Bài: Nhập xuất cơ bản trong C# Console Application

Xem bài học trên website để ủng hộ Kteam: Nhập xuất cơ bản trong C# Console Application

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 hầu hết các ngôn ngữ lập trình, khi mới tiếp cận điều đầu tiên chúng ta quan tâm tới đó là làm sao để nhập dữ liệu từ bàn phím và làm sao để xuất dữ liệu ra màn hình. Như trong bài CẤU TRÚC LỆNH CƠ BẢN chúng ta đã xuất dòng chữ "**Kteam**" lên màn hình console. Vậy thì cấu trúc của các lệnh nhập xuất này là gì và sử dụng chúng như thế nào ?

Chúng ta sẽ cùng tìm hiểu vấn đề này trong bài học hôm nay – **CẤU TRÚC NHẬP XUẤT CƠ BẢN TRONG C#**.

Nội dung

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

- Cấu trúc cơ bản của các lệnh nhập xuất và ý nghĩa của chúng trong C#.
- Ví dụ demo chương trình nhập xuất bằng C#.

Cấu trúc cơ bản của các lệnh nhập xuất và ý nghĩa của chúng trong C#

Trong C# có 5 lệnh dùng để nhập xuất đó là:

:

```
Console.Write();

Console.WriteLine();

Console.Read();

Console.ReadLine();

Console.ReadKey();
```

Bây giờ chúng ta sẽ cùng tìm hiểu lần lượt các lệnh trên.

Console.Write();

Cú pháp:

> Console.**Write**(<*giá trị cần in ra màn hình*>);

Ý nghĩa: In giá trị ra màn hình console. Giá trị này có thể là 1 ký tự, 1 chuỗi, một giá trị có thể chuyển về kiểu chuỗi (Vấn đề này sẽ được trình bày chi tiết ở bài ÉP KIỂU TRONG C# và bài CLASS TRONG C#).

Ví dụ:

Các bạn tạo mới một Project Console Application (cách tạo project các bạn xem lại bài CẤU TRÚC LỆNH CƠ BẢN) và thử đoạn lệnh sau:

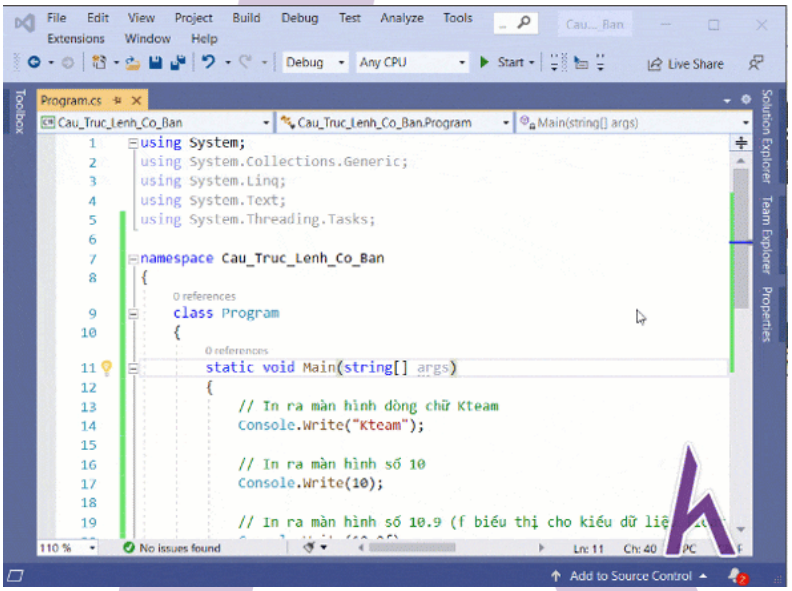
:

```
static void Main(string[] args)
        {
// In ra màn hình dòng chữ Kteam
            Console.Write("Kteam");

// In ra màn hình số 10
            Console.Write(10);

// In ra màn hình số 10.9 (f biểu thị cho kiểu dữ liệu float sẽ được trình bày chi tiết ở bài 5)
            Console.Write(10.9f);

// In ra màn hình chữ true của kiểu dữ liệu bool(sẽ được trình bày chi tiết ở bài 5)
            Console.Write(true);
        }
```

Thực hiện chạy chương trình thì ta thấy màn hình console vừa hiện lên đã tắt. Vậy làm sao để xem được kết quả?

