Лабораторна робота № 6

Обробка структур

Мета роботи – набуття практичних навичок щодо обробки структур та освоєння на їх основі методики опрацювання таблиць.

Дана лабораторна робота сприяє напрацюванню таких **компетентностей** відповідно до Національної рамки кваліфікацій:

знання:

призначення, оголошення й визначення структур;

способів доступу до елементів структур;

способів ініціалізації елементів структур;

типових алгоритмів застосування структур;

уміння:

складати програми, дані в яких надані у вигляді відповідних структур;

виконувати налагодження та покрокове тестування типових програм обробки масивів структур у середовищі системи Visual C# .NET;

розробляти програми формування багаторядкових табличних документів;

комунікації:

обґрунтування рекомендацій команді учасників проекту щодо доцільності застосування поданих даних у вигляді відповідних структур та їх масивів;

робота в команді над окремими частинами складного коду, який містить обробку структур;

автономність і відповідальність:

прийняття рішення щодо доцільності застосування відповідних алгоритмів обробки структур та їх масивів;

самостійне обґрунтування можливих варіантів обробки таблиць на базі структур.

Основні положення

Структури – це складені типи даних, побудовані з використанням інших типів. Вони становлять об'єднаний загальним ім'ям набір даних різних типів.

Окремі дані структури називаються елементами або полями. Елементи однієї й тієї ж структури повинні мати унікальні імена, але дві

різні структури можуть містити неконфліктуючі елементи з однаковими іменами.

Синтаксичний блок визначення структури має такий вид:

[<Специфікатор_доступності>] struct <Ідентифікатор_структури>
[<Список_інтерфейсів>]

```
{
        <Елементи_структури>
};
```

Відповідно до синтаксису мови, опис структури починається зі службового слова struct, услід за яким міститься обране користувачем ім'я типу. Елементи, що входять у структуру, розміщуються в фігурних дужках, після яких ставиться крапка з комою. Елементи структури можуть мати вбудований або похідний тип.

Опис структури не резервує ніякого простору в пам'яті, він тільки створює новий тип даних, що може використовуватися для визначення змінних. У структурі обов'язково повинен бути вказаний хоча б один компонент.

Нехай необхідно створити тип для опису характеристики викладача університету. Цей тип повинен містити ім'я викладача, його кваліфікацію (гарна, задовільна тощо), стаж роботи і поточну якість викладання (за 12-бальною оцінкою). Далі наведено опис структури, що задовольняє ці вимоги:

```
struct Profesor
{
        public string Nombre;           // ім'я
        public string Calificacion;     // кваліфікація
        public int  Aprendizaje;        // стаж
        public double Calidad;          // якість
} ;
```

Ключове слово struct указує на те, що код визначає тип структури. Ідентифікатор Profesor – назва для цього типу. Таким чином, тепер можна створювати змінні типу Profesor так само, як змінні будь-якого базового типу, наприклад int або char.

Між фігурними дужками знаходиться список полів структури. Кожний елемент списку – це оператор визначення. Тут можна використовувати будь-який з типів даних C#, включаючи масиви та інші стуктури. В цьому прикладі використовуються два масиви типу string,

зручні для збереження рядків з атрибутами “Ім'я” і “Кваліфікація”, а також змінні int і double – для зберігання відповідних числових значень.

Тепер, коли структура оголошена, її можна використовувати. Для цього спочатку потрібно створити (визначити) екземпляр структури.

Екземпляр структури створюється за допомогою ключового слова new:

Profesor P_Econom_Inform = new Profesor( );

але, на відміну від класу, екземпляр структури можна створити і без new. Це виглядає в такий спосіб:

Profesor P_Econom_Inform;

Під час створення структури без ключового слова new її конструктори не викликаються. При цьому значення всім її елементам слід присвоїти явно, звернувшись до них через ім'я структури, як показано далі:

```
P_Econom_Inform.Nombre = "Браткевич В'ячеслав";
        P_Econom_Inform.Calificacion = "задовільна";
        P_Econom_Inform.Aprendizaje = 32;
        P_Econom_Inform.Calidad = 7.59;
```

