

Tema 3

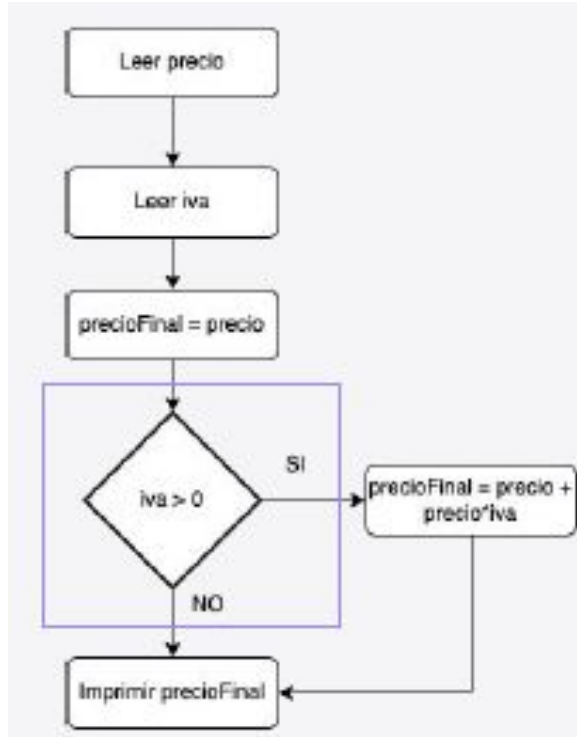
Selecciones

Objetivos

- Implementar el control de selección usando declaraciones if de un solo sentido.
- Implementar el control de selección usando declaraciones if-else bidireccionales.
- Implementar el control de selección usando sentencias anidadas if y multi-way if.
- Evitar errores comunes y trampas en declaraciones if.
- Combinar las condiciones usando operadores lógicos (&&, ||, and!)
- Implementar el control de selección usando instrucciones de conmutación.
- Escribir expresiones utilizando la expresión condicional.
- Aplicar las reglas que rigen la precedencia y asociatividad del operador.
- Aplicar técnicas comunes para depurar errores.

Estructuras de Control Condicionales

- Nos permiten **alterar** el **flujo** de un **programa** en función de la evaluación de una **condición**.



- Por ejemplo:
Para calcular el precio final de un producto hay que sumarle el IVA, que es un % del precio base.
Si el IVA es 0, no hay que sumar nada.

Evaluación condición y operadores

A menudo en un programa necesitamos comparar dos valores, algo como si, i es mayor que j.

Java proporciona seis operadores de comparación (también conocidos como operadores relacionales) que se pueden utilizar para comparar dos valores. El resultado de la comparación es un valor booleano: true o false.

Operador Java	Símbolos Matematicos	Nombre	Ejemplo (radius is 5)	Resultado
<	<	menor que	<code>radius < 0</code>	<code>false</code>
<=	=	menor o igual a	<code>radius <= 0</code>	<code>false</code>
>	>	mayor a	<code>radius > 0</code>	<code>true</code>
>=	=	mayor o igual a	<code>radius >= 0</code>	<code>true</code>
==	=	igual a	<code>radius == 0</code>	<code>false</code>
!=	?	distinto a	<code>radius != 0</code>	<code>true</code>

Sentencia IF-ELSE

- La **sentencia if-else** es la forma en que podemos implementar en Java una estructura condicional simple.

```
if (condicion) {  
    // acciones si se cumple la condición  
} else {  
    // acciones si no se cumple la condición  
}
```

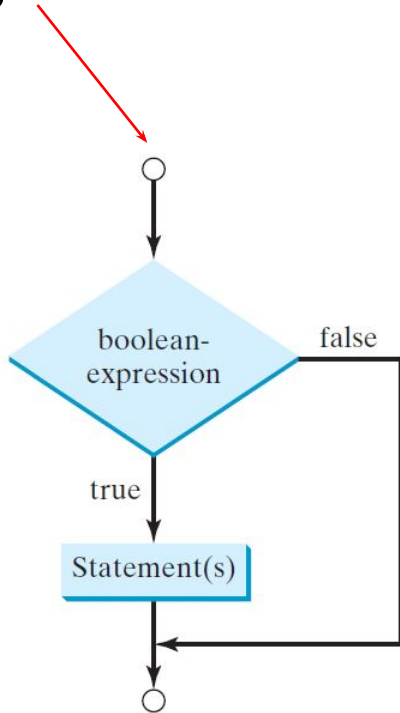
- El bloque else no es obligatorio.
 - Solamente se añade si queremos realizar alguna acción si la condición no se cumple.

