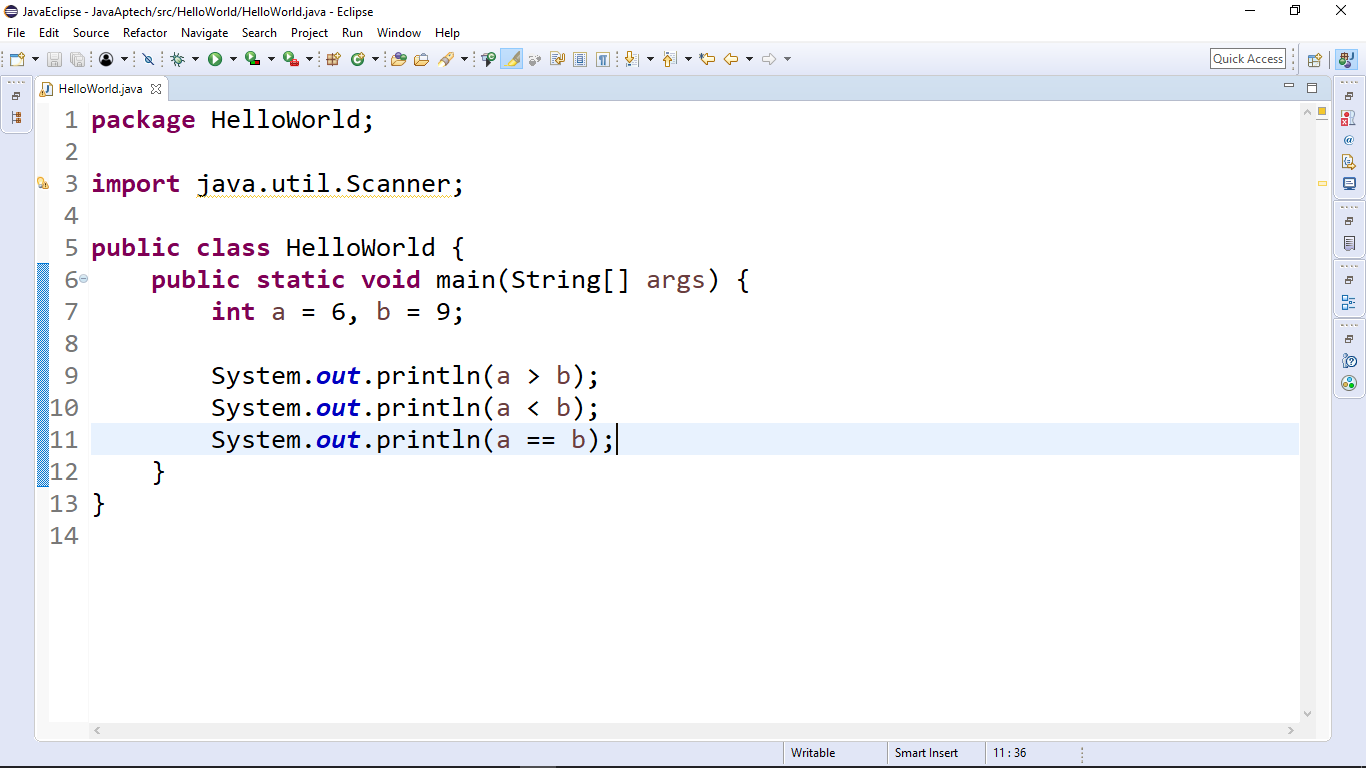
**LỘ TRÌNH ĐÀO TẠO CHƯƠNG 4 PHẦN 1**

1. Giới thiệu

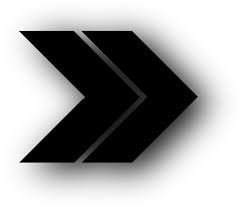
* Câu lệnh điều kiện & câu lệnh rẽ nhánh là 2 câu lệnh thường xuyên gặp trong bất kỳ chương  
  trình nào (dù nhỏ hay lớn).
* Câu lệnh điều kiện: Chương trình sẽ đi thực hiện công việc nếu như có điều kiện gì đó xảy ra ở  
  trước đó (lưu ý là điều kiện phải xảy ra rồi sau đó mới thực hiện công việc).
* Câu lệnh rẽ nhánh: Có chức năng về cơ bản thì giống như câu lệnh điều kiện.
* Thông qua bài học này, các bạn sẽ nắm được những điều sau:
  + Cách sử dụng 2 loại câu lệnh trên.
