

# Bài: Vòng lặp For trong C#.

Xem bài học trên website để ủng hộ Kteam: [Vòng lặp For trong C#](#).

Mọi vấn đề về lỗi website làm ảnh hưởng đến bạn hoặc thắc mắc, mong muốn khóa học mới, nhằm hỗ trợ cải thiện Website. Các bạn vui lòng phản hồi đến Fanpage [How Kteam](#) nhé!

## Dẫn nhập

Ở các bài học trước, chúng ta đã cùng nhau tìm hiểu về [CẤU TRÚC VÒNG LẶP GOTO](#). Ở bài này chúng ta sẽ tiếp tục tìm hiểu chi tiết cách sử dụng **vòng lặp for** trong C#.

## Nội dung

Để đọc hiểu bài này tốt nhất các bạn nên có kiến thức cơ bản về các phần:

- [CẤU TRÚC CƠ BẢN MỘT CHƯƠNG TRÌNH C# console application](#)
- [BIẾN](#) và [KIỂU DỮ LIỆU](#) trong C#
- [TOÁN TỬ TRONG C#](#)
- [CÂU ĐIỀU KIỆN TRONG C#](#)
- [CẤU TRÚC CƠ BẢN CỦA MỘT VÒNG LẶP](#)

Trong bài học này, chúng ta sẽ cùng tìm hiểu các vấn đề:

- Cấu trúc của một vòng lặp **for** trong C#
- Các ví dụ sử dụng **for** trong C#

## Cấu trúc của một vòng lặp for trong C#

Cú pháp:

```
for ([Khởi tạo]; [Điều kiện lặp]; [Bước lặp lại])  
{  
    // Khối lệnh được lặp lại. Có thể bỏ trống  
}
```

Trong đó:

- Các phần **[Khởi tạo]**; **[Điều kiện lặp]**; **[Bước lặp lại]** hoàn toàn có thể để trống như ví dụ sau.
- Mỗi đoạn **[Khởi tạo]**; hay **[Điều kiện lặp]**; hay **[Bước lặp lại]** là một câu lệnh riêng.

**Tiến trình:**

- Ban đầu trình biên dịch sẽ di vào phần **khởi tạo** chạy đoạn lệnh khởi tạo.
- Tiếp theo kiểm tra **điều kiện lặp**. Rồi thực hiện khối code bên trong vòng lặp **for**. Khi đến ký hiệu } thì sẽ quay lên **bước lặp lại**.
- Sau đó lại kiểm tra **điều kiện lặp** rồi tiếp tục thực hiện đoạn code trong khối lệnh. Đến khi **điều kiện lặp** không còn thỏa mãn thì sẽ kết thúc vòng lặp **for**.
- Trường hợp khác:

```
for (; ) // lưu ý dấu ;  
  
{  
  
    // Khối lệnh được lặp lại. Có thể bỏ trống  
  
}
```

Trong đó:

- Vòng lặp **for** này trở thành vòng lặp vô tận.
- Lưu ý dấu ; vẫn phải có.

Chúng ta sẽ cùng tìm hiểu ý nghĩa và cách sử dụng các phần [Khởi tạo]; [Điều kiện lặp]; [Bước lặp lại] nhé.

## Khởi tạo

Khi bắt đầu vào đoạn code của vòng lặp **for**, đoạn lệnh này sẽ được chạy đầu tiên. Và chỉ được gọi duy nhất một lần trong vòng đời của vòng lặp **for**.

