Java Loops & Jump Statements | for, while, do-while, break, continue Explained

made by vijay singh

In Java, loops help us repeat a block of code multiple times.
Jump statements help us control the flow inside loops.

We will cover:

- for loop
- while loop
- do-while loop
- break statement
- continue statement

1. for Loop

A for loop is used when we know **how many times** we want to repeat something.

Syntax

```
for(initialization; condition; update) {
   // code
}
```

How It Works

1. Initialization runs once.
2. Condition is checked.
3. If true → code runs.
4. Update runs.
5. Again condition is checked.

Example 1: Print numbers from 1 to 5

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

Output

```
1
2
3
```

4
5

Example 2: Print even numbers from 2 to 10

```
for(int i = 2; i <= 10; i = i + 2) {
   System.out.println(i);
}
```

2. while Loop

A while loop is used when we don't know exactly how many times the loop should run.
It runs **until the condition becomes false**.

Syntax

```
while(condition) {
   // code
}
```

Example 1: Print numbers from 1 to 5

```
int i = 1;

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

Example 2: Print table of 3

```
int i = 1;

while(i <= 10) {
   System.out.println(3 * i);
   i++;
}
```

3. do-while Loop

A do-while loop executes the code **at least once**, even if the condition is false.

Because condition is checked **after** execution.

Syntax

```
do {
   // code
} while(condition);
```

Example 1

```
int i = 1;

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

Example 2 (Important Concept)

```
int i = 10;

do {
   System.out.println("Hello");
} while(i < 5);
```

Output:

Hello

Even though condition is false, it runs once.

Loop	Condition Checked	Minimum Execution
for	Before execution	0 times
while	Before execution	0 times
do-while	After execution	1 time

4. break Statement

break is used to **stop the loop immediately**.

Example 1: Stop at 5

```
for(int i = 1; i <= 10; i++) {

   if(i == 5) {
```

```
    break;
  }

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

Output

```
1
2
3
4
```

Loop stops when i == 5.

Example 2: Break in while loop

```
int i = 1;

while(i <= 10) {

  if(i == 7) {
    break;
  }

  System.out.println(i);
  i++;
}
```

5. continue Statement

continue skips the current iteration and moves to the next iteration.

Example 1: Skip number 5

```
for(int i = 1; i <= 10; i++) {

  if(i == 5) {
    continue;
  }

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

Output

```
1
2
3
4
6
7
8
9
10
```

Example 2: Print only even numbers

```
for(int i = 1; i <= 10; i++) {

   if(i % 2 != 0) {
      continue;
   }

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

Feature	break	continue
Stops loop completely?	Yes	No
Skips current iteration?	No	Yes
Works in loops?	Yes	Yes
Works in switch?	Yes	No

When To Use What?

Use for → when count is fixed
Use while → when condition-based repetition
Use do-while → when code must run at least once
Use break → when you want to exit early
Use continue → when you want to skip specific cases