Bài: Cấu trúc rẽ nhánh Switch case trong C#

Xem bài học trên website để ủng hộ Kteam: Cấu trúc rẽ nhánh Switch case 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

Cấu trúc rẽ nhánh có 2 loại, ở bài CẤU TRÚC RẼ NHÁNH IF - ELSE TRONG C# chúng ta đã tìm hiểu loại đầu tiên. Và hôm nay chúng ta sẽ tìm hiểu loại còn lại – **Cấu trúc rẽ nhánh Switch case trong C#**.

Nội dung

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

- Cấu trúc lệnh của C# viết trên nền Console Application.
- Cấu trúc nhập xuất của C# trên nền Console Application.
- BIẾN, KIỂU DỮ LIỆU, TOÁN TỬ trong C#.
- ÉP KIỂU TRONG C#.

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

- Cấu trúc switch case dạng thiếu và dạng đủ.
- Ví dụ chương trình sử dụng cấu trúc switch case.

Cấu trúc switch case dạng thiếu và dạng đủ

Dạng thiếu

Cú pháp:

```
switch (<biểu thức>)
{
case <giá trị thứ 1>: <câu lệnh thứ 1>;
                      break;
case <giá trị thứ 2>: <câu lệnh thứ 2>;
                      break;
. . .
case <giá trị thứ n>: <câu lệnh thứ n>;
                      break;
}
```

Trong đó:

- switch, case là từ khóa bắt buộc.
- break là một lệnh nhảy
  - Ý nghĩa của nó là thoát ra khỏi cấu trúc, vòng lặp chứa nó (khái niệm về vòng lặp sẽ được trình bày ở bài CẤU TRÚC LẶP GOTO TRONG C#)
  - Ngoài break ra vẫn còn lệnh nhảy khác như goto nhưng ít được sử dụng (chi tiết về lệnh goto sẽ được trình bày trong bài CẤU TRÚC LẶP GOTO TRONG C#).
  - Vì trong cấu trúc switch. . . case chủ yếu chỉ sử dụng lệnh break nên mình cố tình để lệnh break vào trong cú pháp thay vì ghi chung chung là lệnh nhảy.
- <biểu thức> phải là biểu thức trả về kết quả kiểu:
  - Số nguyên (int, long, byte, . . .)
  - Ký tự hoặc chuỗi (char, string)
  - Kiểu liệt kê (enum, sẽ được trình bày trong bài ENUM TRONG LẬP TRÌNH C#)
- <giá trị thứ i> với i = 1..n là giá trị muốn so sánh với giá trị của <biểu thức>.
- <câu lệnh thứ i> với i = 1..n là câu lệnh muốn thực hiện khi <giá trị thứ i> tương ứng bằng với giá trị của <biểu thức>.

Ý nghĩa: Duyệt lần lượt từ trên xuống dưới và kiểm tra xem giá trị của <biểu thức> có bằng với <giá trị thứ i> đang xét hay không. Nếu bằng thì thực hiện <câu lệnh thứ i> tương ứng.

Lưu ý:

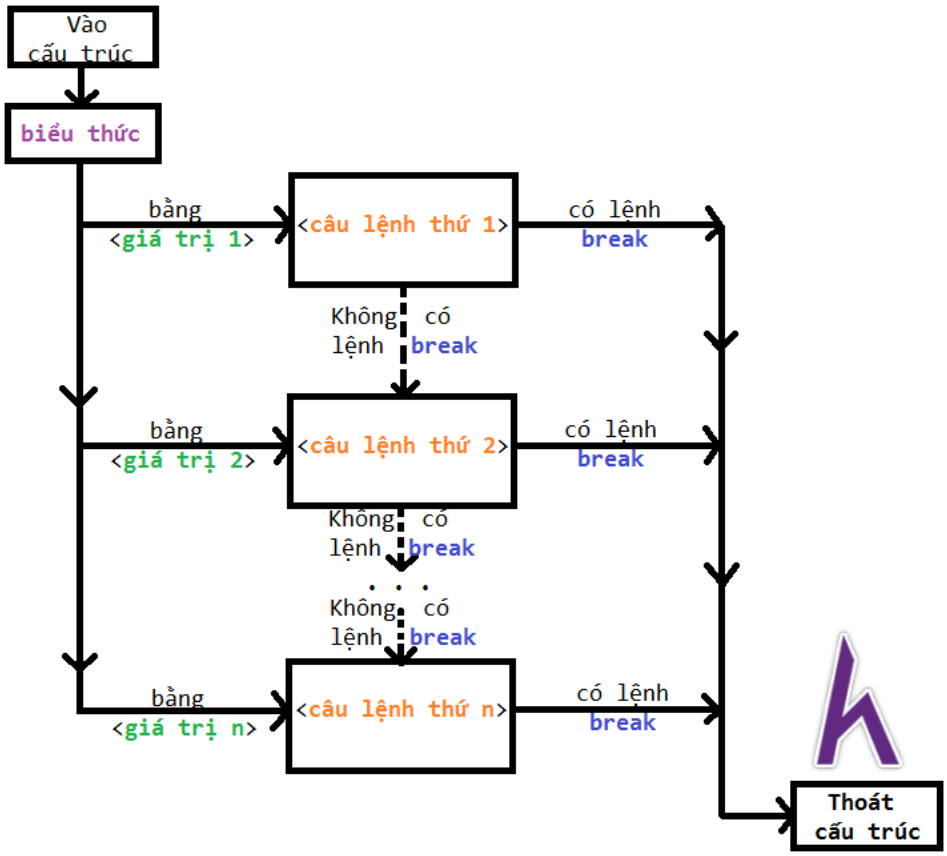
> - <giá trị thứ i> phải có kiểu dữ liệu giống với kiểu dữ liệu của giá trị của biểu thức.
> - <câu lệnh thứ i> có thể gồm nhiều câu lệnh và không nhất thiết phải đặt trong cặp dấu ngoặc nhọn { } nhưng tốt hơn bạn nên đặt trong cặp dấu { } để code được rõ ràng hơn.
> - Nếu case đang xét không rỗng (có lệnh để thực hiện) thì bắt buộc phải có lệnh nhảy (cụ thể là lệnh break) sau đó.

- **Ví dụ:**

C#:

```
int k = 8;

            switch (k)
            {
                case 3:
                    Console.WriteLine("HowKteam");
                    break; // Vì case này có lệnh thực hiện nên phải có lệnh break
                case 9: // case này rỗng (không có lệnh thực hiện) nên không cần lệnh break
                case 10:
                    Console.WriteLine("Free Education");
                    break;
            }
```

Lưu đồ sau sẽ minh họa cho các bạn cách thức hoạt động của cấu trúc switch. . . case dạng thiếu:



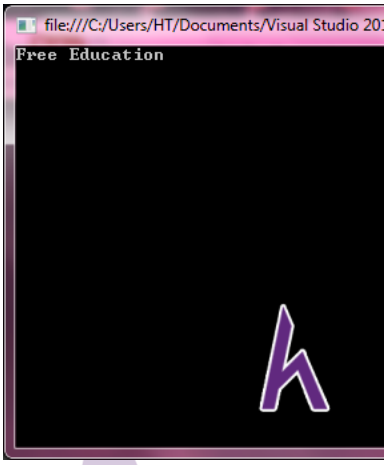
- Chú ý là trường hợp không có lệnh break như trong hình đồng nghĩa với việc case đó rỗng (không có câu lệnh thực hiện).
- Đối với case cuối cùng dù có câu lệnh để thực hiện hay không vẫn phải có lệnh break để thoát khỏi cấu trúc.