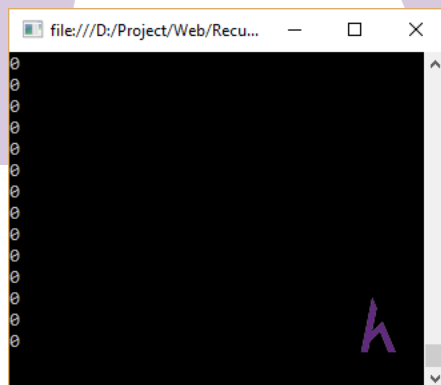
**Ví dụ:**

- **Ví dụ 1:**

**C#:**

```
static void Main(string[] args)  
{  
    // i sẽ được khởi tạo lần đầu tiên tại vòng lặp for  
    // khi vòng đời của vòng lặp for kết thúc bộ nhớ của biến i sẽ được giải phóng  
    // hay nói cách khác i là biến cục bộ của vòng lặp for  
  
    for (int i = 0; ; )  
    {  
        Console.WriteLine(i);  
    }  
    Console.ReadKey();  
}
```

- Kết quả màn hình xuất ra một loạt giá trị 0 vì **i = 0** được khởi tạo tại phần **khởi tạo** của vòng lặp **for** và vòng lặp **for** này không có Điều kiện lặp nên chương trình sẽ chạy vô tận.
- Ở trường hợp này **i** được gọi là biến đếm (thuật ngữ lập trình dùng cho một biến có tác dụng tăng giá trị lên mỗi lần lặp lại).



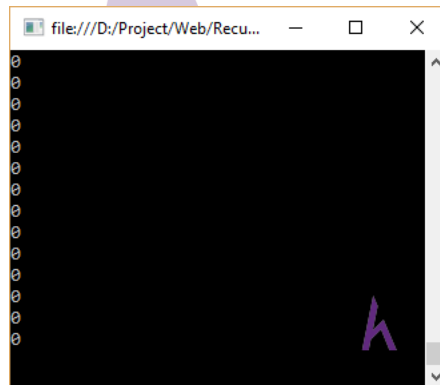
- **Ví dụ 2:**
  - Chúng ta không nhất thiết phải khai báo một biến ngay tại vị trí khởi tạo. Ta có thể chỉ gán giá trị hoặc không làm gì cả (bỏ trống).

**C#:**

```
static void Main(string[] args)
{
    int i;
    // i được gán giá trị bằng 0
    // i lúc này không phải biến cục bộ của vòng lặp for
    // i thuộc hàm main

    for (i = 0; ; )
    {
        Console.WriteLine(i);
    }
    Console.ReadKey();
}
```

- Kết quả tương tự như ví dụ trên



- **Ví dụ 3:**
  - Chỉ có thể có duy nhất một câu lệnh **khởi tạo** trong vòng lặp (lưu ý dấu , và dấu ; ở hai ví dụ sau)

**C#:**

```
for static void Main(string[] args)
{
    int i;
    // lỗi vì chỉ được phép có duy nhất một dòng lệnh khởi tạo trong vòng lặp for

    for (i = 0, int j = 0; ; )
    {
        Console.WriteLine(i);
    }
    Console.ReadKey();
}
```

Hay

**C#:**

```
static void Main(string[] args)
{
    int i;
    // lỗi vì chỉ được phép có duy nhất một dòng lệnh khởi tạo trong vòng lặp for

    for (i = 0; int j = 0; ; )
    {
        Console.WriteLine(i);
    }
    Console.ReadKey();
}
```

## Điều kiện lặp

**Điều kiện lặp** là một biểu thức logic với kết quả trả về bắt buộc là **true** hoặc **false** (có thể bỏ trống sẽ trả về kết quả là **true**).

Điều kiện lặp là dòng lệnh thứ 2 vòng **for** sẽ chạy vào khi chạy lần đầu tiên (Khởi tạo chạy trước). Từ lần lặp thứ 2 của vòng **for**, Điều kiện lặp cũng là dòng lệnh thứ 2 được chạy (sau bước lặp lại). (Cứ nhớ là luôn đứng thứ 2)

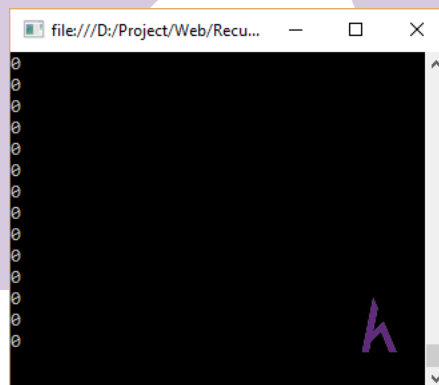
Khi câu điều kiện lặp không còn thỏa mãn (kết quả là **false**) thì vòng lặp **for** sẽ kết thúc.

