



MÓDULO 0485 - PROGRAMACIÓN

UT 3-

ESTRUCTURAS DE CONTROL
(PARTE 1) – CONDICIONALES

Técnico Superior En Desarrollo De Aplicaciones Web

1º DAW (Grupo 1WV) (Vespertino)

Curso 2025-26

Profesor: Javier Rojo

ÍNDICE



PARTE TEÓRICA/CONCEPTUAL

- Estructuras de control: Condicionales
- Estructuras de control: Bucles

PARTE PRÁCTICA

- Ejercicios de condicionales
- Bucles
- Ejercicios de bucles



ESTRUCTURAS DE CONTROL: CONDICIONALES

ESTRUCTURAS DE CONTROL: CONDICIONALES



ÍNDICE

- Estructuras alternativas
 - Estructuras alternativas Simple (if)
 - Estructuras alternativas Doble (if-else)
 - Estructuras alternativas múltiples (switch)
- Ejemplo

The background features several large, overlapping organic shapes in yellow, light purple, pink, red, and blue. A small black smiley face is positioned at the top center, within the light purple shape.

ESTRUCTURAS ALTERNATIVAS

ESTRUCTURAS ALTERNATIVAS

- Como ya vimos, las estructuras alternativas son construcciones que permiten alterar el flujo secuencial de un programa
- En función de una condición o el valor de una expresión, el mismo pueda ser desviado en una u otra alternativa de código.
- Las estructuras alternativas disponibles en Java son:
 1. Alternativa Simple (if)
 2. Alternativa Doble (if-else)
 3. Alternativa Múltiple (switch)

1. ESTRUCTURA ALTERNATIVA SIMPLE (IF)

ESTRUCTURA ALTERNATIVA SIMPLE (IF)

1. La alternativa simple se codifica de la siguiente forma:

Código	Diagrama de flujo
<pre>if (condición) { // Acciones; }</pre> <p>El bloque de Acciones se ejecuta si la condición se evalúa a true (es verdadera).</p> <pre>if (cont == 0) { System.out.println("cont es 0"); // más instrucciones... }</pre> <p>Si dentro del if solo hay una instrucción, no es necesario poner las llaves.</p> <pre>if (cont == 0) System.out.println("cont es 0");</pre>	<pre>graph TD Start(()) --> Condicion{Condicion} Condicion -- Si --> Acciones[Acciones] Condicion -- No --> Merge(()) Acciones --> Merge Merge --> Exit(())</pre>

2. ESTRUCTURA ALTERNATIVA DOBLE (IF-ELSE)

ESTRUCTURA ALTERNATIVA DOBLE (IF-ELSE)

1. La alternativa doble permite indicar qué código ejecutar si la condición es falsa.

Recordad que el operador relacional para comprobar si son iguales es ==, no un solo = que corresponde con el operador de asignación. Este error no lo detecta el compilador y es difícil de averiguar.

Código	Diagrama de flujo
<pre> if (condición) { // AccionesSI; } else { // AccionesNO; } </pre> <p>El bloque AccionesSI se ejecuta si la condición se evalúa a true (verdadera). En caso contrario, se ejecuta el bloque de AccionesNO.</p> <pre> if (cont == 0) { System.out.println("cont es 0"); // más instrucciones... } else { System.out.println("cont no es 0"); // más instrucciones... } </pre> <p>Si dentro del if o el else solo hay una instrucción, no es necesario poner las llaves.</p> <pre> if (cont == 0) System.out.println("cont es 0"); else System.out.println("cont no es 0"); </pre>	<pre> graph TD Entrada(()) --> Condicion{Condicion} Condicion -- NO --> Acciones1[Acciones] Condicion -- SI --> Acciones2[Acciones] Acciones1 --> Salida(()) Acciones2 --> Salida Salida --> Salida </pre>

ESTRUCTURA ALTERNATIVA DOBLE (IF-ELSE)

1. En muchas ocasiones, se anidan estructuras alternativas if-else, de forma que se pregunte por una condición si anteriormente no se ha cumplido otra sucesivamente.
2. Por ejemplo: supongamos que realizamos un programa que muestra la nota de un alumno en la forma (insuficiente, suficiente, bien, notable o sobresaliente) en función de su nota numérica. Podría codificarse de la siguiente forma:

```
1  import java.util.Scanner;
2
3
4  public class Nota {
5
6      public static void main(String[] args) {
7          Scanner entrada = new Scanner(System.in);
8          int nota;
9          //Suponemos que el usuario introduce el número correctamente.
10         //No hacemos comprobación
11         System.out.println("Dame un número entre 0 y 10");
12
13         nota = entrada.nextInt();
14
15         if (nota < 5) {
16             System.out.println("Insuficiente");
17         } else if (nota < 6) {
18             System.out.println("Suficiente");
19         } else if (nota < 7) {
20             System.out.println("Bien");
21         } else if (nota < 9) {
22             System.out.println("Notable");
23         } else {
24             System.out.println("Sobresaliente");
25         }
26     }
27 }
28 }
```

Es muy recomendable usar la tecla tabulador en las instrucciones de cada bloque. Como se puede ver en el ejemplo, cada **else** está alineado con su **if** asociado, de esta forma es más fácil leer el código.

```
run:
Dame un número entre 0 y 10
8
Notable
BUILD SUCCESSFUL (total time: 11 seconds)
```

3. ESTRUCTURA ALTERNATIVA MÚLTIPLE (SWITCH)

ESTRUCTURA ALTERNATIVA MÚLTIPLE (SWITCH)

1. La alternativa múltiple se codifica de la siguiente forma:

Es muy importante entender que en el switch se evalúa una expresión (un valor concreto como 0, 5, 1...) no una condición (verdadera o falsa) como en el if y el ifelse.

