

# Bài: Struct trong lập trình C# căn bản

Xem bài học trên website để ủng hộ Kteam: [Struct trong lập trình C# căn bản](#)

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 [STRING TRONG C#](#). Hôm nay chúng ta sẽ cùng tìm hiểu về **Struct trong C#**.

## Nội dung

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

- [BIẾN](#) và [KIỂU DỮ LIỆU](#) trong C#
- [TOÁN TỬ TRONG C#](#)
- [CÂU ĐIỀU KIỆN TRONG C#](#)
- [CẤU TRÚC CƠ BẢN CỦA VÒNG LẶP TRONG C#](#)
- [CẤU TRÚC CƠ BẢN CỦA HÀM TRONG C#](#)
- [MẢNG 1 CHIỀU TRONG C#](#)

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

- Struct là gì? Đặc điểm của struct.
- Khai báo và sử dụng struct.

## Struct là gì? Đặc điểm của Struct

**Struct** là một kiểu dữ liệu có cấu trúc, được kết hợp từ các kiểu dữ liệu nguyên thủy do người lập trình định nghĩa để thuận tiện trong việc quản lý dữ liệu và lập trình.

Xét bài toán sau:

Ta cần lưu trữ thông tin của 10 sinh viên với mỗi sinh viên gồm có các thông tin như

- Mã số.
- Họ tên.
- Nơi sinh.
- CMND.

Khi đó, để lưu thông tin của 1 sinh viên ta cần 4 biến chứa 4 thông tin trên. Nếu muốn lưu thông tin 10 sinh viên thì cần 40 biến. Chắc không quá nhiều, nhưng nếu muốn lưu thông tin của 1000, 10000 sinh viên thì sao?

- Số lượng biến lúc này rất nhiều khiến cho code dài dòng khó thao tác, khó kiểm soát.

Từ đó người ta mới đưa ra khái niệm kiểu dữ liệu có cấu trúc để giải quyết vấn đề trên.

Ý tưởng là đóng gói các thông tin đó vào 1 đối tượng duy nhất. Như vậy thay vì phải khai báo 40 biến thì ta chỉ cần khai báo 1 mảng 10 phần tử mà mỗi phần tử có kiểu dữ liệu ta đã định nghĩa.

## Đặc điểm của struct:

- Là một kiểu dữ liệu tham trị (**kiểu dữ liệu tham trị** đã được trình bày trong bài [KIỂU DỮ LIỆU](#) )
- Dùng để đóng gói các trường dữ liệu khác nhau nhưng có liên quan đến nhau.
- Bên trong **struct** ngoài các biến có kiểu dữ liệu cơ bản còn có các phương thức, các struct khác.
- Muốn sử dụng phải khởi tạo cấp phát vùng nhớ cho đối tượng thông qua toán tử **new**.
- **Struct** không được phép kế thừa (khái niệm về kế thừa sẽ trình bày trong bài [KẾ THỪA TRONG C#](#) ).

## Khai báo và sử dụng struct

### Khai báo

#### Cú pháp:

```
struct <tên struct>

{

    public <danh sách các biến>;

}
```

Trong đó:

- **<tên struct>** là tên kiểu dữ liệu do mình tự đặt và tuân thủ theo quy tắc đặt tên (đã trình bày trong bài [BIẾN TRONG C#](#)).
- **<danh sách các biến>** là danh sách các biến thành phần được khai báo như khai báo biến bình thường (đã trình bày trong bài [BIẾN TRONG C#](#)).
- Từ khoá **public** là từ khoá chỉ định phạm vi truy cập (sẽ trình bày trong bài [TÍNH ĐỒNG GÓI](#)). Trong ngữ cảnh hiện tại thì có thể hiểu từ khoá này giúp cho người khác có thể truy xuất được để sử dụng.

#### Ví dụ:

**C#:**

```
struct SinhVien
{
    public int MaSo;
    public string HoTen;
    public double DiemToan;
    public double DiemLy;
    public double DiemVan;
}
```

- Với khai báo này ta đã có 1 kiểu dữ liệu mới tên là **SinhVien**. Và có thể khai báo biến, sử dụng nó như sử dụng các kiểu dữ liệu khác.
- Nếu như kiểu **int** có thể chứa số nguyên, kiểu **double** có thể chứa số thực thì kiểu **SinhVien** vừa khai báo có thể chứa 5 trường thông tin con là **MaSo, HoTen, DiemToan, DiemLy, DiemVan**.

