Java Basic Program

Module 1
1. Introduction part, Languages Programing, Java history, JDK, “hello World”
2. Project, package, Class, method
3. Variables. Keyboard input
4. Variables and data types
5. Consultation

Module 2
1. TEST #1 and second part Compilation and constructor
2. Methods and Random ways
3. Practice and examples with methods
4. boolean. Boolean expressions
5. Consultation

Module 3
1. if-else-if
2. Switch, ternary operator
3. Loops, for
4. Loops, while, do while
5. Consultation

Module 4
1. Arrays in Java
2. Arrays search and sort
3. String, StringBuilder, StringBuffer, practice
4. TEST #2 and second part Class and Object
5. Consultation

Module 5 (Optional)
1. Method main() for the test and introduction to JUnit testing
2. Practice, repetitions, console Lottery game
3. Practice, implementation of the distribution of cards in Poker
4. Summarizing and Introduction to the professional course program and the profession Back-end developer in Java
5. Consultation (questions)

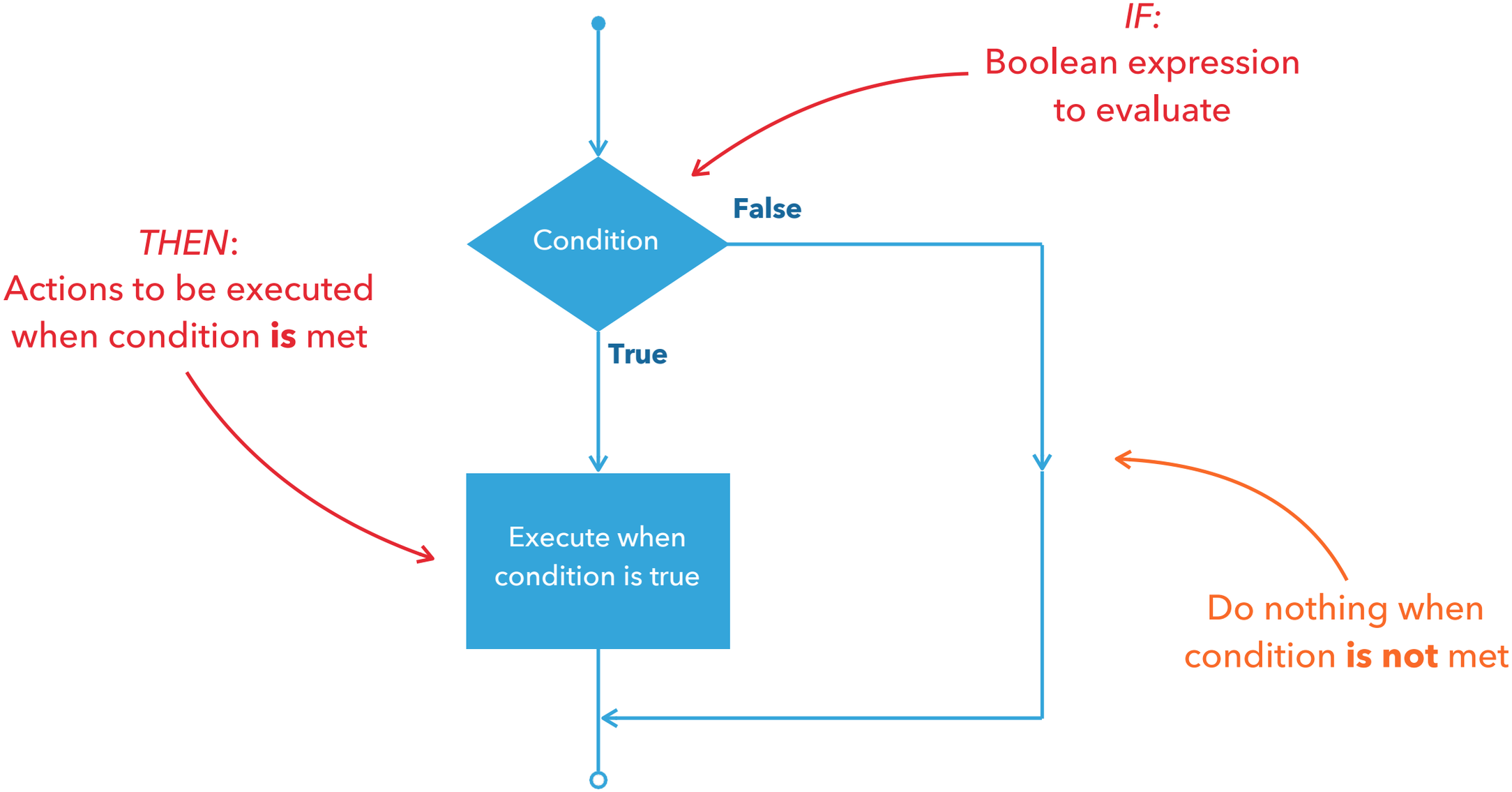
CONDITIONAL

FLOW CONTROL

CONDITIONAL STATEMENTS

- Control code execution by specifying certain conditions
  - When conditional statement is **met** (equals to 'true')
  - When conditional statement is **not met** (equals to 'false')
- There are two main conditional statements:
  - **If** statement
  - **Switch** statement

DECISION MAKING FLOWCHART: IF



IF STATEMENT: SYNTAX

Keyword specifying conditional statement

Variable or expression with boolean result

```
if (statement) {
    //Code to execute
    //When statement is true
}
```

IF STATEMENT: EXAMPLE

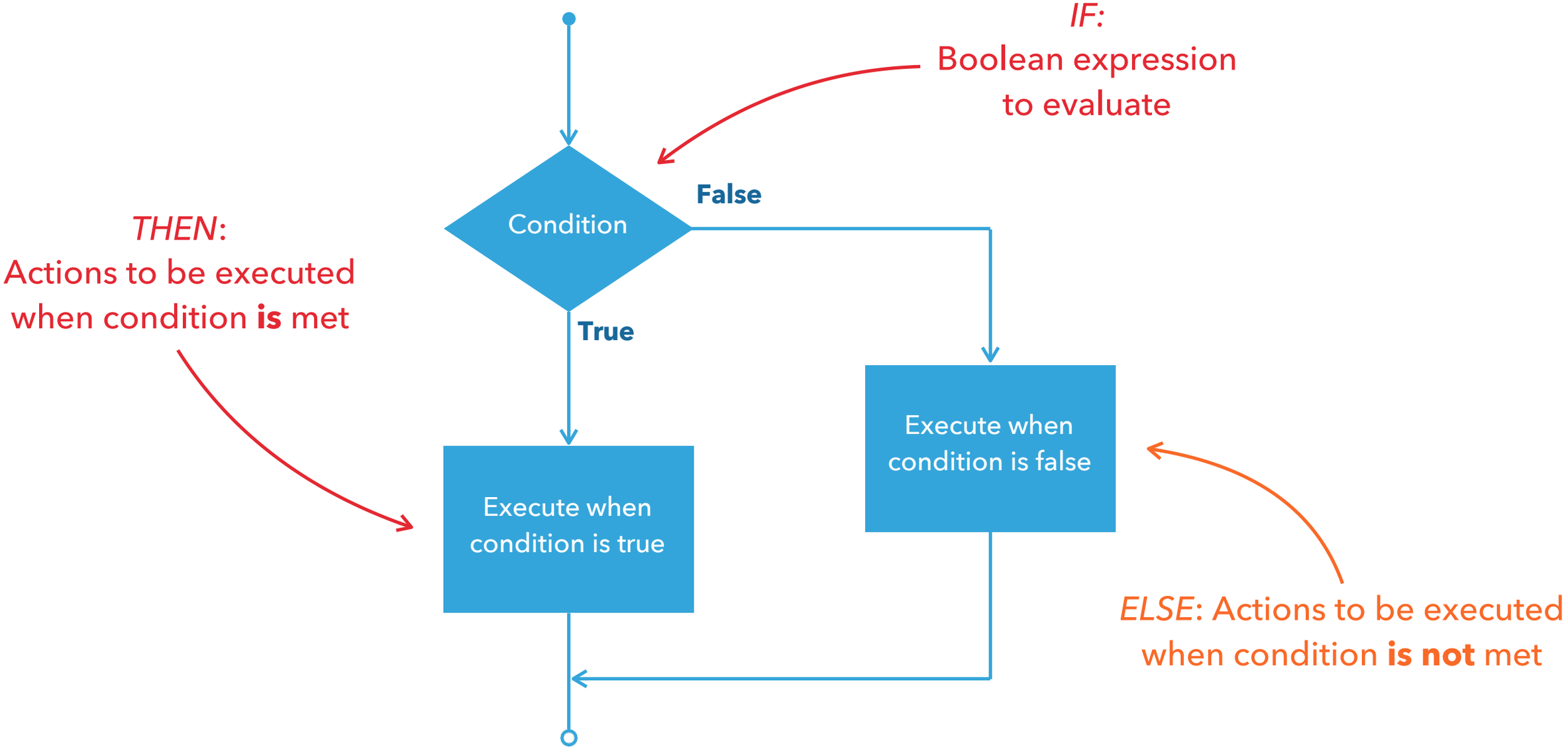
Boolean variable expression

```
boolean flag = true;

if (flag) {
    System.out.print("True");
}
```