Để giải quyết vấn đề này chúng ta có nhiều cách:

- Sử dụng sự trợ giúp của công cụ hỗ trợ lập trình (cụ thể ở đây là Visual Studio):
- Ta thực hiện chạy chương trình bằng cách vào Debug > **Start Without Debugging**.
- Phím tắt **Ctrl + F5**.



Sử dụng mẹo nhỏ để giải quyết:

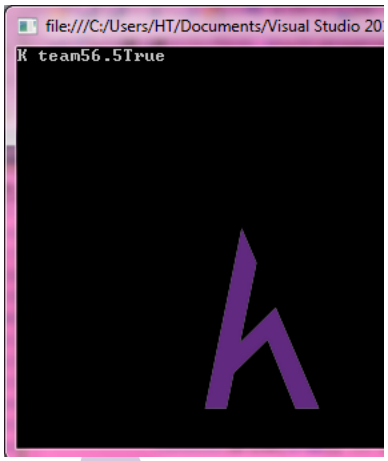
- Ý tưởng: ta sẽ dùng một lệnh nào đó là cho chương trình phải dừng lại đợi mình nhấn một phím bất kỳ mới kết thúc như vậy trước khi chúng ta nhấn một phím bất kỳ thì chúng ta có thể quan sát được kết quả trên màn hình console.
- Lệnh để thực hiện điều này:

:

```
Console.Read();
Console.ReadLine();
Console.ReadKey();
```

Chúng ta chỉ cần thêm 1 trong 3 lệnh trên vào cuối chương trình là xong. Ý nghĩa của 3 lệnh trên sẽ được giải thích chi tiết ở phần sau trong bài học này.

- Cuối cùng ta được kết quả:



Có vẻ như kết quả in ra không như chúng ta mong muốn thì phải. Vấn đề đặt ra bây giờ là **"chúng ta muốn in mỗi giá trị trên một dòng thì phải làm sao?"** Để trả lời cho câu hỏi này chúng ta cùng qua phần tiếp theo

---

Console.WriteLine();

Cú pháp:

```
Console.WriteLine(<giá trị cần in ra màn hình>);
```

Ý nghĩa:

Lệnh này cũng tượng tự như Console.Write()

Nhưng sẽ khác khi in giá trị ra màn hình xong nó sẽ tự động đưa con trỏ xuống dòng. Điều này giúp ta có thể giải quyết được vấn đề đã đặt ra ở phần trên.

Ngoài ra, để xuống dòng ta còn có nhiều cách khác như:

- Sử dụng ký tự đặc biệt: chúng ta sử dụng ký tự "**\n**" trong chuỗi in ra màn hình thì trình biên dịch sẽ tự động đổi nó thành ký tự xuống dòng.
  - Như vậy thay vì dùng Console.WriteLine("K team") ta có thể dùng Console.Write ("K team \n")
  - Các ký tự đặc biệt sẽ được giới thiệu trong phần sau của bài học.
- Sử dụng lệnh xuống dòng: ta sử dụng thêm 1 lệnh xuống dòng là
  :

  ```
  Environment.NewLine
  ```

  Như vậy thay vì dùng 2 cách trên ta sẽ viết
  :

  ```
  Console.Write(Environment.NewLine);
  ```