  + So sánh ưu & nhược điểm của 2 loại câu lệnh trên.
  + Trong trường hợp nào thì nên áp dụng câu lệnh nào ?

1. Câu lệnh điều khiển
   1. So sánh cơ bản:

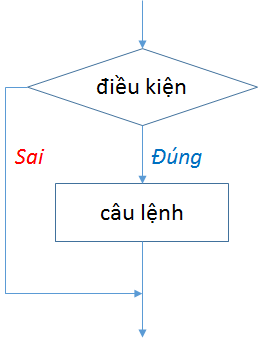


* 1. Câu lệnh điều kiện có trong cuộc sống

Ở ngoài đời ta cũng đã có những câu điều kiện như sau:  
Vd:  
Câu lệnh thiếu:  
- Nếu trời mưa thì tôi nghỉ học.  
- Nếu Ngọc Trinh hạ giá thì tôi sẽ đi … :3  
Câu lệnh đủ:  
- Nếu trời mưa thì tôi nghỉ học, còn không thì tôi đi học.  
- Nếu Ngọc Trinh hạ giá thì tôi sẽ đi, còn không thì tôi không đi.

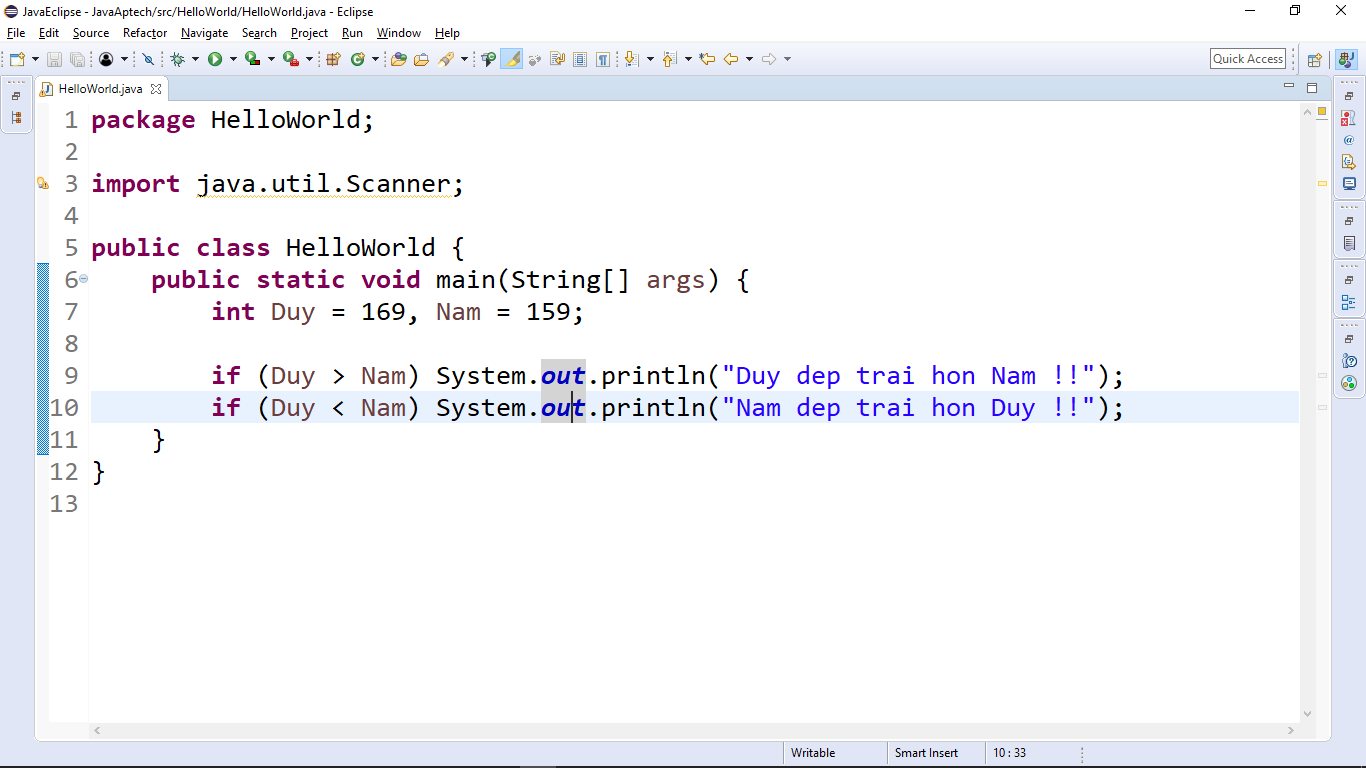
 Hành động chỉ xảy ra khi trước đó có điều kiện gì đó xảy ra.

