NESTED IF STATEMENTS

Nested if statements can be useful for implementing complex conditional logic and handling multiple conditions in a structured manner.

Here's an example of a nested if statement with conditions

```
 public class SimpleIf {
    public static void main(String[] args) {
        int x = 10;
        int y = 5;

        if (x > y) {
            System.out.println("x is greater than y");

            if (x > 0) {
                System.out.println("x is positive");
            }
        }
    }
}
```

In this example, the outer if statement checks if x is greater than y. If the condition is true, the inner if statement checks if x is greater than 0. If both conditions are true, the corresponding messages will be printed.

NESTED IF-ELSE STATEMENTS

Nested if-else statements are permitted in Java. It is perfectly valid to have an if-else statement nested inside another if statement or else block.

Example 1

```
public class SimpleIf {
    public static void main(String[] args) {
        int x = 10;
        int y = 5;

        if (x > y) {
            System.out.println("x is greater than y.");
            if (x % 2 == 0) {
                System.out.println("x is even.");
            } else {
                System.out.println("x is odd.");
            }
        } else {
            System.out.println("x is not greater than y.");
        }
    }
}
```

In this example, we have a nested if-else structure. The outer if statement checks if x is greater than y. If the condition is true, the program enters the block of the outer if. Inside this block, there is another if-else structure that checks if x is even or odd. If x is even, it prints "x is even." If x is odd, it prints "x is odd." If the condition of the outer if is false, the program skips the block of the outer if and directly executes the statements in the else part.

Example 2

```
public class SimpleIf {
    public static void main(String[] args) {
        boolean condition1 = true;
        boolean condition2 = false;

        if (condition1) {
            System.out.println("Condition 1 is true.");
        } else {
            if (condition2) {
                System.out.println("Condition 1 is false and Condition 2 is true.");
            } else {
                System.out.println("Both Condition 1 and Condition 2 are false.");
            }
        }
    }
}
```

In this example, we have a nested if-else structure. The outer if-else statement checks condition1. If condition1 is true, it prints "Condition 1 is true."

If condition1 is false, the program enters the block of the else part. Inside this block, there is another if-else structure that checks condition2. If condition2 is true, it prints "Condition 1 is false and Condition 2 is true."

If both condition1 and condition2 are false, the program enters the block of the inner else part and prints "Both Condition 1 and Condition 2 are false."

NESTED WHILE LOOP

Nested while loops are permitted in Java. You can have a while loop inside another while loop.

```
public class NestedWhileExample {
    public static void main(String[] args) {
        int outerCount = 1;
        while (outerCount <= 3) {
            System.out.println("Outer loop iteration: " + outerCount);

            int innerCount = 1;
            while (innerCount <= 3) {
                System.out.println("Inner loop iteration: " + innerCount);
                innerCount++;
            }

            outerCount++;
        }
    }
}
```

In this example, we have a nested while loop structure. The outer while loop iterates as long as outerCount is less than or equal to 3. Inside the outer loop, there is an inner while loop that iterates as long as innerCount is less than or equal to 3.

During each iteration of the outer loop, the program prints the outer loop iteration number. Then, it enters the inner loop and prints the inner loop iteration number. After the inner loop completes, the outer loop continues to the next iteration.

NESTED WHILE LOOP

Nested do-while loops are permitted in Java. You can have a do-while loop inside another do-while loop

```
public class NestedDoWhileExample {
    public static void main(String[] args) {
        int outerCount = 1;
        do {
            System.out.println("Outer loop iteration: " + outerCount);

            int innerCount = 1;
            do {
                System.out.println("Inner loop iteration: " + innerCount);
                innerCount++;
            } while (innerCount <= 3);

            outerCount++;
        } while (outerCount <= 3);
    }
}
```

In this example, we have a nested do-while loop structure. The outer do-while loop iterates as long as outerCount is less than or equal to 3. Inside the outer loop, there is an inner do-while loop that iterates as long as innerCount is less than or equal to 3.

During each iteration of the outer loop, the program prints the outer loop iteration number. Then, it enters the inner loop and prints the inner loop iteration number. After the inner loop completes, the outer loop continues to the next iteration.