9-шы дәріс. Кластардың негізгі элементтері: өрістерді, әдістерді, конструкторларды, қасиеттерді пайдалану жолдары

1.Кластар туралы жалпы мәліметтер
2. Класс спецификаторлары
3. Класс объектілерін жасау
3.1. Объектілерді меншіктеу және салыстыру
3.2. Мәліметтер: өрістер мен константалар
4. Класс өрісінің спецификаторлары мен константалары
5. Әдістер
5.1.Әдістер параметрлері
5.2. Әдісті шақыру
5.3. Параметрлерді беру әдістері және олардың типтері
5.4. Параметрлерді қолдану ережелері
6. *this* түйінді сөзі
7. Конструкторлар
8. Қасиеттер

Класс сипаттамасы class түйінді сөзі мен класс *атынан* тұрады да, жүйелі жақша ішінде класс *тұлғасы* – оның элементтері тізімі орналасады. Бұларға қоса, кластың базалық элементтерін (тегі-предки) және әртүрлі сипаттары анықталатын, міндетті емес бірсыпыра *атрибуттары* мен *спецификаторларын* беруге болады.

3.2. Мәліметтер: өрістер мен константалар
4. Класс өрісі спецификаторлары мен константалары
Класс мысалы
5. Әдістер

Әдіс – класпен немесе экземплярмен атқарылатын есептеулерді орындайтын функционалдық элемент. Әдістер кластың тәртібін анықтайды да, оның **интерфейсін** құрайды.

Әдіс – аты арқылы пайдалануға болатын, аяқталған код фрагменті. Ол бір рет жазылады да, қанша рет қажет болса, сонша рет шақырылып орындала береді.

Бір әдіс оған аргумент ретінде берілген әртүрлі мәліметтерді өңдей алады. Төмендегі мысалға қараңыз:

Әдіс синтаксисі
Әдістер мысалдары
5.1.Әдістер параметрлері
Қарапайым әдіс мысалы
5.2. Әдісті шақыру
Әдіске параметрлерді беру мысалы

```
using System;
namespace Lisning5_3 {
class Class1
    {   static int Max(int a, int b)        // макс. мәнді таңдау
        {
            if ( a > b ) return a;
            else            return b;
        }
        static void Main()
        {
            int a = 2, b = 4;
            int x = Max( a, b );            // Max әдісін шақыру
```

```
            Console.WriteLine( x );                    // нәтижесі: 4
            short t1 = 3, t2 = 4;
            int y = Max( t1, t2 );          // үйлесімді тип пар-рлері
            Console.WriteLine( y );                    // нәтижесі : 4
            int z = Max( a + t1, t1 / 2 * b );     // өрнек
            Console.WriteLine( z );                    // нәтижесі : 5
}}}
```

5.3. Параметрлерді беру әдістері және олардың типтері

Параметрлерді беру әдістері: мән бойынша және сілтеме бойынша.

Параметрлерді мән бойынша беру кезінде әдіс аргументтер мәнінің көшірмелерін алады да, әдіс операторлары осы көшірмелермен жұмыс істейді.

Параметрлерді сілтеме (адрес) бойынша беру кезінде әдіс аргументтер адресінің көшірмесін алады да, сол (бұрынғы) адрестердегі аргументтерді пайдаланады.

C# тілінде параметрлердің төрт типі бар:

мән-параметрлер;

сілтеме-параметрлер (**ref**);

нәтижелік (шығыс - выходные) параметрлер (**out**);

жиым-параметрлер (**params**).

Түйінді сөздер параметрлер типтерін сипаттаудың алдында тұрады. Егер ол болмаса, параметр мән-параметр болып есептеледі. Мысалы:

public int Calculate(int a, ref int b, out int c, params int[] d) …

Мысал: мән-параметрлер мен ref сілтемелері

```
using System;
namespace Listing5_4
{     class Class1
      {     static void P( int a, ref int b )
            {
                  a = 44; b = 33;
                  Console.WriteLine( "Әдіс ішінде {0} {1}", a, b );
            }
static void Main()
            {
                  int a = 2, b = 4;
                  Console.WriteLine( "шақыруға дейін      {0} {1}", a, b );
                  P( a, ref b );
                  Console.WriteLine("шақырудан кейін    {0} {1}", a, b );
}}}
```

Мысал: out шығыс параметрлері

```
using System;
namespace Listing5_5
{     class Class1
      {     static void P( int x, out int y )
            {
                  x = 44; y = 33;
                  Console.WriteLine( " Әдіс ішінде {0} {1}", x, y );
            }
static void Main()
            {
                  int a = 2, b;              // b-ны инициалдау қажет емес
P( a, out b );
```

```
                Console.WriteLine( "шақырудан кейін    {0} {1}", a, b );
}}}
```

5.4. Параметрлерді қолдану ережелері

1. Мән-параметрлер үшін мән бойынша параметр беру қолданылады. Бұл мүмкіндік әдістің бастапқы мәліметтері үшін қолданылады.

Әдісті шақыру кезінде мән бойынша берілетін параметр орнында өрнек болуы мүмкін (және де оның жеке жағдайы ретінде – айнымалы немесе константа). Мұнда өрнек типі параметр типіне келтірілуі керек.

2. Сілтеме-параметрлер және нәтижелік (шығыс) параметрлер адрес бойынша беріледі. Мұндай мүмкіндік әдістің қосалқы нәтижелерін (побочные результаты) беру кезінде қолданылады.

