

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
МІСЬКОГО ГОСПОДАРСТВА імені О. М. БЕКЕТОВА

МЕТОДИЧНІ РЕКОМЕНДАЦІЇ

до виконання курсової роботи
з навчальної дисципліни

«ОБ'ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ»

*(для здобувачів першого (бакалаврського) рівня вищої освіти
денної форми навчання зі спеціальностей F3, 122 – Комп'ютерні науки,
F6, 126 – Інформаційні системи та технології)*

Харків
ХНУМГ ім. О. М. Бекетова
2025

Методичні рекомендації до виконання курсової роботи з навчальної дисципліни «Об'єктно-орієнтоване програмування» (для здобувачів першого (бакалаврського) рівня вищої освіти денної форми навчання зі спеціальностей F3, 122 – Комп'ютерні науки, F6, 126 – Інформаційні системи та технології) / Харків. нац. ун-т міськ. госп-ва ім. О. М. Бекетова ; уклад. : В. М. Бредіхін, А. Л. Литвинов. – Харків : ХНУМГ ім. О. М. Бекетова, 2025. – 54 с.

Укладачі: канд. техн. наук, доц. В. М. Бредіхін,
д-р техн. наук, проф. А. Л. Литвинов

Рецензент

О. Б. Костенко, кандидат фізико-матиматичних наук, доцент кафедри комп'ютерних наук та інформаційних технологій Харківського національного університету міського господарства імені О. М. Бекетова

Рекомендовано кафедрою комп'ютерних наук та інформаційних технологій, протокол № 1 від 29 серпня 2022 р.

ЗМІСТ

ВСТУП	4
1 ВИМОГИ ДО КУРСОВОЇ РОБОТИ	5
1.1 Проєктування системи класів	5
2 КОРОТКІ ТЕОРЕТИЧНІ ПОЯСНЕННЯ	8
2.1 Ініціалізація об'єктів і списків	13
2.2 Приклади запитань	14
3 ЗАВДАННЯ НА КУРСОВУ РОБОТУ	20
4 ОФОРМЛЕННЯ ЗВІТУ З ВИКОНАНОЇ КУРСОВОЇ РОБОТИ	51
СПИСОК РЕКОМЕНДОВАНИХ ДЖЕРЕЛ	52

ВСТУП

Одним із найважливіших досягнень у технології розробки сучасних додатків є реалізація запитів до баз даних з метою отримання інформації, підготовленої для сприйняття користувачем. Інформацію зазвичай потрібно не тільки вилучати, але й фільтрувати, групувати, сортувати, виконувати обчислення та інші операції. Усе це можна здійснити за допомогою *LINQ*-запитів.

LINQ (Language INtegrated Query – мова інтегрованих запитів) – це вбудована у середовище *Visual Studio* мова, що дозволяє видобувати та обробляти дані з різноманітних джерел, інформація в яких може мати різні формати даних.

Джерелами даних можуть бути як локальні колекції, так і віддалені бази даних. Для отримання даних можуть застосовуватися досить складні умови пошуку. Одночасно з реалізацією запиту можна виконувати проєкцію, фільтрацію, групування та сортування даних, що вилучаються. Можна також реалізовувати арифметичні операції з колекціями: знаходження максимальних та мінімальних значень, підрахунок сум та середніх арифметичних значень тощо.

Під час реалізації запиту, який повертає список, можлива поява нових типів (класів) даних. Зокрема, під час об'єднання даних нова таблиця містить стовпці, витягнуті (частково чи повністю) з вихідних таблиць, що означає поява об'єкта нового класу.

Вихідними даними для операції *LINQ* можуть бути один або два списки будь-якого типу. Результат виконання також може бути списком, але може бути й одиночним значенням. Ця курсова робота призначена для освоєння студентами обробки інформації в програмах користувача за допомогою запитів *LINQ*.

1 ВИМОГИ ДО КУРСОВОЇ РОБОТИ

Метою курсової роботи є розробка у середовищі *Visual Studio* фірми *Microsoft* консольної програми для роботи з локальними узагальненими списками. Інформація для списків повинна зберігатись у файлах, реалізованих за допомогою програми «Блокнот». Кожен зі списків має містити не менше 10-ти ініціалізованих об'єктів класу.

Структура сутностей наведена в описі кожного індивідуального завдання. Реалізація класів може передбачати наявність конструктора з параметрами (залежно від виду запитів), але може бути виконана за допомогою властивостей у формі {get; set;}. Студент може додавати до класу властивості, але їх не може бути менше, ніж зазначено у завданні.

Додаток, що розробляється, повинен обробляти не менше шести запитів. Запити виконуються у вигляді *LINQ to Objects*. Тексти запитів повинні подаватися на екрані у вигляді списку, і кожен із запитів може бути обраний за допомогою вибору номера його позиції у списку. Після виконання запиту на екран знову виводиться список, щоб вибрати інший запит. Одна з позицій у списку має забезпечувати «нормальний» вихід із програми.

Звіт студент надає в електронній формі у вигляді папки, назва якої містить номер завдання та прізвище студента. У папці подається консольна програма, на якій студент демонструє працездатність розробленої програми, і є текстовий файл з описом курсової роботи яка містить: титульний лист; текст завдання; текст програми; пояснення до програми; файли для ініціалізації списків.

1.1 Проєктування системи класів

Проєктування системи класів починається із обробки словника предметної області. Ця обробка полягає у виявленні того, які слова відповідають об'єктам, класам, властивостям та методам. Список

іменників є основою виділення класів та його властивостей, а список дієслів – визначення методів.

Для наведеного вище прикладу можна вказати таку відповідність:

- класи: точка, куля;
- властивості: координата, кут, колір, довжина, ширина;
- методи: намалювати, повернути, масштабувати, перемістити.

Наступний крок є фактично завершальним на етапі проєктування класів. Він полягає в тому, щоб визначити, який із класів містить відповідні властивості та методи. Необхідно звернути увагу, що набори властивостей і методів у різних класів можуть «перетинатися». Наприклад, і у класі «точка», й у класі «куля» наявні методи «намалювати», «масштабувати», «перемістити». Водночас метод «повернути» не має сенсу щодо об'єктів цих класів, зате наявний у класі «паралелепіпед».

Ще одним питанням, що потребує вирішення на цьому етапі, є виявлення відносин між класами. Йдеться про відносини успадкування та включення. Необхідно звернути увагу, що поняття «успадкування» найчастіше виникає тоді, коли різні класи мають частково подібні набори властивостей і методів. Під час складання словника предметної області у список можуть потрапити поняття, яким відповідають базові класи в ієрархії спадкування.

Тому потрібно уважно проаналізувати список класів, властивостей, методів, їх відповідність, і, можливо, виділити низку нових класів, пов'язаних із наявними відносинами спадкування та включення. На цьому етапі можна обрати у якості законів предметної області такі принципи об'єктно-орієнтованого підходу, як абстракція, інкапсуляція, поліморфізм.

У прикладі можна виділити абстрактний клас «фігура» з властивостями «абсциса», «ордината», «апліката», «колір» і методом «намалювати». Класи «точка» і «куля» будуть спадкоємцями класу «фігура», а метод «намалювати» може бути віртуальним, що дає нам поліморфічний кластер, що містить три класи.

Результати такого аналізу мають бути оформлені у вигляді діаграми класів. Переважним є використання нотації мови UML. Зокрема, необхідно дотримуватись таких правил:

- клас позначається прямокутником;
- прямокутник ділиться на три частини, у кожній з яких, відповідно, зазначаються: ім'я класу, список властивостей, список методів;
- імена класів, властивостей та методів можуть бути записані українською мовою, але відповідно до норм написання стандартних ідентифікаторів (одне слово, що містить літери, цифри, символ підкреслення та не починається з цифри);
- імена класів записуються з великої літери, імена властивостей та методів – з малої;
- перед ім'ям властивості або методу ставиться символ, що вказує на режим доступності: закритий (-), захищений (#), відкритий (+);
- після імені методу ставляться круглі дужки, в яких можуть бути перераховані параметри методу;
- успадкування класів позначається стрілкою з трикутним незафарбованим наконечником;
- стрілка під час успадкування прямує від класу-спадкоємця до батьківського класу.

Приклад діаграми класів наведено на рисунку 1.1.



Рисунок 1.1 – Діаграма класів тривимірних графічних примітивів
Діаграма класів із коментарями до неї є результатом цього етапу роботи.

2 КОРОТКІ ТЕОРЕТИЧНІ ПОЯСНЕННЯ

Інформація для заповнення об'єктів класів зберігається у файлах. Файли створюються в додатку «Блокнот». Всі дані для одного об'єкта записують в одному рядку.

Дробова частина відокремлюється від цілої за допомогою коми, тому для розмежування елементів у файлах її застосовувати не можна (якщо передбачається застосування дробових чисел). Можна використовувати пробіл (якщо в цьому проєкті рядкові дані пробілів не мають), точку з комою тощо. Нижче в прикладі кома відокремлює дробову частину числа від цілої, а елементи розмежовані крапкою із комою. Строкові дані в лапки не укладають.