Versión más simple de la sentencia if:

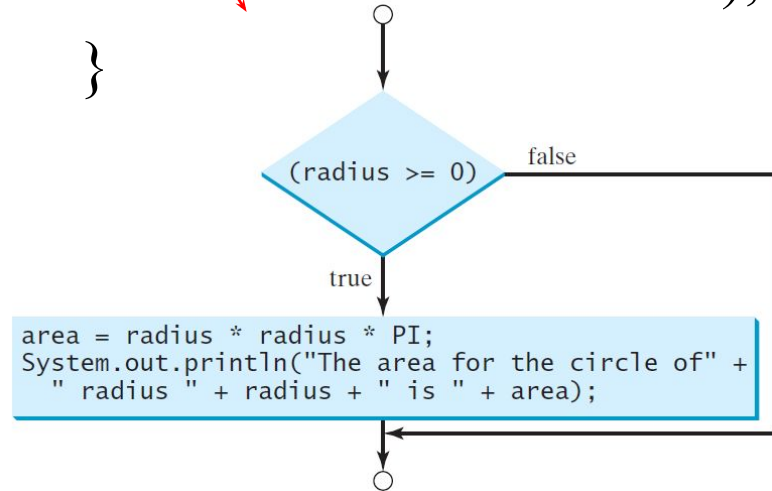
```
if (condicion) {  
    // acciones si se cumple la condición  
}
```

Sentencia if Statements

```
if (expresion -booleana) {  
    sentencia(s);  
}
```



```
if (radius >= 0) {  
    area = radius * radius * PI;  
    System.out.println("The area"  
        + " for the circle of radius "  
        + radius + " is " + area);  
}
```



Nota

```
if i > 0 {  
    System.out.println("i es positivo");  
}
```

(a) Erroneo

```
if (i > 0) {  
    System.out.println("i es positivo");  
}
```

(b) Correcto

```
if (i > 0) {  
    System.out.println("i es positivo");  
}
```

(a)

Equivalente

```
if (i > 0)  
    System.out.println("i es positivo");
```

(b)

Ejemplo: Demostración simple if

Escriba un programa que le pida al usuario que introduzca un número entero. Si el número es un múltiplo de 5, imprima HiFive. Si el número es divisible por 2, imprima HiEven.

```
package ejemplo;
import java.util.Scanner;

public class CondSimple {

    public static void main(String[] args) {
        var scanner = new Scanner(System.in);
        int numero;

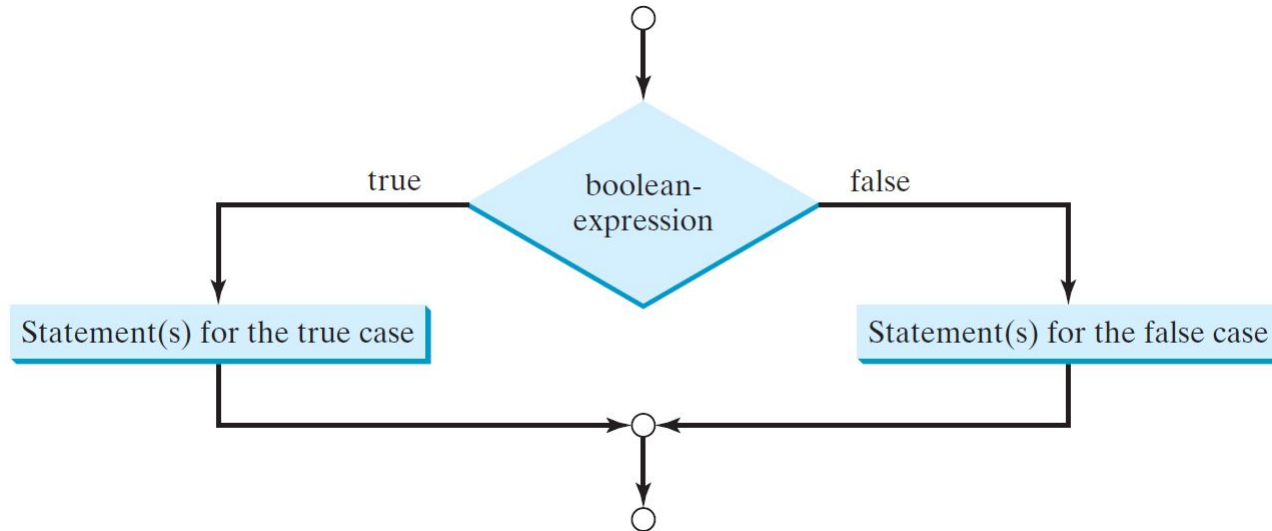
        System.out.println("Introduzca un número entero:");
        numero = scanner.nextInt();
        if (numero % 5 == 0)
            System.out.println("HiFive");

        if (numero % 2 == 0)
            System.out.println("HiEven");

        scanner.close();
    }
}
```