Ví dụ:

C#:

```
int k = 10;

            switch (k) // giá trị biểu thức là giá trị của biến k (kiểu số nguyên)
            {
                case 3: // các giá trị so sánh cũng là kiểu số nguyên
                    Console.WriteLine("HowKteam"); // lệnh thực hiện nếu k = 3
                    break; // lệnh thoát ra khỏi cấu trúc
                case 9:
                    Console.WriteLine("Kteam"); // tương tự
                    break;
                case 10:
                    Console.WriteLine("Free Education"); // tương tự
                    break;
            }
```

Kết quả khi chạy chương trình trên là:



Dạng đủ

Cú pháp:

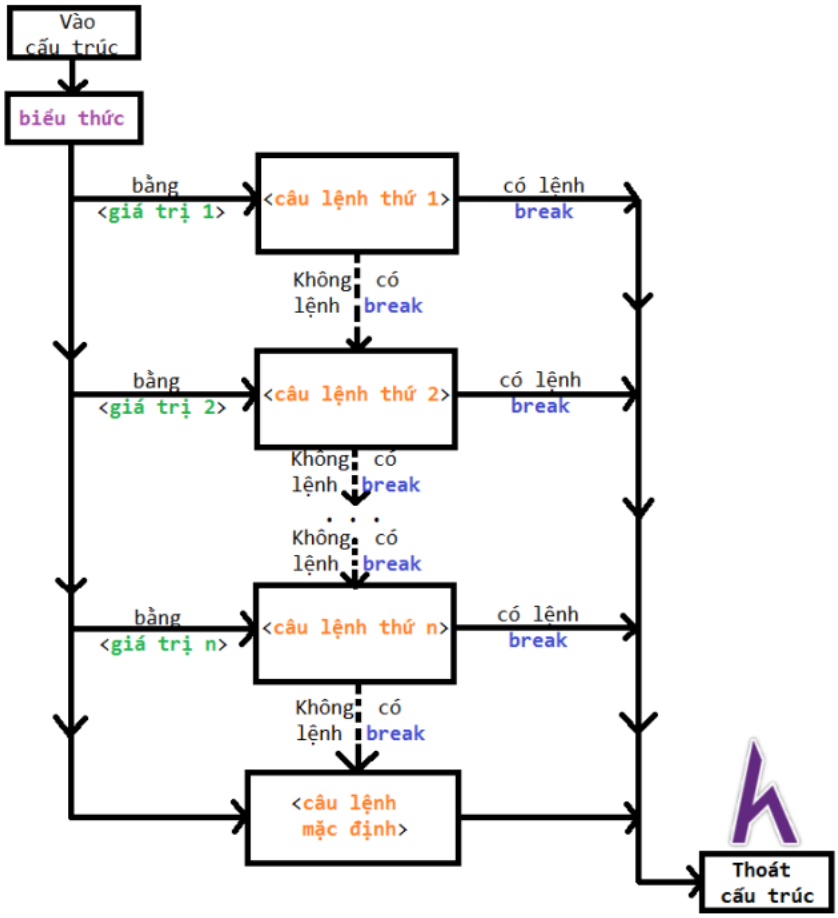
```
switch (<biểu thức>)
{
case <giá trị thứ 1>: <câu lệnh thứ 1>;
                    break;
case <giá trị thứ 2>: <câu lệnh thứ 2>;
                    break;
. . .
case <giá trị thứ n>: <câu lệnh thứ n>;
                    break;
default: <câu lệnh mặc định>;
                break;
}
```

Trong đó:

- switch, case, default là từ khóa bắt buộc.
- <biểu thức> phải là biểu thức trả về kết quả kiểu:
  - Số nguyên (int, long, byte, . . .)
  - Ký tự hoặc chuỗi (char, string)
  - Kiểu liệt kê (enum, sẽ được trình bày trong bài ENUM TRONG LẬP TRÌNH C#)
- <giá trị thứ i> với i = 1..n là giá trị muốn so sánh với giá trị của <biểu thức>.
- <câu lệnh thứ i> với i = 1..n là câu lệnh muốn thực hiện khi <giá trị thứ i> tương ứng bằng với giá trị của <biểu thức>.
- <câu lệnh mặc định> là câu lệnh sẽ được thực hiện nếu giá trị <biểu thức> không bằng với <giá trị thứ i> nào.