Inline expression

```
int x = 5;

if (x > 10) {
    System.out.print("x > 10");
}
```

DECISION MAKING FLOWCHART: IF – ELSE



IF – ELSE STATEMENT: SYNTAX

Keyword specifying conditional statement

Variable or expression with boolean result

```
if (statement) {
    //Code to execute
    //When statement is true
} else {
    //Code to execute
    //When statement is false
}
```

Keyword specifying alternative code block

IF – ELSE STATEMENT: EXAMPLE

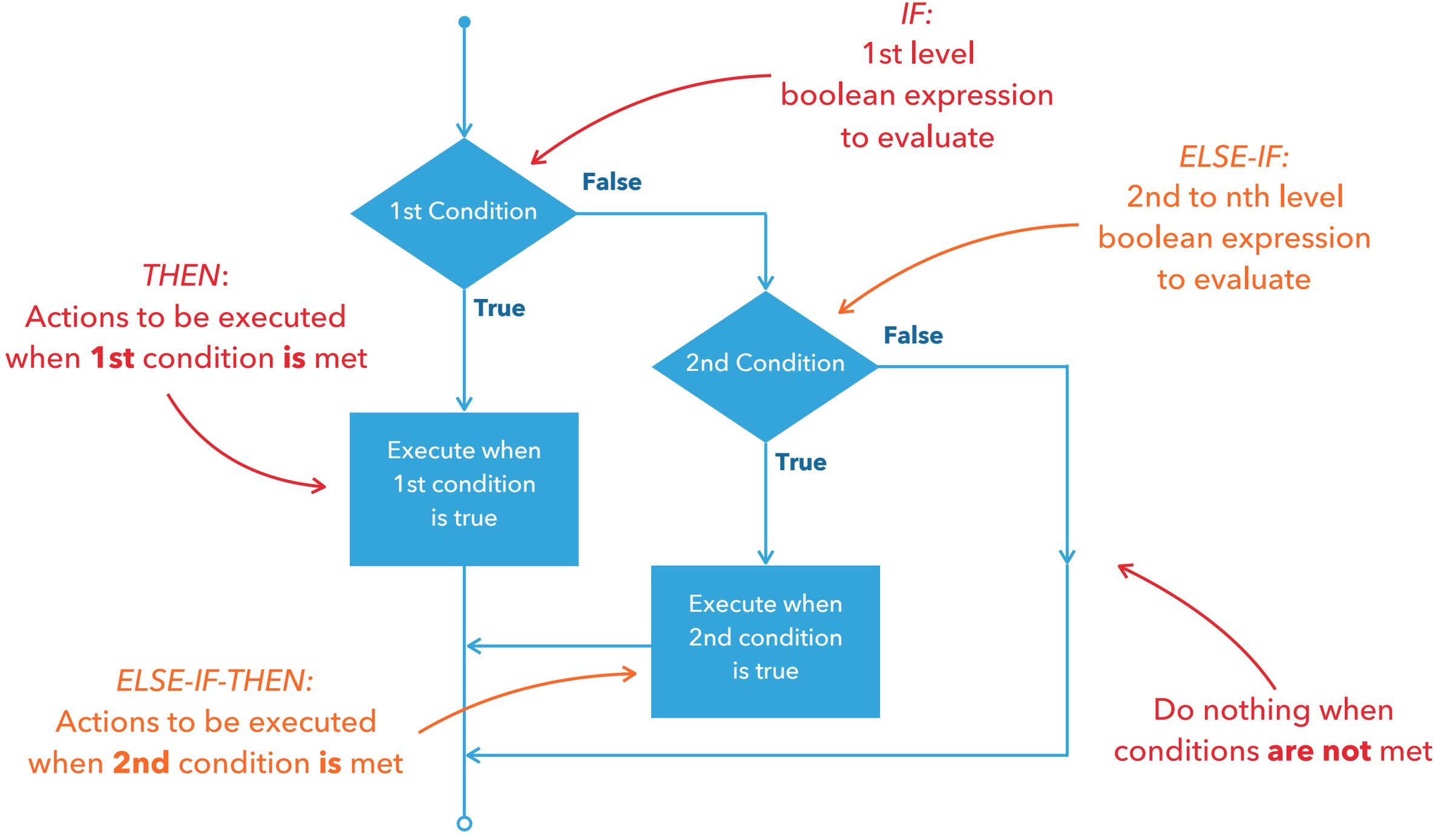
Boolean variable expression

```java
boolean flag = false;

if (flag) {
    System.out.print("True");
} else {
    System.out.print("False");
}
```