**C#:**

```
static void Main(string[] args)
{
    int i;
    // vòng lặp for này vẫn lặp vô tận vì không bao giờ thỏa mãn điều kiện dừng
    // i luôn == 0
    // Điều kiện lặp luôn là true

    for (i = 0; i < 10;)
    {
        Console.WriteLine(i);
    }
    Console.ReadKey();
}
```

- Ta có thể thấy Điều kiện lặp của vòng lặp này luôn là **true**, nên vòng lặp sẽ lặp vô tận.



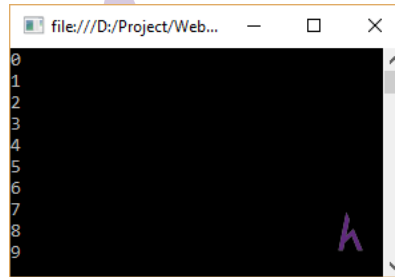
- Để giải quyết vấn đề này và cho vòng lặp kết thúc khi thỏa mãn điều kiện lặp. Chúng ta thêm một đoạn code **i++**; ngay dưới đoạn code **Console.WriteLine(i);**

**C#:**

```
static void Main(string[] args)
{
    int i;

    for (i = 0; i < 10;)
    {
        Console.WriteLine(i);
        i++;
    }
    Console.ReadKey();
}
```

- Kết quả màn hình xuất ra các giá trị số nguyên từ 0 đến 9 (10 lần). Chứng tỏ vòng lặp đã kết thúc sau 10 lần lặp (không còn lặp vô tận).



#### Lưu ý:

- Giá trị in ra từ 0 đến 9 chứ không phải đến 10. Vì Điều kiện lặp là  $i < 10$  ( $10 == 10$  nên câu điều kiện là **false** và kết thúc vòng lặp. Vẫn thỏa mãn lặp 10 lần).
- Sau mỗi lần lặp giá trị  $i$  lại tăng lên 1 đơn vị. Sau 11 lần thì giá trị  $i == 10$ , không còn thỏa mãn Điều kiện lặp nữa nên vòng lặp kết thúc.
- Các bạn có thể xem bảng thử dưới đây:

| Lần       | i         | $i < 10$     |
|-----------|-----------|--------------|
| 1         | 0         | TRUE         |
| 2         | 1         | TRUE         |
| 3         | 2         | TRUE         |
| 4         | 3         | TRUE         |
| 5         | 4         | TRUE         |
| 6         | 5         | TRUE         |
| 7         | 6         | TRUE         |
| 8         | 7         | TRUE         |
| 9         | 8         | TRUE         |
| 10        | 9         | TRUE         |
| <b>11</b> | <b>10</b> | <b>FALSE</b> |

- Bạn hoàn toàn có thể để giá trị **true** hoặc **false** vào phần điều kiện lặp (bỏ trống mặc định là **true**). Hoặc một biểu thức logic phức tạp nhưng kết quả cuối cùng trả về là **true** hoặc **false**.

**C#:**

```
static void Main(string[] args)
{
    int i;

    for (i = 0; (i % 3 == 0) && (i < 10);)
    {
        Console.WriteLine(i);
        i++;
    }
    Console.ReadKey();
}
```

Hay

**C#:**

```
static void Main(string[] args)
{
    int i;

    for (i = 0; false;)
    {
        Console.WriteLine(i);
        i++;
    }
    Console.ReadKey();
}
```

Hoặc

**C#:**

```
static void Main(string[] args)
{
    int i;

    for (i = 0; true;)
    {
        Console.WriteLine(i);
        i++;
    }
    Console.ReadKey();
}
```

## Bước lặp lại

Như ví dụ trên ta thấy. Mỗi lần muốn tăng giá trị của **i** ta phải dùng một đoạn lệnh **i++** ; ở cuối khối lệnh. Vậy trường hợp bất cứ khi nào lặp lại ta cũng cần thực thi đoạn lệnh **i++** ; thì sao? Để tiện hơn cho việc code. Chúng ta có một phần tiếp theo để tìm hiểu. Đó là **bước lặp lại**.