Sentencia `if` de dos vías

```
if (expresion-booleana) {  
    sentencia(s)-caso-de-ser-cierto;  
}  
else {  
    sentencia(s)-caso-de-ser-falso;  
}
```



Ejemplo if-else

```
if (radius >= 0) {  
    area = radius * radius * 3.14159;  
  
    System.out.println("el area para el " +  
        "circulo de radio " + radius +  
        " es " + area);  
}  
else {  
    System.out.println("radio negativo");  
}
```

Ejemplo: Demostración if-else

- 1.- Realiza un programa que indique si el usuario es mayor de edad o no. (El usuario debe indicar su edad).
- 2.- Realiza un programa que acepte dos números desde teclado e indique cuál de los números es mayor.
- 3.- Este ejemplo crea un programa para que realice una suma de primer grado. El programa genera aleatoriamente dos números enteros de un dígito número1 y número2 y muestra una pregunta como "¿Qué es $7 + 9$?" Al estudiante.

Después de que el estudiante escribe la respuesta, el programa muestra un mensaje para indicar si la respuesta es verdadera o falsa.

Sentencia IF-ELSE-IF

- Derivada de if-else
- Nos permite evaluar una segunda (tercera, cuarta, ...) condición si la primera (segunda, tercera, ...) no se cumple.

```
if (score >= 90.0)
    System.out.print("A");
else
    if (score >= 80.0)
        System.out.print("B");
    else
        if (score >= 70.0)
            System.out.print("C");
        else
            if (score >= 60.0)
                System.out.print("D");
            else
                System.out.print("F");
```

(a)