Нехай є таблиці такої структури:

Магазин	Товар
Ключ	Ключ
Назва	Назва
Адреса	Ціна
	Кількість

Оголошення класів цих таблиць показано нижче.

```
class Vegetable
{
    public int vKey {set; get; }           // Ключ
    public string Name { set; get; }       // Назва
    public double Price { set; get; }      // Ціна
    public int Quan { set; get; }          // Кількість
    public Vegetable () { }
    public Vegetable (string n, double p, int q) // Конструктор
    {
        Name = n; Price = p; Quan = q;
    }
}

Клас «Магазин»
class Shop
{
    public string Name {set; get; }        // Назва товару, замість ключа
    public string ShopName {set; get; }    // Назва магазину
    public string Street {set; get; }
    public Shop () { }                     // Порожній конструктор
    public Shop (string s, string c, string n) // Конструктор з параметрами
}
```

```

{ ShopName = s; Street = c; Name = n; }
Public Shop (Shop it) // Конструктор, який приймає об'єкт
{ ShopName = it.ShopName; Street = it.Street; Name = it.Name; }
}

```

Для кожної таблиці створюється автономний файл, і введення даних для неї виконується окремо.

Якщо потрібно, щоб одна таблиця містила вкладену або об'єднану таблицю, то для створення такого списку до обидвох класів, що подають таблиці, додаються цілі ключі, за якими виконується об'єднання класів за допомогою запитів LINQ Join або GroupJoin. У деяких випадках (якщо вміст поля може бути ключем) додавання до класу спеціального ключа не є обов'язковим.

Перевага цього способу – можливість угруповання за елементами вкладеного списку, оскільки за допомогою об'єднання списків можна головним списком зробити вкладений.

Нижче наведено приклад оформлення файлів (на початку кожного рядка цілісний ключ):

Файл 1 (магазини)

- 1; Ашан; вул. Героїв праці;
- 2; Рост; вул. Полтавський шлях;

Файл 2 (товари)

- 1; Помідор; 83,5; 1
- 1; Огірок; 65; 2
- 2; Морква; 35,2; 3
- 2; Буряк; 14,5; 2
- 2; Цибуля; 21,4; 3

(У магазині «Ашан» – 2 товари, в магазині – «Рост» 3).

Нижче наведено метод формування узагальненого списку для одного класу. (У цьому прикладі ім'я класу Vegetable (Овочі). У завданнях на курсову роботу має бути 3 або 4 класи. Відповідно, ви маєте змінити імена класів, шляхи до файлів, оголошення списків та кількість полів у класах). Метод оформлений для формування одного списку і викликається кілька

разів у циклі, передаючи шлях до файлу (path), індекс списку (n) і кількість елементів у рядку файлу (m). Списки оголошують глобально (у класі Program). Метод передається індексу масиву, що містить кількість елементів у рядку файлу (m). Роздільна обробка кожного списку (за індексом) обумовлена тим, що об'єкт кожного класу відрізняється кількістю елементів, їх типами та порядком прямування.

using System.IO; // Не забудьте вставити цю збірку

Основна програма

```
static void Main(string[] args)
{
    File_Load(path); // Зчитування файлу
    // Обробка, зазначена у завданні на курсову роботу
    Console.ReadKey();
}
static List<Shop> lst1=new List<Shop>(); // Перший список
static List<Vegetable> lst2 = new List<>( Vegetable); // Другий список
static string[] path = {«C:\Workc\KP\List1.txt», «C:\Workc\KP\List2.txt»}; //
```

Шляхи

Підпрограма формування n таблиць

```
static void File_Load(string[] path, int n, int m)
{
    if (n == 0) // Перша таблиця
    {
        StreamReader FileIn = New StreamReader(path[0], Encoding.Default);
        string str;
        while ((str = FileIn.ReadLine()) != null) // Поки що не кінець файлу
        {
            string[] ms=new string[m]; // m – кількість полів у класі
            ms = str.Split(';'); // Розщеплення на масив рядків
            // Додавання рядка через конструктор з параметрами
            lst1.Add(new Shop (ms[0], ms[1], ms[2]));
            // Перетворення враховує опис класу
        } // Від while
    } // Від if
    // У разі повторного виклику методу File_Load
    // n = 1 (Друга таблиця, опущені оператори if та while)
    {string[] ms = new string[m]; // m – кількість полів у цьому класі
      ms = str.Split(';'); // Розщеплення на масив рядків
      lst2.Add(new Vegetable (Convert.ToInt32(ms[0]), ms[1],
                              Convert.ToDouble(ms[2]))); // Другий список
    }
}
```

Для конкретного завдання потрібно описати класи та відповідно змінити поля та оператори додавання елементів.

Запити у кожному завданні постійні. Для їхньої обробки оформлюється оператор switch. Реалізація запиту оформлюється як групи case, що викликає метод обробки запиту.

Для моделювання роботи з базою даних бажано запитувати номер з описом запиту. Після вибору номера натискають клавішу Enter. Бажаючі можуть спробувати змінити програму так, щоб натискати клавішу Enter після вибору номера було не потрібно. (Підказка: треба обробити подію `keybd_event`).

Для виклику запитів за номером рекомендується реалізувати наведену нижче програму. Перед запуском циклу обробки запитів у циклі формуються узагальнені списки. З цією метою об'єднуються списки (їхня кількість залежить від завдання, у прикладі 3) і створені масиви, що містять шляхи до трьох файлів та кількість полів у кожному класі. Спискам бажано надавати продумані імена.

```
static List<Vegetable> lst1 = new List<Vegetable>();// Оголосить 3 списки
static string [ ] p = { «C:\\Workc17 Vis\\KP\\List1.txt», «Інші шляхи»}; // 3 шляхи
static int [ ] m = {3, 6, 5}; // Кількість полів у класі для кожного класу
static void Main(string[] args)
{
    for (int i = 0; i < 3; ++i)
    {
        File_Load(p[i], n, m[i]);
        // p[i] – шляхи, n – індекс списку, m[i] – кількість полів у класі
    }
    int n = -1; // Позиція у switch
    while (true)
    {
        Frame();// Виведення списку для вибору запиту
        if (! int.TryParse(Console.ReadLine(), out n))
        {
            Console.WriteLine("Це не цифра, натисніть на Enter");
            Console.ReadLine();
            Console.Clear();
            continue;
        }
        switch (n)
```

```

{
    case 1:
        Console.Clear();
        // Виклик методу реалізації запиту позиції 1
        Console.WriteLine («Запит закінчено, натисніть будь-яку клавішу»);
        Console.ReadKey();
        Console.Clear();
        continue;
    case 2:
        Console.Clear();
        // Виклик підпрограми реалізації запиту позиції 2
        Console.WriteLine («Запит закінчено, натисніть будь-яку клавішу»);
        Console.ReadKey();
        Console.Clear();
        continue;
        //      Обробка інших запитів
    case 9:
        Console.WriteLine
        («Вихід із програми – так, ні – будь-яка клавіша»);
        string st = Console.ReadLine();
        if (st== “так”) Environment.Exit(0);
        else
        {
            Console.Clear();
            continue;
        }
        continue; // Без цього оператора виникає помилка default:
        Console.Clear();
        Console.WriteLine
        («Введений № відсутній, натисніть будь-яку клавішу»);
        Console.ReadKey();
        Console.Clear();
        continue;
    }
}

static void Frame()          // Метод для вибору списку запитів
{
    Console.WriteLine («\tВиберіть № завдання\n»);
    Console.WriteLine («1 – Отримати довідку про конкретного клієнта\n»);
    Console.WriteLine («2 – Сумарна вартість товару у відділі\n»);
        Console.WriteLine («\t Інші запити»);
    Console.WriteLine («\n9 – Вихід із програми»);
}
} // Дужка від class Program

```

Приклади опису класів подані на початку параграфу.

2.1 Ініціалізація об'єктів і списків

Ініціалізація об'єкта класу може виконуватися або за допомогою властивостей ({get; set;}), або за допомогою конструктора. Якщо ініціалізація виконується за допомогою конструктора, описи властивостей не потрібні. Якщо для ініціалізації застосовуються властивості, конструктор не потрібен. У деяких випадках система вимагає опису порожнього конструктора, який наведено в описах класів.

Якщо клас містить масив, ініціалізація такого масиву виконується лише за допомогою конструктора. Нижче наведено приклад оголошення класу, що містить масив, та конструктор для ініціалізації цього класу.

```
class AZS//
{
    public string Firm;
    public string [] Benz = new string [3]; //
    public AZS (string f, params string [] b) //
    {
        Firm = f; Address = a; GasKey = g;
        'Benz[0] = ""; Benz[1] = ""; Benz [2] = ""; // Цей рядок не є обов'язковим
        Benz[0] = b[0]; Benz[1] = b[1]; Benz[2] = b[2];
    }
}
```

Зверніть увагу на модифікатори params. Такий модифікатор може бути лише один, тому масив у класі зазвичай також один. Але якщо два масиви мають однаковий тип, то індекси змінної з модифікатором params (у прикладі g) можна продовжити для ініціалізації другого масиву (у якому індекси від 0, а індекси g прямують за максимальним індексом першого масиву).

