Java Loops - Comprehensive Notes

Java Loops - Comprehensive Notes

Loops in Java are used to execute a block of code repeatedly as long as a given condition is true. Java provides several loop constructs:

- for loop
- while loop
- do-while loop
- for-each loop (enhanced for loop)
- Nested loops (any loop inside another loop)

1. for Loop

Syntax:
```
for (initialization; condition; update) {
    // block of code
}
```

Terms Explained:
- initialization: Executes once before the loop starts. Usually used to initialize a loop control variable.
- condition: Boolean expression checked before each iteration. If true, the loop continues; if false, it stops.
- update: Updates the loop control variable, executed after each iteration.

Example:
```
for (int i = 1; i <= 5; i++) {
    System.out.println("i = " + i);
}
```
Output:
```
i = 1
i = 2
i = 3
i = 4
i = 5
```

2. Nested for Loop

Use Case: Often used for printing patterns or working with multi-dimensional data.

Example:
```
for (int i = 1; i <= 3; i++) {
    for (int j = 1; j <= 2; j++) {
        System.out.println("i = " + i + ", j = " + j);
    }
}
```
Output:
```
i = 1, j = 1
i = 1, j = 2
```

```
i = 2, j = 1
i = 2, j = 2
i = 3, j = 1
i = 3, j = 2
```

3. while Loop

Syntax:

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

Explanation:
- The condition is checked before the loop executes. If it's false initially, the loop may never run.

Example:

```
int i = 1;
while (i <= 3) {
    System.out.println("i = " + i);
    i++;
}
```

Output:

```
i = 1
i = 2
i = 3
```

4. do-while Loop

Syntax:

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

Explanation:
- Executes the block at least once, even if the condition is false initially.

Example:

```
int i = 1;
do {
    System.out.println("i = " + i);
    i++;
} while (i <= 3);
```

Output:

```
i = 1
i = 2
i = 3
```

Java Loops - Comprehensive Notes

5. for-each Loop (Enhanced for loop)

Use Case: Best for iterating over arrays or collections when index is not required.

Syntax:

```
for (type element : array) {
    // block of code
}
```

Example:

```
int[] numbers = {10, 20, 30};
for (int num : numbers) {
    System.out.println("num = " + num);
}
```

Output:

```
num = 10
num = 20
num = 30
```

Summary Table

Loop Type	Entry Condition	Guaranteed Execution	Best Use Case
for	Yes	No	Known number of iterations
Nested for	Yes	No	Patterns, matrices
while	Yes	No	Unknown iterations, sentinel loops
do-while	No	Yes	At least one iteration needed
for-each	Yes	No	Arrays or collections

End of Notes.