КЛАСТАРДЫҢ КОМПОЗИЦИЯСЫ ЖӘНЕ КОЛЛЕКЦИЯСЫ

Гульназ Жомартrsps
ВКГТУ им. Д.Серикбаева

КЛАСТАРДЫҢ КОМПОЗИЦИЯСЫ ЖӘНЕ КОЛЛЕКЦИЯСЫ

Егер класс өз өрістерінде басқа класс объекттерін пайдаланатын болса, онда кластардың осындай бірлестігі **композиция** деп аталады.

Класс композициясы бұл - бұрын жазылған қосымша кодын қайта пайдалану түрлерінің бірі. Мысалы, «дәріхана» класын «дәрілер» класының композициясы, яғни «дәрілер» класының түрлі объекттерінің массиві ретінде қарастыруға болады.

Композиция мысалы ретінде «гараж» класында «автомобиль» объекттерінің бірлестігін келтіруге болады.

КЛАСТАРДЫҢ КОМПОЗИЦИЯСЫ ЖӘНЕ КОЛЛЕКЦИЯСЫ

Деректердің құрылымда бір типті объекттердің бірлесуі **коллекция** деп аталады.

Коллекция объекттерді өңдеудің көптеген функцияларын қамтамасыз етуі керек, яғни объекттерді сақтау мен жою және объекттерді қосу, жаңарту бойынша операцияларды ұсыну.

Әдетте коллекция жеке класпен анықталады.

КЛАСТАРДЫҢ КОМПОЗИЦИЯСЫ ЖӘНЕ КОЛЛЕКЦИЯСЫ

Бүкіл коллекцияны сипаттайтын класты **коллекция класы** деп атайды.

Кластар композициясы бойынша қарастырылған мысалдарда екінші класс (бірінші кластың) **объекттер коллекциясы** қызметінде болады.

Мысалы, «гараж» класы «автомобиль» класының объекттер коллекциясы болып табылады. «Дәріхана» класы – «дәрілер» класы объекттерінің коллекциясы.

КЛАСТАРДЫҢ КОМПОЗИЦИЯСЫ ЖӘНЕ КОЛЛЕКЦИЯСЫ

C# бағдарламалау ортасынада басқа кластардың бір типті объекттерін коллекциялау үшін көптеген түрлі кластар қолданылады, мысалы, тізімдер, стектер, кезектер, сөздіктер, бұтақтар, және көптеген басқа да коллекциялар.

КЛАСТАРДЫҢ КОМПОЗИЦИЯСЫ ЖӘНЕ КОЛЛЕКЦИЯСЫ

Барлық коллекциялар кластарын шартты түрде **сызықты** және **сызықты емес** деп бөлуге болады.

Сызықты коллекцияларды келесі коллекциялардың кластары ұсынады:

– индекстелген қол жеткізу, мысалы, сөздіктер мен анықтама кітаптары;

– тікелей қол жеткізу (с прямым доступом), мысалы, массивтер;

– ретті қол жеткізу (с последовательным доступом), мысалы, стектер, кезектер, тізімдер. Көптеген «тізімдер» динамикалық массивтермен анықталған;

КЛАСТАРДЫҢ КОМПОЗИЦИЯСЫ ЖӘНЕ КОЛЛЕКЦИЯСЫ

Сызықты емес коллекцияларды келесі коллекциялардың кластарын ұсынады:

– иерархиялық, мысалы, әр түрлі бұтақтар тәрізді құрылымдар. Классификацияның иерархиялық жүйесі немесе құжаттарды іздестіру массивін ұйымдастыру;

– топтық, мысалы, түрлі жиынтықтар, желілік құрылымдар, графтар.

Кластар композициясы бұрын құрылған кластарды немесе қосымша үзінділерін қолдануда бағдарламалаудың маңызды құралы болып табылатының ерекше атап өту керек.

Коллекциялар

Көптеген қосымшаларда байланыстырылған объекттер тобын құру және басқару қажет. Объекттерді топтаудың екі жолы бар (https://docs.microsoft.com/ru-ru/dotnet/csharp/programming-guide/concepts/collections дерегінен):

- **объекттер масссивін құру**
- **коллекцияларды құру**

Массивтер қатаң типтелген объекттердің белгіленген санын құру және онымен жұмыс жасау үшін ыңғайлы.

