

Bài 4.2: Tìm hiểu về tham số

- ✓ Truyền tham chiếu vs giá trị
- ✓ Keyword ref
- ✓ Keyword in
- ✓ Keyword out
- ✓ Ví dụ minh họa
- ✓ Bài tập thực hành

Truyền tham chiếu vs truyền giá trị

- ✓ Khi gọi phương thức, ta có thể truyền các đối số vào cho các tham số của phương thức bằng 2 cách: truyền tham chiếu và truyền giá trị.
- ✓ Mặc định tất cả các đối số kiểu value type như các kiểu số, char, bool, struct khi truyền vào phương thức là truyền giá trị.
- ✓ Truyền giá trị tức là tạo một bản sao của giá trị, biến gửi vào cho tham số.
- ✓ Do đó khi có thay đổi xảy ra với các tham số, giá trị gốc của nó tức các đối số không bị ảnh hưởng.
- ✓ Nhược điểm của truyền giá trị là làm giảm hiệu năng chương trình khi giá trị truyền vào có kích thước lớn như các struct.
- ✓ Truyền tham chiếu là truyền địa chỉ của biến vào cho tham số của phương thức.
- ✓ Truyền tham chiếu cho phép phương thức làm thay đổi giá trị gốc của các đối số.
- ✓ Truyền tham chiếu làm tăng hiệu năng chương trình vì không phải sao chép dữ liệu.

Truyền tham chiếu

- ✓ C# cho phép ta truyền tham chiếu với cả các kiểu value type và các kiểu tham chiếu.
- ✓ Để truyền tham chiếu ta sử dụng 3 keyword:
 - ✓ Keyword **ref**: cho phép phương thức được quyền thay đổi giá trị của đối số.
 - ✓ Keyword **in**: cho phép truyền tham chiếu nhưng không cho phép phương thức thay đổi giá trị đối số.
 - ✓ Keyword **out**: cho phép truyền tham chiếu trong đó phương thức được gọi phải gán giá trị mới cho đối số.
- ✓ Tất cả các đối số sử dụng để truyền tham chiếu phải là các biến.
- ✓ Không thể dùng các hằng số, thuộc tính làm đối số trong lời gọi truyền tham chiếu.

Sử dụng keyword ref

- ✓ Keyword **ref** sử dụng để chỉ ra rằng một giá trị đang được truyền tham chiếu.
- ✓ Keyword **ref** sử dụng trong 4 ngữ cảnh khác nhau:
 - ✓ Sử dụng trong định nghĩa tham số phương thức và trong lời gọi phương thức để truyền tham chiếu.
 - ✓ Sử dụng trong kiểu trả về và trước giá trị trả về của phương thức để cho phép trả về tham chiếu.
 - ✓ Trước các biến nhận giá trị tham chiếu được trả về bởi một phương thức.
 - ✓ Trong khai báo của một struct để khai báo một ref struct hoặc một readonly ref struct.
- ✓ Biến được sử dụng để truyền tham chiếu vào phương thức bắt buộc phải được khởi tạo trước đó.
- ✓ Để truyền tham chiếu với ref, ta thêm keyword này vào trước tên biến trong đối số và trước kiểu của tham số trong khai báo phương thức.

Sử dụng keyword ref

- ✓ Ví dụ: phương thức Swap sử dụng truyền tham chiếu để trao đổi giá trị của hai biến a, b.

```
0 references
static void Main()
{
    int a = 20;
    int b = 50;
    Console.WriteLine($"Before change. a = {a}, b = {b}");
    Swap(ref a, ref b);
    Console.WriteLine($"After change. a = {a}, b = {b}");
}
```

```
1 reference
static void Swap(ref int first, ref int second)
{
    int tmp = first;
    first = second;
    second = tmp;
}
```

```
C:\WINDOWS\system32\cmd.exe
Before change. a = 20, b = 50
After change. a = 50, b = 20
Press any key to continue . . .
```

Sử dụng keyword ref

✓ Cũng là ví dụ trên nhưng không sử dụng tham chiếu ref:

```
0 references
static void Main()
{
    int a = 20;
    int b = 50;
    Console.WriteLine($"Before change. a = {a}, b = {b}");
    Swap(a, b);
    Console.WriteLine($"After change. a = {a}, b = {b}");
}
```

```
1 reference
static void Swap(int first, int second)
{
    int tmp = first;
    first = second;
    second = tmp;
}
```

```
C:\WINDOWS\system32\cmd.exe
Before change. a = 20, b = 50
After change. a = 20, b = 50
Press any key to continue . . .
```

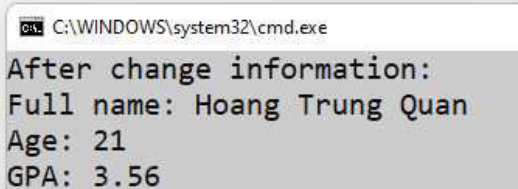
Sử dụng keyword ref

- ✓ Không sử dụng **ref**, **in**, **out** trong các loại phương thức sau:
 - ✓ Phương thức đồng bộ với keyword **async** trong định nghĩa phương thức.
 - ✓ Phương thức iterator chứa **yield return** hoặc **yield break**.
 - ✓ Sử dụng giới hạn trong phương thức mở rộng(extension methods).
- ✓ Truyền tham chiếu còn cho phép một phương thức làm thay đổi nhiều giá trị. Ví dụ:

```
var fullName = "Tran Van Quang";
var age = 20;
var gpa = 3.22f;
ChangeInfo(ref fullName, ref age, ref gpa);
Console.WriteLine($"After change information: ");
Console.WriteLine($"Full name: {fullName}");
Console.WriteLine($"Age: {age}");
Console.WriteLine($"GPA: {gpa}");
}
```