Ініціалізація списку з елементами, що містять масив, наведена нижче.

```
static List<AZS> az = new List<AZS>() // Оголошення порожнього списку
{
    new AZS(«Роснефть», «98», «», «92»), // Перший елемент списку //
    new AZS(«ТНК», «98», «95», «»), // Другий елемент списку
}; // Крапка з комою тут обов'язкова
```

2.2 Приклади запитань

Зберігати результати запитів не потрібно. Після реалізації кожного запиту дані виводяться на екран. Після перегляду виконується стирання екрана, й інформація буде відновлена під час наступного запиту.

Нижче наведено найбільш поширені запити LINQ.

2.2.1 Сорткування, проєкція, фільтрація

У прикладах запитів: `lst` узагальнений список для класу **Vegetable** (див. вище). `w` – ім'я змінної для лямбда-вираження (це ім'я може бути будь-яким іншим).

Вибірка всіх стовпців:

```
var qr = lst.Select(w => w);
```

Вибірка всіх стовпців із сортуванням (за зростанням):

```
var qr = lst.Select(w => w).OrderBy(w => w.Name);
```

Вибірка двох шпальт (проєкція):

```
var qr = lst.Select(w => new {w.Name, w.Price});
```

Під час виконання запиту **GroupJoin** зазвичай виходить владений список. Запит **Select** виконує вибірку лише на одному рівні. Тому під час запиту **Select** вкладений список видно, але доступу до його полів немає. Щоб отримати доступ до полів вкладеного списку, всередині анонімного методу застосовують запит **SelectMany**:

```
var qr = lst.Select(rl => new {k = rl.Key, slst = rl.SelectMany(c => c.lst) });
```

Тут `ls` містить вкладений список, `k` – нове ім'я для ключа угруповання, `slst` – нове ім'я вкладеного списку; тепер його поля доступні в операторі **foreach**.

2.2.2 Фільтрування (вибірка рядків за умовою)

В одному запиті можна задати умову лише для одної властивості (поля).

Попадання в діапазон значень:

```
var qr = lst.Where( w => w.num > 2 && w.num <5); //
```

Додати умову для другої властивості.

(Порівняння для другої властивості записується через точку після першого порівняння. Це забезпечує операцію «І» (&&) між логічними операціями для кожного поля).

```
var qr=lst.Where(w=>w.Quan>2&&w.Quan<5).Where(b=>b.Price<4);
```

Якщо у другій перевірці b замінити на a, нічого не зміниться.

2.2.3 Виділення частини списку та пошук

Оператори **Take**, **Skip**, **TakeWhile** і **SkipWhile** дозволяють або вибирати рядки з початку списку або пропускати.

Взяти перші три записи від початку списку:

```
var qr = lst.Take(3);
```

Вибрати з початку списку за умовою:

```
var qs = lst.TakeWhile(w => w.Price < 50);
```

Пошук одиночного значення:

```
var qr = lst.FirstOrDefault( w => w.Name=="Помідор");
```

```
Console.WriteLine("{0}\t{1}\t{2}", qr.Name, qr.Price, qr.Quan);
```

Запит **FirstOrDefault** вибирає перший елемент у списку із заданим для пошуку значенням. Якщо такого елемента не знайдено, то виводиться перший елемент зі списку. Зверніть увагу, що під час пошуку вибирається об'єкт класу, а не одиночне значення.

Запит **LastOrDefault** вибирає останній елемент у списку з заданим для пошуку значенням. Якщо такий елемент не знайдено, виводиться останній елемент зі списку.

Для пошуку всіх елементів із заданим для пошуку значенням можна застосовувати запит **FindAll**, але, напевно, краще в цьому випадку застосовувати запит **Where**.

2.2.4 Операції з колекціями

Під час роботи з колекціями виконуються операції, характерні для операцій з множинами, а також операції з угруповання та об'єднання колекцій. Операції можуть стосуватися колекцій з однаковим типом даних

та колекцій з різними типами даних. До першої групи належать такі операції:

Concat	Об'єднання (конкатенація) двох колекцій
Distinct	Виняток зі списку повторюваних елементів
Except	Видалення з одного списку елементів іншого списку
Intersect	Перетин двох колекцій
Union	Об'єднання двох колекцій з видаленням повторюваних елементів

2.2.5 Операції з множинами

Для виконання операцій з колекціями типів даних користувача потрібно порівнювати між собою об'єкти класу. Тому клас необхідно змінити (додати клас інтерфейсу та додати методи для порівняння властивостей). Клас набуде вигляду:

```
public class Vegetable : IEquatable<Vegetable> // Овочі
{
    public string Name { get; set; } // Назва
    public double Price { get; set; } // Ціна
    public int Quan { get; set; } // Кількість
    public Vegetable(string n, double p, int u)
    {
        Name = n; Price = p; Quan = u;
    }
    public bool Equals(Vegetable other) // Перевизначення Equals
    {
        // Перевірити, чи порівнюваний об'єкт значення null.
        if (Object.ReferenceEquals(other, null)) return false;
        // Перевірити, чи посилаються порівнювані об'єкти на одні й ті самі дані
        if (Object.ReferenceEquals(this, other)) return true;
        // Перевірити, чи рівні властивості товарів
        return name.Equals(other.Name) && Price.Equals(other.Price);
    }
    public override int GetHashCode() {
        //Отримати хеш-код для поля Name, якщо воно не дорівнює null.
        int hashVegName = Name == null? 0 : name.GetHashCode();
        // Отримати хеш-код для поля price.
        int hashVegPrice = Price.GetHashCode(); // Обчислити хеш-код товару.
        return hashVegName ^ hashVegPrice;
    }
}
```

Тепер запит, наприклад, на видалення повторюваних елементів має вигляд:

```
var qr = lst.Distinct();
```

У цьому прикладі видаляються елементи, у яких збігається назва та ціна.

2.2.6 Об'єднання та угруповання колекцій

Під час об'єднання колекцій з даними користувача можливе як внутрішнє, так і зовнішнє об'єднання. Для об'єднання таблиць має бути обраний ключ. У разі внутрішнього об'єднання кожному рядку в одній таблиці відповідає один рядок в іншій таблиці. Зовнішнє ліве об'єднання дозволяє зібрати нову таблицю зі стовпців так, що в новій таблиці повторюються рядки з однієї таблиці з однаковим значенням ключа в іншій таблиці. Якщо після об'єднання виконати угруповання (за рядками, що повторюються), виходить аналог відношення «один до багатьох». Нижче наведено приклад лівого зовнішнього об'єднання, рисунок 2.1.



Рисунок 2.1 – Ліве зовнішнє об'єднання

Праве зовнішнє об'єднання (відношення багатьох до одного) не розглядається.

Нижче наведено приклад об'єднання двох класів (**Shop** і **Vegetable**) за ключами **cKey** (магазин), **oKey** (товар). (**Shop** – лівий список, змінна **w** для лівого списку, змінна **b** для правого списку):

```
var qr = cst.Join(lst, w => w.cKey, b => b.oKey, (w, b) => new
{
    w.ShopName,
    b.Name,
```

```

        b.Price,
        w.Quan,
    }); // Можна відсортувати .OrderBy( w => w.ShopName);
    string[ ] st = { «Магазин», «Овочі», «Ціна», «Кількість» }; // Заголовки
    Console.WriteLine("{0}\t{1}\t{2}\t{3} \n", st[0], st[1], st[2], st[3]);
    foreach (var v in qr) // Виведення нової таблиці
        Console.WriteLine("{0} \t{1,-10} \t{2} \t{3}",v.Name, v.Price, v.Quan,);

```

Під час виведення таблиці зверніть увагу на змінну *v*, вона однакова для властивостей з лівої та правої таблиць, оскільки вся таблиця тепер єдина (об'єднана).

Принциповим є отримання нового типу даних (нової таблиці), запити LINQ до якої дозволяють реалізувати досить складні операції, в яких беруть участь дві таблиці.

Нижче наведено приклад застосування групування до підсумкової таблиці (у прикладі операції **Join** вище була отримана таблиця з ім'ям **qr**).

Зверніть увагу: після угруповання з результату вилучається ключ та список для кожного ключа.

```

var grp = qr.GroupBy(w => w.ShopName).Select(rs =>new
{
    kgr = rs.Key, // Ключ групи
    sng = rs      // Список групи
});
foreach (var res in grp)
{
    Console.WriteLine("\nЗамовник: " + res.kgr); // Виведення
ключів
    foreach (var sp in res.sng) // Цикл виведення списку групи
        Console.WriteLine("{0}\t{1}\t{2}", sp.Name, sp.Price, sp.Quan);
}

```

Імена змінних *res* і *sp* в операторах *foreach* довільні, але змінна *res* із зовнішнього циклу дозволяє за допомогою точкової нотації звернутися до списку, що належить до групи.

До списку групи за допомогою точкової нотації можна застосовувати запити, що агрегують.

Угруповання може виконуватися з різних ключів і може мати кілька рівнів.

2.2.7 Агрегуючі запити

Основні операції для агрегувальних запитів: Average (середнє значення), Aggregate (операції одного елемента з іншими), Count (кількість елементів у списку), Max, Min, Sum. Ці операції повертають поодинокі значення для конкретного списку даних. Наприклад, оператор:

```
int n = qs.Count();
```

підраховує кількість елементів у списку QS.