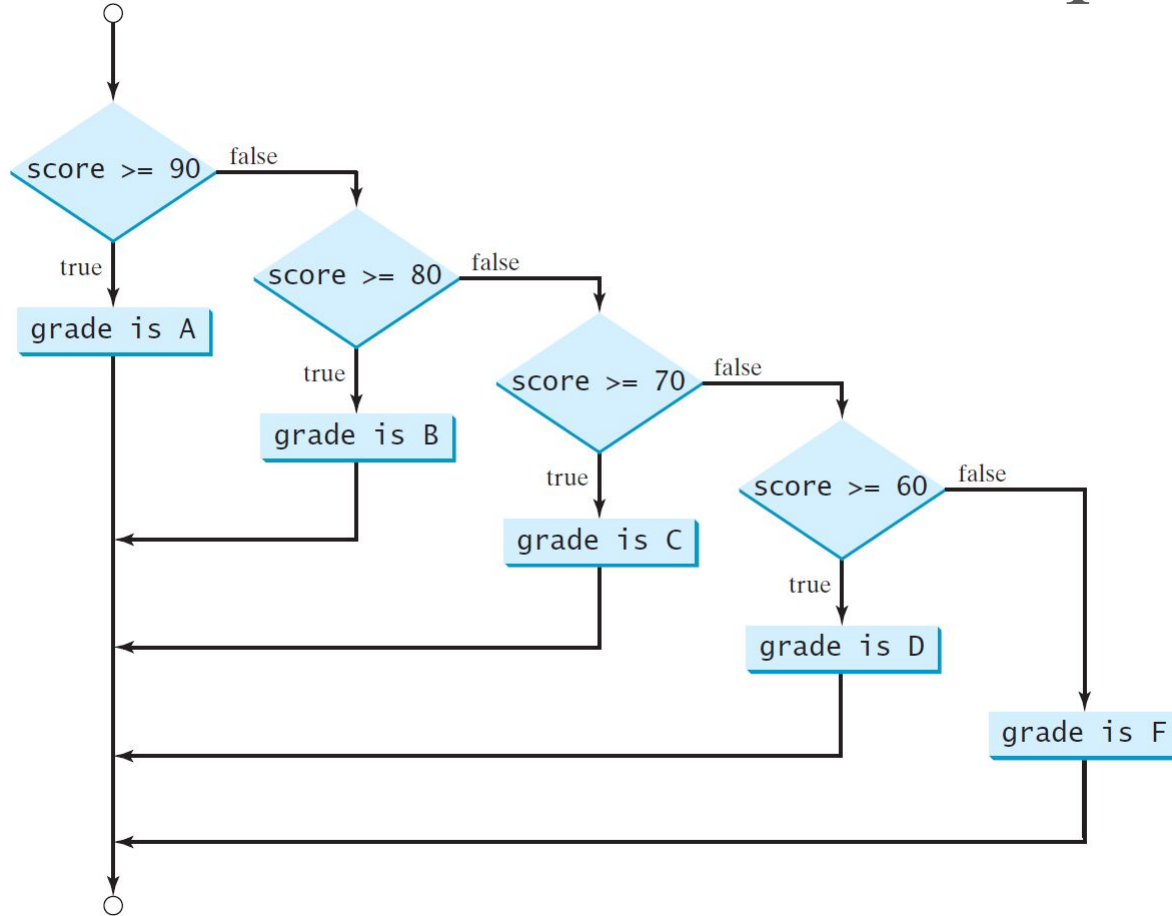
Equivalente

Así es mejor

```
if (score >= 90.0)
    System.out.print("A");
else if (score >= 80.0)
    System.out.print("B");
else if (score >= 70.0)
    System.out.print("C");
else if (score >= 60.0)
    System.out.print("D");
else
    System.out.print("F");
```

(b)

Declaraciones if-else de múltiples vías



Trace if-else statement

Supongamos que la
puntuación es 70.0

La condición es falsa

```
if (score >= 90.0)
```

```
    System.out.print("A");
```

```
else if (score >= 80.0)
```

```
    System.out.print("B");
```

```
else if (score >= 70.0)
```

```
    System.out.print("C");
```

```
else if (score >= 60.0)
```

```
    System.out.print("D");
```

```
else
```

```
    System.out.print("F");
```

Trace if-else statement

Supongamos que la
puntuación es 70.0

La condición es falsa

```
if (score >= 90.0)
    System.out.print("A");
else if (score >= 80.0)
    System.out.print("B");
else if (score >= 70.0)
    System.out.print("C");
else if (score >= 60.0)
    System.out.print("D");
else
    System.out.print("F");
```

Trace if-else statement

Supongamos que la
puntuación es 70.0

La condición es cierta

```
if (score >= 90.0)
    System.out.print("A");
else if (score >= 80.0)
    System.out.print("B");
else if (score >= 70.0)
    System.out.print("C");
else if (score >= 60.0)
    System.out.print("D");
else
    System.out.print("F");
```


Trace if-else statement

Supongamos que la
puntuación es 70.0

La nota es C

```
if (score >= 90.0)
    System.out.print("A");
else if (score >= 80.0)
    System.out.print("B");
else if (score >= 70.0)
    System.out.print("C");
else if (score >= 60.0)
    System.out.print("D");
else
    System.out.print("F");
```