Xét đoạn code sau:

**C#:**

```
static void Main(string[] args)
{
    int i;

    for (i = 0; i < 10;)
    {
        Console.WriteLine(i);
        i++;
    }
    Console.ReadKey();
}
```

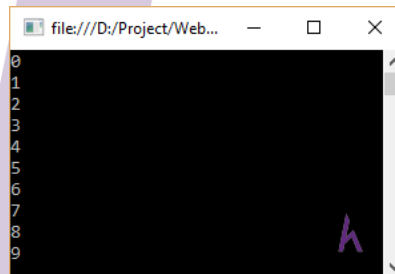
- Ta có thể viết gọn lại bằng cách đưa **i++**; vào phần bước lặp lại của khối **for**.

**C#:**

```
static void Main(string[] args)
{
    int i;

    for (i = 0; i < 10; i++)
    {
        Console.WriteLine(i);
    }
    Console.ReadKey();
}
```

- Kết quả tương tự như bình thường



- Chúng ta có thể thực hiện nhiều đoạn lệnh trong bước lặp.

**C#:**

```
static void Main(string[] args)
{
    int i;
    int j = 0;

    for (i = 0; i < 10; i++, j += 3)
    {
        Console.WriteLine(i);
    }
    Console.ReadKey();
}
```

- Ta thấy đoạn **i++** và **j += 3** được cách nhau bởi dấu phẩy (,)
- Với mỗi đoạn lệnh trong bước lặp. Chúng được phân cách nhau bởi dấu phẩy (,)

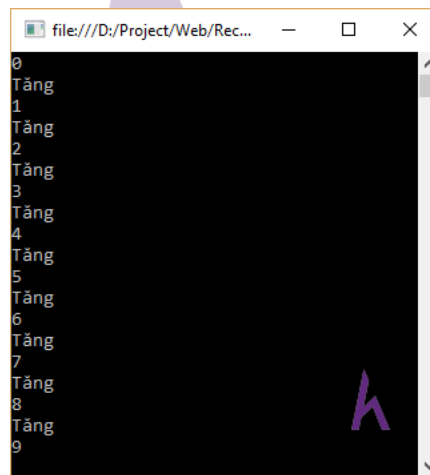
- **Lưu ý:** Đoạn code trong bước lặp còn có thể thêm cả **Console.WriteLine("Tăng")** vào (**khuyến cáo không nên**). Nhưng không thể thực hiện đoạn code có chứa từ khóa (như **if**, **for** ...).

**C#:**

```
static void Main(string[] args)
{
    int i;
    int j = 0;

    for (i = 0; i < 10; i++, j += 3, Console.WriteLine("Tăng"))
    {
        Console.WriteLine(i);
    }
    Console.ReadKey();
}
```

- Kết quả xuất dòng chữ "Tăng" mỗi khi lặp lại.



- Không thể thêm câu điều kiện

**C#:**

```
static void Main(string[] args)
{
    int i;
    int j = 0;
    // lỗi đoạn , if (i % 2 == 0) j += 3

    for (i = 0; i < 10; i++, if (i % 2 == 0) j += 3)
    {
        Console.WriteLine(i);
    }
    Console.ReadKey();
}
```

## Các ví dụ sử dụng for trong C#

Chúng ta cùng thử một ví dụ đầy đủ về các phần của vòng lặp for nhé!