**Lưu ý:** bên trong vẫn còn 2 khai báo chưa được nhắc đến đó là:

- Constructor (hàm khởi tạo).
- Các phương thức mà mình muốn cung cấp để hỗ trợ người dùng khi thao tác với dữ liệu bên trong **struct**.

Hai phần này sẽ được trình bày trong bài [CLASS TRONG C#](#). Còn trong bài học này ta chỉ tìm hiểu cú pháp cơ bản của struct thôi.

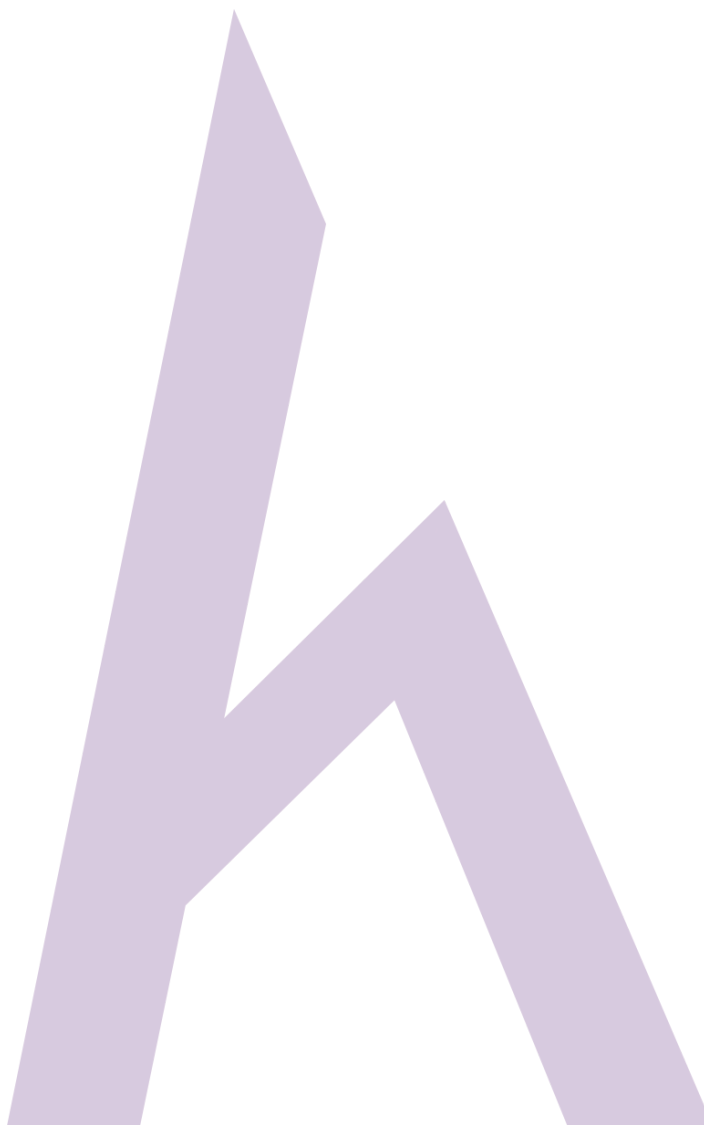
### Sử dụng

Ta có thể truy xuất đến từng thành phần dữ liệu của struct thông qua toán tử "." Kèm theo tên thành phần muốn truy xuất.

**Xét bài toán sau:** Viết chương trình lưu trữ thông tin của sinh viên bao gồm: mã số, họ tên, điểm toán, điểm lý, điểm văn. Thực hiện nhập thông tin cho 1 sinh viên và tính điểm trung bình theo công thức  $(\text{toán} + \text{lý} + \text{văn})/3$ .