3 ЗАВДАННЯ НА КУРСОВУ РОБОТУ

Завдання на курсову роботу з дисципліни «Об'єктно-орієнтоване програмування» реалізуються у формі консольних програм у середовищі Visual Studio мовою C#. Узагальнені списки та колекції є локальними. Вони формуються з файлів, реалізованих у додатку «Блокнот».

Завдання № 1 (Готель)

Потрібно розробити програмну систему, призначену для адміністратора готелю. Така система повинна забезпечувати збереження відомостей про наявні в готелі номери, про клієнтів, що проживають у готелі, і про працівників, які прибирають у номерах. Кількість номерів у готелі відома, і є номери трьох типів: одномісний, двомісний та тримісний, що відрізняються вартістю проживання за добу. Клієнту надається місце у номері. Оплата за місце становить частку вартості номера залежно від кількості місць. Номер сплачується після прибуття. Працівник готелю прибирає всі номери на одному поверсі у певні дні тижня, при цьому в різні дні він може прибирати різні поверхи.

Готель	Клієнт	Працівник
Номер	Ключ	Ключ
Поверх	ПІБ	ПІБ
Тип номера	Номер паспорта	Список поверхів
Вартість за добу	Звідки	Список днів тижня
Список клієнтів	Номер	
Список співробітників	Місце	
	Коли прибув	
	Оплата (доба)	

Повинні бути створені узагальнені списки:

- відомості про готель;
- відомості про клієнтів;

– відомості про працівників.

Повинні бути реалізовані такі запити:

– вартість місця для заданого поверху та номера;

– список клієнтів, що прибули із заданого міста;

– хто з працівників прибирав номер вказаного клієнта у заданий день тижня;

– скільки у готелі вільних місць та вільних номерів;

– клієнти, які мешкають в одномісних номерах;

– загальна сума, виплачена всіма клієнтами.

Адміністратор повинен мати можливість (використовуючи запити LINQ) виконати таку операцію: взяти на роботу або звільнити працівника готелю (видалити зі списку).

Завдання № 2 (Бібліотека)

Потрібно розробити програмну систему, призначену для працівників бібліотеки. Така система повинна забезпечувати збереження відомостей про наявні в бібліотеці книги, читачів бібліотеки та читальні зали.

Бібліотека має кілька читальних залів, які характеризуються номером, спеціалізацією та кількістю місць. Читачі закріплюються за певною залом (загальною, фізико-математичною тощо) і можуть записуватися та виписуватися з бібліотеки. Один читач може взяти кілька книжок. Шифр книги задається під час отримання нової книги, а номер квитка читача обирається під час реєстрації.

Книга	Читач	Зал
Ключ	Ключ	Ключ
Назва	ПІБ	Назва бібліотеки
Автор	Номер квитка	Зал
Рік видання	Дата народження	Спеціалізація
Шифр книги	Телефон	Кількість місць

Дата видачі	Освіта	Список читачів
Дата отримання	Зал	
Кількість екземплярів	Список книг	

Необхідно створити узагальнені списки:

- відомості про книги;
- відомості про читачів;
- відомості про читальні зали.

Бібліотекарю можуть знадобитися такі відомості про поточний стан бібліотеки:

- які книжки видано кожному читачеві;
- скільки вільних місць у кожному залі;
- чи можна видати книгу читачеві (є вільні екземпляри);
- кількість книг заданого автора у читальному залі;
- читачі, які взяли книги, що наявні в одному примірнику;
- книги з максимальним рейтингом.

Бібліотекар може виконувати такі операції:

- записати до бібліотеки нового читача;
- списати стару чи втрачену книгу;
- прийняти книгу до фонду бібліотеки.

Завдання № 3 (Школа)

Потрібно розробити програмну систему, призначену для завуча школи. Вона повинна забезпечувати зберігання відомостей про кожного вчителя, предмети, які він викладає, номер закріпленого за ним кабінету. Про учнів повинні зберігатися такі відомості: прізвище та ім'я, у якому класі вчиться, яку оцінку має у поточній чверті з кожного предмета. Завуч повинен мати можливість додати відомості про нового вчителя або учня, внести до бази даних четверті оцінки учнів кожного класу з кожного предмета, видалити дані про звільненого вчителя та відрахованого зі школи учня.

Вчитель	Учень	Успішність
Ключ	Ключ	Ключ
ПІБ	Учень	Учень
Предмет	Клас	Предмет 1
Номер кабінету		Предмет 2
		Предмет 3

Успішність задається числами від 2 до 5. Кількість предметів може бути збільшена.

Необхідно створити узагальнені списки:

- відомості про вчителів;
- відомості про учнів;
- відомості про успішність.

Завуч може вимагати таких відомостей про поточний стан успішності:

- успішність із заданого предмета;
- кількість неуспішних учнів з усіх класів;
- у якого вчителя найнижча успішність;
- середня оцінка з усіх предметів у кожному класі;
- клас із найвищою успішністю з усіх предметів;
- клас із найнижчою успішністю з усіх предметів.

Завуч може виконувати такі операції:

- записати до школи нового учня;
- відрахувати зі школи учня.

Завдання № 4 (Виставка собак)

Потрібно розробити програмну систему, призначену для організаторів виставки собак. Вона повинна забезпечувати збереження відомостей про собак – учасників виставки та експертів. Для кожної собаки повинні зберігатися відомості про те, до якого клубу він належить, прізвисько, порода та вік, відомості про родовід (номер документа,

прізвиська батьків), дату останнього щеплення, прізвище, ім'я, по батькові та паспортні дані господаря.

Відомості про експерта повинні містити прізвище та ім'я, спеціалізацію за породою, номер рингу, який він обслуговує, назва клубу, де він працює. Кожен ринг має спеціалізацію пород, і його можуть обслуговувати кілька експертів. Кожна порода собак виступає на своєму рингу, але на тому самому рингу в різний час можуть виступати різні породи.

Підсумком виставки є визначення медалістів з кожної породи. Організатор виставки повинен мати можливість додати до бази нового учасника або нового експерта, зняти експерта з суддівства, замінивши його іншим, усунути собаку від участі у виставці.

Собака	Ринг	Експерт
Ключ	Ключ	Ключ
Прізвисько	Номер (назва)	ПІБ
Порода	Адреса	Спеціалізація
Вік	Масив спеціалізацій	Номер рингу
Клуб	Клуб	Клуб
ПІБ господаря	Список експертів (ключ)	
Місце (медаль)	Список собак (ключ)	

Необхідно створити узагальнені списки:

- відомості про собак;
- відомості про ринги;
- відомості про експертів;

Організатору виставки можуть знадобитися такі відомості:

- на якому рингу виступає заданий господар зі своїм собакою;
- які породи є у заданому клубі;
- які медалі та скільки має клуб;
- які фахівці обслуговують породу;
- розподіл спеціалізації за рингами;

– рекордсмени за породами та кількістю медалей.

Голова клубу може виконувати такі операції:

- взяти до клубу нового собаку;
- усунути від змагань собаку;
- взяти до клубу нового експерта.
- звільнити з клубу експерта.

Завдання № 5 (Прийом до ВНЗ)

Потрібно розробити програмну систему, призначену для працівників приймальної комісії вищого навчального закладу. Вона повинна забезпечувати зберігання, перегляд та зміну відомостей про абітурієнтів, а також розклад іспитів та консультацій. Результатом роботи приймальної комісії має бути список абітурієнтів, зарахованих до інституту.

Спеціальність задається назвою кафедри та факультету, на які вступає абітурієнт. Під час реєстрації абітурієнту видають екзаменаційний лист. Абітурієнти на період вступних іспитів об'єднуються в групи. Кожна група складає по три іспити, за якими формується екзаменаційна відомість. Для кожного потоку формується розклад консультацій та іспитів з предметів.

Медалістам зараховують усі іспити. Відомо кількість місць на кожній кафедрі. Приймальна комісія за результатами екзаменів має сформувати списки абітурієнтів, зарахованих до інституту. Для обчислення прохідного бала абітурієнта долучають до списку зарахованих зі списку результатів іспиту, відсортованого за зменшенням оцінок, доки не буде досягнуто кількості наявних місць на обраній кафедрі.

Абітурієнт	Екз. лист	Іспити
Ключ	Ключ	Ключ
Абітурієнт	ПІБ	Розклад
ПІБ	Предмет 1	Група

Група	Предмет 2	Іспит / Консультація
Закінчив	Предмет 3	Число
Медаль		Час
Кафедра		Аудиторія
Факультет		

Необхідно створити узагальнені списки:

- відомості про абітурієнтів;
- відомості про розклад;
- відомості про екзамени;
- масив місць на кафедрах.

Секретарю приймальної комісії можуть знадобитися такі відомості:

- список абітурієнтів із груп з урахуванням кафедри;
- прохідний бал за всіма спеціальностями;
- консультація та іспит із заданого предмета;
- списки зарахованих абітурієнтів;
- конкурс на кожну спеціальність;
- списки абітурієнтів, які не пройшли за конкурсом;
- кафедри, за якими був недобір абітурієнтів.