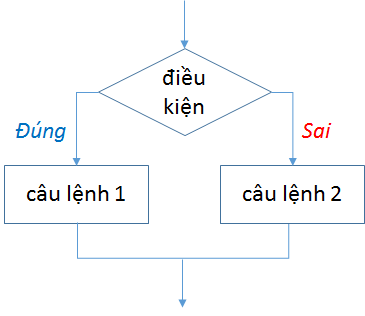
* 1. Câu lệnh điều kiện if/else
     1. Cậu lệnh điều kiện thiếu

- Nhìn vào hình vẽ ta hiểu quá trình như sau: Chương trình xét  
điều kiện xảy ra, nếu điều kiện đó là đúng thì sẽ đi thực hiện  
câu lệnh được cài đặt ở dưới. Nếu điều kiện đó là sai thì sẽ bỏ  
qua không thực hiện câu lệnh, chương trình sẽ tiếp tục chạy  
xuống dưới.  
- Điều kiện luận lý: Cho kết quả đúng hoặc sai. Đúng sẽ là true  
hoặc hiểu là số khác 0. Sai sẽ là false hoặc hiểu là số 0.  
- Câu lệnh: Có thể là “câu lệnh đơn” (Tức là chỉ có duy nhất 1  
câu lệnh) hoặc là “câu lệnh phức” (Từ 2 câu lệnh trở lên). Nếu  
là câu lệnh phức thì phải đặt bên trong cặp ngoặc nhọn “{}” .  
Nếu là câu lệnh đơn thì không cần thiết.   
if (Điều kiện luận lý)