Chương trình tham khảo:

**C#:**



```
struct SinhVien
{
    public int MaSo;
    public string HoTen;
    public double DiemToan;
    public double DiemLy;
    public double DiemVan;
}

static void NhapThongTinSinhVien(out SinhVien SV)
{
    Console.Write(" Ma so: ");
    SV.MaSo = int.Parse(Console.ReadLine());
    Console.Write(" Ho ten: ");
    SV.HoTen = Console.ReadLine();
    Console.Write(" Diem toan: ");
    SV.DiemToan = Double.Parse(Console.ReadLine());
    Console.Write(" Diem ly: ");
    SV.DiemLy = Double.Parse(Console.ReadLine());
    Console.Write(" Diem van: ");
    SV.DiemVan = Double.Parse(Console.ReadLine());
}

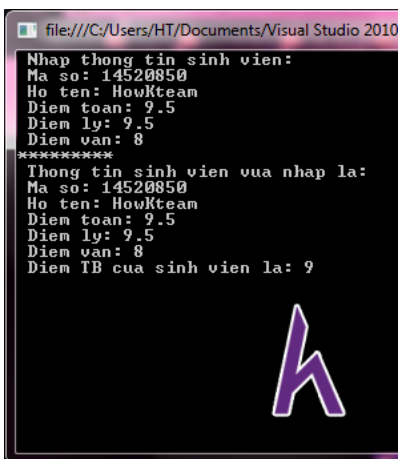
static void XuatThongTinSinhVien(SinhVien SV)
{
    Console.WriteLine(" Ma so: " + SV.MaSo);
    Console.WriteLine(" Ho ten: " + SV.HoTen);
    Console.WriteLine(" Diem toan: " + SV.DiemToan);
    Console.WriteLine(" Diem ly: " + SV.DiemLy);
    Console.WriteLine(" Diem van: " + SV.DiemVan);
}

static double DiemTBSinhVien(SinhVien SV)
{
    return (SV.DiemToan + SV.DiemLy + SV.DiemVan) / 3;
}

static void Main(string[] args)
{
    /*
    * Khai báo 1 kiểu dữ liệu SinhVien với các trường thông tin như đề bài.
    * Khai báo và khởi tạo 1 đối tượng SV1 kiểu SinhVien.
    */
    SinhVien SV1 = new SinhVien();
    Console.WriteLine(" Nhập thông tin sinh viên: ");
    /*
    * Đây là hàm hỗ trợ nhập thông tin sinh viên.
    * Sử dụng từ khoá out để có thể cập nhật giá trị nhập được ra biến SV1 bên ngoài
    * khi kết thúc gọi hàm (có thể xem lại bài Hàm trong C#).
    */
    NhapThongTinSinhVien(out SV1);
    Console.WriteLine("*****");
    Console.WriteLine(" Thông tin sinh viên vừa nhập là: ");
    XuatThongTinSinhVien(SV1);
    Console.WriteLine(" Diem TB của sinh viên là: " + DiemTBSinhVien(SV1));

    Console.ReadLine();
}
```

Kết quả khi chạy chương trình trên là:



```
file:///C:/Users/HT/Documents/Visual Studio 2010/
Nhap thong tin sinh vien:
Ma so: 14520850
Ho ten: HowKteam
Diem toan: 9.5
Diem ly: 9.5
Diem van: 8
*****
Thong tin sinh vien vua nhap la:
Ma so: 14520850
Ho ten: HowKteam
Diem toan: 9.5
Diem ly: 9.5
Diem van: 8
Diem TB cua sinh vien la: 9
```

Trong chương trình trên ta có thể thấy:

- Kiểu dữ liệu **SinhVien** có thể dùng làm kiểu dữ liệu cho biến, parameter cho phương thức. Ngoài ra còn có thể làm kiểu trả về cho phương thức (các bạn có thể khám phá thử, có thể tham khảo bài [HÀM TRONG C#](#)).
- Các thành phần dữ liệu bên trong được truy xuất thông qua dấu ".".
- Vì **struct** là kiểu tham trị nên khi truyền vào các phương thức thì giá trị của nó sau khi kết thúc phương thức sẽ không thay đổi. Do đó cần sử dụng từ khóa **out** để có thể cập nhật giá trị thay đổi khi kết thúc phương thức.

## Kết luận

Nội dung bài này giúp các bạn nắm được:

- Struct là gì? Đặc điểm của struct.
- Khai báo và sử dụng struct.

Bài sau chúng ta sẽ tìm hiểu về [ENUM 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**".