Необхідно також передбачити можливість отримання документа зі згрупованими за кафедрами списками абітурієнтів, зарахованих до інституту, із зазначенням набраних ними балів з кожного предмета. Звіт повинен містити прохідний бал за кожною кафедрою, факультетом та інститутом в цілому, а також кількість абітурієнтів, які вступають на кафедру та інститут.

Завдання № 6 (Пошта)

Потрібно розробити програмну систему, що дозволяє відстежувати розподіл за поштовими відділеннями газет, що друкуються у друкарнях міста. Система повинна забезпечувати зберігання, перегляд та зміну відомостей про газети, поштові відділення, які отримують газети та про

друкарні, що випускають газети. Відомості про газети містять: назву газети, індекс видання, прізвище, ім'я та по батькові редактора, ціну екземпляра газети. Ціни можуть змінюватись. Можлива поява нових газет та зміна індексу існуючого видання. Для друкарень вказуються їх назви та адреси. У друкарні різними накладками друкуються газети кількох найменувань. Поштове відділення має номер та адресу. На кожне поштове відділення надходять у певних кількостях газети різних найменувань,

Газета	Друкарня	Пошта
Ключ	Ключ	Ключ
Назва	Назва	Номер
Індекс	Адреса	Адреса
Редактор	Список газет	Список газет
Наклад		Список типографій
Ціна		

Необхідно створити узагальнені списки:

- відомості про газети;
- відомості про друкарні;
- відомості про поштові відділення.

Користувачам системи може знадобитися така інформація:

- у яких друкарнях друкуються газети цієї назви;
- прізвище редактора газети, яке друкується у зазначеній друкарні найбільшим накладом;
- якою є загальна вартість накладу заданої газети;
- до якого поштового відділення надходить найбільше газет;
- до яких поштових відділень надходить ця газета, що друкується у визначеній друкарні;
- у якому поштовому відділенні максимальна вартість надрукованих газет.

Повинна бути передбачена можливість додавання нової газети, видалення газети та зміни її індексу.

Завдання № 7 (Птахофабрика)

Потрібно розробити програмну систему, орієнтовану на адміністрацію птахофабрики, що дозволяє працювати з інформацією про працівників фабрики та про наявних на ній птахів.

Про кожну курку повинна зберігатися така інформація: вага, вік, кількість яєць, що щомісяця отримують від курки, а також інформація про місцезнаходження курки. На птахофабриці за кожною куркою закріплено окрему клітку.

Про працівників птахофабрики у базі даних повинна зберігатись така інформація: паспортні дані, зарплата, закріплені за працівником клітки. Не повинно бути курей, які не обслуговуються жодним працівником. Кількість курей може змінюватися як у більший, так і в менший бік, в окремі моменти часу частина кліток може бути порожньою. Ціна яєць однакова для всіх курей. Як дати потрібно застосовувати дні одного місяця.

Курка	Птахофабрика	Працівник
Ключ	Номер клітки	ПІБ
Клітина	Дата	Клітки (ключ)
Вік	Яйце знесене?	Зарплата
Вага		
Несучість		
Порода		

Необхідно створити узагальнені списки:

- відомості про курок;
- відомості про птахофабрику;
- відомості про працівників.

Директору можуть знадобитися такі відомості:

- середня кількість яєць, що отримується від кожної курки заданої ваги та віку;

- загальна кількість отриманих яєць за діапазон дат та їхня спільна вартість;
- кількість яєць, зібраних кожним працівником.
- курки, що знесли кількість яєць нижче середньої несучості по фабриці;
- у якій клітці знаходиться курка, від якої отримали більше всього яєць;
- яку кількість курей обслуговує кожен працівник.

Повинна бути передбачена можливість додавання та видалення курки.

Завдання № 8 (Продмаг)

Потрібно розробити програмну систему, що призначена для директора продовольчого магазину. Така система повинна забезпечувати зберігання відомостей про магазин, наявні в ньому товари, відділи та продажі. Кожен відділ спрямований на продаж певного виду товарів (кондитерський, гастроном, бакалія тощо). Товари характеризуються ціною та сортом. За кожним товаром задається кількість проданих одиниць і наявність, що залишилася.

Товар	Відділ	Продажі
Ключ	Ключ	Ключ
Назва	Назва	Товар
Відділ	Товар	В наявності
Сорт		Продано
Ціна		

Необхідно створити узагальнені списки:

- відомості про товари;
- відомості про відділи;
- відомості про продаж.

Директору можуть знадобитися такі відомості:

- список товарів у магазині відділів;
- яких товарів у відділах немає;
- відділи з максимальним та мінімальним прибутком;
- продаж товарів у відділах;
- який товар дав максимальний прибуток у магазині;
- вартість товарів для поповнення.

Необхідно передбачити можливість додавання, видалення та зміни товару.

Завдання № 9 (Автобусний парк)

Потрібно розробити програмну систему, що призначена для диспетчера автобусного парку. Така система повинна забезпечувати збереження відомостей про водіїв, про маршрути та характеристики автобусів.

Кожен водій характеризується паспортними даними, класом, стажем роботи та платнею, причому платня залежить від класу та стажу роботи. Маршрут автобуса характеризується номером маршруту, назвою початкового та кінцевого пункту руху, часом початку та кінця руху, інтервалом руху та протяжністю в хвилинали (часом руху по всьому маршруту).

Характеристиками автобуса є: його тип, місткість та державний номерний знак. Кожен водій закріплений за окремим автобусом і працює на діловому маршруті, але у разі поломки свого автобуса або хвороби іншого водія може пересісти на іншу машину. У базі повинен зберігатись графік роботи водіїв у вигляді масиву днів виходу на роботу. Необхідно передбачити можливість коригування списків у разі надходження на роботу нового водія, списання старого автобуса, запровадження нового маршруту або зміни старого тощо.

Водій	Маршрут	Автобус
Ключ	Ключ	Ключ
ПІБ	Номер	Держ. номер
Стаж	Початок	Тип
Клас	Кінець	Місткість
Оклад	Час відправлення	Справний
Маршрут	Час прибуття	
Графік	Автобус (держ. номер)	

Необхідно створити узагальнені списки:

- інформацію про водіїв;
- відомості про маршрути;
- відомості про автобуси.

Графік задається вкладеним масивом, що містить номери днів тижня.

Диспетчеру автопарку можуть знадобитися такі відомості:

- список водіїв, які працюють на певному маршруті з вказівкою графіка їх роботи;
- які автобуси обслуговують цей маршрут;
- протяжність маршрутів та їхня мінімальна та максимальна довжина;
- на якому маршруті працює водій із максимальним стажем;
- яка загальна протяжність маршрутів, що обслуговуються автопарком.
- які водії не вийшли на лінію через несправність автобуса.

Директор автопарку повинен мати можливість додати водія чи автобус та видалити водія чи автобус.

Завдання № 10 (Кінотеатри)

Потрібна розробка програмної системи, призначеної для працівників довідкової служби кінотеатрів міста. Така система повинна забезпечувати

збереження відомостей про кінотеатри міста, про фільми, що в них демонструються, про сеанси та квитки на ці сеанси. На різних сеансах в одному кінотеатрі можуть демонструватися різні фільми, а якщо в кінотеатрі кілька залів, то й на одному.

Кінотеатр	Фільм	Репертуар
Ключ	Ключ	Ключ
Назва	Назва	Дата
Адреса	Режисер	Сеанс
Категорія	Оператор	Ціна
Кількість місць	Список акторів	Вільних місць
Кількість залів	Жанр	
Стан	Кіностудія	

Необхідно створити узагальнені списки:

- відомості про кінотеатри;
- відомості про фільми;
- відомості про репертуар.

Довідковій службі можуть знадобитися такі відомості про поточний стан прокату фільмів у місті:

- репертуар кінотеатру;
- у яких кінотеатрах можна переглянути бойовики;
- кількість вільних місць на цей сеанс у заданому кінотеатрі;
- ціна квитків на заданий сеанс у вказаному кінотеатрі;
- які фільми заданого режисера демонструються у кінотеатрах;
- у яких кінотеатрах демонструються комедії.

Необхідно передбачити можливість додавання та видалення фільму.

Завдання № 11 (Футбол)

Потрібно розробити програмну систему, призначену для організаторів змагань із футболу в рамках першості країни. Така система повинна забезпечувати збереження відомостей про команди, які беруть

участь у першості, про гравців команд, про розклад зустрічей та їхні результати, про ціну квитків на ігри.

Відомості про команду: назва команди, місто, де вона базується, ім'я головного тренера, місце у таблиці минулого сезону, розпис зустрічей. В один день команда може брати участь лише в одній зустрічі.

Відомості про гравця містять прізвище та ім'я гравця, його вік, номер і амплуа в команді.

Відомості про стадіон, на якому відбувається зустріч, містять місто, в якому він знаходиться, назву стадіону та його місткість. Ціна квитка на матч залежить від місткості стадіону та положення команд, що зустрічаються в турнірній таблиці минулого сезону (найбільша – у разі гри трійки призерів, найменша – у разі гри трійки аутсайдерів). Організатори змагань повинні мати можливість внести зміни до даних про склад команд, перенести зустріч.

