Java

Tema 2

Estructuras básicas de control

Sentencias

Sentencias de **selección**

- if
- Operador ?:
- switch

Sentencias de **iteración/repetición** (bucles)

- for
- while
- do .. while

Java

Sentencias de selección: If y switch

3

La sentencia `if .. else`

- La sentencia if-else permite elegir qué bloque de código se ejecutará entre dos posibilidades, según se cumpla o no la condición, es decir sea verdadera (true) ó falsa (false).

```
if (condición){
    sentencias1;//si es true
} else {
    sentencias2; //si es false
}
```

```
if (condición){
     //si es true
     sentencias1;
}
```

- Si la expresión condicional es verdadera/true, se ejecutarán las sentencias del if; sino, se ejecutarán, si existe, las sentencias del else.
- Las llaves { } solo son obligatorias si hay más de una sentencia pero es recomendable utilizarlas

4

La sentencia `if .. else:` ejemplo

```
import java.util.*;
class Main {
  public static void main ( String args [ ] ) {
   int num = 100;

   if (num > 0){
      System.out.println(num + " es positivo");
   }else{
      System.out.println(num + " es negativo");
   }
  }
}
```

- **¿Qué pasa con el número 0?**

If anidados

- Cuando se anidan if, cada else corresponde al if más próximo dentro del mismo bloque que no este asociado ya con otro else.

```
if ( i == 10 ){
    if (j < 20)
        a = b;
    if (k > 100)
        c = d;
    else // se refiere a (k <= 100)
        a = c;
}else // se refiere a (i!=10)
        a = d;
```

La escalera `if-else-if`

```
import java.util.*:
class Main {
 public static void main(String args[]){
   int x =  2;

   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
            System.out.println("x debe estar entre 1 y 3");
 }
}
```

Escribe este código sin usar if-else-if

7

Operador ? :

- Existe un operador ternario, usa tres operandos, que abrevia una sentencia if. Si la condición se cumple (es true) asigna valor1 a resultado, sino asigna valor2.

```
resultado = (condicion) ? valor1 : valor2
```

- Es equivalente a la siguiente sentencia IF:

```
if (condicion){
   resultado = valor1;
} else {
   resultado = valor2;
}
```

8

Operador ? :

- Ejemplo

```
class Main {
 public static void main(String args[]){
   int numero;
   String resultado;

    numero = -25;

   resultado = (numero>0) ? "positivo" : "negativo";

   System.out.println(resultado);
 }
}
```

Escribe este código usando un IF

La sentencia switch

- La sentencia switch es un caso particular de if-else.

```
switch (expresión){
    case valor_1:
        // sentencias
        break;
    case valor_2:
        // sentencias
        break;
    . . .
    case valor_N:
        // sentencias
        break;
    default:
      //sentencias por defecto. Si no se cumple ningún case
} //fin del switch
```

- Expresión debe ser tipo entero, carácter, String ó enumerado.
- Cada valor debe ser compatible con el de expresión.
- Cuando aparece break, la ejecución del código continua en la primera línea tras el fin del switch. Si se omite break, la ejecución continúa secuencialmente hacia abajo.

La sentencia `switch`

Ejemplo de uso sentencia switch

```
import java.util.*:
class Main {
 public static void main(String args[]){
   Scanner sc = new Scanner(System.in);
   System.out.println("Dame un valor entre 1 y 3: ");
   int x =  sc.nextInt();

   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;
     default:  System.out.println("x NO esta entre 1 y 3");
   }
 }
}
```


Ejercicios

1. **Programa que recoja un número por teclado que representa la nota de un examen.**
   - **Controla que sea de 0 a 10**
   - **Indica con texto que nota le corresponde (0,1,2,3,4→ insuficiente, 5→suficiente, 6→ bien ….).**
2. **Haz dos versiones del ejercicio anterior una usando sentencias if y otra con sentencia switch**
3. **Programa que recoja el día de la semana en texto y nos indique el número que lo representa ("lunes"→ 1, "martes" → 2 …)**