Ініціалізацію не можна виконати через методи або властивості, оскільки жоден з елементів-функцій не може бути викликаний, поки не будуть ініціалізовані елементи-дані. Тому останні потрібно оголошувати як public**.**

Приклад 1. Оголошення, визначення (без new), ініціалізація та виведення на екран структури Profesor.

```
using System;
// Опис структури Profesor
struct Profesor
{
    public string Nombre;          // ім'я
    public string Calificacion;    // кваліфікація
    public int  Aprendizaje;       // стаж
    public double Calidad;         // якість
} ;
class Class1
{
    static void Main()
    {
```

```
                Profesor  P_Econom_Inform; // Оголошення екземпляра
структури
                //      Роздільна ініціалізація полів структури
                P_Econom_Inform.Nombre = "Браткевич В'ячеслав";
                P_Econom_Inform.Calificacion = "задовільна";
                P_Econom_Inform.Aprendizaje = 32;
                P_Econom_Inform.Calidad = 7.59;
                // Контрольне виведення
                Console.WriteLine("Викладач {0}: \nКваліфікація - {1};"+
                "\nСтаж - {2};\nЯкість - {3}", P_Econom_Inform.Nombre,
                P_Econom_Inform.Calificacion,
P_Econom_Inform.Aprendizaje,
                P_Econom_Inform.Calidad);
            }
        }
```

Результат роботи програми:

Викладач Браткевич В'ячеслав:
Кваліфікація – задовільна;
Стаж – 32;
Якість – 7,59

Приклад 2. Оголошення, визначення (з new), ініціалізація та виведення на екран структури Profesor.

```
using System;
// Опис структури Profesor
struct Profesor
{
        public string Nombre;           // ім'я
        public string Calificacion;     // кваліфікація
        public int  Aprendizaje;        // стаж
        public double Calidad;          // якість
} ;
        class Class1
{
        static void Main()
        {
                // Ініціалізації елементів структури за замовчуванням
```

```
                Profesor  P_Econom_Inform = new Profesor();
                // Контрольне виведення
                Console.WriteLine("Викладач {0}: \nКваліфікація - {1};"+
"\nСтаж - {2};\nЯкість - {3}\n", P_Econom_Inform.Nombre,
        P_Econom_Inform.Calificacion, P_Econom_Inform.Aprendizaje,
        P_Econom_Inform.Calidad);
                // Роздільна ініціалізація полів структури
                P_Econom_Inform.Nombre = "Браткевич В'ячеслав";
                P_Econom_Inform.Calificacion = "задовільна";
                P_Econom_Inform.Aprendizaje = 32;
                P_Econom_Inform.Calidad = 7.59;
                // Контрольне виведення
                Console.WriteLine("Викладач {0}: \nКваліфікація - {1};"+
"\nСтаж - {2};\nЯкість - {3}\n", P_Econom_Inform.Nombre,
        P_Econom_Inform.Calificacion, P_Econom_Inform.Aprendizaje,
                                                P_Econom_Inform.Calidad);
                Console.Read();  // для паузи
            }
        }
```

Результат роботи програми:

Викладач :

Кваліфікація –

Стаж – 0;

Якість – 0

Викладач Браткевич В'ячеслав:

Кваліфікація – задовільна;

Стаж – 32;

Якість – 7,59

Масиви структур.