Коллекциялар объектілер топтарымен жұмыс істеудің неғұрлым икемді әдісін ұсынады. Массивтерден айырмашылығы, сіз жұмыс істейтін коллекция қажет болғанда динамикалық түрде өсуі немесе азаюы мүмкін.

Кластар коллекциясын қолдану

Кластар коллекциясын қолдану

Сан алуан коллекциялар кластарының ішінен ең қарапайым кластар коллекциясын – массивті қарастырайық.

Есеп. Массив элементі болып КІТАП класының объектісі есептеледі. Класс деректері: автр, атауы және бағасы. Класс әдісі ретінде параметрлері берілген конструктор қолданылады.

Бағдарламада объекттер массивін файлға жазуды және файлдан деректерді массивке оқуды қарастыру керек (объекттердің сериализациясы және кері сериализациясы/десериализация көмегімен).

Кластар коллекциясын қолдану

```
using System.Collections.Generic;
using System.Runtime.Serialization.Formatters.Binary;
using System.Runtime.Serialization;
using System.IO;
. . .
namespace WindowsFormsApplication1
{ public partial class Form1 : Form
   {  public Form1()
      {    InitializeComponent();       }
      [Serializable]
       public class Kniga
      {  public string Naz;
         public string Avtor;
         public int Ctoimoct;
         public Kniga(string sa, string sb, int sc)
         {  Avtor = sa; Naz = sb; Ctoimoct = sc;     }
      };
       public static Kniga[] Polka= new Kniga[20];
```

Кластар коллекциясын қолдану

```
private void button1_Click(object sender, EventArgs e)
    {
     string a, b;
     int c;
     a = textBox1.Text;
     b = textBox2.Text;
     c = Convert.ToInt32(textBox3.Text);
     Kniga Tom = new Kniga(a, b, c);
     Polka[kol] = Tom;
     kol++;
    }
```

Кластар коллекциясын қолдану

```
private void button2_Click(object sender, EventArgs e)
      {
         textBox4.Text = "";
         for(int i=0;i<kol;i++)
             {
                ss = Polka[i].Avtor + "  " + Polka[i].Naz + "  " +
                Convert.ToString(Polka[i].Ctoimoct) + "\r\n";

                textBox4.AppendText(ss);
             }
       }
```