Әдісті шақыру кезінде **ref** сілтемесі бойынша берілетін параметр орнында тек сол типтегі инициалданған айнымалы аты ғана болуы мүмкін. Параметр аты алдында ref түйінді сөзі көрсетіледі.

Әдісті шақыру кезінде **out** нәтижелік (шығыс) параметр орнында тек сол типтегі айнымалы аты ғана болуы мүмкін. Оны инициалдау қажет емес. Параметр аты алдында out түйінді сөзі көрсетіледі.

6. this түйінді сөзі

Әдістің ол өңдеуге тиіс объект өрісімен жұмыс істеуін қамтамасыз ету үшін, сол әдіске автоматты түрде **this** жасырын параметрі беріледі, ол функцияны шақырған объектіге сілтеме жасап тұрады.

this сөзін нақты түрде пайдалану

this параметрі тікелей түрде мынадай жағдайларда қолданылады:

```
// Әдістен оны шақырған объектіге сілтеме қайтару үшін:
class Demo
{               double y;
                public Demo T()        { return this; }
//    аты Әдіс параметрі атымен бірдей өрісті идентификациялау үшін:
                public void Sety( double y ) { this.y = y; }
}
```

7. Конструкторлар

Конструктор объектіні инициалдау үшін қажет. Ол класс объектісін жасау кезінде ***new*** операциясы арқылы шақырылады. Конструктор аты класс атымен бірдей болады.

Конструкторлар қасиеттері:

Конструктор ешқандай мән (тіпті void типін де) қайтармайды.

Кластың, инициалдаудың әр түрлері үшін әртүрлі параметрлері бар, бірнеше конструкторлары болуы мүмкін.

Егер программалаушы бірде-бір конструкторды көрсетпесе немесе кейбір өрістер инициалданбаса, мәндік типтегі өрістерге – нөл, ал сілтемелік типтегі өрістерге – null мәні меншіктеледі.

Параметрсіз шақырылатын конструктор (үнсіз) келісім бойынша алынған конструктор деп аталады.

Конструкторы бар класс мысалы

```
using System;
namespace Listing5_6    {
class Demo
        {
                public Demo( int a, double y )                // конструктор
                {
                        this.a = a;
```

```
            this.y = y;
        }
        int a;
        double y;
    }

    class Class1
    {   static void Main()
        {
            Demo a = new Demo( 300, 0.002 );   // конструкторды шақыру
            Console.WriteLine( a.Gety() );     // нәтиже: 0,002
            Demo b = new Demo( 1, 5.71 );      // конструкторды шақыру
            Console.WriteLine( b.Gety() );     // нәтиже : 5,71
  } } }
```

Екі конструкторы бар класс мысалы

```
class Demo
    {
        public Demo( int a )                   //   1 конструктор
        {
            this.a = a;
            this.y = 0.002;
        }
        public Demo( double y )                // 2 конструктор
        {
            this.a = 1;
            this.y = y;
        }
        ...
    }
        ...
            Demo x = new Demo( 300 );      // 1 конструкторды шақыру
            Demo y = new Demo( 5.71 );     // 2 конструкторды шақыру
```

Жалпы класс мысалы

```
class Monster {
public Monster()          // конструктор
    {
            this.name   = "Noname";
            this.health = 100;
            this.ammo = 100;
    }
  public Monster( string name ) : this()
    {
            this.name = name;
     }
  public Monster( int health, int ammo, string name )
     {
            this.name   = name;
            this.health = health;
            this.ammo = ammo;
     }
```

```
public int GetName()                // Әдіс
    {   return name;     }
public int GetAmmo()                // Әдіс
     {   return ammo;}
```

8. Қасиеттер

Қасиеттер класс өрістерімен қатынас құру үшін пайдаланылады. Көбінесе, қасиет жабық өрістермен қатынас құру әдістерін анықтайды.

Қасиет синтаксисі:

[спецификаторлар] типі қасиет_аты
{
[get қатынасу_коды]
[set қатынасу_коды]
}

Қасиеттерді пайдалану кезінде, автоматты түрде онда көрсетілген оқу (**get**) және орнату (**set**) блоктары шақырылады.

Мұнда не get, не set бөлігі болмауы мүмкін, бірақ екеуінің де бірден жоқ болуы мүмкін емес. Егер set бөлігі болмаса, қасиет тек оқылады (read-only), ал егер get бөлігі болмаса – тек жазылады (write-only).

Қасиеттерді сипаттау мысалы

```
public class Button: Control
{   private string caption;        // қасиетпен байланысты   өріс
      public string Caption {        // қасиет
          get { return caption; }    // қасиетті алу жолы
          set                        // қасиетті орнату жолы
                { if (caption != value) { caption = value; }
 }}       ...
```

Классмысалы

```
class Monster {
      public Monster()          // конструктор
      {
                this.name    = "Noname";
                this.health = 100;
                this.ammo = 100;
      }
   public Monster( string name ) : this()
      {
                this.name = name;
       }
   public Monster( int health, int ammo, string name )
       {
                this.name    = name;
                this.health = health;
                this.ammo = ammo;
       }
   public int GetName()                // Әдіс
      {   return name;     }
   public int GetAmmo()                // Әдіс
       {   return ammo;}
```