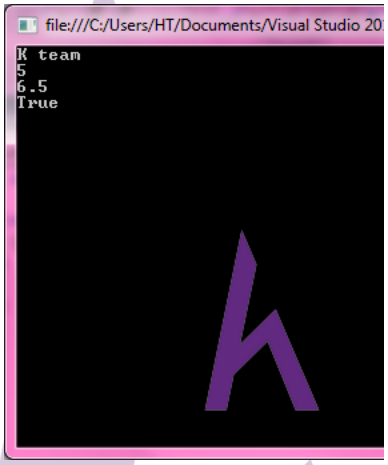
  Cách này khá dài dòng so với 2 cách trên và cũng rất ít người sử dụng. Hầu hết khi xuống dòng ta sử dụng Console.WriteLine() hoặc "\n"

Ví dụ:

:

```
static void Main(string[] args)
{
    Console.Write("K team \n"); // Sử dụng ký tự đặc biệt để xuống dòng
    Console.WriteLine(5); // Sử dụng lệnh in ra màn hình có xuống dòng
    Console.Write(6.5f); // In ra giá trị nhưng không xuống dòng
    Console.Write(Environment.NewLine); // sử dụng lệnh xuống dòng
    Console.Write(true);

    Console.ReadLine();
}
```

Kết quả khi chạy chương trình là:



Như vậy chúng ta đã tìm hiểu qua 2 lệnh xuất dữ liệu ra màn hình rồi. Điểm khác biệt cơ bản giữa 2 lệnh là:

- Console.**Write**(<giá trị cần in ra màn hình>): in giá trị ra màn hình nhưng không đưa con trỏ xuống dòng.
- Console.**WriteLine**(<giá trị cần in ra màn hình>): in giá trị ra màn hình và đưa con trỏ xuống dòng.

> **Lưu ý:** Giá trị in ra màn hình có thể được cộng dồn và có thể in ra giá trị của biến (khái niệm và ý nghĩa của biến sẽ được trình bày chi tiết trong bài BIẾN TRONG C#).

Cộng dồn chuỗi in ra màn hình

Thay vì chúng ta viết:

:

```
int a = 5; // khai báo biến kiểu nguyên có tên là a và khởi tạo giá trị là 5.
Console.Write("a = "); // In ra màn hình giá trị "a = ".
Console.Write(a); // In ra giá trị của a là 5

// Kết quả màn hình là: a = 5
```

Thì ta có thể viết gọn lại là Console.Write("a = " + a); vẫn in ra màn hình a = 5.

Như vậy để cho chương trình ngắn gọn, trực quan ta có thể cộng trực tiếp như vậy thay vì viết ra nhiều dòng Console.Write().

In ra giá trị của biến

Cộng dồn là một cách in ra giá trị của biến.

Ngoài ra ta cũng có thể chỉ định vị trí in ra giá trị của biến trong chuỗi bằng cú pháp {<số đếm>}.

Ví dụ:

:

```
int a = 5; // khai báo biến kiểu nguyên có tên là a và khởi tạo giá trị là 5.
Console.Write("a = {0}", a); // In ra màn hình giá trị "a = 5".
```

Cú pháp chung:

> Console.**Write**("{0} {1} {2} {...}", <giá trị 0>, <giá trị 1>, <giá trị 2>, <giá trị n>);

Trong đó:

- <giá trị 0> sẽ được điền tương ứng vào vị trí số 0 tương tự như vậy cho các giá trị còn lại.

Với 2 cách trên ta đã có thể thao tác biến hóa làm cho code trở nên gọn gàng, trực quan hơn rồi.

Console.Read();

Cú pháp:

> Console.**Read**();

Ý nghĩa:

Đọc **1 ký tự** từ bàn phím và trả về **kiểu số nguyên** (sẽ được trình bày chi tiết ở bài KIỂU DỮ LIỆU TRONG C#) là mã ASCII (American Standard Code for Information Interchange - Chuẩn mã trao đổi thông tin Hoa Kì,là bộ kí tự và bộ mã kí tự dựa trên bảng chữ cái La Tinh được dùng trong tiếng Anh hiện đại và các ngôn ngữ Tây Âu khác) của ký tự đó.

> **Chú ý:** lệnh này không đọc được các phím chức năng như **Ctrl**, **Shift**, **Alt**, **Caps Lock**, **Tab**, . . .