Команда	Гравець	Стадіон	Матчі
Назва	Ключ	Ключ	Ключ
Місто	ПІБ	Назва	Команда
Тренер	Вік	Місто	Дата
Місце у рейтингу	Амплуа	Місткість	Рахунок
Гравці	Номер	Ціна квитка	

Стадіони

Необхідно створити узагальнені списки:

- відомості про гравців;
- відомості про команду;
- відомості про стадіон;
- відомості про матчі.

Повинні бути реалізовані такі запити:

- дати зустрічей зазначеної команди, її супротивники та рахунок;
- номери та прізвища гравців команд, що брали участь у зустрічі, яка проходила у вказаний день на стадіоні;

- ціна квитка на матч між вказаними командами;
- команди, які мають найкращу та найгіршу різницю забитих та пропущених м'ячів;
- команди, які посіли призові місця;
- розклад ігор на стадіонах.

Повинна бути передбачена можливість додавання та видалення гравців та матчів.

Завдання № 12 (Інститут)

Потрібно розробити програмну систему, призначену для працівника методичного відділу інституту. Така система має забезпечувати зберігання відомостей про спеціальності, за якими веде підготовку кафедра, про дисципліни, що входять до переліку підготовки з кожної спеціальності. Відомості про спеціальність: код та назва спеціальності, кваліфікація та форма навчання (денна, вечірня, заочна).

Відомості про кафедру містять її назву, телефон (телефони), факультет, до якого належить кафедра. Відомості про дисципліну: назва дисципліни, семестр (семестри) та спеціальності, для яких вона читається, структура занять (скільки годин для кожної спеціальності відводиться на лекції, лабораторні та практичні заняття з цієї дисципліни; види звітності: залік, екзамен, курсова робота).

Спеціальність	Кафедра	Дисципліна
Ключ	Назва	Назва
Назва	Факультет	Семестр
Спеціальність	Телефон	Годинник
Форма навчання	Спеціальності	Структура
		Звітність

Необхідно створити узагальнені списки:

- відомості про спеціальність;
- відомості про кафедру;

- відомості про дисципліну.

Структура дисципліни задається вкладеним масивом, що містить три значення (кількість годин на тиждень: лекції, семінари, лабораторні роботи, наприклад, 2, 1, 4).

Співробітнику методичного відділу можуть знадобитися такі відомості:

- список дисциплін для заданої спеціальності.
- кількість годин лабораторних робіт для заданої спеціальності.
- семестрове навантаження кафедри зі спеціальності.
- дисципліни, з яких веде заняття кафедра.
- дисципліни з мінімальною та максимальною тривалістю занять.
- загальна кількість іспитів та заліків на кафедрі (включно з курсовою роботою).

Має бути передбачена можливість додавання або видалення спеціальності.

Завдання № 13 (Деканат)

Потрібно розробити програмну систему, призначену для працівника деканату. Така система повинна забезпечувати збереження відомостей про групи та студентів, а також результати поточної сесії.

Студент	Група	Предмет
Ключ	Ключ	Ключ
ПІБ	Факультет	Весняна / Зимова
Номер заліковки	Група	Назва
Адреса	Спеціальність	Звітність (залік, прим.)
Стипендія	Кафедра	Оцінка

Необхідно створити узагальнені списки:

- відомості про студента;
- відомості про групу;
- відомості про предмет;

Працівнику деканату можуть знадобитися такі відомості:

- середній бал студентів з кожної групи зазначеного факультету;
- середній бал за кожним предметом;
- група з максимальним середнім балом за заданим предметом;
- список факультетів студентів, які підлягають відрахуванню (не складено іспити з більш як двох предметів);
- з якого предмета найбільше незадовільних оцінок.

Необхідно передбачити можливість додавання та відрахування студента.

Завдання № 14 (Поліклініка)

Потрібно розробити програмну систему, призначену для лікарів та працівників реєстратури поліклініки. Така система повинна зберігати відомості про дільниці, які належать до поліклініки, про розклад роботи дільничних лікарів, інформацію про лікарів, а також картки пацієнтів.

Картка має номер, до неї додаються відомості про кожне відвідування поліклініки пацієнтом: дата відвідування, скарги, попередній діагноз, призначення, виписаний чи ні лікарняний лист, і якщо виписаний, то на який термін, ім'я лікаря. У картці на першій сторінці вказано також прізвище, ім'я, по батькові пацієнта, його домашню адресу, стать та вік, номер страхового полісу, дату заповнення картки. У розкладі роботи лікарів зазначається, на якій ділянці працює лікар, дні та години прийому, номер кабінету. Лікар може обслуговувати більше однієї ділянки. У разі звільнення лікаря, його ділянка передається іншим лікарям.

Дані про лікаря, які зберігаються у базі даних: це прізвище, ім'я та по батькові, категорія, стаж роботи, дата народження. У картку хворого під час кожного відвідування поліклініки лікар додає черговий запис. Працівники реєстратури реєструють пацієнта, заповнюючи першу сторінку його картки. У разі звільнення лікаря його хворих передають іншому лікарю.

Ділянка	Пацієнт	Медпрацівник
Ключ	Ключ	Ключ
Назва	ПІБ	ПІБ
Адреса	Адреса	Категорія
	Підлога	Стаж
	Вік	Вік
	Номер полісу	Спеціалізація
	Дата відвідування	Ділянка
	Скарга	Розклад
	Діагноз	Кабінет
	Номер полісу	Розклад прийому

Необхідно створити узагальнені списки:

- відомості про ділянки.
- картки пацієнтів.
- відомості про лікарів.

«Розклад» прийому задається у вигляді масиву днів тижня (не більше трьох разів на тиждень). Дати задаються у межах днів одного місяця.

Працівникам поліклініки можуть знадобитися такі відомості:

- списки пацієнтів, які обслуговує цей лікар;
- скільки медсестер на кожній ділянці;
- розклад прийому всіх лікарів заданої спеціалізації;
- адреса хворого, дата останнього відвідування поліклініки та діагноз;
- прізвище та ініціали лікаря цього хворого;
- яку кількість хворих обслуговував за минулий місяць який з лікарів поліклініки.

Повинна бути передбачена можливість звільнення лікаря та додавання лікаря.

Завдання № 15 (СТО)

Потрібно розробити програмну систему, призначену для диспетчера станції техобслуговування. Така система повинна забезпечувати збереження відомостей про послуги, що надаються станцією та їхню вартість, про клієнтів станції, про працівників станції та про автомобілі, які вони ремонтують у поточний момент.

Клієнт станції – це людина, яка хоча б раз скористалася послугами станції. Про клієнта повинні зберігатися відомості з зазначенням автомобілів, які він здавав у ремонт. Після повернення автомобіля клієнту дані про проведений ремонт поміщаються в архів, клієнт отримує рахунок, у якому міститься перелік усунених несправностей із зазначенням часу роботи, вартості роботи та вартості запчастин.

Власник	Автомобіль	Працівник
Ключ	Ключ	Ключ
ПІБ	Марка	ПІБ
Номер паспорта	Колір	Спеціальність
Адреса	Рік випуску	Розряд
Вік	Держ. номер	Список дефектів
Марка авто		Дата
Дата звернення		Ціна роботи
Дата отримання		Ціна запчастин

Необхідно створити узагальнені списки:

- відомості про власників автомобілів;
- відомості про автомобілі;
- відомості про працівників.

Список дефектів може бути поданий як вкладений масив з номерами несправностей.

Диспетчеру можуть знадобитися такі відомості:

- хто обслуговував власника цього автомобіля;
- марка та рік випуску автомобіля даного власника;

– перелік усунених несправностей у автомобілі цього власника та їхня вартість;

– власники, які здали у ремонт автомобілі з зазначеним типом несправності;

– найпоширеніша несправність в автомобілях указаної марки;

– працівник із максимальним виробленням.

Має бути передбачена можливість додавання автомобіля (для заданого клієнта) та видалення клієнта.

Завдання № 16 (Менеджер музичних гуртів)

Потрібно розробити програмну систему, призначену для менеджера музичних груп. Така система повинна забезпечувати збереження відомостей про групи, що містять назву групи, рік створення та країну, склад виконавців, становище в останньому хіт-параді, репертуар групи. Відомості про кожну пісню з репертуару гурту: її назву, композитора, автора тексту.

Необхідно також зберігати дані про останню гастрольну поїздку кожної групи: назву гастрольної програми, назви населених пунктів, дату початку та закінчення виступів, середню ціну квитка (залежить від місця виступу та розташування групи в хіт-параді).

Група	Пісня	Гастролі
Ключ	Ключ	Ключ
Назва	Назва	Місто
Рік створення	Композитор	Дата початку
Країна	Автор тексту	Дата кінця
Репертуар	Рік створення	
Рейтинг	Співак	

Необхідно створити узагальнені списки:

– відомості про групи;

– відомості про пісні;

– відомості про гастролі.