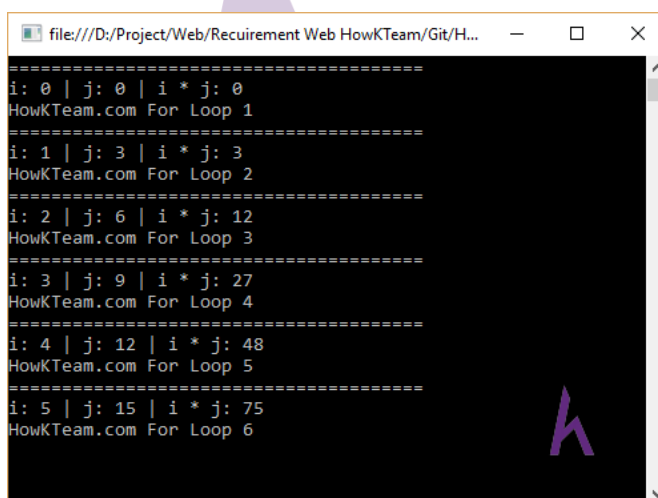
**C#:**



```
static void Main(string[] args)
{
    int n = 100;
    int j = 0;

    for (int i = 0; i * j < n; i++, j += 3, Console.WriteLine("HowKTeam.com For Loop {0}", i))
    {
        Console.WriteLine("=====");
        Console.WriteLine("i: {0} | j: {1} | i * j: {2}", i, j, i * j);
    }
    Console.ReadKey();
}
```

- Kết quả



```
file:///D:/Project/Web/Recuirement Web HowKTeam/Git/H...
=====
i: 0 | j: 0 | i * j: 0
HowKTeam.com For Loop 1
=====
i: 1 | j: 3 | i * j: 3
HowKTeam.com For Loop 2
=====
i: 2 | j: 6 | i * j: 12
HowKTeam.com For Loop 3
=====
i: 3 | j: 9 | i * j: 27
HowKTeam.com For Loop 4
=====
i: 4 | j: 12 | i * j: 48
HowKTeam.com For Loop 5
=====
i: 5 | j: 15 | i * j: 75
HowKTeam.com For Loop 6
```

- Chúng ta cũng có thể vẽ một hình chữ nhật rỗng **NxM** với vòng lặp **for**:

C#:

```

static void Main(string[] args)
{
    int N = 10;
    int M = 20;

    char drawChar = '*';
    char insideChar = ' ';

    // Vẽ từ trên xuống
    for (int i = 0; i < N; i++)
    {
        // Vẽ từ trái sang
        for (int j = 0; j < M; j++)
        {
            /*
             * Nếu đang ở tọa độ là cạnh trên hoặc dưới (i % (N - 1) == 0
             * hoặc đang ở cạnh trái hoặc phải (j % (M - 1) == 0)
             * mà không nằm ở cạnh trên hoặc dưới (i % (N - 1) != 0)
             * ((i % (N - 1) != 0) && (j % (M - 1) == 0))
             * thì vẽ ra ký tự của hình chữ nhật
             * ngược lại vẽ ra ký tự không thuộc hình chữ nhật
             */

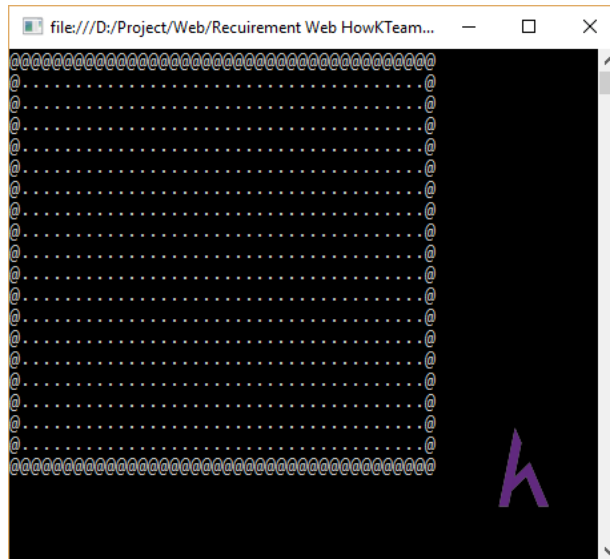
            if (i % (N - 1) == 0 || ((i % (N - 1) != 0) && (j % (M - 1) == 0)))
            {
                Console.Write(drawChar);    // lúc này là ký tự *
            }
            else
            {
                Console.Write(insideChar); // lúc này là ký tự rỗng ' '
            }
        }
        //mỗi lần vẽ xong một hàng thì xuống dòng
        Console.WriteLine();
    }
    Console.ReadKey();
}