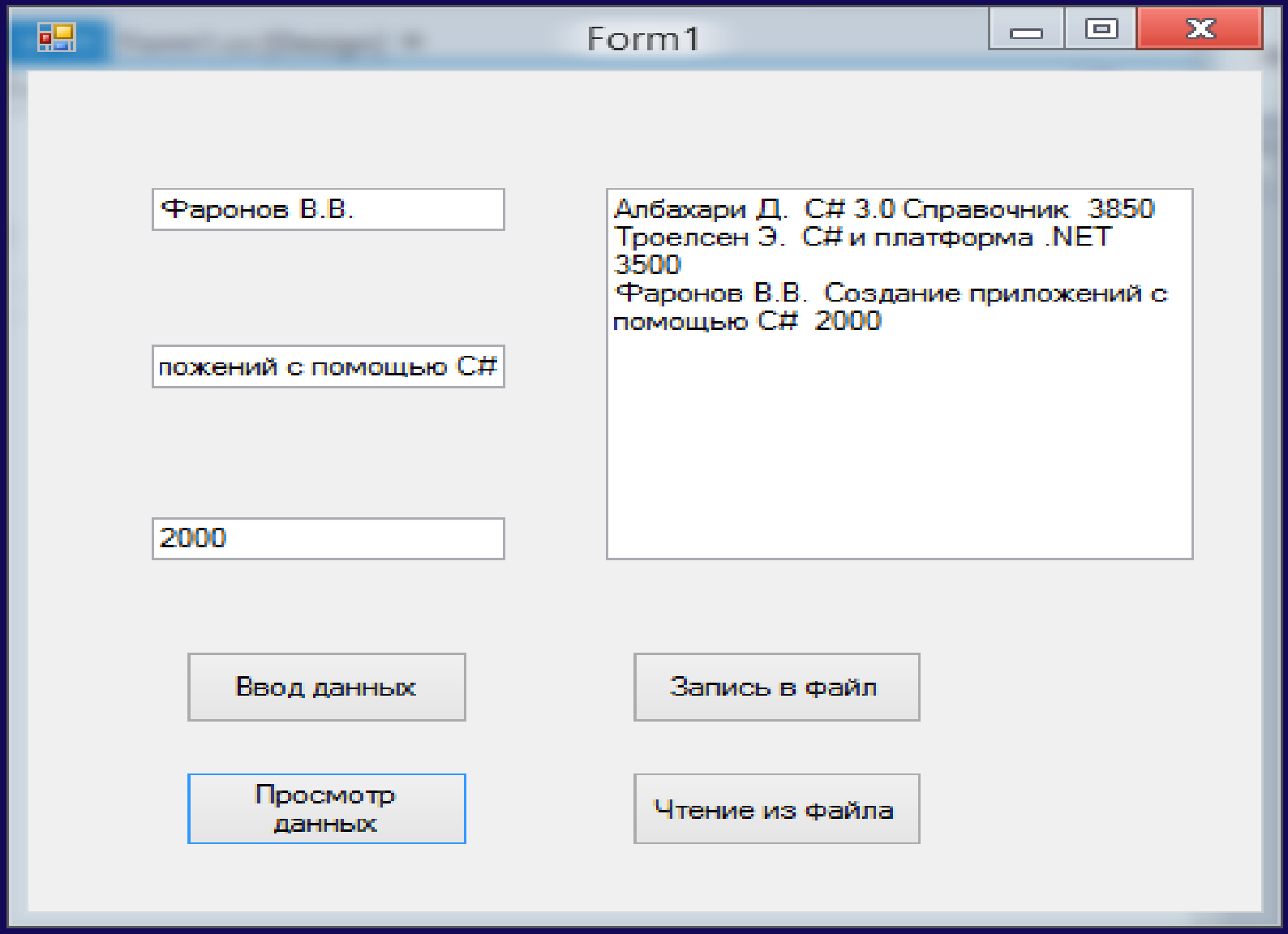
Кластар коллекциясын қолдану

```
private void button3_Click(object sender, EventArgs e)
    {
        // Создаем поток для сериализации:
        FileStream StreamOut = new FileStream("knigi.dat",
FileMode.Create, FileAccess.Write);
        // Используем двоичный формат:
        BinaryFormatter fmt = new BinaryFormatter();
        for (int i = 0; i < kol; ++i)
        {
            fmt.Serialize(StreamOut, Polka[i]);  // Сериализуем
объекты
        }
        StreamOut.Close();              // Закрываем поток
    }
```

Кластар коллекциясын қолдану

```
  private void button4_Click(object sender, EventArgs e)
      {
         textBox4.Text = "Вывод после чтения из файла: \r\n";
         // Создаем поток для десериализации:
         FileStream StreamIn = new FileStream("knigi.dat", FileMode.Open,
FileAccess.Read);
         // Используем двоичный формат:
         BinaryFormatter fmt = new BinaryFormatter();
         kol = 0;   bool ok = true;   int i = 0;
         while (ok)
         {
           try
            {   Polka[i] = (Kniga)fmt.Deserialize(StreamIn);//Десериализуем
                kol++; i++; textBox4.AppendText(kol.ToString() + "\r\n");
            }
            catch
            {  MessageBox.Show("Failed to deserialize");
              break;
            }
         }
         StreamIn.Close();  // Закрываем поток
         textBox4.AppendText(kol.ToString() + "\r\n");
      }
```

Кластар коллекциясын қолдану



Коллекциялар түрлері

Коллекциялар түрлері

Коллекциялар түрлері

Көптеген коллекциялар .NET Framework платформасында қамтамасыз етілген. Коллекцияның әрбір түрі белгілі бір мақсатқа арналған.