Lời gọi

```
1 reference
static void ChangeInfo(ref string fullName, ref int age, ref float gpa)
{
    fullName = "Hoang Trung Quan";
    age = 21;
    gpa = 3.56f;
}
```



```
C:\WINDOWS\system32\cmd.exe
After change information:
Full name: Hoang Trung Quan
Age: 21
GPA: 3.56
```

Ref return và ref local

- ✓ Để một phương thức có thể trả về tham chiếu, ta thêm **ref** vào trước kiểu trả về của phương thức.
- ✓ Trước giá trị cần trả về trong phương thức trả về tham chiếu ta thêm **ref**.
- ✓ Để lưu trữ giá trị tham chiếu trả về bởi phương thức trên ta sử dụng **ref** local. Đây là một biến có keyword **ref** trước kiểu dữ liệu và được gán thêm **ref** vào trước lời gọi tới phương thức trả về một tham chiếu.

Ví dụ

```
int[] numbers = { 1, 3, 2, 5, 8, 4, 6, 9, 7 };  
ref var maxValue = ref FindMax(numbers);  
Console.WriteLine($"Max value in array: {maxValue}");  
// change max value by double it  
maxValue *= 2;  
// find new max value in the array  
ref var newMax = ref FindMax(numbers);  
Console.WriteLine($"New max value in array: {maxValue}");  
}
```

2 references

```
static ref int FindMax(int[] arr)  
{  
    int max = arr[0];  
    int maxValueIndex = 0;  
    for (int i = 0; i < arr.Length; i++)  
    {  
        if (max < arr[i])  
        {  
            max = arr[i];  
            maxValueIndex = i;  
        }  
    }  
    return ref arr[maxValueIndex];  
}
```

C:\WINDOWS\system32\cmd.exe

```
Max value in array: 9  
New max value in array: 18  
Press any key to continue .
```

Sử dụng keyword in

- ✓ Keyword **in** cho phép ta truyền tham chiếu nhưng không cho phép thay đổi giá trị của biến gốc dùng làm đối số.
- ✓ Có thể hiểu tham chiếu **in** là chức năng giúp nâng cao hiệu năng chương trình bằng cách truyền tham chiếu ở chế độ chỉ cho phép đọc.
- ✓ Lý do nâng cao hiệu năng là vì khi dữ liệu truyền vào có kích thước lớn, truyền tham chiếu chỉ truyền địa chỉ biến gốc vào phương thức chứ không tạo bản sao của biến rồi mới truyền giá trị bản sao này vào lời gọi.
- ✓ Biến sử dụng với keyword **in** bắt buộc phải được khởi tạo trước khi truyền đi.
- ✓ Trong khai báo tham số của phương thức ta thêm keyword **in** trước kiểu của tham số muốn truyền tham chiếu kiểu này.
- ✓ Ở lời gọi ta có thể bỏ qua việc chỉ rõ **in** trước đối số tham chiếu in.

Ví dụ

- ✓ Ví dụ sau sử dụng truyền tham chiếu in. Ta có thể chỉ định rõ (hoặc bỏ qua) keyword `in` trong lời gọi phương thức. Ta không thể làm thay đổi giá trị của tham số truyền tham chiếu in:

```
string msg = "Hello message";  
PrintMessage(msg); // ok  
PrintMessage(in msg); // still ok  
}  
  
2 references  
static void PrintMessage(in string message)  
{  
    Console.WriteLine(message);  
    message = "New Message"; // error, don't do this here!  
}
```

Sử dụng keyword out

- ✓ Keyword out đặc biệt được sử dụng để đẩy nhiệm vụ thay đổi giá trị của đối số cho phương thức được gọi.
- ✓ Ta sử dụng keyword này trong trường hợp giá trị của biến chỉ được biết sau khi phương thức hoàn thành nhiệm vụ.
- ✓ Keyword này cần được chỉ rõ cả trong tham số và trong đối số khi gọi phương thức.
- ✓ Biến sử dụng với keyword out không bắt buộc phải được khởi tạo trước.
- ✓ Trước C# 7 ta phải khai báo biến trước khi truyền tham chiếu out vào phương thức.
- ✓ Từ C# 7 ta có thể gộp việc khai báo tham số tham chiếu out vào lời gọi để tránh rườm rà.

Sử dụng keyword out

- ✓ Ví dụ sau nhập vào thông tin sinh viên từ bàn phím. Dữ liệu chỉ được cung cấp khi người dùng nhập vào. Trước đó ta không biết cụ thể. Do đó sử dụng tham chiếu out:

```
GetStudentData(out string fullName, out int age, out float gpa); // from C# 7
Console.WriteLine($"After change information: ");
Console.WriteLine($"Full name: {fullName}");
Console.WriteLine($"Age: {age}");
Console.WriteLine($"GPA: {gpa}");
}

1 reference
static void GetStudentData(out string fullName, out int age, out float gpa)
{
    Console.Write("Full name: ");
    fullName = Console.ReadLine();
    Console.Write("Age: ");
    age = int.Parse(Console.ReadLine());
    Console.Write("Gpa: ");
    gpa = float.Parse(Console.ReadLine());
}
```



Nội dung tiếp theo

Nạp chồng phương thức