```

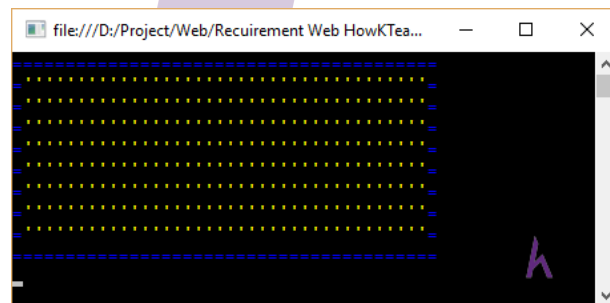
- Kết quả sẽ thấy một hình chữ nhật rỗng chiều ngang 10 chiều dài 20 được vẽ lên màn hình.



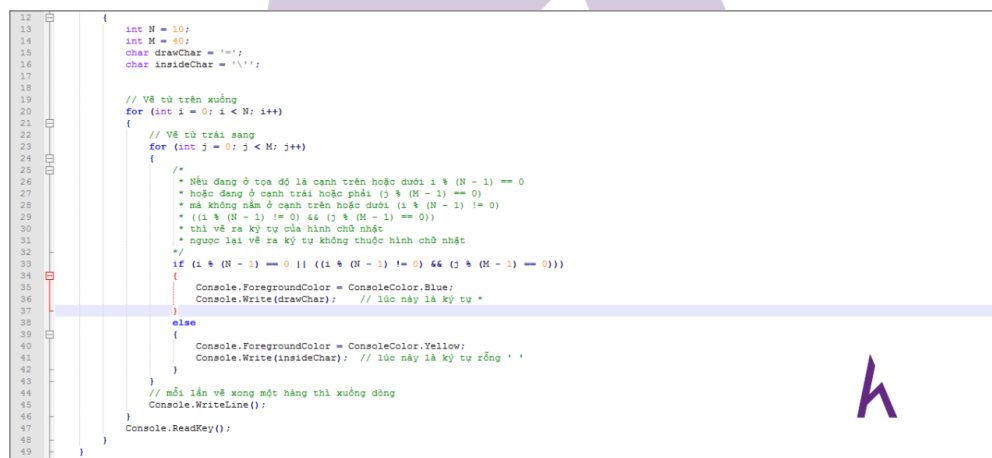
- Ta có thể thay đổi giá trị của M, N, drawChar và insideChar để có những hình chữ nhật màu mè cùng kích thước khác nhau.



- Bạn hãy thử thay đổi màu sắc hoặc vẽ hình chữ nhật ở bất cứ đâu trên màn hình console nhé. Thử vẽ hình chữ nhật đặc hay hình tam giác, hình tròn cũng là một thử thách thú vị đấy. Chúc các bạn thành công!



- Code gợi ý cho trường hợp trên:



## Kết luận

Qua bài này chúng ta đã nắm được cách sử dụng vòng lặp **for**. Một cấu trúc rất mạnh mẽ và tần xuất sử dụng cực kỳ nhiều trong lập trình. Những đặc điểm của vòng lặp **for**. Cùng những điều cần lưu ý.

Bài sau chúng ta sẽ đi sâu hơn vào cách sử dụng của [CẤU TRÚC VÒNG LẶP WHILE TRONG C#](#).

Cảm ơn các bạn đã theo dõi bài viết. Hãy để lại bình luận hoặc góp ý của mình để phát triển bài viết tốt hơn. Đừng quên “**Luyện tập – Thử thách – Không ngại khó**”.