Дата задається не більше одного місяця. Менеджеру можуть знадобитися такі відомості:

- які пісні виконували на гастролях заданої групи;
- які гурти виконують пісні заданого композитора;
- автор тексту, композитор та дата створення пісні з цією назвою, та група, до репертуару якої вона входить;
- репертуар найпопулярнішої групи;
- місце та тривалість гастролей групи з заданою назвою;
- які пісні виконує певний співак.

Повинна бути передбачена можливість додавання та видалення пісень заданої групи.

Завдання № 17 (Архів)

Потрібно розробити програмну систему, призначену для працівників технічного архіву підприємства. Технічний архів містить стелажі, полиці та осередки, в яких зберігається документація. Осередок архіву може бути порожнім або зберігати всі екземпляри одного документа. Кожен екземпляр документації має інвентарний номер та назву.

У базі даних повинна зберігатися така інформація про кожен документ архіву: номер стелажа, номер полиці, номер осередку, де зберігається документ, назва документа та назва теми, до якої він належить, його інвентарний номер, кількість екземплярів документа, що містяться в осередку, дата надходження документа до архіву. Документ може бути затребуваний абонентом архіву. Працівники архіву, надаючи документ абоненту, фіксують, коли і кому він виданий.

Архів	Абонент	Документ
Ключ	Ключ	Ключ
Стелаж	ПІБ	Назва

Полиця	Відділ	Тема
Комірка	Телефон	Інвентарний номер
Код осередку	дата отримання	Код осередку
Заповнення		Кількість
		Дата надходження
		Дата видачі (список)

Необхідно створити узагальнені списки:

- відомості про архів;
- інформація про абонентів;
- відомості про документи.

Дата задається вдень у межах місяця, дати видачі (у кількості до 4-х) оформлені як вкладений масив.

Працівнику архіву можуть знадобитися такі відомості:

- у яких осередках зберігаються затребувані абонентом документи;
- хто із абонентів отримав документи на задану тему;
- максимально заповнений осередок;
- абонент, який брав зазначений документ останнім;
- чи є в архіві порожні стелажі, полиці, осередки та в якій кількості;
- список документів, які не потрібні протягом заданого терміну.

Необхідно передбачити можливість додавання та видалення документа в заданий осередок.

Завдання № 18 (Турфірма)

Потрібно розробити програмну систему, призначену для працівників туристичної фірми. Така система повинна забезпечувати зберігання відомостей про наявні у продажу путівки та про клієнтів фірми. Відомості про путівку містять її вартість, час відправлення (початок) та повернення, екскурсії та інші послуги, наприклад, у вартість путівки повністю або частково може входити харчування.

Путівка передбачає відвідування одного населеного пункту. Туристична фірма продає стандартні путівки. Клієнт може висловити свої побажання щодо термінів поїздки, її вартості, міст, які він хотів би відвідати. Кожен клієнт купує одну путівку. Для постійних клієнтів існує система знижок. Фірма веде також журнал про продаж путівок (дата, які путівки продані, скільки).

Клієнт	Путівка	Продаж
Ключ	Ключ	Ключ
ПІБ	Назва	ПІБ співробітника
Телефон	початок	Назва путівки
Номер паспорта	Кінець	Кількість продаж
		Кількість вільних
Знижка	Місто	путівок
	Послуги	Поточна дата
	Ціна	

Необхідно створити узагальнені списки:

- відомості про клієнтів;
- відомості про путівки;
- інформація про продаж путівок.

Дата задається вдень близько одного місяця.

Працівнику туристичної фірми можуть знадобитися такі дані:

- хто з клієнтів вибрав задане місто;
- чи є путівки з заданою датою початку;
- які путівки є «палаючими», тобто дата відправлення, зазначена в них, не більше ніж на 5 днів більше поточної;
- скільки втрачає фірма через знижки для постійних клієнтів;
- що є найдорожчою путівкою з наявних на поточний день;
- які путівки мають найбільший попит.

Має бути передбачена можливість додавання та видалення клієнтів для заданої путівки.

Завдання № 19 (Аптека)

Потрібно розробити програмну систему, що призначена для директора аптеки. Така система повинна забезпечувати зберігання відомостей про аптеку, про наявні в ній товари, про постачальників та товари, що ними постачаються. Аптека здійснює закупівлю товарів у різних постачальників, воліючи при цьому закуповувати одні види товару в одних постачальників, інші – в інших. Перевага постачальника визначається цифрою (1, 2 або 3). Список товарів у постачальника містить наявну кількість.

Товари, що є в аптеці, характеризуються найменуванням, ціною, фасуванням (5 мг, 10 мг тощо), терміном придатності та кількістю. Директор аптеки закуповує у постачальників товари, яких бракує та списує прострочені товари.

Аптека	Товар	Постачальник
Ключ	Ключ	Ключ
Назва	Назва	Назва
Поточна дата	Масив фасувань	Товар
Постачальник	Ціна	
Товар	Постачальник	
	Кількість	
	Термін придатності	
	Бажаний постачальник	

Необхідно створити узагальнені списки:

- відомості про аптеку.
- відомості про товари.
- відомості про постачальників.

Фасування задається у вигляді вкладеного масиву.

Директору можуть знадобитися такі відомості:

- які товари та скільки є в кожній аптеці;

- у яких товарів та в яких аптеках закінчився термін придатності товару;
- які продукти можна замовити у постачальників;
- сумарна вартість товарів у аптеці;
- у яких постачальників та в якій кількості є товар потрібної назви;
- який постачальник має товар із заданим фасуванням.

Необхідно передбачити можливість додавання та видалення товару в задану аптеку.

Завдання № 20 (Прилади)

Потрібно розробити програмну систему, призначену для відділу метрологічного забезпечення підприємства. Така система повинна забезпечувати збереження відомостей про всі наявні вимірювальні прилади, які беруться на облік та закріплюються за конкретним структурним підрозділом, який має профіль робіт (біологія, хімія, виробництво тощо). Під час взяття на облік кожному приладу надається свій індивідуальний номер. Кожен вимірювальний прилад характеризується найменуванням, призначенням та іншими метрологічними характеристиками, а також вартістю.

Кожен прилад може бути поставлений підрозділом на консервацію, повірятися (лише робітниками), ремонтуватись або списуватись.

Під час перевірки приладів встановлюється дата наступної перевірки, а після її закінчення фіксується дата перевірки, номер свідоцтва про повірку, найменування організації, яка здійснює перевірку, висновки за придатністю (придатний, не придатний), вартість перевірки.

Списання приладу здійснюється за результатами перевірки.

Прилад	Підрозділ	Перевірка
Ключ	Ключ	Ключ
Назва	Найменування	Організація

Інвентарний номер	Профіль	Прилад
Профіль	Поточна дата	Інвентарний номер
Вартість		Дата
Стан		Результат
Дата перевірки		Дата перевірки

Необхідно створити узагальнені списки:

- інформація про прилад;
- інформація про закріплення приладів за підрозділами;
- відомості про ремонт.

Відділу метрологічного забезпечення можуть знадобитися такі відомості:

- які прилади якого профілю та в якій кількості є у підрозділі;
- які прилади, що є у підрозділі, знаходяться на консервації та на перевірці;
- які прилади було списано;
- яким приладам (за підрозділами) необхідно пройти перевірку (що залишився менше 5-ти днів);
- які організації здійснюють перевірку приладів заданого профілю;
- у яких приладів настав термін чергової перевірки.

Необхідно передбачити можливість додавання та видалення приладу.

Завдання № 21 (Хлібозавод)

Потрібно розробити програмну систему, призначену для обробки даних про хлібозаводи. Така система повинна забезпечувати збереження відомостей про вироби та їх вартість. Відомості про вироби містять: найменування виробу, обсяг виробництва, дату виробництва, термін придатності інгредієнти, відпускну ціну.

Кожен склад характеризується найменуванням, кількістю, датою постачання, терміном придатності. Склад виробу містить список

можливих інгредієнтів: вода, борошно пшеничне, борошно житнє, висівки, цукор, олія соняшникова, сіль, дріжджі, добавки. На складах може не бути деяких інгредієнтів (цукру, олії).

Хлібозавод	Виріб	Склад
Ключ	Ключ	Ключ
Назва	Назва	Інгредієнт 1
Дата виробництва	Вага виробу	Інгредієнт 2
Поточна дата	Термін придатності	Інгредієнт 3
	Об'єм виробництва	Інгредієнт 4
	Ціна	Інгредієнт 5
		Інгредієнт 6
		Вага
		Дата постачання
		Термін придатності

Повинні бути створені узагальнені списки:

- відомості про хлібозавод.
- відомості про продукт.
- відомості про склад.

Керівництву комбінату можуть знадобитися такі відомості:

- які вироби випускає заданий хлібозавод;
- яка сумарна вартість всіх виробів, що випускаються за даним хлібозаводом;
- чи не було виробів, виробництво яких було виконано після закінчення терміну будь-якого інгредієнта;
- відсортувати хлібозаводи у порядку зменшення обсягу виробництва;
- у якому виробі найбільше інгредієнтів.
- який виріб забезпечить під час реалізації максимальний прибуток.

Необхідно передбачити можливість додавання та видалення виробу для заданого хлібозаводу.

