Практическая работа № 4.  
Работа с коллекциями данных  
с помощью LINQ

**Цель практической работы**

Закрепление теоретических знаний по основам работы с коллекциями данных при помощи LINQ.

**Постановка задачи**

**Language Integrated Query (LINQ)** — это технология, интегрированная в язык C#, которая предоставляет унифицированный синтаксис для выполнения запросов к различным источникам данных: коллекциям в памяти, базам данных, XML-документам и другим. LINQ объединяет возможности декларативного программирования с императивным стилем C#, что повышает читаемость кода и снижает вероятность ошибок.

**Основные концепции LINQ**

**Определение и назначение**

LINQ позволяет выражать запросы непосредственно в коде на C#, используя синтаксис, напоминающий SQL. Это достигается за счет:

* **Расширяющих методов (extension methods)**, таких как Where(), Select(), OrderBy(), определенных в пространстве имен System.Linq.
* **Выражений запросов (query expressions)**, которые компилируются в вызовы этих методов.

**Источники данных LINQ**

LINQ поддерживает несколько провайдеров для работы с разными типами данных:

* **LINQ to Objects** — запросы к коллекциям в памяти (массивы, списки, словари).
* **LINQ to Entities / Entity Framework Core** — запросы к базам данных через ORM.
* **LINQ to XML** — работа с XML-документами.
* **LINQ to SQL** (устаревший) — ранняя реализация для SQL Server.
* 2. Синтаксис LINQ

**Методный синтаксис (Method Syntax)**

Основан на цепочках вызовов методов-расширений. Пример:

record Sample(int Id,

string Name,

int Age,

int Property,

string Department);

var collection = new List<Sample>();

var result = collection

.Where(item => item.Property > 10)

.OrderBy(item => item.Name)

.Select(item => new { item.Id, item.Name });

**Синтаксис запросов (Query Syntax)**

Напоминает SQL и компилируется в методные вызовы:

var result = from item in collection

where item.Property > 10

orderby item.Name

select new { item.Id, item.Name };

Различия и выбор синтаксиса

* **Query Syntax** удобен для простых запросов, особенно с join и group by.
* **Method Syntax** более гибкий, поддерживает все операции и часто используется в комбинации с лямбда-выражениями.

Основные операции LINQ

Фильтрация

* Where() — выбирает элементы, удовлетворяющие условию.

var filtered = collection.Where(x => x.Age > 18);

Проекция

* Select() — преобразует элементы в новую форму.

var names = collection.Select(x => x.Name);

Сортировка

* OrderBy(), OrderByDescending(), ThenBy() — сортировка по ключу.

var sorted = collection.OrderBy(x => x.Age).ThenBy(x => x.Name);

Группировка

* GroupBy() — группирует элементы по ключу.

var groups = collection.GroupBy(x => x.Department);

Агрегация

* Count(), Sum(), Average(), Max(), Min() — статистические операции.

var total = collection.Sum(x => x.Price);

Соединения (Joins)

* Join() и GroupJoin() — аналоги SQL INNER JOIN и LEFT JOIN.

var joined = customers.Join(orders,

c => c.Id,

o => o.CustomerId,

(c, o) => new { c.Name, o.Amount });

**Задание на практическую работу**

**Задание 1: Фильтрация и сортировка списка чисел**

**Условие:**

Дан список целых чисел, например: 5, 12, 3, 8, 15, 7, 20, 1, 9, 10

**Задачи:**

1. Выбрать все числа больше 5.
2. Отсортировать их по возрастанию.
3. Вывести результат в консоль.

**Задание 2: Работа с коллекцией объектов**

**Условие:**

Дана коллекция студентов:

public class Student

{

public string Name { get; set; }

public int Age { get; set; }

public string Group { get; set; }

}

List<Student> students = new List<Student>

{

new Student { Name = "Алексей", Age = 20, Group = "CS101" },

new Student { Name = "Мария", Age = 19, Group = "CS102" },

new Student { Name = "Иван", Age = 21, Group = "CS101" },

new Student { Name = "Ольга", Age = 18, Group = "CS103" },

};

**Задачи:**

1. Выбрать всех студентов из группы "CS101".
2. Отсортировать их по возрасту (от младшего к старшему).
3. Вывести их имена и возраст.