Коллекциялар класстары келесі атаулар кеңістігінде орналасқан:

- **System.Collections** - қарапайым, біртипті емес коллекциялар үшін коллекция класстары (простые необобщенные классы коллекций);
- **System.Collections.Generic** – типтелген коллекция класстары (обобщенные или типизированные классы коллекций).
- **System.Collections.Concurrent -** коллекция класстары (для обеспечения параллельного выполнения задач и многопоточного доступа)

Некоторые коллекции Framework

Реализация классов коллекций осуществляется с помощью интерфейсов. Платформа .NET, а именно библиотека Framework, предоставляет разработчика коллекций дополнительные интерфейсы и их обобщенные версии для организации коллекций, доступа к ее элементам по индексам, поиск в коллекции и т.д.

Интерфейс **ICollection** являются стандартным для счетных коллекций объектов. Он допускает перебор своих элементов с использованием интерфейсов **IEnumerable** и **IEnumerator**, позволяет определять размер коллекции, копировать ее в массив для более сложной обработки и т.д.

Интерфейс **IList** являются стандартным для создания коллекций типа массив. Он также допускает перебор своих элементов с использованием интерфейсов **IEnumerable** и **IEnumerator**, но и обеспечивает возможность прямого доступа к элементу с помощью перегружаемого индексатора. Интерфейс **IList** обеспечивает добавление, удаление и редактирование элемента коллекции по его индексу.

Некоторые коллекции Framework

Существует множество интерфейсов для работы с нестандартными коллекциями, например, интерфейс **IDictioaryEnumerator** позволяет работать с коллекциями типа словари и т.д.

Использование интерфейсов позволило в библиотеке Framework создать несколько классов коллекций, например, **Array, ArrayList. LinkedList. Queue. Stack** и другие.

Для визуального представления коллекций (фактически это тоже классы коллекций) в языке C# имеется несколько управляющих элементов, например, известный нам **DataGridView** и другие подобные ему элементы, **TreeView** – предназначен для отображения иерархических коллекций и т.д.

Более подробно Вы будете изучать коллекции в дисциплинах «Прикладное программирование» и «Базы данных» в нашей дисциплине мы рассмотрим работу с классом коллекции на примере коллекции **ArrayList.**

Коллекциялар түрлері

System.Collections (атаулар кеңістігінің) класстары

System.Collections атаулар кеңістігіндегі класстар нақты типтелген объекттер түріндегі элементтерді емес Object типіндегі объекттерді сақтайды.

Төмендегі кестеде **System.Collections** атаулар кеңістігіндегі ең жиі пайдаланылатын кейбір класстар берілген:

Коллекциялар түрлері

System.Collections (атаулар кеңістігінің) класстары

Класс	Сипаттама
ArrayList	Қажет болған жағдайда өлшемі динамикалық түрде өсетін объекттер массивын сипаттайды.
Hashtable	Кілттің хэш-код арқылы реттелетін «кілт-мән» жұптарының коллекциясын сипаттайды.
Queue	Представляет коллекцию объектов, которая обслуживается в порядке поступления (FIFO).
Stack	Представляет коллекцию объектов, которая обслуживается в обратном порядке (LIFO).

ArrayList коллекциясы

ArrayList Класс System.Collections.ArrayList атаулар кеңістігіне тиісті және кез-келген типтегі объекттерді сақтау үшін қолданылады. ArrayList класының негізгі қасиеттері мен әдістері (үзінді, кестенің толық нұсқасын дәрістен қара)

Іске асырады: IList, ICloneable

Қасиеттер мен әдістер	Сипаттама
public static ArrayList (IList: List);	**List коллекциясының негізінде ArrayList объектін құрады**
public virtual int Add(Oblect Value);	**Тізім соңына жаңа объектті қосады және индексті қайтарады.**
public virtual viod AddRange (ICollection coll)l	**Тізім соңына бірнеше объекттерді қосады.**

ArrayList коллекциясы

Задача 12.1 Использовать класс ArrayList для организации коллекции «Гараж», объединяющей объекты «Автомобиль». Для работы с коллекцией использовать свойства и методы таблицы 12.1.

```
…
public class Avto
  {  public string Marka;
     public int Cena;
     public Avto(string sm, int sc)
     {     Marka = sm; Cena = sc;     }
  };
  public ArrayList Garaj = new ArrayList();
…
 private void button1_Click(object sender, EventArgs e)
  {  panel2.Visible = true;   panel3.Visible = false;   panel1.Visible = false;
     string n;    int c;
     n = textBox1.Text;
    c = Convert.ToInt32(textBox2.Text);
    Avto at = new Avto(n, c);
     Garaj.Add(at);
  }
```

Коллекциялар түрлері

System.Collections.Generic (атаулар кеңістігінің) класстары

Коллекциядағы барлық элементтер бірдей деректер типінде болса, әмбебап коллекция (универсальная коллекция) қолданылады.