Java

Sentencias de iteración/repetición: Bucles

13

Tipos de Bucles

- for
- while
- do ... while

14

El bucle `for`

```
for (inicialización; condición; iteración) {
  // cuerpo del bucle, conjunto de sentencias
}
```

- Permite que se ejecute un conjunto de sentencias hasta que la condición deja de cumplirse, se suele utilizar una variable contador que se modifica su valor según lo expresado en iteración.
  - inicialización: valor inicial de la variable/s contador/es
  - condición: expresión lógica de condición de ejecución, mientras sea true el bucle se ejecutará, si es false termina.
  - iteración: se ejecuta en cada vuelta. Suele ser incremento / decremento del contador/es.
- Tanto la inicialización como condición como iteración, son opcionales
- Las variables definidas en la inicialización son **locales** al bucle. Por lo tanto, dejan de existir una vez finaliza el bucle.

15

El bucle `for`

Ejemplo: Mostrar los 100 primeros números naturales. Del **0 al 99**

```
for (int i = 0; i < 100; i++) {
      System.out.println(i);
}
```

Ejercicio: ¿Cómo mostrar los números del 1 al 100?

16

El bucle for

- Pueden usarse **varias variables de control** del bucle for:

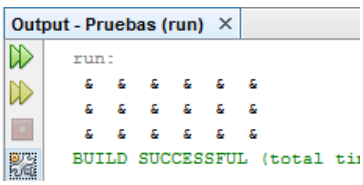
```
Class Main {
  public static void main(String args[ ]) {
  // dos variables de control
    for (int i = 1, j = 4;  i < j ;  i++, j-- ){
      System.out.println ("i y j : " + i + " " + j);

    }
  }
}
```

17

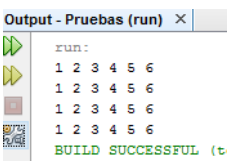
El bucle for

- Pueden **anidarse un bucle dentro de otro**:

```
for (int i = 1;  i <= 3 ;  i++ ){  //FILAS
   for (int j = 1;  j <= 6 ;  j++ ){   //COLUMNAS
    System.out.print(" & " );
   }
   System.out.println();
}
```



¿Cómo conseguimos esto?



18

El bucle `while`

```
while (condición) {
  // cuerpo del bucle, conjunto de sentencias
}
sentencia1;
```

- Se **ejecuta el cuerpo del bucle mientras se cumple la condición**, es decir, mientras es true/verdadera
- La condición puede ser cualquier expresión booleana.
- El cuerpo del bucle se ejecutará cero o más veces, mientras la expresión condicional sea verdadera.
- Cuando la condición sea falsa, la ejecución pasa a la siguiente línea de código que va inmediatamente después del bucle (sentencia1)

19

El bucle `while`

// Ejemplo: Mostrar los 100 primeros números naturales. Del **0 al 99**

```
int i = 0;

while (i < 100) {
    System.out.println(i);
    i++;
}

System.out.println("FIN");
```

Ejercicio: ¿Cómo mostrar los números del 100 al 1?

20

El bucle do.. while

```
do {
   // cuerpo del bucle, conjunto de sentencias
} while (condición);
```

- **Se ejecuta el cuerpo del bucle mientras se cumple la condición**.
- Este bucle ejecuta siempre al menos una vez el cuerpo, ya que la expresión condicional se evalúa al final del mismo.

El bucle do.. while

- do-while es útil cuando se quiere limitar el valor de una variable leída por teclado.

```
public class Main {
 public static void main(String args[]){
   int numero;
   Scanner teclado = new Scanner(System.in);

   System.out.println("Dame un numero entre 1 y 100: ");
  // leemos un numero mientras sea < 1 o > 100
   do {
       numero = teclado.nextInt();
   } while (numero < 1 || numero > 100);
 }
  System.out.println("El valor es " + numero);
}
```