**Задание 3: Агрегация данных (поиск среднего и максимального значения)**

**Условие:**

Дан массив температур за неделю, например: 15, 18, 12, 22, 17, 20, 14

**Задачи:**

1. Найти среднюю температуру.
2. Найти максимальную температуру.
3. Вывести оба значения.

**Задание 4: Группировка товаров по категориям**

**Условие:**

Дана коллекция товаров:

public class Product

{

public string Name { get; set; }

public string Category { get; set; }

public decimal Price { get; set; }

}

List<Product> products = new List<Product>

{

new Product { Name = "Ноутбук", Category = "Электроника", Price = 50000 },

new Product { Name = "Смартфон", Category = "Электроника", Price = 30000 },

new Product { Name = "Футболка", Category = "Одежда", Price = 1500 },

new Product { Name = "Книга", Category = "Книги", Price = 800 },

new Product { Name = "Джинсы", Category = "Одежда", Price = 3500 },

};

**Задачи:**

1. Сгруппировать товары по категориям.
2. Для каждой категории вывести:
   * Название категории.
   * Количество товаров в ней.
   * Среднюю цену товаров.
3. Отсортировать категории по убыванию средней цены.

**Задание 5: Соединение двух коллекций (аналог SQL JOIN)**

**Условие:**

Даны две коллекции: **заказы** и **клиенты**.

public class Customer

{

public int Id { get; set; }

public string Name { get; set; }

}

public class Order

{

public int Id { get; set; }

public int CustomerId { get; set; }

public decimal Amount { get; set; }

}

List<Customer> customers = new List<Customer>

{

new Customer { Id = 1, Name = "Иван" },

new Customer { Id = 2, Name = "Анна" },

new Customer { Id = 3, Name = "Петр" },

};

List<Order> orders = new List<Order>

{

new Order { Id = 101, CustomerId = 1, Amount = 5000 },

new Order { Id = 102, CustomerId = 2, Amount = 3000 },

new Order { Id = 103, CustomerId = 1, Amount = 2000 },

new Order { Id = 104, CustomerId = 3, Amount = 7000 },

};

**Задачи:**

1. Соединить коллекции customers и orders по полю Id (клиента) и CustomerId (заказа).
2. Вывести список заказов в формате:
   * Имя клиента.
   * Сумма заказа.
3. **Дополнительно:**
   * Найти общую сумму заказов для каждого клиента.
   * Вывести клиентов, у которых сумма заказов больше 5000 ₽.

**Пример разработки приложения**

Шаг 1: Подключение необходимых пространств имен

Для работы с LINQ нужно добавить:

using System;

using System.Collections.Generic;

using System.Linq; // Это пространство имен необходимо для работы LINQ

Шаг 2: Фильтрация чисел (выбор чисел > 5)

Используем метод Where():

List<int> numbers = new List<int> { 5, 12, 3, 8, 15, 7, 20, 1, 9, 10 };

var filteredNumbers = numbers.Where(n => n > 5);

* Where() - метод расширения LINQ для фильтрации
* n => n > 5 - лямбда-выражение, где n представляет каждый элемент коллекции
* Возвращает IEnumerable<int> с числами, удовлетворяющими условию

Промежуточный результат (после фильтрации):

12, 8, 15, 7, 20, 9, 10

Шаг 3: Сортировка по возрастанию

Применяем метод OrderBy():

var sortedNumbers = filteredNumbers.OrderBy(n => n);

* OrderBy() - сортирует элементы по возрастанию
* n => n - сортировка по самим числам (можно изменить для сложных объектов)

Промежуточный результат (после сортировки):  
7, 8, 9, 10, 12, 15, 20

Шаг 4: Вывод результатов в консоль

Используем цикл foreach:

Console.WriteLine("Отфильтрованные и отсортированные числа:");

foreach (var num in sortedNumbers)

{

Console.Write(num + " ");

}

Шаг 5: Оптимизация (объединение операций в цепочку)

Можно объединить все операции в один LINQ-запрос:

var result = numbers

.Where(n => n > 5)

.OrderBy(n => n);

# Содержание пояснительной записки

1. Постановка задачи. Приводится теоретический материал, использованный при написании приложения.

