CODEGYM

Java Developer in 12 months

Mentor-supported training program from CodeGym

MODULE 1. JAVA SYNTAX

Lesson 5 Loops

CODEGYM

Lesson plan

- while loop
- for loop
- Reverse loop (do while)
- break statement
- continue statement
- Loop within a loop

CODEGYM

While loop

A while loop consists of only two parts: a condition and a loop body.

```
while (condition)
  statement;
```

The statement or block of statements is executed over and over again as long as the loop condition equals true.

```
while (condition){
    block of
statements
}
```

First, the condition is checked, then the loop body is executed, and then the condition is checked again and the loop body is executed again. And so on until the condition becomes false.

Code	Explanation

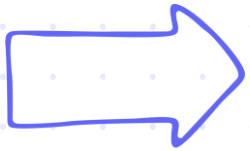
`int n = 5;`
`while(n >0) {`
`  System.out.println(n);`
`  n--;`
`}`
	5 lines will be displayed on the screen:
	5
	4
	3
	2
	1

CODEGYM

For loop

This is another way to write a while loop, just more compact and convenient

```
for (statement1; condition; statement2){
    BLOCK OF STATEMENTS
}
```



```
for (int i = 0; i < 20; i++) {
    System.out.println(i);
}
```

I n a for loop, **statement 1** is executed just once before the loop itself begins.

Statement 2 is executed the same number of times as the body of the loop, and each time it is executed after the entire body of the loop has been executed

CODEGYM

Do-while loop

A **do-while** loop is very similar to the ordinary while loop and also consists of only two parts: a "**condition**" and a "**loop body**". The loop body is executed over and over again as long as the condition is **true**.

```
do {
   BLOCK
OF STATEMENTS
} while
(condition);
```

while	do while

`String s = console.nextLine();`
`while (!"exit".equals(s)) {`
`    s = console.nextLine();`
`}`
	`String s;`
	`do {`
	`    s = console.nextLine();`
	`}`
	`while (!"exit".equals(s));`

CODEGYM

Break statement

If a break statement is executed inside a loop, then the loop ends immediately. The program will start executing the statement that follows the loop.

Code	Explanation

`Scanner console = new Scanner(System.in);`
`while (true) {`
`  String s = console.nextLine();`
`  if ("exit".equals(s)) {`
`  break;`
`  }`
`}`
	The program will read a line from the keyboard until you enter "exit".

Continue statement

Executing the loop body once is called an iteration of the loop. The **continue** statement interrupts the current iteration of the loop, but unlike the break statement, it doesn't end the loop itself.

The **continue** statement is convenient to use in a loop if you need to "skip" execution of the loop body

Code	Explanation

`int i = 1;`
`while (i <= 20) {`
`  if ((i % 7) == 0) {`
`  continue;`
`  }`
`  System.out.println(i);`
`  i++;`
`}`
	The program displays numbers from 1 to 20. If the number is divisible by 7 (the remainder of division by 7 is 0), then we skip displaying the number.

Actually, this code will not work, because i will be forever stuck at the number 7. After all, the continue statement skips two other statements: **`System.out.println(i)`** and **`i++.`**

As a result, once we reach the value 7, the variable i will stop changing and we'll be in an infinite loop.

CODEGYM

Loop within a loop

To write a loop within a loop, you need to write the second loop inside the body of the first loop.

```
while (condition for outer loop) {
    while (condition for inner loop) {
        block of statements
    }
}
```

Code	Explanation

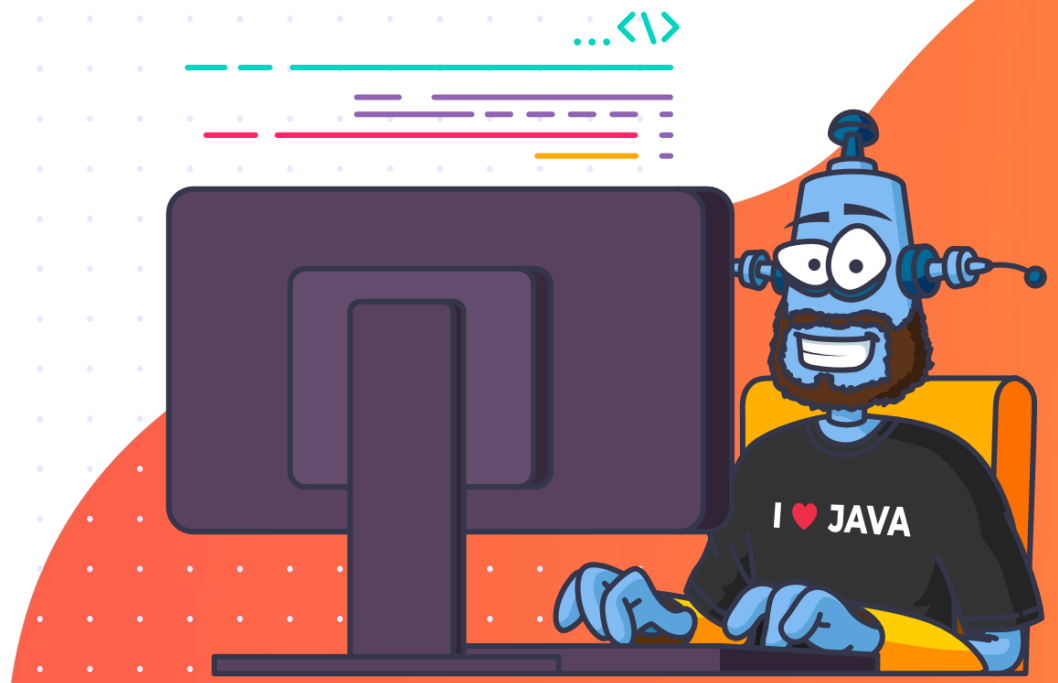
`int n = 0;`
`while (n < 4) {`
`  int m = 0;`
`  while (m < 5) {`
`  System.out.print("A");`
`  m++;`
`  }`
`  System.out.println();`
`  n++;`
`}`
	The ***outer loop*** is purple. It uses the **n** variable to count the number of iterations of the loop.
	
	The ***inner loop*** is green. It uses the **m** variable to count the number of loop iterations.
	
	We have to explicitly **move the cursor** to the next line after the inner loop is complete. Otherwise, all the letters that the program prints will end up on one line.
	
	The screen output will be:
	AAAAA
	AAAAA
	AAAAA
	AAAAA

CODEGYM

Homework

MODULE 1. JAVA SYNTAX

Complete Level 6



CODEGYM

Answers to questions

MODULE 1. JAVA SYNTAX