINQ.

3. Відмінності масивів та колекцій. Розмір: масиви мають фіксований розмір при створенні, тоді як колекції можуть динамічно змінювати свій розмір. Функціональність: колекції надають широкий спектр методів для додавання, видалення, пошуку та сортування елементів, тоді як масиви мають більш обмежений набір операцій. Тип елементів: масиви можуть зберігати тільки елементи одного типу, тоді як колекції можуть зберігати елементи різних типів (включаючи значення null).

4. LINQ to Objects - це технологія, яка дозволяє запитувати та маніпулювати даними в об'єктних колекціях за допомогою мови запитів, схожої на SQL.

5. Основні типи колекцій: List, Dictionary, HashSet, Queue, Stack.

6. LINQ для обробки даних. Вибір елементів, які відповідають певним критеріям. Сортування. Перетворення однієї колекції в іншу, змінюючи структуру або тип елементів. Об'єднання елементів за певними критеріями.

7. Метод розширення - це статичний метод, який дозволяє додавати нові методи до існуючих типів без зміни їхнього вихідного коду.

8. LINQ забезпечує більш декларативний і читабельний синтаксис для роботи з колекціями, що спрощує написання коду та зменшує кількість помилок.

9. Переваги LINQ: Висока читабельність коду. Зменшення кількості помилок. Інтеграція з різними джерелами даних. Підтримка функціонального стилю програмування.

10. SortedList, SortedSet.

11. Компілятор перевіряє типи елементів при додаванні їх до колекції.

12. Фільтрація та сортування за допомогою LINQ:

Фільтрація: Where(), FirstOrDefault(), SingleOrDefault().

Сортування: OrderBy(), OrderByDescending().

13. Конкуренція за доступ до колекції, необхідність синхронізації доступу.

14. TDD - це методологія розробки, яка передбачає написання тестів перед написанням коду. Це дозволяє переконатися, що код працює правильно і забезпечує високу якість.

15. NUnit, xUnit, MSTest.

16. Забезпечення якості коду, виявлення помилок на ранніх етапах розробки, спрощення рефакторингу.

**Завдання**

1. Опрацювати теорію (див. додаток А)

2. Реалізувати методи розширення:

для класу String:

* інвертування рядка;
* підрахунок кількості входжень заданого у параметрі символа у рядок.

для одновимірних масивів:

* метод, що визначає скільки разів зустрічається задане значення у масиві (метод має працювати для одновимірних масивів усіх типів, для реалізації даного методу розширення використайте узагальнення та їх обмеження за допомогою “where”);
* метод, що повертає новий масив такого ж типу і формує його з унікальних елементів (видаляє повтори);

Реалізувати *unit-тести* для демонстрації роботи реалізованих методів розширення.

3. Реалізувати узагальнений клас для зберігання «розширеного словника» де для ключа передбачається зв'язок з двома значеннями:

ExtendedDictionary<T, U, V>

де T - тип даних ключа,

U - тип даних першого значення,

V - тип даних другого значення.

Подання елемента словника реалізувати у вигляді окремого класу:

ExtendedDictionaryElement<T, U, V>

передбачивши властивості для доступу до ключа, першого та другого значення.

Словник повинен мати можливість використання у циклах foreach:

foreach (var elem in array) { … }

Необхідно передбачити наступні операції:

* додавання елемента у словник;
* видалення елемента зі словника за заданим ключем;
* перевірка наявності елемента із заданим ключем;
* перевірка наявності елемента із заданим значенням (значення1 та значення2);
* повернення елемента за заданим ключем (реалізувати операцію індексування);
* властивість, що повертає кількість елементів;

Реалізувати *unit-тести* для демонстрації роботи з реалізованим словником.

4. За допомогою синтаксису запитів [LINQ to Objects](http://habrahabr.ru/post/81592/) реалізувати обробку послідовностей згідно одного з варіантів, наведених нижче. **Номер варіанта необхідно обирати за останньою цифрою залікової книжки.**

Дани масиви наступного змісту:

|  |
| --- |
| String[] A = {  "Falluot 3", "Daxter 2", "System Shok 2", "Morrowind", "Pes 2013"  };  int[] B = { 2, -7, -10, 6, 7, 9, 3 };  String[] C = {  "Light Green", "Red", "Green", "Yellow", "Purple", "Dark Green",  "Light Red", "Dark Red", "Dark Yellow", "Light Yellow"  }; |

Класи:

|  |
| --- |
| Car, содержащий максимальную скорость, цвет и производителя, число пассажиров.  Product, содержащий название продукта, количество упаковок на складе, описание. |

Списки (контейнери):

|  |
| --- |
| List<string> myCars = new List<String> { "Yugo", "Aztec", "BMW" };  List<string> yourCars = new List<String> { "BMW", "Saab", "Aztec" }; |

**Варіант 4**

1. Написати запит для знаходження елементів масиву А, що включають пробіл.
2. Написати запит для негайного знаходження негативних елементів масиву В як List[].
3. Написати запит для знаходження всіх відтінків зеленого кольору в масиві С.
4. Створити перелік автомобілів класу Car. Вибрати зі списку автомобілі Audi та вивести всю інформацію про них.
5. Створити перелік продуктів класу Product. Вибрати продукти, які закінчилися і вивести всю інформацію про них.
6. Вивести результуючий набір списків myCars та yourCars, що містить спільні елементи.

**Хід роботи**

**Висновки**