Референтни и вредносни параметри

Све досадашње обрађене функције су имале вредносне параметре.
То значи, када се користи параметар, даје се вредност у промењиву коју користи функција.
Било која промена над том промењивом унутар функције нема никакав ефекат на аргумент специфициран у позиву функције.
Нпр, функција која дуплира и приказује вредност коришћеног параметра:

```
static void Main(string[] args)
{
      int mojBroj = 5;
      Console.WriteLine($"moj broj je {mojBroj}");
      PrikazDuplo(mojBroj);
      Console.WriteLine($"moj broj je {mojBroj}");
}
static void PrikazDuplo(int x)
{
      x *= 2;
      Console.WriteLine($"x je dupliran, pa je sada {x}");
}
```

Даје:
moj broj je 5
x je dupliran, pa je sada 10
moj broj je 5

Позивање функције PrikazDuplo() коришћењем mojBroj као аргумента нема утицаја на вредност промењиве mojBroj унутар функције Main(), иако је параметар који је повезан (x) дуплиран.
Овако је добро али ако је потребно променити и вредност у mojBroj, то представља проблем.
Може се користити функција која враћа вредност за промењиву mojBroj:

```
using System;
namespace Proba
{
    class Program
    {
        static void Main(string[] args)
        {
            int mojBroj = 5;
            Console.WriteLine($"moj broj je {mojBroj}");
            mojBroj = PrikazDuplo(mojBroj);
            Console.WriteLine($"moj broj je {mojBroj}");
        }
        static int PrikazDuplo(int x) => x *= 2;
    }
}
```

Даје:
moj broj je 5
moj broj je 10

Ипак, овај код је мало интуитиван и не би се слагао са променама вредности код више коришћених промењивих као аргумената (пошто функција враћа само једну вредност).

Уместо тога, потребно је користити параметар по референци, што значи да ће функција радити са истом промењивом као што је она коришћена у позиву функције, а не са неком промењивом која има исту вредност.
Било која промена на тој промењивој ће се осликати на вредност промењиве коришћене као аргумент.
Да би се то реализовало, користи се службена реч ref приликом спецификације параметара:

```
using System;
namespace Proba
{
    class Program
    {
        static void Main(string[] args)
        {
            int mojBroj = 5;
            Console.WriteLine($"moj broj je {mojBroj}");
            PrikazDuplo(ref mojBroj);
            Console.WriteLine($"moj broj je {mojBroj}");
        }
        static void PrikazDuplo(ref int x)
        {
            x *= 2;
            Console.WriteLine($"x je duplirano pa je sada {x}");
        }
    }
}
```

Даје:
moj broj je 5
x je duplirano pa je sada 10
moj broj je 10

Постоје два ограничења на промењивој која је употребљена као параметар по референци.
Први, функција може да резултује у промени вредности параметра по референци, па се мора користити неконстантана (nonconstant) промењива у позиву функције.
Зато је следећи пример нелегалан:

```
const int mojBroj = 5;
Console.WriteLine($"moj broj je {mojBroj}");
PrikazDuplo(ref mojBroj);
Console.WriteLine($"moj broj je {mojBroj}");
```

Други, мора се користити иницијализована промењива.
C# не допушта претпоставку да ће параметар по референци бити иницијализован у функцији која га и користи.
Зато је следећи пример нелегалан:

```
int mojBroj;
PrikazDuplo(ref mojBroj);
Console.WriteLine($"moj broj je {mojBroj}");
```

До сада се службена реч ref примењивала само на параметре функције, али могуће је да се примени и на локалне промењиве и на враћене вредности.
У следећем примеру, mojRefBroj има референцу на mojBroj, па промена над mojRefBroj резултује променом над mojBroj.
Ако је вредност обе промењиве приказана, она ће бити идентична (6) за обе промењиве:

```
int mojBroj = 5;
ref int mojRefBroj = ref mojBroj;
mojRefBroj = 6;
```

Такође је могуће користити ref службену реч као повратни тип.
У следећем примеру службена реч ref идентификује повратни тип као ref int и такође је у телу кода, чиме се указује функцији да врати ref x:

```
static ref int PrikazDuplo(int x)
{
     x *= 2;
     return ref x;}
```

Стартовање кода би изазвало грешку у компајлирању.
Разлог за то је што се не може додати тип промењиве као параметар функције преко референце без уписа службене речи ref као префикса испред декларације промењиве.
У следећем коду је ref додато да би функција могла да се копајлира:

```
static ref int PrikazDuplo(ref int x)
{
     x *= 2;
     return ref x;}
```

Промењиве попут стрингова и низова су референтног типа, а низови се могу вратити са ref службеном речи без декларације параметара:

```
static ref int VracenSaRef()
{
     int[] niz = { 2 };
     return ref niz[0];}
```

Иако су стрингови референтног типа, они су посебан случај пошто су непромењиви.
То значи да се не могу мењати пошто било каква модификација резултује појавом новог стринга; стари стринг је заборављен у меморији.
C# компајлер неће допустити покушај повратка стринга са ref.