Trace if-else statement

Supongamos que la
puntuación es 70.0

Fin de la sentencia

```
if (score >= 90.0)
    System.out.print("A");
else if (score >= 80.0)
    System.out.print("B");
else if (score >= 70.0)
    System.out.print("C");
else if (score >= 60.0)
    System.out.print("D");
else
    System.out.print("F");
```

Nota

La cláusula else coincide con la cláusula if más reciente en el mismo bloque.

```
int i = 1, j = 2, k = 3;

if (i > j)
    if (i > k)
        System.out.println("A");
    else
        System.out.println("B");
```

(a)

Equivalent

This is better
with correct
indentation

```
int i = 1, j = 2, k = 3;

if (i > j)
    if (i > k)
        System.out.println("A");
    else
        System.out.println("B");
```

(b)

Nota, continuación.

Nada se imprime de la declaración anterior. Para forzar la cláusula else para que coincida con la primera cláusula if, debe agregar un par de llaves :


```
int i = 1;
int j = 2;
int k = 3;
if (i > j) {
    if (i > k)
        System.out.println("A") ;
}
else
    System.out.println("B") ;
```

Esta declaración imprime B.

Errores comunes

Agregar un punto y coma al final de una cláusula if es un error común.

```
if (radius >= 0);  
{  
    area = radius*radius*PI;  
    System.out.println("The area for the circle of radius " +  
        radius + " is " + area);  
}
```



Este error es difícil de encontrar, ya que no es un error de compilación o un error de tiempo de ejecución, es un error lógico.

Este error a menudo se produce cuando utiliza el estilo de bloque siguiente línea.

TIP

```
if (number % 2 == 0)
    par = true;
else
    par = false;
```

(a)

Equivalente

```
boolean par
    = number % 2 == 0;
```

(b)

CAUTION

```
if (par == true)
    System.out.println(
        "Esto es par.");
```

(a)

Equivalente

```
if (par)
    System.out.println(
        "Esto es par.");
```

(b)

Funciones Matemáticas

Java proporciona muchos métodos útiles en la clase Math para realizar funciones matemáticas comunes (busca información sobre la clase).

Métodos de clase:

- Métodos Trigonométricos
 - sin, cos, tan,...
- Métodos exponentes;
 - exp(double a), pow(double a, double b), sqrt(double a)
- Métodos de redondeo:
 - floor(double x), round(float x), round(double x)
- Métodos min, max, abs y random:
 - max(a, b) y min(a, b)
 - abs(a)
 - random()

Funciones Matemáticas: Método random

Genera un valor aleatorio double mayor que o igual a 0,0 y menor que 1.0 ($0 \leq \text{Math.random()} < 1.0$).

Si queremos generar números enteros, basta con hacer un casting

Ejemplo:

`(int) (Math.random() * 10)` → Devuelve un entero aleatorio entre 0 y 9.

`50 + (int) (Math.random() * 50)` → Devuelve un entero aleatorio entre 50 y 99.

En general, para generar un número aleatorio en un rango es:

`a + Math.random() * b` → Devuelve un número aleatorio entre a y a + b, excluyendo a + b.

Ejemplo: genera aleatoriamente el número del día de la semana (entre 1 y 7):

```
int dia = (int)(Math.random()*7) + 1;
```


Ejercicios Funciones matemáticas

Ejercicio1: Introduce 3 notas desde teclado y muestra para cada nota, la nota redondeada con un decimal a la alza, a la baja, la nota truncada en las unidades y al final muestra la media e indica el máximo y mínimo.

Ejercicio2: Crea un programa para enseñar a un niño de primaria cómo aprender las sustracciones. El programa genera aleatoriamente dos números enteros de un dígito número1 y número2 con $\text{número1} \geq \text{número2}$ y muestra una pregunta como "¿Cuanto es 9-2?" Al estudiante. Después de que el estudiante escribe la respuesta, el programa muestra si la respuesta es correcta.

Ejercicio3: El lanzamiento de dados, que muestre el resultado y señale al ganador.

