4 дәріс. Әдістерден объектілерді қайтару.

Дәрістің мақсаты: студенттерде әдістердің құрамында объектілермен жұмыс істеу ерекшеліктері бойынша түсініктерін көрсетуге қабілет қалыптастыру.

Осы дәрісті меңгеру нәтижесінде студенттер келесі қабілеттерге ие болады:

- Әдістен объектіні қайтару форматын білетінін көрсету;
- Әдістен ағымдағы объектіні қайтару форматын білетінін көрсету.

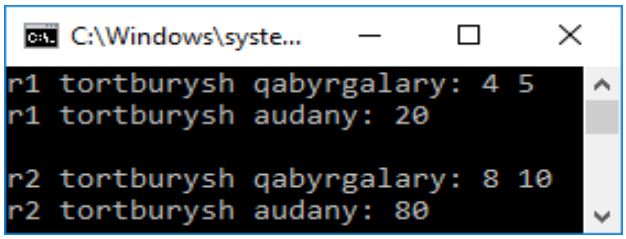
Әдіс кез келген типтегі мәліметті қайтара алады, ол класс типін де қайтара алады. Төменде құрамында Enlarge() әдісі бар Rect класының бір нұсқасы берілген, онда тіктөртбұрыш тұрғызылады, оның қабырғалары әдісті шақыратын объектідегі тіктөртбұрышпен бірдей, бірақ тұрғызылатын тіктөртбұрыш көрсетілген коэффициентке көбейтіліп, үлкейтіледі.

```
// Әдістен объектіні қайтару.
using System;
class Rect {
    int width;
    int height;

    public Rect(int w, int h) {
       width = w;
       height = h;
    }

    public int Area() {
       return width * height;
    }

    public void Show() {
        Console.WriteLine(width + " " + height);
    }


    /* Әдіс шақырылған объектідегі төртбұрышпен салыстырып
         қарағанда, қабырғалары берілген  коэффициентке
         пропорционал үлкейтілген төртбұрышты қайтарады. */
    public Rect Enlarge(int factor) {
        return new Rect(width * factor, height * factor);
    }
 }

class RetObj {
static void Main() {
    Rect r1 = new Rect(4, 5);

    Console.Write("r1 tortburysh qabyrgalary: ");
    r1.Show();
    Console.WriteLine("r1 tortburysh audany: " +  r1.Area());
```

```
        Console.WriteLine();

        // r1 tortburyshynan eki ese ulken tortburysh turgyzu.
        Rect r2 = r1.Enlarge(2);
        Console.Write("r2 tortburysh qabyrgalary : ");
        r2.Show();
        Console.WriteLine("r2 tortburysh audany : " + r2.Area());
    }
}
```

Бұл программа нәтижесі:

```
r1 tortburysh qabyrgalary:  4  5
r1 tortburysh audany : 20

r2 tortburysh qabyrgalary:  8  10
r2 tortburysh audany : 80
```



Әдіс объектіні қайтарғанда, объект оған сілтеме болғанша жұмыс істей береді. Сонан кейін объект "қоқыс" ретінде жойылады. Сол себепті объектіні құрған әдіс аяқталғанмен, объект жойылмайды. Объект типіндегі қайтарылатын мәліметтердің практикалық мысалы ретінде *класс фабрикасы* қолданылады. Ол өз класындағы объектілер тұрғызуға арналған әдіс болып табылады.

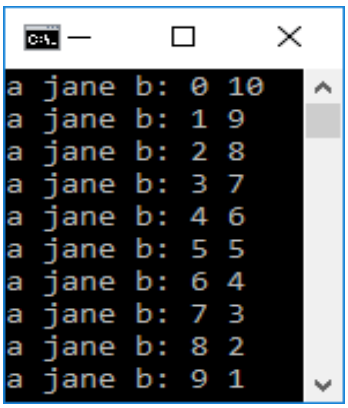
Кейбір кездерде класс қолданушыларына сол класс конструкторларын пайдалануға мүмкіндік беру қауіпсіздік талаптарына қарай қажет болмайды, ол объектіні тұрғызудың кейбір сыртқы факторларға тәуелді болатындығына байланысты да болып жатады. Мұндай жағдайларда объект құру үшін класс фабрикасы қолданылады. Мысал қарастырайық.

```
// Класс фабрикасын пайдалану.

using System;

class MyClass {
    int a, b;               // кластың жабық мүшелері

    // MyClass класы үшін фабрика құру.
    public MyClass Factory(int i, int j) {
      MyClass t = new MyClass();

      t.a = i;
      t.b = j;

      return t;             // объектіні қайтару
    }

    public void Show() {
      Console.WriteLine("a jane b: " + a + " " + b);
    }
}

class MakeObjects {
    static void Main() {
```

```
        MyClass ob = new MyClass();
        int i,  j;
        // Класс фабрикасы арқылы объектілер қалыптастыру.
        for(i=0, j =10; i < 10; i++, j --) {
            MyClass anotherOb = ob.Factory(i, j);   // объект құру
            anotherOb.Show();
        }
        Console.WriteLine() ;
    }
}
```