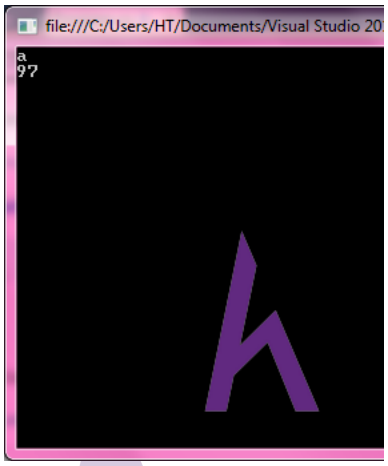
Ví dụ:

Để biết chắc rằng máy tính có đọc được ký tự mình vừa nhấn hay không thì chúng ta sẽ thử viết chương trình đọc 1 ký tự và in ký tự đó ra màn hình như sau:

:

```
static void Main(string[] args)
{
     Console.WriteLine(Console.Read()); // đọc 1 ký tự từ bàn phím bằng lệnh Console.Read() sau đó in ra ký tự vừa đọc được.
     Console.ReadKey(); // lệnh này dùng với mục đích dừng màn hình để xem kết quả.
}
```

Kết quả khi chạy chương trình ta được:



Như đã giải thích lệnh Console.**Read**() dùng để đọc 1 ký tự và trả về 1 số nguyên là mã ASCII của ký tự đó nên khi ta nhập a thì màn hình sẽ in ra số 97 (là mã ASCII của ký tự a).

Console.ReadLine();

Cú pháp:

> Console.**ReadLine**();

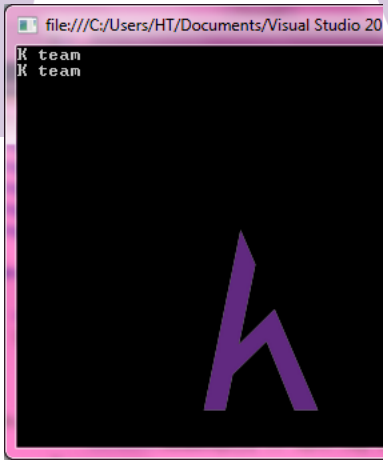
Ý nghĩa: Đọc dữ liệu từ bàn phím cho đến khi gặp ký tự xuống dòng thì dừng (Nói cách khác là đọc cho đến khi mình nhấn enter thì dừng) và giá trị đọc được luôn là một chuỗi.

Ví dụ:

:

```
static void Main(string[] args)
{
    Console.WriteLine(Console.ReadLine()); // đọc dữ liệu từ bàn phím cho đến khi gặp ký tự xuống dòng thì dừng. Sau đó in giá trị đã nhập ra màn hình.
    Console.ReadKey(); // lệnh này dùng với mục đích dừng màn hình để xem kết quả.
}
```

Kết quả khi chạy chương trình:



- Màn hình có 2 chữ "**K team**" là vì chữ đầu tiên do người dùng nhập từ bàn phím chữ thứ 2 là máy tính in ra bằng lệnh Console.WriteLine()

Console.ReadKey();

Cú pháp:

```
Console.ReadKey(<tham số kiểu bool>)
```

Ý nghĩa:

- Lệnh này cũng dùng để đọc một ký tự từ bàn phím nhưng trả về kiểu **ConsoleKeyInfo** (là một kiểu dữ liệu có cấu trúc được định nghĩa sẵn để chứa những ký tự của bàn phím bao gồm các phím chức năng).
- **Tham số kiểu bool** bao gồm 2 giá trị: true hoặc false. Nếu truyền vào true thì phím được ấn sẽ không hiển thị lên màn hình console mà được đọc ngầm ngược lại thì phím được ấn sẽ hiển thị lên màn hình console (kiểu bool sẽ được trình bày chi tiết ở bài KIỂU DỮ LIỆU TRONG C#). Nếu không truyền tham số vào thì mặc định sẽ là false.