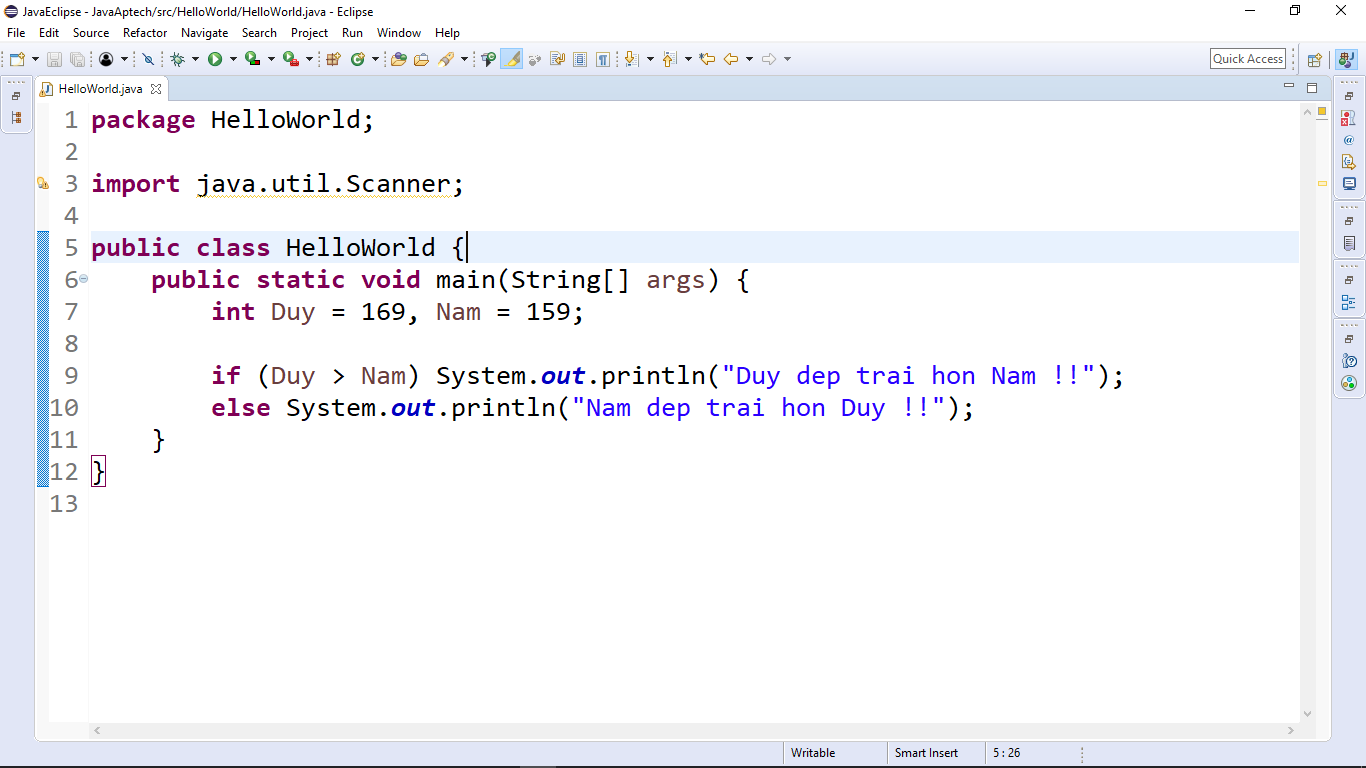
Câu lệnh;

VD:

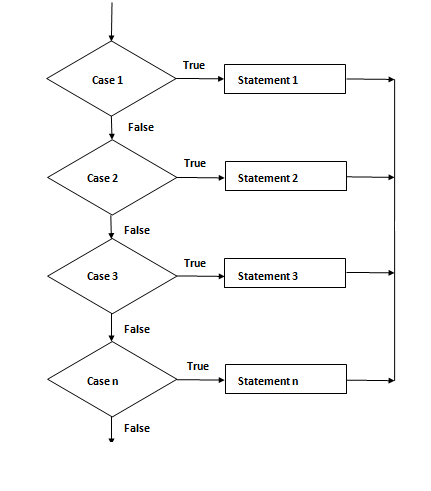


* + 1. Câu lệnh điều kiện đủ  
       - Nhìn vào hình vẽ ta hiểu quá trình như sau: Chương trình xét điều kiện xảy ra, nếu điều kiện đó là đúng thì sẽ đi thực hiện câu lệnh 1. Nếu điều kiện đó là sai thì sẽ đi thực hiện câu lệnh 2, xong rồi chương trình sẽ tiếp tục chạy xuống dưới.  
       - Điều kiện luận lý: Cho kết quả đúng hoặc sai. Đúng sẽ là true hoặc hiểu là số khác 0. Sai sẽ là false hoặc hiểu là số 0.  
       - Câu lệnh: Có thể là “câu lệnh đơn” (Tức là chỉ có duy nhất 1 câu lệnh) hoặc là “câu lệnh phức” (Từ 2 câu lệnh trở lên). Nếu là câu lệnh phức thì phải đặt bên trong cặp ngoặc nhọn “{}” . Nếu là câu lệnh đơn thì không cần thiết.