2. Формулировка задания и вариант. Приводится задание на лабораторную работу и вариант этого задания.

3. Описание выполняемых действий. Необходимо привести описание последовательности разработки программы, реализации используемых методов, алгоритмов, блок-схем.

4. Анализ результатов. Привести анализ входных и выходных данных. Показать результаты выполнения программного кода. Предоставить скриншоты обработки тестовых примеров. Сделать выводы.

5. Листинг программы. Привести листинг разработанного программного кода, содержание файлов входных и выходных данных.

# Используемое программное обеспечение

1. Среда программирования MS Visual Studio Community 2022 (Свободно распространяемое программное обеспечение (в учебных целях));
2. Microsoft Office Standard 2007 (Open License: 42267924);
3. Open Office (Свободно распространяемое программное обеспечение).
4. Браузер (Свободно распространяемое программное обеспечение).

# Список литературы

* + - 1. Мейер Б. Объектно-ориентированное программирование и программная инженерия [Электронный ресурс]/ Мейер Б. – Электрон. текстовые данные. – М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016. – 285 c.
      2. Биллиг, В. A. Основы объектного программирования на С# (C# 3.0, Visual Studio 2008) [Электронный ресурс] : учебное пособие / В. A. Биллиг. — Электрон. текстовые данные. — Москва, Саратов : Интернет-Университет Информационных Технологий (ИНТУИТ), Вузовское образование, 2017. — 583 c. — 978-5-4487-0145-0. — Режим доступа: http://www.iprbookshop.ru/72339.html
      3. Павловская, Т. А. Программирование на языке высокого уровня C# [Электронный ресурс] / Т. А. Павловская. — 2-е изд. — Электрон. текстовые данные. — М. : Интернет-Университет Информационных Технологий (ИНТУИТ), 2016. — 245 c. — 2227-8397. — Режим доступа: http://www.iprbookshop.ru/73713.html
      4. Агапов, В. П. Основы программирования на языке С# [Электронный ресурс] : учебное пособие / В. П. Агапов. — Электрон. текстовые данные. — М. : Московский государственный строительный университет, ЭБС АСВ, 2012. — 128 c. — 978-5-7264-0576-6. — Режим доступа: http://www.iprbookshop.ru/16366.html
      5. Медведев, М. А. Программирование на СИ# [Электронный ресурс] : учебное пособие / М. А. Медведев, А. Н. Медведев ; под ред. А. В. Присяжный. — Электрон. текстовые данные. — Екатеринбург : Уральский федеральный университет, ЭБС АСВ, 2015. — 64 c. — 978-5-7996-1561-1. — Режим доступа: http://www.iprbookshop.ru/69667.html
      6. Казанский А.А. Объектно-ориентированное программирование на языке Microsoft Visual С# в среде разработки Microsoft Visual Studio 2008 и .NET Framework. 4.3 [Электронный ресурс]: учебное пособие и практикум/ Казанский А.А.— Электрон. текстовые данные.— М.: Московский государственный строительный университет, ЭБС АСВ, 2011.— 180 c
      7. Уйманова Н.А. Основы объектно-ориентированного программирования [Электронный ресурс]: практикум/ Уйманова Н.А., Таспаева М.Г.— Электрон. текстовые данные.— Оренбург: Оренбургский государственный университет, ЭБС АСВ, 2017.— 156 c.— Режим доступа: http://www.iprbookshop.ru/78808.html.— ЭБС «IPRbooks»
      8. Новиков П.В. Объектно-ориентированное программирование [Электронный ресурс]: учебно-методическое пособие к лабораторным работам/ Новиков П.В.— Электрон. текстовые данные.— Саратов: Вузовское образование, 2017.— 124 c.— Режим доступа: http://www.iprbookshop.ru/64650.html.— ЭБС «IPRbooks»