Programación en Java – Clase 4

Estructuras de control

La clase pasada vimos que las estructuras de control en Java son herramientas fundamentales para controlar el flujo de ejecución de un programa y permiten tomar decisiones y repetir acciones según se requiera. Esta clase trabajaremos sobre las estructuras de control de repetición.

Estructuras de control de repetición

Estas estructuras permiten repetir una acción varias veces mientras se cumpla una determinada condición. Las estructuras de control de repetición en Java son el while, do-while y el for.

Estructura for

La estructura for se utiliza para ejecutar una serie de sentencias de forma repetida un número determinado de veces.

Su sintaxis es la siguiente:

```
for (inicialización; condición; incremento) {
   // Instrucciones a repetir
}
```

Los tres elementos que se utilizan en la estructura for son:

- **inicialización:** En este paso, se inicializa una variable de control que se utilizará en el for. Por lo general, se suele declarar una variable con un valor inicial que se utilizará en el ciclo.

- **condición:** La condición establece una expresión que debe evaluarse en cada iteración del ciclo. Si la condición es verdadera, se ejecutan las instrucciones dentro del ciclo. Si es falsa, se sale del ciclo.

- **incremento:** Este paso se ejecuta después de cada iteración del ciclo. Se utiliza para modificar la variable de control y, de esta forma, modificar la condición del ciclo.

Veamos un ejemplo:

```
for (int i = 1; i <= 5; i++) {
   System.out.println(i);
}
```

En este ejemplo, la variable de control i se inicializa con un valor de 1, y el ciclo se repetirá mientras i sea menor o igual a 5. Después de cada iteración, se incrementa el valor de i en 1.

Es importante tener en cuenta que el for es una estructura de control muy versátil que permite controlar el número de iteraciones y la forma en que se realizan. Se puede utilizar para iterar sobre una colección de datos, para realizar operaciones matemáticas complejas o para cualquier otra tarea que requiera repetición controlada.

Estructura while

La estructura while se utiliza para ejecutar una serie de sentencias mientras se cumpla una determinada condición.

Su sintaxis es la siguiente:

```
while (condición) {
   // Instrucciones a repetir
}
```

La condición del while es una **expresión booleana** que se evalúa en cada iteración del ciclo. Si la condición es verdadera, se ejecutan las acciones dentro del ciclo. Si es falsa, se sale del ciclo y se continúa con la ejecución del programa.

A diferencia del ciclo for, el ciclo while no requiere una inicialización previa de la variable de control, y el incremento o decremento de dicha variable se realiza dentro de las sentencias del ciclo.

Veamos un ejemplo:

```
int i = 1;
while (i <= 5) {
   System.out.println(i);
   i++;
}
```

En este ejemplo, la variable i se inicializa con un valor de 1 y se evalúa la condición i <= 5. Si la condición es verdadera, se imprime el valor de i y se incrementa en 1. El ciclo se repetirá mientras la condición siga siendo verdadera, es decir, mientras i sea menor o igual a 5.

Es importante tener en cuenta que el ciclo while puede ser muy útil en situaciones en las que **no se conoce de antemano cuántas iteraciones se van a realizar,** o cuando se requiere una condición de salida más compleja que la simple iteración de un contador. Sin embargo, es importante asegurarse de que la condición de salida se cumpla en algún momento, para evitar caer en un ciclo infinito.

Estructura do-while

La estructura do-while es similar a la estructura while, con la diferencia de que la condición se evalúa después de ejecutar las sentencias del ciclo. Esto significa que las acciones se ejecutarán al menos una vez, independientemente de si la condición es verdadera o falsa.

Su sintaxis es la siguiente:

```
do {
   // sentencias a repetir
} while (condición);
```

Las acciones dentro del ciclo se ejecutan primero, y luego se evalúa la condición. Si la condición es verdadera, se vuelve a ejecutar el ciclo, y así sucesivamente hasta que la condición sea falsa.

Veamos un ejemplo:

```
int numero;
do {
   System.out.print("Carga un número entre 1 y 10: ");
   numero = scanner.nextInt();
} while (numero < 1 || numero > 10);
```

En este ejemplo, el programa pide al usuario que introduzca un número entre 1 y 10 utilizando un ciclo do-while. Las sentencias dentro del ciclo incluyen una llamada al método nextInt() de la clase Scanner para leer el número introducido por el usuario. Recuerden que vimos esto en la clase 2. La condición de salida del ciclo es numero < 1 || numero > 10, lo que significa que el ciclo seguirá ejecutándose mientras el número introducido sea menor que 1 o mayor que 10.

Es importante tener en cuenta que el ciclo do-while es útil en situaciones en las que se requiere que se ejecute al menos una vez antes de evaluar la condición de salida. Sin embargo, al igual que con cualquier ciclo, es importante asegurarse de que la condición de salida se cumpla en algún momento, para evitar caer en un ciclo infinito.