Problema: Índice de masa corporal

Índice de masa corporal (IMC) es una medida de la salud en el peso. Se puede calcular tomando su peso en kilogramos y dividiendo por el cuadrado de su altura en metros.

La interpretación del IMC para las personas mayores de 16 años es la siguiente:

IMC	Interpretación
IMC <18.5	Bajo de peso
18.5 >= IMC <25.0	Normal
25.0 >= IMC <30.0	Sobrepeso
IMC >=30.0	Obeso

Granja de caracoles

Realice un programa que calcule el precio final de la compra de una cantidad de kg con un determinado descuento y aplicando el IVA.

Se debe indicar si el comprador es particular o empresa y en función de la tabla calcular el descuento. El programa mostrará el precio sin descuento precio con descuento IVA y precio final

<u>Descuento</u>	<u>Empresa</u>	<u>Particular</u>
10%	mas de 10kg	mas de 12Kg
15%	mas de 25kg	mas de 30Kg
25%	mas de 50kg	mas de 60Kg
40%	mas de 100kg	mas de 120Kg

(El precio de un kg 3,5€)

Repaso Operadores lógicos

Operador	Nombre	Descripción
!	not	Negación lógica
&&	and	Conjunción lógica
	or	Disyunción lógica

Tabla de la verdad para el operador !

p	!p	Ejemplo (supongamos que la edad = 24, peso = 140)
true	false	!(edad > 18) es falsa, porque (edad > 18) es cierto.
false	true	! (peso == 150) es cierto, porque (peso == 150) es falso.

Tabla de la verdad para el operador &&

p_1	p_2	$p_1 \ \&\& \ p_2$	Ejemplo (supongamos que la edad = 24, peso = 140)
false	false	false	(edad \leq 18) && (peso $<$ 140) es falsa, porque ambas condiciones son falsas.
false	true	false	
true	false	false	(edad $>$ 18) && (peso $>$ 140) es falso, porque (peso $>$ 140) es falso.
true	true	true	(edad $>$ 18) && (peso \geq 140) es cierto, ya que ambos (edad $>$ 18) y (peso \geq 140) son verdaderos.

Tabla de la verdad para el operador ||

p_1	p_2	$p_1 \parallel p_2$	Ejemplo (supongamos que la edad = 24, peso = 140)
false	false	false	
false	true	true	$(\text{edad} > 34) \parallel (\text{peso} \leq 140)$ es true, porque $(\text{edad} > 34)$ es falso, pero $(\text{peso} \leq 140)$ es cierto.
true	false	true	$(\text{edad} > 14) \parallel (\text{peso} \geq 150)$ is false, porque $(\text{edad} > 14)$ is cierto.
true	true	true	

Repaso Operadores & y |

El operador **&** funciona exactamente igual que el operador **&&**, y el operador **|** funciona exactamente igual que el operador **||** con una excepción: los operadores **&** y **|** siempre evalúan ambos operandos. Por lo tanto, **&** se conoce como los operadores *AND* y *OR incondicional*.

Usar cuando sea necesario garantizar que el operando de la derecha se evalúa independientemente de si el operando de la izquierda es verdadero o falso.

Los beneficios de los operadores **&** y **|** son marginales. El uso de ellos hará que el programa sea difícil de leer y podría causar errores.

Problema: Determinación del año bisiesto?

Este programa solicita primero al usuario que introduzca un año como un valor int y compruebe si es un año bisiesto.

Un año es bisiesto **si es divisible por 4** pero **no** por 100, o es divisible por 400.

(año % 4 == 0 && año % 100 != 0) || (año % 400 == 0)

Problema: Lotería

Escriba un programa que genere aleatoriamente un número de dos dígitos y que le pida al usuario que introduzca un número de dos dígitos y determine si el usuario gana de acuerdo con la siguiente regla:

- Si la entrada del usuario coincide con el número aleatorio en el orden exacto, el premio es de 10,000€.
- Si la entrada del usuario coincide con el número aleatorio en orden inverso, el premio es 3,000€.
- Si un dígito en la entrada del usuario coincide con un dígito del número aleatorio, el premio es de 1,000€.