Завдання № 22 (Судохідна компанія)

База даних судноплавної компанії містить відомості про рейси пасажирських суден, дати та час відплиття, кількість пасажирів на рейсі, про назви суден, їх місткість та номери рейсів, про порти, в які заходить судно під час виконання рейсу, про час стоянок у портах тощо. Один рейс може обслуговувати кілька суден. Система повинна забезпечувати пошук та видачу відомостей за різними запитами. Дата задається в межах одного місяця. Розробити інформаційну систему для судноплавної компанії.

Необхідно створити узагальнені списки:

- відомості судів;
- відомості про рейси;
- відомості про компанії.

Судно	Рейс	Стоянка
Ключ	Ключ	Ключ
Назва	Номер	Порт
Тип	Дата	Час прибуття
Кількість місць	Судно	Час стоянки
		Час відплиття
		Вартість
		Зайнято місць

Менеджеру компанії можуть знадобитися такі відомості:

- розклад рейсів за датами;
- найдорожчий рейс;
- рейс, який обслуговується максимальною кількістю суден;
- найзавантаженіші рейси (дата, кількість пасажирів);
- заданий рейс з перерахуванням відвідуваних портів та час стоянки;
- у яких портах найдовші стоянки.

Необхідно передбачити можливість видачі документа, що подає звіт про виконані рейси, а також звіт про прибутки.

Завдання № 23 (Магазин музичних записів)

Розробити інформаційну систему для магазину аудіо- та відеозаписів. Магазин музичних та відеозаписів веде для покупців каталог наявних у продажу записів із зазначенням їх роздрібних цін, жанру, різновидів жанру, виду носія запису (диск, флешка), імені виконавця або автора, назви твору чи запису, року випуску, виробника тощо.

Крім того, магазин веде внутрішній облік поточних оптових цін на записи, кількості екземплярів, проданих за окремі періоди минулого, кількості ще не розпроданих (наявних) екземплярів записів. У реальній ситуації також потрібно вести облік замовлень на записи, яких бракує, а також підрахунок прибутків / збитків за минулі періоди тощо.

Запис	Каталог	Магазин
Ключ	Ключ	Ключ
Назва	Запис	Список записів
Жанр	Ціна (роздріб)	Ціна (опт)
Виконавець	Носій	Кількість продажів
Рік випуску		Наявність
Фірма		
Вид носія		

Необхідно створити узагальнені списки:

- відомості про записи.
- відомості про твори.
- відомості про магазини.

Повинні бути реалізовані такі запити:

- отримання списку всіх записів заданого жанру;
- отримання списку найпопулярніших записів;
- виконавець найпопулярніших творів;
- перелік записів, яких немає в магазині;
- вартість всіх проданих записів;

– запис з максимальною різницею між роздрібною та оптовою ціною.

Необхідно передбачити можливість додавання та видалення пісень у задану групу.

Завдання № 24 (ДТП)

Потрібно розробити програмну систему, що призначена для аналізу дорожньо-транспортних пригод (ДТП). Така система має забезпечувати збереження відомостей про ДТП.

Для кожної ДТП мають бути збережені: вид ДТП (наїзд на пішохода, наїзд на перешкоду, зіткнення, перекидання тощо), дата, державні номери автомобілів (якщо в ДТП брали участь кілька автомобілів), дані про водія та причина (виїзд на смугу зустрічного руху, стан водія, несправність автомобіля, порушення правил дорожнього руху тощо). Дата задається в межах одного місяця.

Система має зберігати повну інформацію про ДТП.

Водій	Автомобіль	Відділ ДІБДР
Ключ	Ключ	Ключ
ПІБ	Фірма	Назва
Стаж	Марка	Номер акта про ДТП
Держ. номер авто	Тип кузова	Водій
Посвідчення	Держ. номер авто	Держ. номер авто
Дата		Дата
Номер акта про		
ДТП		Місце
		Кількість
		постраждалих
		Вид ДТП
		Причина ДТП

Необхідно створити узагальнені списки:

– інформація про водіїв;

- відомості про автомобілі;
- відомості про відділ ДАІ.

Керівництву МВС можуть знадобитися такі відомості:

- список водіїв, які вчинили більше ДТП;
- список водіїв, які беруть участь у ДТП у заданому місці;
- список водіїв, які беруть участь у ДТП на цю дату;
- ДТП із максимальною кількістю потерпілих;
- список водіїв, які беруть участь у ДТП із наїздом на пішоходів;
- причини ДТП у порядку зменшення їх кількості.

Необхідно передбачити можливість додавання та видалення інформації про ДТП.

4 ОФОРМЛЕННЯ ЗВІТУ З ВИКОНАНОЇ КУРСОВОЇ РОБОТИ

Звіт курсової роботи повинен містити такі структурні елементи:

1. Титульний аркуш.
2. Зміст.
3. Вступ.
4. Основний розділ.
5. Текст програм консольних додатків з реалізацією запитів LINQ.
6. Файли для реалізації списків.
7. Висновок.
8. Список джерел.
9. Програми.

Обсяг звіту має становити 15–20 сторінок.

У змісті вказуються всі розділи звіту з зазначенням сторінок.

У вступі відображаються: мета, завдання курсової роботи, актуальність обраної теми, необхідність вирішення проблеми в сучасних умовах.

Основний розділ містить опис основних класів, типів узагальнених колекцій та списків, опис запитів LINQ з оформленням лямбда-виразів та анонімних методів, а також текст програми для реалізації заданих у завданні запитів.

Необхідно подавати всі файли, необхідні для ініціалізації списків, включно зі символами розмежування між елементами.

До висновку додаються скріншоти (бажано у вигляді негативів від виведення на екран у консольних додатках) реалізації виведення на екран результатів запитів LINQ.

Список джерел повинен містити список книг і джерел з мережі «Інтернет», дійсно використаних під час підготовки та написання звіту, і складатися не менше ніж із 10-ти позицій.

СПИСОК РЕКОМЕНДОВАНИХ ДЖЕРЕЛ

1. Об'єктно-орієнтоване програмування [Електрон. ресурс] : метод. посіб. до лекц. та лаб. занять для студентів ф-ту математики, фізики та інформ. технологій першого (бакалавр.) рівня освіти спец. 122 – Комп'ютерні науки / уклад. : А. Л. Рачинська, О. А. Недєва, К. С. Палій. – Одеса : Одес. нац. ун-т ім. І. І. Мечникова, 2024. – 338 с. – Електрон. текст. дані. – Режим доступу: <https://dspace.onu.edu.ua/bitstreams/68babdee-36dd-42e9-aeaa-ecb0a23abd2b/download>, вільний (дата звернення: 29.03.2025). – Назва з екрана.
2. NET 6 books, will update.net 7 version [Electronic resource]. – Electronic text data. – Regime of access: <https://github.com/pty819/csharpbooks/tree/main>, free (date of the application: 02.03.2025). – Header from the screen.
3. Tutorial: Get started with ASP.NET Core [Electronic resource]. – Electronic text data. – Regime of access: <https://learn.microsoft.com/en-us/aspnet/core/getting-started/?view=aspnetcore-9.0>, free (date of the application: 14.05.2025). – Header from the screen.
4. Start building with these Microsoft products [Electronic resource]. – Electronic text data. – Regime of access: <https://developer.microsoft.com/en-us/>, free (date of the application: 12.01.2025). – Header from the screen.
5. Desktop Guide for .NET and .NET Framework [Electronic resource]. – Electronic text data. – Regime of access: <https://learn.microsoft.com/uk-ua/dotnet/desktop/>, free (date of the application: 03.03.2025). – Header from the screen.
6. Resources for Developers, by Developers [Electronic resource]. – Electronic text data. – Regime of access: <https://developer.mozilla.org/en-US/>, вільний (дата звернення: 03.04.2025). – Назва з екрана.
7. C# Tutorial [Electronic resource]. – Electronic text data. – Regime of access: <https://www.w3schools.com/cs/index.php>, free (date of the application: 02.03.2025). – Header from the screen.
8. Офіційний сайт Object Management Group [Електрон. ресурс]. –

Електрон. текст. дані. – Режим доступу: <http://www.omg.org/>, вільний (дата звернення: 10.04.2025). – Назва з екрана.

Електронне навчальне видання

Методичні рекомендації
до виконання курсової роботи
з навчальної дисципліни

«ОБ’ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ»

*(для здобувачів першого (бакалаврського) рівня вищої освіти
денної форми навчання зі спеціальностей F3, 122 – Комп’ютерні науки,
F6, 126 – Інформаційні системи та технології)*

Укладачі: **БРЕДІХІН** Володимир Михайлович,
ЛИТВИНОВ Анатолій Леонідович

Відповідальний за випуск *М. В. Булаєнко*
Редактор *Б. О. Хільська*
Комп’ютерне верстання *В. М. Бредіхін*

План 2022, поз. 238М

Підп. до друку 30.10.2025. Формат 60 × 84/16
Ум. друк. арк. 3,1

Видавець і виготовлювач:
Харківський національний університет
міського господарства імені О. М. Бекетова,
вул. Чорноглазівська, 17, Харків, 61002
Електронна адреса: office@kname.edu.ua
Свідоцтво суб’єкта видавничої справи:
ДК № 8386 від 14.07.2025