VD:



* + 1. Có thể bạn chưa biết
       1. Câu lệnh else là phủ định trường hợp ngược lại của if ở trên nó và gần với nó nhất.
       2. Các câu lệnh if else có thể lồng vào nhau với số lượng không giới hạn.
       3. Đằng sau biểu thức điều kiện không được có “;”. Nếu như ta thêm “;” vào đằng sau if thì chương trình sẽ hiểu if đó bị “rỗng”.
       4. Nên biết khi nào sử dụng toàn if, khi nào cần phải có else.
  1. Câu lệnh rẽ nhánh switch
     1. Câu lệnh rẽ nhánh thiếu.

- <Biến hoặc Biểu Thức>:  
Phải có 1 giá trị cụ thể và  
giá trị đó chỉ có thể là số  
nguyên hoặc là ký tự  
hoặc là chuỗi.  
- <case: Giá Trị>: Kiểu dữ  
liệu của giá trị phải trùng  
với kiểu dữ liệu của  
<Biến hoặc Biểu Thức>  
và đặc biệt là các Giá Trị  
của các case không được  
trùng nhau.  
- Câu Lệnh: Có thể là câu  
lệnh đơn hoặc câu lệnh  
phức {}.

**switch(<Biến hoặc biểu thức>) {**

**case <Giá trị 1>:**

**Câu lệnh 1;**

**break;**

**case <Giá trị 1>:**

**Câu lệnh 1;**

**break;**

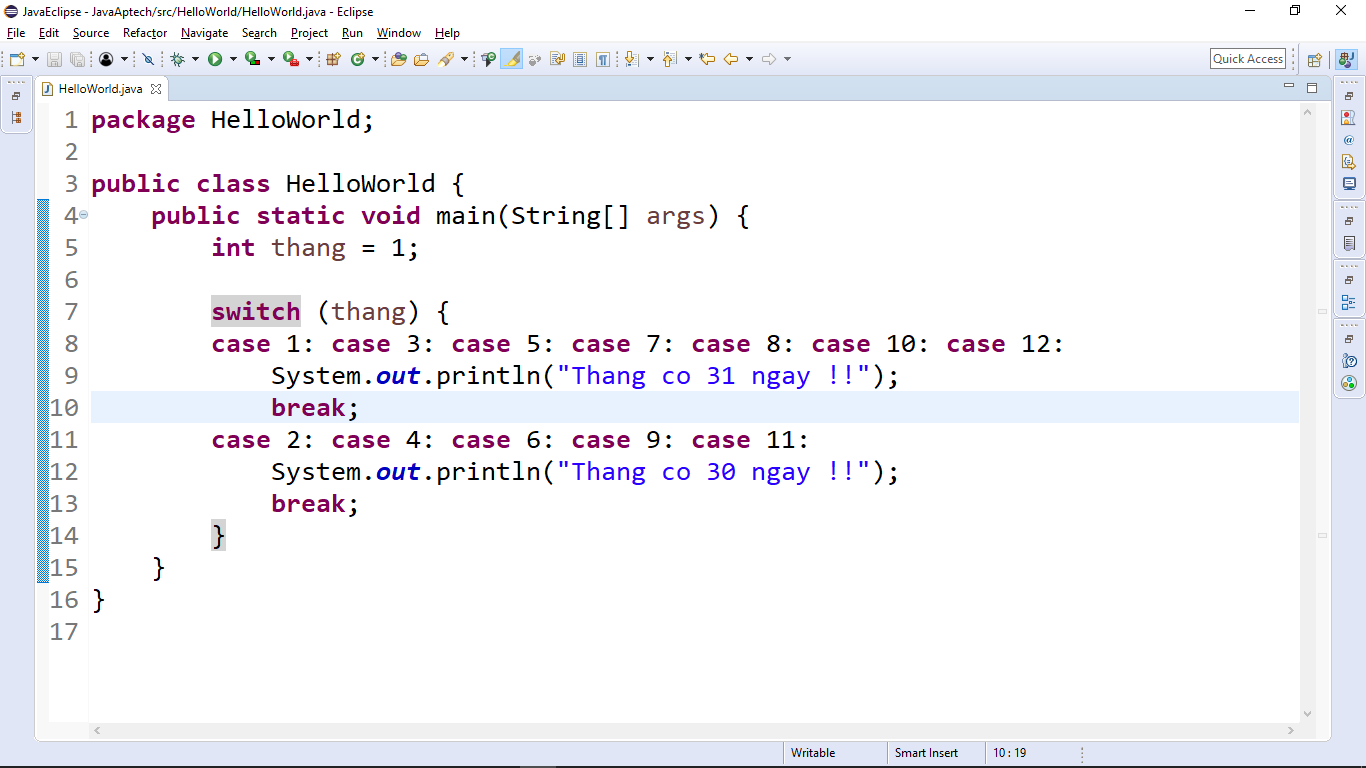
**case <Giá trị 1>:**

**Câu lệnh 1;**

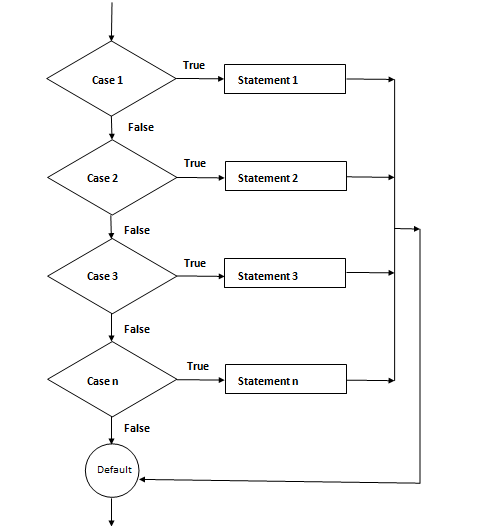
**break;**

**}**

**VD:**



* + 1. Câu lệnh rẽ nhánh đủ.



- <Biến hoặc Biểu Thức>:  
Phải có 1 giá trị cụ thể và  
giá trị đó chỉ có thể là số  
nguyên hoặc là ký tự  
hoặc là chuỗi.  
- <case: Giá Trị>: Kiểu dữ  
liệu của giá trị phải trùng  
với kiểu dữ liệu của  
<Biến hoặc Biểu Thức>  
và đặc biệt là các Giá Trị  
của các case không được  
trùng nhau.  
- Câu Lệnh: Có thể là câu  
lệnh đơn hoặc câu lệnh  
phức {}.