Declaraciones `switch`

- Nos permite **evaluar varias posibilidades** sin tener que pasar por todas las anteriores.
- Desventaja: las **condiciones son siempre de igualdad**.
- Funciona con los **tipos primitivos** y **String**, entre otros.
- Dos versiones
 - `switch` como sentencia.
 - `switch` como expresión.

SWITCH COMO SENTENCIA

- Es la forma tradicional.
- Para cada caso podemos ejecutar una o varias sentencias.
- Al terminar un caso añadimos `break`.
- Si no lo hacemos, continúa ejecutando el siguiente caso.
- `default` por si no coincide con ningún caso (similar a `else`).

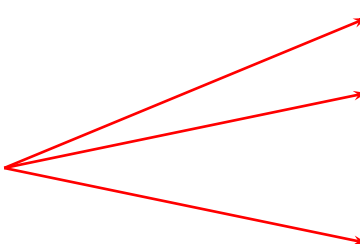
Sintaxis switch

```
switch(expresion) {  
    case v1: accion1;  
        accion2;  
        break;  
    case v2: accion3;  
        break;  
    // más casos  
    default: accionN;  
}
```

Reglas sentencia switch

La expresión-switch debe llevar un valor de tipo char, byte, short o int y debe estar siempre entre paréntesis.

El valor1, ... y el valorN deben tener el mismo tipo de datos que el valor de la expresión-switch. Las sentencias resultantes en la instrucción case se ejecutan cuando el valor en la instrucción case coincide con el valor de la expresión-switch. Tenga en cuenta que value1, ..., y valueN son expresiones constantes, lo que significa que no pueden contener variables en la expresión, como $1 + x$.



```
switch (switch-expression) {  
    case value1: statement(s)1;  
        break;  
    case value2: statement(s)2;  
        break;  
    ...  
    case valueN: statement(s)N;  
        break;  
    default: statement(s)-for-default;  
}
```

Reglas sentencia switch

La palabra clave break es opcional, pero debe utilizarse al final de cada caso para terminar el resto de la instrucción switch. Si la sentencia break no está presente, la siguiente declaración de case se ejecutará

El default, que es opcional, puede utilizarse para realizar acciones cuando ninguno de los casos especificados coincide con la expresión de conmutación.

```
switch (switch-expression) {  
    case value1: statement(s)1;  
        break;  
    case value2: statement(s)2;  
        break;  
    ...  
    case valueN: statement(s)N;  
        break;  
    default: statement(s)-for-default;  
}
```

Cuando el valor de una instrucción **case** coincide con **switch-expression**, las sentencias que comienzan a partir de este caso se ejecutan hasta que se alcanza una instrucción **break** o el final de la instrucción switch.

Seguimiento de la sentencia switch

Supongamos que el día es 2 :

```
switch (day) {  
    case 1:  
    case 2:  
    case 3:  
    case 4:  
    case 5: System.out.println("Weekday"); break;  
    case 0:  
    case 6: System.out.println("Weekend");  
}
```

Seguimiento de la sentencia switch

Coincide case 2

```
switch (day) {  
    case 1:  
    case 2:  
    case 3:  
    case 4:  
    case 5: System.out.println("Weekday"); break;  
    case 0:  
    case 6: System.out.println("Weekend");  
}
```


Seguimiento de la sentencia switch

baja a través de case 3

```
switch (day) {  
    case 1:  
    case 2:  
    case 3:  
    case 4:  
    case 5: System.out.println("Weekday"); break;  
    case 0:  
    case 6: System.out.println("Weekend");  
}
```

Seguimiento de la sentencia switch

baja a través de case 4

```
switch (day) {  
    case 1:  
    case 2:  
    case 3:  
    case 4:  
    case 5: System.out.println("Weekday"); break;  
    case 0:  
    case 6: System.out.println("Weekend");  
}
```

Seguimiento de la sentencia switch

baja a través de case 5

```
switch (day) {  
    case 1:  
    case 2:  
    case 3:  
    case 4:  
    case 5: System.out.println("Weekday"); break;  
    case 0:  
    case 6: System.out.println("Weekend");  
}
```