Inline expression

```java
int x = 5;

if (x > 10) {
    System.out.print("x > 10");
} else {
    System.out.print("x =< 10");
}
```

DECISION MAKING FLOWCHART: IF – ELSE IF



IF – ELSE IF STATEMENT: SYNTAX

Keyword specifying conditional statement

Variable or expression with boolean result

```
if (statement1) {
    //Code to execute
    //When statement1 is true
} else if (statement2) {
    //Code to execute
    //When statement2 is true
}
```

Keyword specifying alternative conditional code block

IF – ELSE IF STATEMENT: EXAMPLE

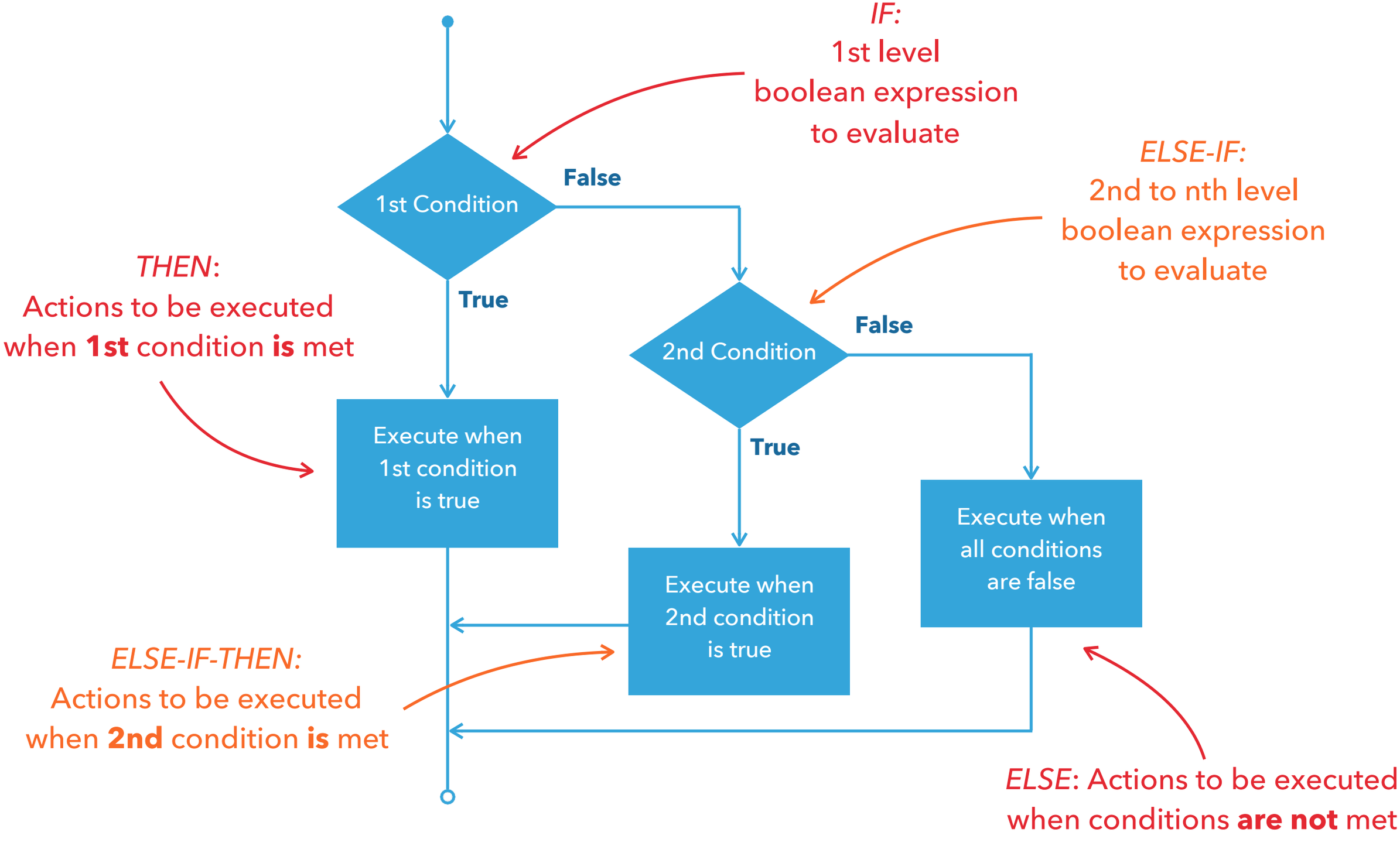
Boolean variable expression

```
boolean flag1 = false;
boolean flag2 = true;

if (flag1) {
    System.out.print("flag1");
} else if (flag2) {
    System.out.print("flag2");
}
```

