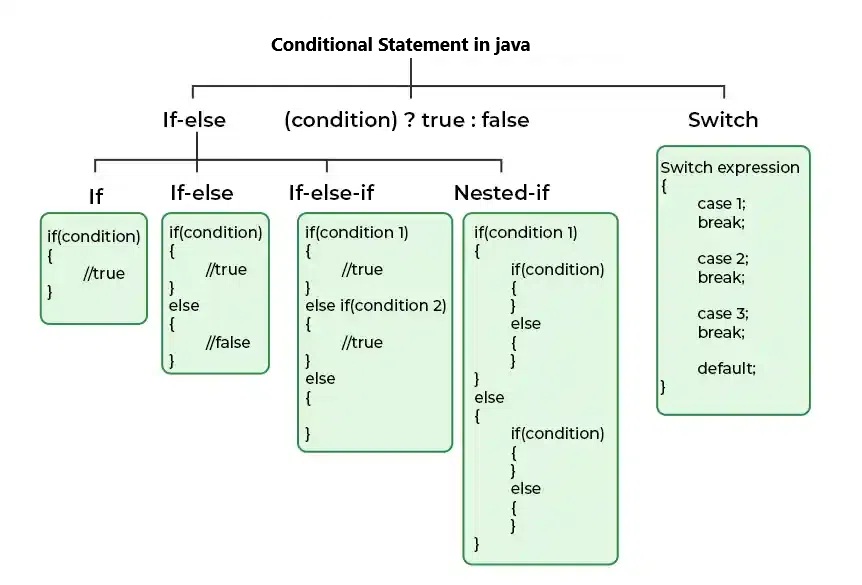
Day 3:🔑 Control Structures: Directing the Flow in Java



Control structures are the foundation of decision-making in programming. They enable conditional execution of code blocks based on certain conditions or logic.

1️⃣ `if` Condition

To create an `if` condition, use the `if` keyword:

if (condition) {

// statement

}

Key Points:

- Requires a boolean (`true` or `false`) condition to execute.

- The code inside the `if` block executes only if the condition is true.

- Logical or relational operators can be used to make the condition.

2️⃣ `if-else` Condition

To create an `if-else` condition, use the `if` and `else` keywords:

if (condition) {

// statement

} else {

// statement

}

Key Points:

- Executes the `if` block when the condition is true.

- Executes the `else` block when the condition is false.

3️⃣ `if-else-if` Condition

This is a combination of `if` and `else` conditions:

if (condition1) {

// statement

} else if (condition2) {

// statement

} else {

// statement

}

Key Points:

- Evaluates conditions from top to bottom.

- Executes the first block where the condition is true.

- If none of the conditions match, the `else` block (optional) is executed.

- Known as the "ladder `if` condition" for its sequential checking.

- Suitable for handling multiple conditions.

4️⃣ Nested `if` Condition

Layers of conditions create a nested structure:

if (condition1) {

if (condition2) {

// statement

} else {

// statement

}

} else {

// statement

}

Key Points:

- Useful for comparisons that depend on other conditions.

- Helps achieve finer decision control, but excessive nesting can affect readability.

5️⃣ `switch` Control Structure

To create a `switch` condition, use the `switch`, `case`, `break`, and `default` keywords:

switch (operand) {

case value1:

// statement

break;

case value2:

// statement

break;

default:

// statement

}

Key Points:

- The `switch` condition requires a single operand to compare.

- Each case contains a specific value and executes the corresponding block if it matches.

- The `default` block executes when no cases match.

- The `break` statement prevents fall-through to subsequent cases.

- Considered the \*\*fastest control structure\*\* as it directly compares the operand to the cases.

NOTE: When using `switch`, if a case is satisfied and the `break` is not used, subsequent cases execute implicitly. Always include `break` unless a fall-through behavior is desired.

Control structures streamline your program’s logic, making your code efficient and easier to understand. Start practicing these concepts to become proficient in Java programming! 🚀