Código	Ordinograma
<pre> switch (expresión) { case valor1: // Acciones1; break; case valor2: // Acciones2; break; case valorN: // AccionesN; break; default: // Acciones por defecto; } </pre>	<pre> graph TD Entry(()) --> Exp{expresión} Exp --> Circle(()) Circle -- valor 1 --> Acc1[Acciones 1] Circle -- valor 2 --> Acc2[Acciones 2] Circle -- valor n --> Accn[Acciones n] Circle -- Defecto --> AccDef[Acciones] Acc1 --> Merge((X)) Acc2 --> Merge Accn --> Merge AccDef --> Merge Merge --> Exit(()) </pre>

ESTRUCTURA ALTERNATIVA MÚLTIPLE (SWITCH)

- El programa comprueba el valor de la expresión
 - Saltará al 'case' que corresponda con dicho valor (valor1 o valor2 o ...) ejecutando el código de dicho 'case' (Acciones1 o Acciones2 o ...).
 - Si no coincide ningún valor, saltará al 'default' y ejecutará las acciones por defecto.
- Es importante añadir la sentencia **break;** al final de cada 'case',
 - De lo contrario el programa seguirá ejecutando el código de las demás acciones y normalmente no queremos que haga eso (aunque Java permite hacerlo, es confuso y por ello está desaconsejado).

ESTRUCTURA ALTERNATIVA MÚLTIPLE (SWITCH)

Un ejemplo sería el siguiente:

```
run:
Dame un número entre 1 y 7:
4
Jueves
BUILD SUCCESSFUL (total time: 2 seconds)
```

```
7 import java.util.Scanner;
8
9 public class Ejemplo1 {
10
11     public static void main(String[] args) {
12
13         Scanner reader = new Scanner(System.in);
14
15         System.out.println("Dame un número entre 1 y 7: ");
16         int dia = reader.nextInt();
17
18         switch (dia) {
19             case 1:
20                 System.out.println("Lunes");
21                 break;
22             case 2:
23                 System.out.println("Martes");
24                 break;
25             case 3:
26                 System.out.println("Miércoles");
27                 break;
28             case 4:
29                 System.out.println("Jueves");
30                 break;
31             case 5:
32                 System.out.println("Viernes");
33                 break;
34             case 6:
35             case 7:
36                 System.out.println("Fin de semana");
37                 break;
38             default:
39                 System.out.println("Error. El número debe estar entre 0 y 7");
40         }
41     }
42 }
```



EJEMPLO



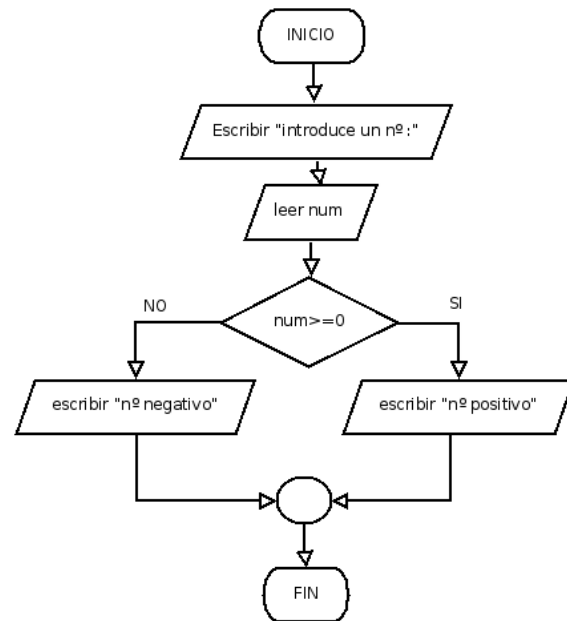
EJEMPLO

- Enunciado del ejemplo: Programa que lee un número y me dice si es positivo o negativo. Consideraremos el cero como positivo.



EJEMPLO

- Enunciado del ejemplo: Programa que lee un número y me dice si es positivo o negativo. Consideraremos el cero como positivo.
- Diagrama de flujo:



EJEMPLO

- Código de la solución y salida esperada:

```
1 package ejemplo2;
2
3 import java.util.Scanner; // Importamos la clase Scanner
4
5 public class Ejemplo2 {
6
7     public static void main(String[] args) {
8
9         // Declaramos la variable num
10        int num;
11
12        // Creamos el objeto Scanner para leer por teclado
13        Scanner reader = new Scanner(System.in);
14
15        // Pedimos y leemos x
16        System.out.print("Introduce un nº: ");
17        num = reader.nextInt();
18
19        // Aestructura alternativa doble
20        if (num >= 0)
21            System.out.println("Número positivo");
22        else
23            System.out.println("Número negativo");
24    }
25 }
```

```
run:
Introduce un nº: 10
Número positivo
```



BIBLIOGRAFÍA

Apuntes actualizados y adaptados a partir de la siguiente documentación:

1. [1] Apuntes Programación de José Antonio Díaz-Alejo. IES Camp de Morvedre.
 2. [2] Apuntes Programación de Javier Valero Lionel Tarazón. Ceedcv.
- 



U

¿DUDAS?



FIN

Javier Rojo
fjrojom001@educarex.es