Bài 4.5:
Hàm cục bộ
Local functions

- ✓ Khái niệm và mục đích sử dụng
- ✓ Cú pháp khai báo
- ✓ Ví dụ minh họa
- ✓ Bài tập thực hành

Khái niệm và mục đích sử dụng

- ✓C# 7 giới thiệu local functions – hàm cục bộ.
- ✓Hàm cục bộ là một phương thức riêng tư(private) của một kiểu.
- ✓Hàm cục bộ được định nghĩa bên trong một thành phần của lớp và chỉ được gọi bởi các câu lệnh tại nơi chứa nó.
- ✓Hàm cục bộ có thể được khai báo và gọi từ:
  - ✓ Phương thức, đặc biệt các phương thức iterator và async
  - ✓ Phương thức khởi tạo
  - ✓ Các property accessor
  - ✓ Event accessor
  - ✓ Phương thức vô danh
  - ✓ Biểu thức lambda
  - ✓ Finalizer
  - ✓ Các hàm cục bộ khác

Khái niệm và mục đích sử dụng

- ✓Định nghĩa của hàm cục bộ có thể được đặt trước hoặc sau lời gọi đều được chấp nhận.
- ✓Mục đích sử dụng của local function là tạo ra các phương thức sao cho nó chỉ được gọi từ trong ngữ cảnh bên trong của một phương thức khác.
- ✓Điều này sẽ tránh được các lời gọi không đúng mục đích sử dụng cũng như ẩn giấu chi tiết thực thi của một phương thức vào bên trong bản thân nó.

Cú pháp khai báo local function

- ✓Cú pháp của local function giống với một phương thức thông thường nhưng lược bỏ access modifier.
- ✓Cú pháp:

```
<modifier> returnType MethodName(params) {
    // statements
}
```

- ✓Trong đó:
  - ✓ <modifier> có thể bỏ qua hoặc là async, unsafe, static(C# 8), extern(C# 9).
  - ✓ returnType là kiểu trả về của hàm cục bộ.
  - ✓ MethodName là tên hàm
  - ✓ Phần params là các tham số của hàm.
  - ✓ Các câu lệnh cần thực hiện đặt trong phần statements.
- ✓Hàm cục bộ không static có thể truy cập vào các biến và tham số của thành phần chứa nó.

Ví dụ local function

```
1 reference
static void ListedPrimeNumbers(int from, int to)
{
    for (int i = from; i <= to; i++)
    {
        if (IsPrime(i))
        {
            Console.Write($"{i} ");
        }
    }
    Console.WriteLine();

    bool IsPrime(int number) // hàm cục bộ
    {
        if (number < 2)
        {
            return false;
        }
        int bound = (int)Math.Sqrt(number);
        for (int i = 2; i <= bound; i++)
        {
            if (number % i == 0)
            {
                return false;
            }
        }
        return true;
    }
}
```

```
1 reference
static void ListedPrimeNumbers(int from, int to)
{
    bool IsPrime(int number) // hàm cục bộ
    {
        if (number < 2)
        {
            return false;
        }
        int bound = (int)Math.Sqrt(number);
        for (int i = 2; i <= bound; i++)
        {
            if (number % i == 0)
            {
                return false;
            }
        }
        return true;
    }

    for (int i = from; i <= to; i++)
    {
        if (IsPrime(i)) // call to local function
        {
            Console.Write($"{i} ");
        }
    }
    Console.WriteLine();
}
```

Ví dụ local function static vs non-static

✓Hàm cục bộ static và non-static:

```
0 references
static void OtherMethod(int number)
{
    int localVariable = 10;
    LocalFunc(); // call to local function
    static void LocalFunc() // static local function
    {
        Console.WriteLine("Local function inside other method");
        Console.WriteLine(localVariable); // error
        Console.WriteLine(number); // still error
    }
}
```

```
0 references
static void OtherMethod(int number)
{
    int localVariable = 10;
    LocalFunc(); // call to local function
    void LocalFunc()
    {
        Console.WriteLine("Local function inside other method");
        Console.WriteLine(localVariable); // ok
        Console.WriteLine(number); // still ok
    }
}
```

Nội dung tiếp theo

Phương thức Main