Ý nghĩa: Duyệt lần lượt từ trên xuống dưới và kiểm tra xem giá trị của <biểu thức> có bằng với <giá trị thứ i> đang xét hay không. Nếu bằng thì thực hiện <câu lệnh thứ i> tương ứng. Nếu không bằng tất cả các <giá trị thứ i> thì sẽ thực hiện <câu lệnh mặc định>.

Lưu đồ sau sẽ minh họa cách thức hoạt động của cấu trúc switch. . . case. . . default. . . :



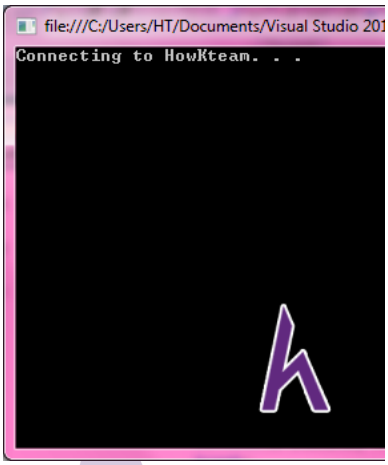
Về cơ bản cách thức hoạt động của 2 cấu trúc switch. . . case dạng đủ và dạng thiếu là như nhau, chỉ khác nhau ở một điểm là dạng đủ có thêm dòng default. . . (tương tự là lệnh else trong bài CẤU TRÚC RẼ NHÁNH IF ELSE) nên các bạn xem lại lưu ý của dạng thiếu để tránh mắc lỗi.

Ví dụ:

C#:

```
int k = 8;

            switch (k)
            {
                case 3:
                    Console.WriteLine("HowKteam");
                    break;
                case 9:
                    Console.WriteLine("Kteam");
                    break;
                case 10:
                    Console.WriteLine("Free Education");
                    break;
                default: // Nếu không thỏa các trường hợp trên sẽ thực hiện lệnh sau đây
                    Console.WriteLine("Connecting to HowKteam. . .");
                    break;
            }
```

Kết quả khi chạy chương trình trên là:



- Vì không tìm thấy case nào có giá trị bằng với giá trị biến k nên sẽ thực hiện câu lệnh trong default. Do đó màn hình in ra "Connecting to HowKteam. . .".

Ví dụ chương trình sử dụng cấu trúc switch case

Ví dụ: Viết chương trình tính năm âm lịch từ năm dương lịch đã nhập.

Thuật toán tính năm âm lịch:

- Năm âm lịch = Can + Chi. Vì thế cần tính được Can và Chi sau đó ghép lại là xong.
- Tính Can bằng cách:
  - Tìm phần dư của phép chia năm dương lịch cho 10.
  - Tra bảng sau để tìm ra Can tương ứng

Năm dương % 10	0	1	2	3	4	5	6	7	8	9
Can	Canh	Tân	Nhâm	Quý	Giáp	Ất	Bính	Đinh	Mậu	Kỷ

- Tìm Chi bằng cách:
  - Tìm phần dư của phép chia năm dương lịch cho 12.
  - Tra bảng sau để tìm ra Chi tương ứng:

ăm dương % 12	0	1	2	3	4	5	6	7	8	9	10	11
Can	Thân	Dậu	Tuất	Hợi	Tý	Sửu	Dần	Mẹo	Thìn	Tỵ	Ngọ	Mùi

- Nối Can và Chi lại để được kết quả.