Приклад 3. Обробка масиву структур.

```
using System;
struct Stroka
{
        public string name;                 // Автор книги
        public double stoimost;             // Вартість виданої книги
```

```
        public int kolich;                  // Кількість виданих книг
        public double sum_stoimost;         // Вартість виданих книг
    };
    class Class1
    {
        static void Main()
        {
            //      Введення вихідних даних
            Console.WriteLine("Введіть кількість рядків у документі");
            int kol = Convert.ToInt32(Console.ReadLine());
            Stroka[ ] Tabl = new Stroka[kol];
            for( int i=0; i < Tabl.Length; i++)
            {
                Console.WriteLine("Автор книги?");
                Tabl[i].name = Console.ReadLine();
                Console.WriteLine("Вартість книги?");
            Tabl[i].stoimost = Convert.ToDouble(Console.ReadLine());
                Console.WriteLine("Кількість книг?");
                Tabl[i].kolich = Convert.ToInt32(Console.ReadLine());
            }
            //  Виконання розрахунків:
            double s1=0, s2=0, s3=0;
            for( int i=0; i < Tabl.Length; i++)
            {
                Tabl[i].sum_stoimost = Tabl[i].stoimost * Tabl[i].kolich;
                s1 += Tabl[i].stoimost;
                s2 += Tabl[i].kolich;
                s3 += Tabl[i].sum_stoimost;
            }
        //  Побудова "шапки" таблиці
        Console.WriteLine("\nВідомості про вартість виданих книг\n");
        Console.WriteLine("|-----------------------------------------------------------|");
```

```
            Console.WriteLine("|  n/n |  Автор  |  Вартість  | Видано |
Витрати  |");
            Console.WriteLine("|-----------------------------------------------------
-----|");
                  // Заповнення таблиці даними:
                  for( int i=0; i < Tabl.Length; i++)
                  {
                        Console.WriteLine("|{0,5}{1,20}{2,14}{3,9}{4,10:N2}
|",
                              i+1,        Tabl[i].name,        Tabl[i].stoimost,
Tabl[i].kolich,
                              Tabl[i].sum_stoimost);
                  }
            Console.WriteLine("|-----------------------------------------------------
-----|");
            Console.WriteLine("| Разом: {0,31} {1,8} {2,9:N2}  |",s1, s2, s3);
            Console.WriteLine("|-----------------------------------------------------
-----|");
            }
      }
      Результат роботи програми:
      Введіть кількість рядків у документі
      3
      Автор книги?
      Рубіна
      Вартість книги?
      34,45
      Кількість книг?
      3
      Автор книги?
      Улицька
      Вартість книги?
      45,78
      Кількість книг?
      10
      Автор книги?
      Пушкін
```

Вартість книги?
75,23
Кількість книг?
3

Відомості про вартість виданих книг

n/n	Автор	Вартість	Видано	Витрати
1	Рубіна	34,45	3	103,35
2	Улицька	45,78	10	457,80
3	Пушкін	75,23	3	225,69
Разом:		155,46	16	786,84

Порядок виконання лабораторної роботи

Загальна частина.

1. Набрати, відкомпілювати і запустити на виконання приклади програм, які були наведені в розділі «Основні положення» даної лабораторної роботи.

2. Проекспериментувати з програмами:

змінити вихідні дані;

дослідити, як впливають синтаксичні помилки на результат компіляції програми. Які при цьому виникають помилки компіляції?

Індивідуальна частина.

Написати програму, яка на базі відповідної структури формує в результаті діалогу табличний документ і здійснює його обробку. Варіанти табличних документів взяти з додаткового файла з індивідуальними завданнями.

Розробити графічну схему (блок-схему) відповідного алгоритму.

Набрати і відкомпілювати тексти програми, усуваючи у разі необхідності помилки.

Дослідити роботу програми, аналізуючи виконання контрольних чисельних прикладів.

Зміст звіту

1. Титульний лист.

2. Цілі лабораторного заняття і вказівка, які навички та вміння передбачається отримати в результаті його виконання.

3. Тексти налагоджених програм загальної частини лабораторного заняття з необхідними коментарями і результатом виконання.

4. Постановка задачі індивідуального завдання і словесний опис його виконання, який супроводжується чисельним прикладом; графічну схему (блок-схему) відповідного алгоритму.

5. Текст налагодженої програми з результатом виконання контрольних чисельних прикладів індивідуального завдання.

6. Висновки.

Контрольні запитання

1. Коли доцільно використовувати структури?

2. Як реалізується призначення, оголошення й визначення структур?

3. Наведіть приклади оголошення, визначення, ініціалізації та виведення на екран елементів заданої структури (без застосування операції **new)**.

4. Наведіть приклади оголошення, визначення, ініціалізації та виведення на екран елементів заданої структури з застосуванням операції **new**.

5. Які особливості обробки елементів структур?