Seguimiento de la sentencia switch

Encuentra break

```
switch (day) {  
    case 1:  
    case 2:  
    case 3:  
    case 4:  
    case 5: System.out.println("Weekday"); break;  
    case 0:  
    case 6: System.out.println("Weekend");  
}
```

Seguimiento de la sentencia switch



Finaliza la sentencia

```
switch (day) {  
    case 1:  
    case 2:  
    case 3:  
    case 4:  
    case 5: System.out.println("Weekday"); break;  
    case 0:  
    case 6: System.out.println("Weekend");  
}
```

Variante Switch como expresión

- Desde Java 14
- En lugar de ejecutar una serie de acciones por cada caso, se realiza una asignación.
- Se pueden agrupar varios casos:
case v1, v2, ... → resultado

- **Sintaxis**

```
variable = switch(expresión) {  
    case valor1 -> resultado1;  
    case valor2 -> resultado2;  
    // ...  
    case valorn -> resultadon;  
    default -> resultadoPorDefecto;  
};
```

Seguimiento de switch como expresión

Supongamos que el día es 2 :

```
switch (day) {  
    case 1, 2, 3, 4, 5 -> System.out.println("Weekday");  
    case 0, 6 -> System.out.println("Weekend");  
}
```

Seguimiento de switch como expresión

Coincide case 2

```
switch (day) {  
  case 1, 2, 3, 4, 5 -> System.out.println("Weekday");  
  case 0, 6 -> System.out.println("Weekend");  
}
```


Seguimiento de switch como expresión

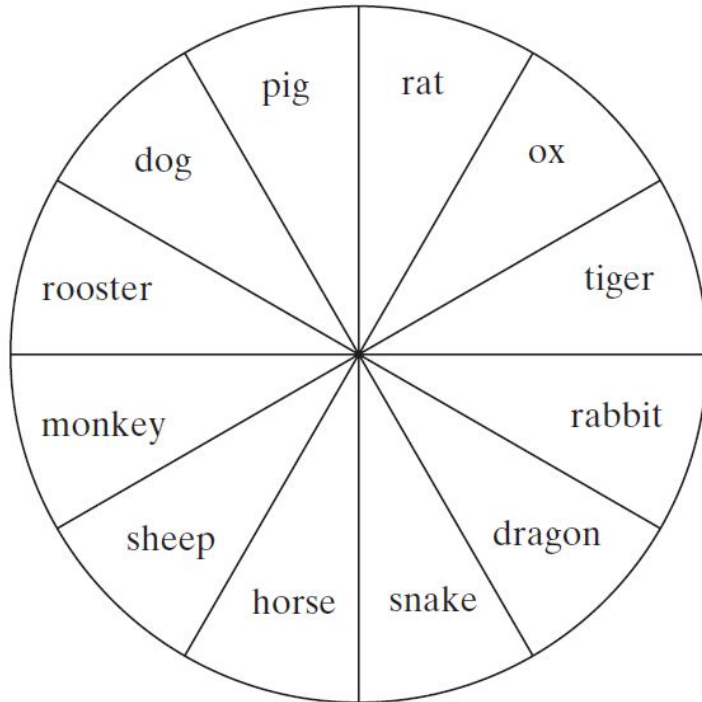


Finaliza la sentencia

```
switch (day) {  
    case 1, 2, 3, 4, 5 -> System.out.println("Weekday");  
    case 0, 6 -> System.out.println("Weekend");  
}
```

Problema: Zodiaco chino

Escriba un programa que le pida al usuario que introduzca un año y muestre el animal para el año.



$$\text{year} \% 12 = \left\{ \begin{array}{l} 0: \text{monkey} \\ 1: \text{rooster} \\ 2: \text{dog} \\ 3: \text{pig} \\ 4: \text{rat} \\ 5: \text{ox} \\ 6: \text{tiger} \\ 7: \text{rabbit} \\ 8: \text{dragon} \\ 9: \text{snake} \\ 10: \text{horse} \\ 11: \text{sheep} \end{array} \right.$$