**Державний вищий навчальний заклад  
Ужгородський національний університет  
Факультет інформаційних технологій**

ЛАБОРАТОРНА РОБОТА № 16  
**Тема:** Робота з колекціями.

Виконав студент  
І курсу спеціальності «Інженерія програмного забезпечення»

Романюк Артем

**Ужгород-2025**

**Мета:** Вивчення принципів утворення та реалізації колекцій різних типів.

**Вивчити питання:** Абстрактні структури даних: список, черга, стек, множина, словник, хеш-таблиця. Простір імен System.Collections. Огляд колекцій. Неузагальнені колекції. Спеціальні колекції. Узагальнені колекції. Паралельні колекції.

**Хід роботи:**

**Варіант 13 (2)**

Завдання до роботи:

Скласти програму на мові C#, в якій визначити клас, що містить не менше 3 полів і характеризує предметну область «Студент» (характеристики визначити самостійно довільним чином). Із об’єктів спроектованого класу утворити неузагальнену колекцію «черга» і продемонструвати такі методи роботи з нею:

1) додавання елемента до колекції;

2) вилучення елемента;

3) пошук елементів за заданим значенням другого поля;

4) перегляд елементів.

Оформити звіт про виконання роботи та завантажити його в системі електронного навчання ДВНЗ «УжНУ» в установлений термін.

Теоретичні відомості про неузагальнену колекцію «черга» (Queue):

У мові C# існує декілька типів черг серед яких є неузагальнена черга: Queue з простору імен System.Collections. Колекції з цього простору імен оперують об’єктами, а не value type змінними. Ці колекції вже дещо застарілі, на противагу узагальненим колекціям, проте їх використовують для підтримки старого коду.

Основні методи та властивості класу Queue:

Enqueue(object obj) – Додає об'єкт до кінця черги

Dequeue() – Видаляє і повертає об'єкт з початку черги

Peek() – Повертає перший елемент без видалення

Contains(object obj) – Перевіряє, чи містить черга вказаний об’єкт

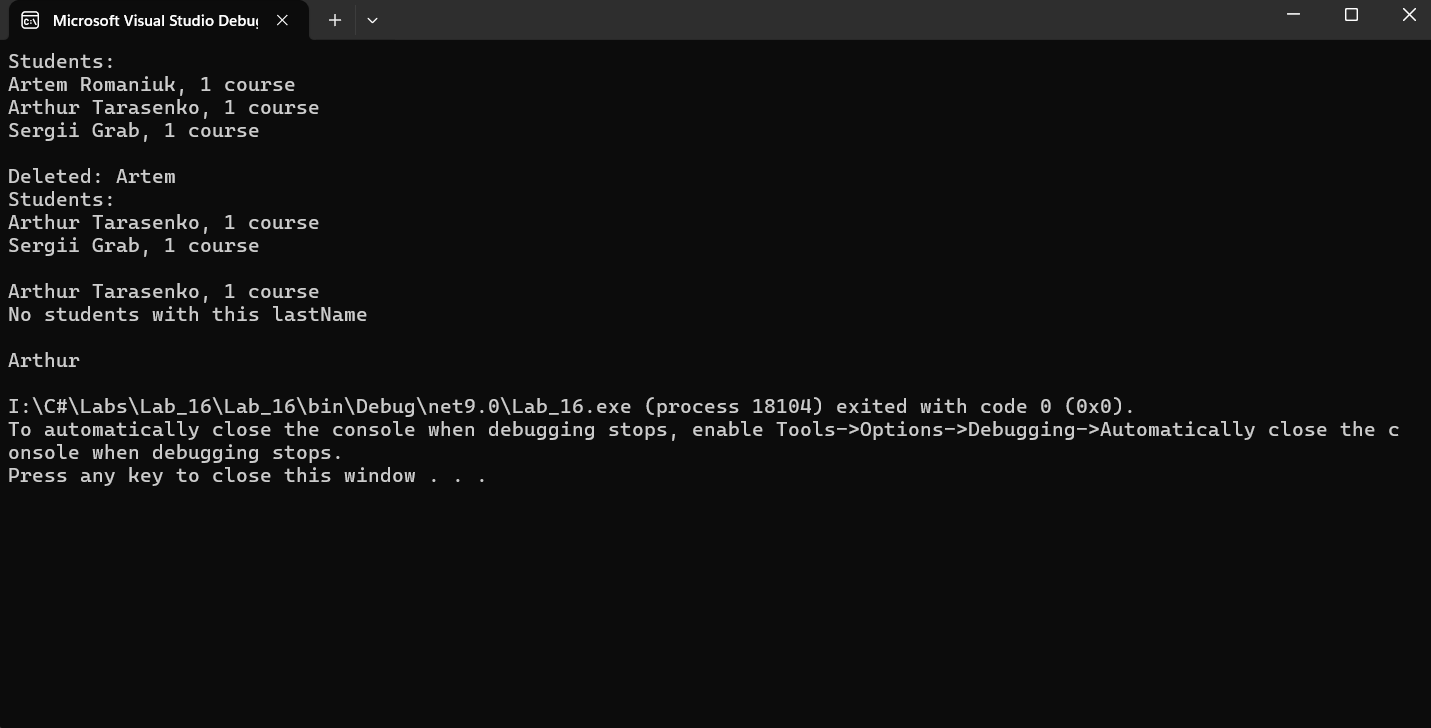
ToArray() – Повертає масив з елементів черги

Clear() – Очищає чергу

Count – властивість, яка показує кількість елементів у черзі

Реалізація в коді:

| using System.Collections;  class Student {  public string firstName;  public string lastName;  public int course;   public Student(string firstName, string lastName, int course)  {  this.firstName = firstName;  this.lastName = lastName;  this.course = course;  } }  class Program {  public static void Main(string[] args)  {  Queue studentsQueue = new Queue();  studentsQueue.Enqueue(new Student("Artem", "Romaniuk", 1));  studentsQueue.Enqueue(new Student("Arthur", "Tarasenko", 1));  studentsQueue.Enqueue(new Student("Sergii", "Grab", 1));  ShowStudentsQueue(studentsQueue);   Console.WriteLine("Deleted: " + ((Student)studentsQueue.Dequeue()).firstName);  ShowStudentsQueue(studentsQueue);   FindByLastName(studentsQueue, "Tarasenko");  FindByLastName(studentsQueue, "Romaniuk");   Console.WriteLine(((Student)studentsQueue.Peek()).firstName);  }  public static void ShowStudentsQueue(Queue queue)  {  Console.WriteLine("Students:");  foreach (Student student in queue)  {  Console.WriteLine($"{student.firstName} {student.lastName}, {student.course} course");  }  Console.WriteLine();  }  public static void FindByLastName(Queue queue, string lastName)  {  foreach (Student student in queue)  {  if (student.lastName.ToLower() == lastName.ToLower())  {  Console.WriteLine($"{student.firstName} {student.lastName}, {student.course} course");  return;  }  }  Console.WriteLine("No students with this lastName");  Console.WriteLine();  } } |
| --- |



**Висновок:**

У ході виконання лабораторної роботи я ознайомився з принципами роботи неузагальнених колекцій у C#, зокрема з класом Queue із простору імен System.Collections. Було розроблено програму, що імітує чергу студентів, реалізовано основні методи роботи з чергою: додавання, видалення, перегляд та пошук елементів за значенням другого поля.

Я навчився вручну реалізовувати пошук за заданим критерієм, оскільки неузагальнені колекції не підтримують методи на кшталт Find() або Where(). Отримані знання можуть бути корисними при роботі зі старим кодом або при використанні абстрактних структур даних у майбутньому.