Inline expression

```
int x = 7;

if (x == 3) {
    System.out.print("x == 3");
} else if (x == 7) {
    System.out.print("x == 7");
}
```

DECISION MAKING FLOWCHART: IF – ELSE IF – ELSE



IF – ELSE IF – ELSE STATEMENT: SYNTAX

Keyword specifying conditional statement

Variable or expression with boolean result

```
if (statement1) {
    //Code to execute
    //When statement1 is true
} else if (statement2) {
    //Code to execute
    //When statement2 is true
} else {
    //Code to execute
    //When all statements are false
}
```

Keyword specifying alternative conditional code block

Keyword specifying alternative code block

IF – ELSE IF – ELSE STATEMENT: EXAMPLE

Boolean variable expression

```
boolean flag1 = false;
boolean flag2 = false;

if (flag1) {
    System.out.print("flag1");
} else if (flag2) {
    System.out.print("flag2");
} else {
    System.out.println("none");
}
```

Inline expression

```
int x = 7;

if (x == 3) {
    System.out.print("x == 3");
} else if (x == 7) {
    System.out.print("x == 7");
} else {
    System.out.print("NOTA");
}
```

IF – ELSE IF – ELSE STATEMENT RULES RECAP

- An if can have **zero** or **one** else's and its must come after any else if's
- An if can have **zero** to **many** else if's and they must come **before** else
- Once an else if **succeeds**, **none** of the **remaining** else if's or else's will be tested

SWITCH STATEMENT OVERVIEW

- Provides an **effective** way to deal with a section of code that could branch in **multiple directions** based on **single variable**
- **Doesn't** support the conditional operators that the **if statement** does
- **Can't** handle **multiple** variables

SWITCH STATEMENT: SYNTAX

Keyword specifying switch statement

Single variable to test against

```
switch (singleVariable) {
    case value1:
        //Code to execute
        //When value1 matches
        break;

    case value2:
        //Code to execute
        //When value2 matches
        //Without break it
        //Falls through cases
        //Until first break

    default:
        //Code to execute
        //When none of the above
        //Values match
}
```

Variable must match one of these (else - if)

Keyword telling to stop switch execution right here

Fallback case (else)

SWITCH STATEMENT: EXAMPLE

```
String drink = "coffee";

switch (drink) {
    case "coffee":
        System.out.println("I would go for Java!");
        break;

    case "tea":
        System.out.println("Everything but Lipton");
        break;

    default:
        System.out.println("Ugh.. What?");
}
```

BUILDING BOOLEAN EXPRESSIONS

THE EQUALITY AND RELATIONAL OPERATORS

Operator	Operation
==	Equal to
!=	Not equal to
>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to

CONDITIONAL OPERATORS

Operator	Operation
&&	Conditional AND
||	Conditional OR
!	Conditional NOT

COMPLEX BOOLEAN STATEMENT EXAMPLE

Make sure that BOTH statements are true

Check if x is greater than 5

Check if x is lesser than 15

```
int x = 10;

if ((x > 5) && (x < 15)) {
    System.out.print("Within bounds!");
}
```