INSTRUCCIÓN SWITCH

La escalera if de un ejercicio del boletín anterior:

```
class Unidad3{
   public static void main(String args[]){
      int x;
      x=3;
      if (x==1)
         System.out.println("x es uno");
      else if (x==2)
         System.out.println("x es dos");
      else if (x==3)
         System.out.println("x es tres");
      else if (x==4)
         System.out.println("x es cuatro");
      else if (x==5)
         System.out.println("x es cinco");
      else
         System.out.println("x no se encuentra entre uno y cinco");
   }
}
```

Se puede reescribir con la instrucción switch

```
class Unidad3{
   public static void main(String args[]){
      int x = 3;
      switch (x){
        case 1:
          System.out.println("x es uno");
          break;
        case 2:
          System.out.println("x es dos");
          break;
        case 3:
          System.out.println("x es tres");
          break;
        case 4:
          System.out.println("x es cuatro");
          break;
        case 5:
          System.out.println("x es cinco");
          break;
        default:
          System.out.println("x no está entre uno y cinco" );

      }
   }
}
```

La instrucción *switch* fuerza a que el control del programa derive al primer *case* que "encaja" con la expresión que va entre paréntesis del switch, y se procede a ejecutar todas las instrucciones que se encuentran a continuación de dicho *case*, incluso las de los otros *case*, a no ser que nos encontremos con una instrucción *break*. La instrucción break provoca que "se salga fuera" del bloque que la contiene, en este caso, el break está dentro del bloque *switch* y obliga a salir de dicho bloque.

Observa que no es necesario poner brackets {} para cada case, sólo se usarían si por alguna razón necesitamos definir una variable con alcance limitado a las llaves. Si este no es el caso, no se usan por cuestión de estilo.

Ejemplo Eliminamos el break del case 3 y 4 y observamos el nuevo flujo del programa con el depurador

```
class Unidad3{
  public static void main(String args[]){
    int x = 3;
    switch (x){
      case 1:
        System.out.println("x es uno");
        break;
      case 2:
        System.out.println("x es dos");
        break;
      case 3:
        System.out.println("x es tres");
        //sin break;
      case 4:
        System.out.println("x es cuatro");
        //sin break;
      case 5:
        System.out.println("x es cinco");
        break;
      default:
        System.out.println("x no está entre uno y cinco" );
    }
  }
}
```

Al ejecutar el código anterior observamos: Como x vale 3 el flujo de programa "salta" a case 3, imprime "x es tres", pero como ahora no hay break prosigue el flujo del programa linealmente entrando en case 4 e imprime "x es cuatro", avanza hacia case 5 donde imprime "x es cinco" y a continuación encuentra un break y sale de bloque del switch y por tanto, no se llega a imprimir el mensaje de *default*.

Observa que, como casi siempre un case lleva asociado un break, si no lo lleva, es conveniente dejar un comentarios aclaratorio de que eso es justo lo que se quiere

No toda "escalera if" se puede convertir en un switch, ya que,la expresión del switch tiene que ser:

- de tipo char, byte, short o int (es decir, int o más pequeños que int)
- tipo enumerado
- un String (novedad del JDK 7, por eso, en muchos libros y apuntes no viene esto)

Ejemplo: Comprobamos error de compilación

```
class Unidad3{
  public static void main(String args[]){
    double x = 3;
    switch (x){
      case 1.0:
        System.out.println("x es uno");
        break;
      case 2.0:
        System.out.println("x es dos");
        break;
      case 3.0:
        System.out.println("x es tres");
        //sin break;
      case 4.0:
        System.out.println("x es cuatro");
        //sin break;
      case 5.0:
        System.out.println("x es cinco");
        break;
      default:
        System.out.println("x no está entre uno y cinco" );
```

```
    }
  }
}
```

Java la expresión del switch la intenta a pasar a int y no es posible de double a int si hacemos nosotros cast

```
class Unidad3{
  public static void main(String args[]){
    double x = 3;
    switch ((int)x){
      case 1.0:
        System.out.println("x es uno");
        break;
      case 2.0:
        System.out.println("x es dos");
        break;
      case 3.0:
        System.out.println("x es tres");
        //sin break;
      case 4.0:
        System.out.println("x es cuatro");
        //sin break;
      case 5.0:
        System.out.println("x es cinco");
        break;
      default:
        System.out.println("x no está entre uno y cinco" );
    }
  }
}
```

observaremos ahora el error de javac en las constantes del Case

Ejemplo: con long un tanto de lo mismo ...

```
class Unidad3{
  public static void main(String args[]){
    long x = 3;
    switch (x){
      case 1L:
        System.out.println("x es uno");
        break;
      case 2L:
        System.out.println("x es dos");
        break;
      case 3L:
        System.out.println("x es tres");
        //sin break;
      case 4L:
        System.out.println("x es cuatro");
        //sin break;
      case 5L:
        System.out.println("x es cinco");
        break;
      default:
        System.out.println("x no está entre uno y cinco" );
    }
  }
}
```