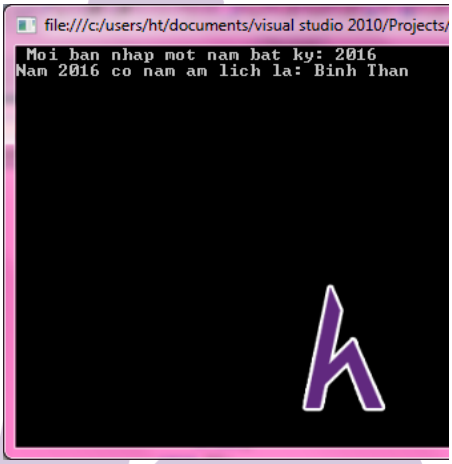
Các bạn tham khảo đoạn chương trình sau:

C#:

```
int Year; // Biến chứa giá trị năm cần tính.
            string Can = "", Chi = ""; // Biến chứa kết quả.

            Console.Write(" Moi ban nhap mot nam bat ky: ");
            Year = Int32.Parse(Console.ReadLine()); // Nhập năm dương lịch và ép kiểu về kiểu số nguyên

            switch (Year % 10) // Tìm Can như thuật toán đã trình bày.
            {
                case 0: // Mỗi case này tương ứng một kết quả cần tra cứu trong bảng tra cứu Can
                    Can = "Canh"; // Giá trị tương ứng với mỗi case
                    break;
                case 1:
                    Can = "Tan";
                    break;
                case 2:
                    Can = "Nham";
                    break;
                case 3:
                    Can = "Quy";
                    break;
                case 4:
                    Can = "Giap";
                    break;
                case 5:
                    Can = "At";
                    break;
                case 6:
                    Can = "Binh";
                    break;
                case 7:
                    Can = "Dinh";
                    break;
                case 8:
                    Can = "Mau";
                    break;
                case 9:
                    Can = "Ky";
                    break;
            }

            switch (Year % 12) // Tìm Chi như thuật toán đã trình bày
            {
                case 0: // Mỗi case này tương ứng một kết quả cần tra cứu trong bảng tra cứu Chi
                    Chi = "Than"; // Giá trị tương ứng với mỗi case
                    break;
                case 1:
                    Chi = "Dau";
                    break;
                case 2:
                    Chi = "Tuat";
                    break;
                case 3:
                    Chi = "Hoi";
                    break;
                case 4:
                    Chi = "Ty";
                    break;
                case 5:
                    Chi = "Suu";
                    break;
                case 6:
                    Chi = "Dan";
                    break;
                case 7:
                    Chi = "Meo";
                    break;
```

```
            case 8:
                Chi = "Thin";
                break;
            case 9:
                Chi = "Ti";
                break;
            case 10:
                Chi = "Ngo";
                break;
            case 11:
                Chi = "Mui";
                break;
        }

        Console.WriteLine("Nam {0} co nam am lich la: {1} {2}", Year, Can, Chi); // Nối Can và Chi lại để được năm âm lịch

        Console.ReadLine();
```

Kết quả khi chạy chương trình trên là:



- Ở ví dụ trên mình đã bỏ qua việc kiểm tra dữ liệu nhập vào có đúng hay không nên các bạn có thể áp dụng kiến thức đã học để thực hiện (tham khảo ví dụ trong CẤU TRÚC RẼ NHÁNH IF ELSE TRONG C#).

Bài tập tham khảo

Tương tự phần bài tập của bài CẤU TRÚC RẼ NHÁNH IF ELSE TRONG C# nhưng sử dụng cấu trúc Switch case.

Hãy khoe thành quả của bạn vào fanpage hoặc phần bình luận nhé.

Kết luận

Nội dung bài này giúp các bạn nắm được:

- Cấu trúc switch case dạng thiếu và dạng đủ.
- Viết chương trình sử dụng cấu trúc switch case.

Bài học sau chúng ta sẽ cùng tìm hiểu một khái niệm tiếp theo đó là **KIỂU DỮ LIỆU OBJECT 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**".