Bài 4 (2): Câu lệnh điều kiện switch trong C

Lệnh `switch-case` cũng gần tương tự như `if-else` mà chúng ta đã được tìm hiểu ở bài trước. Nghĩa là nó có nhiều điều kiện, chương trình chúng ta duyệt từng điều kiện từ trên xuống dưới, nếu thỏa mãn điều kiện nào thì đoạn code bên trong điều kiện đó sẽ được thực thi.

- `switch-case` là một cấu trúc điều khiển và rẽ nhánh hoàn toàn có thể thay thế được `if-else`.
- Việc sử dụng `switch-case` sẽ giúp code của chúng ta dễ viết và dễ đọc hơn.
- Sử dụng `switch-case` sẽ cho hiệu năng tốt hơn so với sử dụng `if-else`.

Dưới đây là cú pháp của lệnh `switch-case`

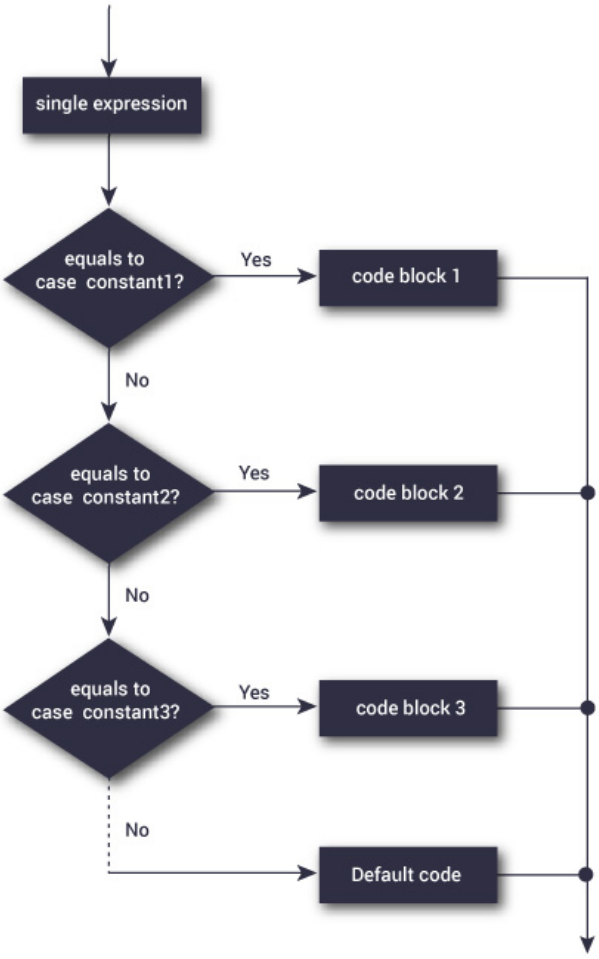
```
switch (expression) {
    case constant1:
        // statements 1
        break;
    case constant2:
        // statements 2
        break;
        .
        .
        .
    default:
        // default statements
}
```

- `switch` sẽ so sánh giá trị của `expression` với mỗi `case (trường hợp)` bên trong nó.
- Từ khóa `break` được sử dụng để kết thúc một `case` trong câu lệnh `switch`.
- Case `default` sẽ được thực hiện nếu không có `case` nào khớp giá trị với `expression`. Trong khối lệnh `switch` có thể có một case `default` hoặc không có.
- Nếu có một case nào đó khớp giá trị với `expression`, các khối lệnh tương ứng của case đó sẽ được thực hiện cho tới khi gặp từ khóa `break`.

Lưu ý:

- Các giá trị của mỗi case phải cùng kiểu dữ liệu với giá trị của `expression`.
- `expression` phải bắt buộc là giá trị hằng, có thể là biểu thức nhưng kết quả cần là hằng số.
- Giá trị của các case là một hằng số và các giá trị của các case phải khác nhau.
- Số lượng các case là không giới hạn nhưng chỉ có thể có duy nhất một `default`.
- Từ khóa `break` có thể sử dụng hoặc không. Nếu không được sử dụng thì chương trình sẽ không kết thúc khi đã thực hiện hết khối lệnh của case đó. Thay vào đó, nó sẽ thực hiện tiếp các khối lệnh của case tiếp theo cho đến khi gặp từ khóa `break` hoặc dấu `}` cuối cùng của cấu trúc `switch-case`.
- Cho phép `switch-case` lồng nhau, tuy nhiên không khuyến khích vì nó làm cho chương trình chúng ta phức tạp và khó đọc hơn thôi.

Dưới đây là sơ đồ khối mô tả hoạt động của lệnh `switch-case`



Ví dụ minh họa:

```
#include <stdio.h>
int main() {
    int day;
    printf("Enter the day: ");
    scanf("%d", &day);
    switch (day) {
        case 2:
            printf("Monday");
            break;
        case 3:
            printf("Tuesday");
            break;
        case 4:
            printf("Wednesday");
            break;
        case 5:
            printf("Thursday");
```

```
            break;
        case 6:
            printf("Friday");
            break;
        case 7:
            printf("Saturday");
            break;
        case 8:
            printf("Sunday");
            break;
        default:
            printf("Only enter 2 -> 8.");
    }
}
```

Kết quả

Enter the day: 6 Friday

Nếu nhập một giá trị không có trong các case, case `default` sẽ được chạy:

Kết quả

Enter the day: 12 Only enter 2 -> 8.

Cùng xem thêm một ví dụ khác sử dụng kiểu `char`.

```
#include <stdio.h>
int main () {
    char grade;
    printf("Enter grade: ");
    scanf("%c", &grade);
    switch(grade) {
        case 'A':
            printf("Excellent!\n");
            break;
        case 'B':
        case 'C':
            printf("Well done.\n");
            break;
        case 'D':
            printf("You passed.\n");
            break;
        case 'F':
            printf("Better try again.\n");
            break;
        default:
            printf("Invalid grade.\n");
    }
```

```
    printf("Your grade is %c\n", grade);
}
```

Kết quả

Enter grade: B Well done. Your grade is B

Ở ví dụ này case `B` chúng ta không dùng `break;` - vì vậy khi chọn `B` (không có `break;` chặn lại) nó sẽ chạy xuống thực hiện code trong case `C` và in ra `Well done.`. Có nghĩa là khi chọn `B` hay `C` đều cho ra một kết quả giống nhau.