Ejemplo: En el switch pueden escribirse expresiones todo lo complejas que queramos, siempre y cuando finalmente se evalúen y obtengamos un valor de un tipo permitido en el switch

```
class Unidad3{
  public static void main(String args[]){
    int x = 3;
    switch (x%2+ 1){
      case 1:
```

```
        System.out.println(" case uno");
        break;
      case 2:
        System.out.println(" case dos");
        break;
      case 3:
        System.out.println(" case tres");
        //break;
      case 4:
        System.out.println(" case cuatro");
        //break;
      case 5:
        System.out.println(" case cinco");
        break;
      default:
        System.out.println(" no está entre uno y cinco" );
    }
  }
}
```

Ejercicio 1: Crea un programa que pida por teclado un número de mes y responda con el nombre del mes. Ejemplo de ejecución

```
teclea un número de mes(1-12):
6
mes en String: Junio
```

Desde Java 7 la expresión del switch también puede devolver un String, con lo cual podemos hacer el siguiente ejercicio.

Ejercicio 2: Consigue el efecto contrario al ejercicio anterior, es decir

```
teclea un mes:
Octubre
mes en número: 10
```

Una observación importante es que la expresión del switch no puede ser boolean, esto limita bastante el switch de Java ya que código del tipo.

```
if (x>0)
  System.out.println("x es mayor que 0);
else if (y>0)
  System.out.println("y es mayor que 0);
else if (z ==8)
  System.out.println("z es igual a 8);
```

No se puede escribir con switch

Ejercicio 3 Repite el ejercicio de las vocales hecho el boletín anterior con la instrucción *if*, ahora con la instrucción *switch*.

Truco: cuando varios "casos" comparten instrucciones, los agrupas y sólo escribes instrucciones para el último caso, por ejemplo:

```
        case 'a':
```

```
            case 'A':
                System.out.println("A de Abejita");
                break;
```

Ejercicio 4: El siguiente programa, dado un número x entre 1 y 5, cuenta hasta el número x. Escríbelo con if.

```
class Unidad3{
   public static void main(String args[]){
      int x = 3;
      switch (x){
         case 1:
            System.out.println(1);
            //break;
         case 2:
            System.out.println(2);
            //break;
         case 3:
            System.out.println(3);
            //break;
         case 4:
            System.out.println(4);
            //break;
         case 5:
            System.out.println(5);
            break;
         default:
            System.out.println("el número inicial no está entre uno y cinco" );
      }
   }
}
```

Ejercicio 5: Escribe un programa que lea un mes y un año en formato numérico e indique el número de días de ese mes. Para realizar este ejercicio hay que tener en cuenta que un año es bisiesto si es divisible por cuatro, excepto cuando es divisible por 100, a no ser que sea divisible por 400.

```
class Unidad3{
   public static void main(String[] args) {

      int month = 2;//cambiamos month para probar switch
      int year = 2000;
      int numDays = 0;

      switch (month) {
         Etc…
    }
      System.out.println("Number of Days = " + numDays);
   }
}
```

Novedades con Lambda:

Se permite con Switch el "patter matching" (->) y valores como resultado. Con esta nueva estructura de switch no es necesario el uso de break para prevenir la ejecución de otras sentencias case.
Ejemplo:

```
int numLetters = switch (dia) {
                    case MONDAY, FRIDAY, SUNDAY -> 6;
                    case TUESDAY                -> 7;
                    case THURSDAY, SATURDAY     -> 8;
                    case WEDNESDAY              -> 9;
                    default -> 0;
                };
```

Siempre será necesario el uso de default salvo cuando lo uséis con enumerados (rango de valores fijos). Haz la prueba con el anterior ejemplo.

Versión clásica:

```
switch (day) {
    case MONDAY:
    case FRIDAY:
    case SUNDAY:
        System.out.println(6);
        break;
    case TUESDAY:
        System.out.println(7);
        break;
    case THURSDAY:
    case SATURDAY:
        System.out.println(8);
        break;
    case WEDNESDAY:
        System.out.println(9);
        break;
}
```

Versión más actual:

```
switch (day) {
    case MONDAY, FRIDAY, SUNDAY -> System.out.println(6);
    case TUESDAY                -> System.out.println(7);
    case THURSDAY, SATURDAY     -> System.out.println(8);
    case WEDNESDAY              -> System.out.println(9);
}
```

Versión clásica:

```
int numLetters;
switch (day) {
    case MONDAY:
    case FRIDAY:
    case SUNDAY:
        numLetters = 6;
        break;
    case TUESDAY:
        numLetters = 7;
        break;
    case THURSDAY:
    case SATURDAY:
        numLetters = 8;
        break;
    case WEDNESDAY:
        numLetters = 9;
        break;
    default:
        throw new IllegalStateException("Wat: " + day);
}
```

Versión actual:

```
int numLetters = switch (day) {
    case MONDAY, FRIDAY, SUNDAY -> 6;
    case TUESDAY                -> 7;
    case THURSDAY, SATURDAY     -> 8;
    case WEDNESDAY              -> 9;
};
```

Ejercicio 6: implementa una función que nos devuelva si un día es laborable, fin de semana o día no reconocido. La firma de la funcion es la siguiente:

```
private static String obtenerTipoDiaDeLaSemana(String day)
```

day tendrá los siguientes valores: “L”, “M”, “X”, “J”, “V”, “S”, “D”
Debes realizarlo usando el “nuevo” switch.