FOR-EACH

El bucle for en Java, junto con el for-each, son estructuras lógicas que te permiten realizar una y otra vez una acción, siempre y cuando se cumpla una condición booleana

El bucle for-each en Java se utiliza para iterar a lo largo de una lista de objetos o variables primitivas. Es decir que, para el examen de certificación OCAJ8P, se utilizará para objetos de tipo List, ArrayList y para array[], tanto de objetos como de variables primitivas.

En este tipo de bucle no tienes tanto control como en el bucle for que acabamos de ver.

Estructura del bucle for-each

La estructura de este bloque es muy básica. Se compone, de nuevo, de la palabra clave for seguida de un paréntesis. Este contiene tres partes, pero que son más sencillas que las del bucle for anterior.

- Para empezar, aparece el tipo del objeto que se va a extraer, o el tipo de la variable primitiva (en caso de que se trate de un array[]).
- A continuación, el nombre que recibe la variable o referencia local, con la que va a operar el bucle.
- Por último, el nombre de la lista de la que vamos a extraer los componentes.
- No se puede prescindir de ninguna de las tres partes.

Aquí tienes un ejemplo más gráfico de la estructura del for each:

```
public class BucleFor {

  public static void main(String args[]) {

    int[] numbers = { 1, 2, 3 };

    for(int    // Tipo de variable primitiva de la lista que vamos a iterar

        number  // Nombre de la variable local

        : numbers) {  // Nombre de la lista que vamos a iterar

                System.out.print(number + " ");

          }

  }

}
```

// Resultado: 1 2 3

El único símbolo que aparece dentro del paréntesis son los dos puntos (:) que separan el nombre de la variable local del nombre de la lista que vamos a iterar.

Bucles for-each anidados

Recuerda que, si el array tiene más de una dimensión, la variable local será también un array, pero con una dimensión menos. A continuación, te muestro una serie de bucles for-each anidados, partiendo de un array de tres dimensiones. Como ves, conforme vamos avanzando, el array iterado tiene una dimensión menos que su padre. ¿Adivinas por qué?

```
public class BucleFor {
```

```
public static void main(String args[]) {
  int[][][] numbers = {
    { { 1, 2 }, { 3, 4, 8 }, { 5, 6 } },
    { { 7, 8, 9, 11 }, { 9, 10 }, { 11, 12, 13 } }
  };
  for(int[][] number2 : numbers) {
                    for(int[] number1 : number2) {
      for(int number : number1) {
        System.out.print(number + " ");
      }
    }
          }
  }
}
```

// Resultado: 1 2 3

Con el bucle for en Java tienes la posibilidad de controlar el bucle de muchas maneras. Puedes, por ejemplo, controlar cuántas veces puede ejecutarse. O incluso hacer que, si itera una lista, se salte un elemento. Esto se consigue, por ejemplo, modificando el índice de control de manera que con cada bucle se sumen dos unidades, en lugar de una (i += 2). El bucle for-each está creado de manera específica para iterar listas. Viene a ser un bucle for resumido, con una función específica.

Debido a esta limitación, no te rompas la cabeza si necesitas un control más fino del bucle: utiliza un for clásico, y ya está.

Break y continue en un for-each

El uso de break y continue, tanto por sí solos como con etiquetas, es exactamente igual al uso que hemos visto más arriba en un bucle for normal:

```
public class BucleFor {

  public static void main(String args[]) {

    int[][][] numbers = {

      { { 1, 2 }, { 3, 4, 8 }, { 5, 6 } },

      { { 7, 8, 9, 11 }, { 9, 10 }, { 11, 12, 13 } }

    };

    primero:

    for(int[][] number2 : numbers) {

      segundo:

                    for(int[] number1 : number2) {

        if(number[0] == 11) break primero;  // Salida de los bucles.

        for(int number : number1) {

          if(number == 8) continue segundo; // Fuerza que el segundo bucle pase al siguiente elemento.

          System.out.print(number + " ");

        }

      }

          }

  }

}
```

// Resultado: 1 2 3 4 5 6 7 9 10