Day 5

Loops and Jumping Statements

- **Loops/Looping statements** in Java are used to **repeatedly execute a block of code** as long as a certain condition is true.
- This allows you to perform repetitive tasks without writing the same code multiple times.
- For example, if you want to print numbers from 1 to 10, instead of writing 10 separate print statements, you can use a loop to do it in just a few lines of code.
- Loops help in automating tasks, making code more efficient and easier to manage.

Java supports 3 types of looping statements:

1. while loop
2. do-while loop
3. for loop

1) While Loop

The while loop repeatedly executes a block of code as long as the given condition is true. If the condition is false from the start, the loop will not execute at all.

Syntax:

```
while (condition) {
    // code to be executed
}
```

Example:

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

2) Do-While Loop

The do-while loop is similar to the while loop, but the main difference is that the code block inside the loop will execute at least once, even if the condition is false at the start.

Syntax:

```
do {
    // code to be executed
} while (condition);
```

Example:

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

3) For Loop

The for loop is typically used when the number of iterations is known beforehand. It is more compact and includes initialization, condition checking, and increment/decrement in a single line.

Syntax:

```
for (initialization; condition; increment/decrement) {
    // code to be executed
}
```

Example:

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

Differences Between While, Do-While, and For Loops

1. **Execution at least once:**
    - **While Loop:** The condition is checked first. If it's false, the loop may not execute at all.
    - **Do-While Loop:** The loop executes at least once because the condition is checked after the code block is executed.

- **For Loop:** The loop may not execute if the condition is false initially, similar to the while loop.

2. **Use Case:**
   - **While Loop:** Best used when the number of iterations is not known in advance, and the loop might not execute at all.
   - **Do-While Loop:** Best used when you want the loop to execute at least once regardless of the condition.
   - **For Loop:** Best used when the number of iterations is known beforehand, making it compact and easy to read.

Jumping Statements:

break Statement:

The break statement is used to terminate the loop or switch statement immediately when it's encountered. The control is transferred to the first statement after the loop or switch.

Example:

```
public class BreakExample {
    public static void main(String[] args) {
        for (int i = 1; i <= 10; i++) {
            if (i == 5) {
                break; // Terminates the loop when i is 5
            }
            System.out.println(i);
        }
    }
}
```

Output:

```
1
2
3
4
```

The loop terminates when i equals 5, so numbers 5 to 10 are not printed.

continue Statement:

The continue statement skips the current iteration of the loop and proceeds with the next iteration. It's typically used to skip certain conditions inside a loop.

Example:

```
public class ContinueExample {
    public static void main(String[] args) {
        for (int i = 1; i <= 10; i++) {
            if (i == 5) {
                continue; // Skips the iteration when i is 5
            }
            System.out.println(i);
        }
    }
}
```

Output:

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

The number 5 is skipped, and the loop continues printing numbers from 6 to 10.

Lab Assignments

While loop

1. Write a Java program to print numbers from 1 to 10 using a while loop.
2. Write a Java program to calculate the sum of the first 10 natural numbers using a while loop.
3. Write a Java program to calculate the factorial of a given number using a while loop.
4. Write a Java program to reverse a given number using a while loop.
5. Write a Java program to count the number of digits in a given number using a while loop.
6. Write a Java program to check if a given number is a prime number using a while loop.
7. Write a Java program to print all even numbers between 1 and 20 using a while loop.
8. Write a Java program to calculate the sum of digits of a given number using a while loop.
9. Write a Java program to find the largest digit in a given number using a while loop.
10. Write a Java program to generate the first 10 numbers of the Fibonacci series using a while loop.
11. Write a Java program to check if a given number is a palindrome using a while loop.
12. Write a Java program to calculate the sum of all even numbers from 1 to N using a while loop.
13. Write a Java program to calculate the sum of all odd numbers from 1 to N using a while loop.

Do-while loop

1. Write a Java program to print numbers from 1 to 10 using a do-while loop.
2. Write a Java program to calculate the sum of digits of a given number using a do-while loop.
3. Write a Java program to reverse a given number using a do-while loop.
4. Write a Java program to check if a given number is a palindrome using a do-while loop.
5. Write a Java program that performs basic arithmetic operations (addition, subtraction, multiplication, and division) using a do-while loop until the user chooses to exit.

For loop

1. Print Numbers 1 to 10
2. Print Even Numbers from 1 to 20
3. Print Odd Numbers from 1 to 20
4. Print Multiples of 5 from 5 to 50
5. Calculate Sum of Numbers from 1 to 10
6. Print Squares of Numbers from 1 to 5
7. Print Cube of Numbers from 1 to 5
8. Print Numbers in Reverse Order from 10 to 1

9. Print Prime Numbers between 1 and 50

10. Print Sum of Even Numbers between 1 and 20

11. Print Sum of Odd Numbers between 1 and 20

12. Print Table of 7

13. Print Numbers Divisible by 3 and 5 from 1 to 100

14. Count Number of Digits in a Number

15. Print Multiples of 7 between 1 and 100

Continue

1. Write a program to print the odd numbers from 1 to 20 using a for loop. Use the continue statement to skip even numbers.

2. Write a program to print numbers from 1 to 30, but skip numbers that are multiples of 5. Use the continue statement within a while loop.

3. Write a program to print numbers from 1 to 50. Skip numbers that are divisible by both 3 and 4 using the continue statement within a for loop.

Break

1. Write a program to find and print the first even number between 1 and 20 using a for loop. Use the break statement to exit the loop as soon as you find the first even number.

2. Write a program to print numbers from 1 to 50, but stop and exit the loop as soon as you encounter a multiple of 7. Use the break statement within a while loop.

3. Write a program to print numbers from 1 to 30. Stop printing and exit the loop when you find a number greater than 15. Use the break statement within a for loop.