**switch(<Biến hoặc biểu thức>) {**

**case <Giá trị 1>:**

**Câu lệnh 1;**

**break;**

**case <Giá trị 1>:**

**Câu lệnh 1;**

**break;**

**case <Giá trị 1>:**

**Câu lệnh 1;**

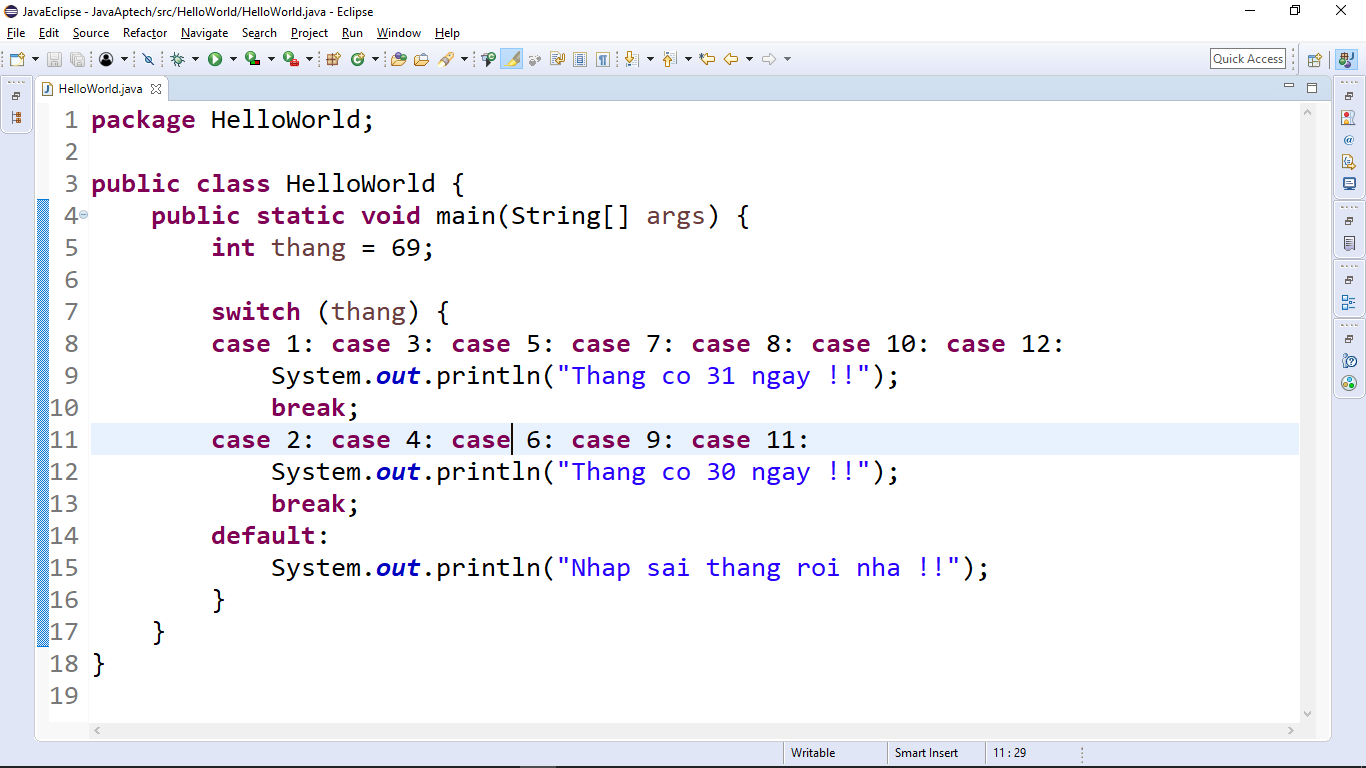
**break;**

**default:**

**Câu lệnh mặc định;**

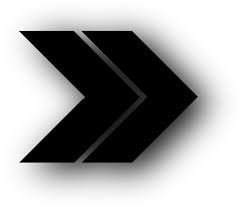
**}**

**VD:**



1. Chia sẽ kinh nghiệm khi lập trình với câu lệnh điều kiện, rẽ nhánh:

Nhận biết khi nào nên sử dụng câu lệnh điều kiện, khi nào nên sử dụng câu lệnh rẽ nhánh ?  
- Khi ta cần so sánh “bằng” giữa 2 giá trị với nhau thì sử dụng cái nào cũng được. Còn các loại so sánh còn lại (>, <, >=, <=, !=) thì chỉ có thể sử dụng câu lệnh điều kiện.  
- Nếu giá trị so sánh “bằng” không thuộc 1 trong 3 kiểu dữ liệu quy định của câu lệnh rẽ nhánh (ký tự, chuỗi, số nguyên) thì không thể sử dụng được.  
- **Vậy nếu như chỉ có thể sử dụng mỗi so sánh bằng giữa 2 giá trị với nhau (kiểu ký tự, số nguyên, chuỗi) vậy thì switch case hơn được if chỗ nào ???**

Khi chương trình được khởi chạy, nếu các giá trị trong các case được xếp “liên tục” nhau (vd: 1, 2, 3, 4, 5 …) thì trình biên dịch sẽ tạo ra 1 bảng ánh xạ để ánh xạ công việc cần làm với giá trị tương ứng, nếu các giá trị này cạnh nhau, trình biên dịch sẽ đưa nó vào 1 mảng. Do tổ chức đặc biệt như thế nên lệnh sau biên dịch ra sẽ nhanh hơn dùng if nhiều (vì if mỗi lần phải biên dịch lại). Như thế này sẽ giúp chúng ta tối ưu hóa được mã nguồn.