Ứng dụng của lệnh này rất mạnh nhưng trong phạm vi bài học hôm nay chúng ta chỉ tìm hiểu cú pháp và ý nghĩa cơ bản. Trong những bài học sau này sẽ giải thích chi tiết khi gặp lệnh trên. Vì thế phần ví dụ mình chỉ trình bày minh hoạ cho việc truyền tham số cho các bạn hiểu trước.

:

```
static void Main(string[] args)
{
	Console.WriteLine("K team");

	Console.ReadKey(); // không truyền tham số vào thì mặc định là false.
	Console.ReadKey(false); // hiển thị phím ấn lên màn hình.
	Console.ReadKey(true); // Không hiển thị phím ấn lên màn hình.
}
```

Các bạn chạy chương trình bằng cách ấn **Ctrl + F5**. Kết quả khi chạy ta được:

Khi chạy chương ta thử ấn 3 ký tự bất kỳ, ở đây mình ấn 3 lần phím '**K**' nhưng trên màn hình chỉ có 2 chữ k được hiển thị.

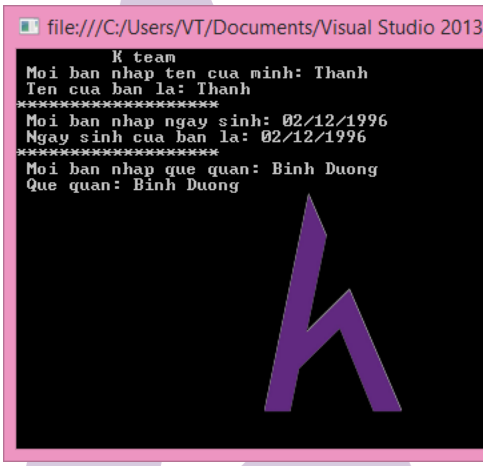
Ví dụ chương trình nhập xuất cơ bản trong C#

Để hiểu kỹ hơn về các lệnh nhập xuất, chúng ta cùng xem thử ví dụ sau:

:

```
static void Main(string[] args)
{
     Console.WriteLine("         K team"); // In chữ "K team" sau đó xuống dòng.
     Console.Write(" Moi ban nhap ten cua minh: "); // In không xuống dòng.
     Console.WriteLine("Ten cua ban la: " + Console.ReadLine()); // Quy tắc chung trong thực hiện lệnh là lệnh bên trong thực hiện trước rồi đến lệnh bên ngoài chứa nó. Do đó chạy đến đây chương trình sẽ thực hiện lệnh Console.ReadLine() sau đó thực hiện cộng chuỗi và cuối cùng in chuỗi ra màn hình.
     Console.Write(" Moi ban nhap ngay sinh: ");
     Console.WriteLine(" Ngay sinh cua ban la: " + Console.ReadLine()); // Tương tự như trên
     Console.Write(" Moi ban nhap que quan: ");
     Console.WriteLine(" Que quan: " + Console.ReadLine()); // Tương tự như trên.

     Console.ReadKey();
}
```

Kết quả khi chạy chương trình trên là:



Bài tập củng cố

1. Viết chương trình cho phép người dùng nhập tên của mình và hiển thị câu: HowKteam.com xin chào <Tên vừa nhập>.
2. Viết chương trình nhập vào các thông tin:
    - Tên
    - Tuổi
    - Địa chỉ

    Xuất ra màn hình theo định dạng: Bạn tên <Tên>, <Tuổi> tuổi, ở <Địa chỉ>

Giải và đăng bài giải của bạn trong phần bình luận bên dưới để mọi người cùng tham khảo nhé!

Kết luận

Nội dung bài học giúp các bạn nắm được:

- Các lệnh nhập xuất trong Console của C#.
- Hiểu được cú pháp và ý nghĩa của từng lệnh.
- Viết chương trình thực hiện các lệnh nhập xuất và chạy thử.

Bài học sau chúng ta sẽ tìm hiểu khái niệm và chi tiết về **BIẾN 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**"