Әмбебап коллекция қажетті типтегі деректерді ғана қосуға мүмкіндік береді, сондықтан қатаң типтелуді қамтамасыз етеді.

Төмендегі кестеде **System.Collections.Generic** атаулар кеңістігіндегі ең жиі пайдаланылатын кейбір класстар сипатталған:

Коллекциялар түрлері

System.Collections.Generic (атаулар кеңістігінің) класстары

Класс	Сипаттама
Dictionary<TKey, TValue>	Предоставляет коллекцию пар «ключ-значение», которые упорядочены по ключу.
List<T>	Представляет список объектов, доступных по индексу. Предоставляет методы для поиска по списку, его сортировки и изменения.
Queue<T>	Представляет коллекцию объектов, которая обслуживается в порядке поступления (FIFO).
SortedList<TKey,TValue>	Представляет коллекцию пар "ключ-значение", упорядоченных по ключу на основе реализации IComparer<T>.
Stack<T>	Представляет коллекцию объектов, которая обслуживается в обратном порядке (LIFO).

System.Collections.Generic (атаулар кеңістігінің) класстары

Список List<T>

Класс **List<T>** представляет простейший список однотипных объектов. Среди его **методов** можно выделить следующие:

- **void Add(T item)**: добавление нового элемента в список
- **void AddRange(ICollection collection)**: добавление с список коллекции или массива
- **int BinarySearch(T item)**: бинарный поиск элемента в списке. Если элемент найден, то метод возвращает индекс этого элемента в коллекции. При этом список должен быть отсортирован.
- **int IndexOf(T item)**: возвращает индекс первого вхождения элемента в списке
- **void Insert(int index, T item)**: вставляет элемент item в списке на позицию index
- **bool Remove(T item)**: удаляет элемент item из списка, и если удаление прошло успешно, то возвращает true
- **void RemoveAt(int index)**: удаление элемента по указанному индексу index
- **void Sort()**: сортировка списка

System.Collections.Generic (атаулар кеңістігінің) класстары

List<T> классы Іске асырады: ICollection<T>, IEnumerable<T>, IList<T>, IReadOnlyCollection<T>, IReadOnlyList<T>, IList

Мысал,

```
using System.Collections.Generic
…
// тізімді құру, Part- класс атауы
List<Part> parts = new List<Part>();

// тізімге элементтерді қосу
parts.Add(new Part() {PartName="crank arm", PartId=1234});
parts.Add(new Part() { PartName = "chain ring", PartId = 1334   }   );

// циклды қолдану
foreach (Part aPart in parts)
{  Console.WriteLine(aPart);
}
// индекс бойынша  элементті жою
parts.RemoveAt(2);
```

Использование LINQ для доступа к коллекции

Использование LINQ для доступа к коллекции

Для доступа к коллекции можно использовать язык LINQ. Запросы LINQ обеспечивают возможности фильтрации, упорядочения и группировки.

В приведенном ниже примере выполняется запрос LINQ применительно к универсальной коллекции List. Запрос LINQ возвращает другую коллекцию, содержащую результаты.

Использование LINQ для доступа к коллекции

```
private static void ShowLINQ()
{
   List<Element> elements = BuildList(); //   BuildList() - метод
// LINQ Query.
   var subset = from theElement in elements
                where theElement.AtomicNumber < 22
                orderby theElement.Name
                select theElement;

   foreach (Element theElement in subset)
   {  Console.WriteLine(theElement.Name + " " +
                                   theElement.AtomicNumber);
   }
   // Output:
   //  Calcium 20
   //  Potassium 19
   //  Scandium 21
}
```