Программа нәтижесі:

```
a jane b: 0 10
a jane b: 1 9
a jane b: 2 8
a jane b: 3 7
a jane b: 4 6
a jane b: 5 5
a jane b: 6 4
a jane b: 7 3
a jane b: 8 2
a jane b: 9 1
```



Мұнда MyClass класы үшін конструктор анықталмайды, сондықтан тек келісім бойынша құрылатын конструктор қолданылады. Бұл a және b айнымалыларының мәнін конструктор арқылы беруге болмайтынын білдіреді. Бірақ Factory() класы фабрикасында a және b айнымалыларының мәндері берілетін объектілер құруға болады. Оған қоса, a және b айнымалылары жабық болғандықтан, олардың мәндерін тек қана Factory() класы фабрикасының көмегімен беруге болады.

Main() әдісінде MyClass класы объектісінің экземпляры алынады, ал оның фабрикалық әдісі for циклінде басқа 10 объектіні құру үшін қолданылады.

Төмендегі жолда сол объектілер құрылатын код көрсетілген.

```
MyClass anotherOb = ob.Factory(i, j); // объект құру
```

циклдің әрбір қадамында anotherOb объектісіне сілтеме жасайтын айнымалы жасалады, оған класс фабрикасында қалыптасатын объектіге сілтеме меншіктеледі. Циклдің әрбір қадамы аяқталған сайын anotherOb айнымалысы өз әсер ету аймағынан шығып кетеді де, ол сілтеме жасап тұрған объект жойылады.

Әдістен жиым қайтару

C# тілінде жиымдар объектілер түрінде жасалған, бұл әдістердің жиымды қайтара алатындығын білдіреді. Басқа тілдердің көбісінде әдістерден жиымдар қайтарылмайды. Төменде FindFactors() әдісі оған берілген аргументтің көбейткіштерін жиым түрінде қайтаратын мысал берілген.

```
// Әдістен жиым қайтару.

using System;

class Factor {
   /* Әдіс num аргументінің көбейткіштерінен тұратын facts жиымын қайтарады. Әдістен
   out типіндегі numfactors параметрі - анықталған көбейткіштер санын қайтарады. */
   public int[] FindFactors(int num, out int numfactors) {
```

```
        int[] facts = new int[80];          //жиым ені алдын ала
                                            // артығымен 80 болып алынған
        int i, j;

        // Көбейткіштерді тауып, facts жиымына орналастыру.
        for(i=2, j=0; i < num/2 + 1; i++)
             if( (num%i)==0 ) {
                 facts[j] = i;
                j++;
             }

        numfactors = j ;
        return facts;
    }
}

class FindFactors {
    static void Main() {
        Factor f = new Factor();
        int numfactors;
        int[ ] factors;

        factors = f.FindFactors(1000, out numfactors);

        Console.WriteLine("1000 sanynyng kobeitkishteri: ");
        for(int i=0; i < numfactors; i++)
             Console.Write(factors[i] + " ") ;
        Console.WriteLine() ;
    }
}
```

Жоғарыдағы Factor класындағы FindFactors() әдісі былай жарияланған:

```
public int[] FindFactors(int num, out int numfactors) {
```

Мұндағы int типіндегі қайтарылатын жиым қалай көрсетілгеніне назар салыңыздар. Бұл синтаксисті жалпылама етуге болады. Әдіс жиым қайтарған сайын, ол осылай көрсетіледі, бірақ оның типі мен өлшемі есепке алынуы тиіс. Мысалы, келесі жолда double типіндегі екі өлшемді жиым қайтаратын someMeth() әдісі жарияланып отыр:

```
public doublet,] someMeth() { // ...
```