In Java, operators are special symbols or keywords that are used to perform operations on variables and values.

## Types of operators

### 1. Arithmetic Operators

These operators are used to perform mathematical operations like addition, subtraction, multiplication, division, and modulus.

| Operator | Description | Example |
|---|---|---|
| + | Addition | a + b |
| - | Subtraction | a – b |
| * | Multiplication | a * b |
| / | Division | a / b |
| % | Modulus (Remainder) | a % b |

**Example:**

```
1
2
3  public class OperOperation {
4
5      public static void main(String args[]){
6  //          Arithmetic Operators
7          int a = 11, b = 5;
8          System.out.println(a + b);  // Output: 16
9          System.out.println(a - b);  // Output: 6
10         System.out.println(a * b);  // Output: 55
11         System.out.println(a / b);  // Output: 2
12         System.out.println(a % b);  // Output: 1
13
14     }
15 }
16
```

### 2. Relational (Comparison) Operators

These operators are used to compare two values and return a boolean result (true or false).

| Operator | Description | Example |
|---|---|---|
| == | Equal to | a == b |

| Operator | Description | Example |
|---|---|---|
| != | Not equal to | a != b |
| > | Greater than | a > b |
| < | Less than | a < b |
| >= | Greater than or equal to | a >= b |
| <= | Less than or equal to | a <= b |

**Example:**

```java
public class OperOperation {

    public static void main(String args[]){
        int a=10;
        int b=7;
        //Relational (Comparison) Operators
        System.out.println( a == b);//Output : false
        System.out.println( a <= b);//Output : false
        System.out.println( a >= b);//Output : true
        System.out.println( a != b);//Output : true
        System.out.println( a >  b);//Output : true
        System.out.println( a <  b);//Output : false

    }
}
```

**3. Logical Operators**

Logical operators are used to combine multiple Boolean expressions or values and return a Boolean result.

| Operator | Description | Example |
|---|---|---|
| && | Logical AND (true if both operands are true) | a && b |
| `\|\| | Logical OR | `a\|\|b |
| ! | Logical NOT (inverts the boolean value) | !a |

**Example:**

```java
public class OperOperation {

    public static void main(String args[]){

        boolean a = true, b = false;
        System.out.println(a && b);  // Output: false
        System.out.println(a || b);  // Output: true
        System.out.println(!a);      // Output: false


    }
}
```

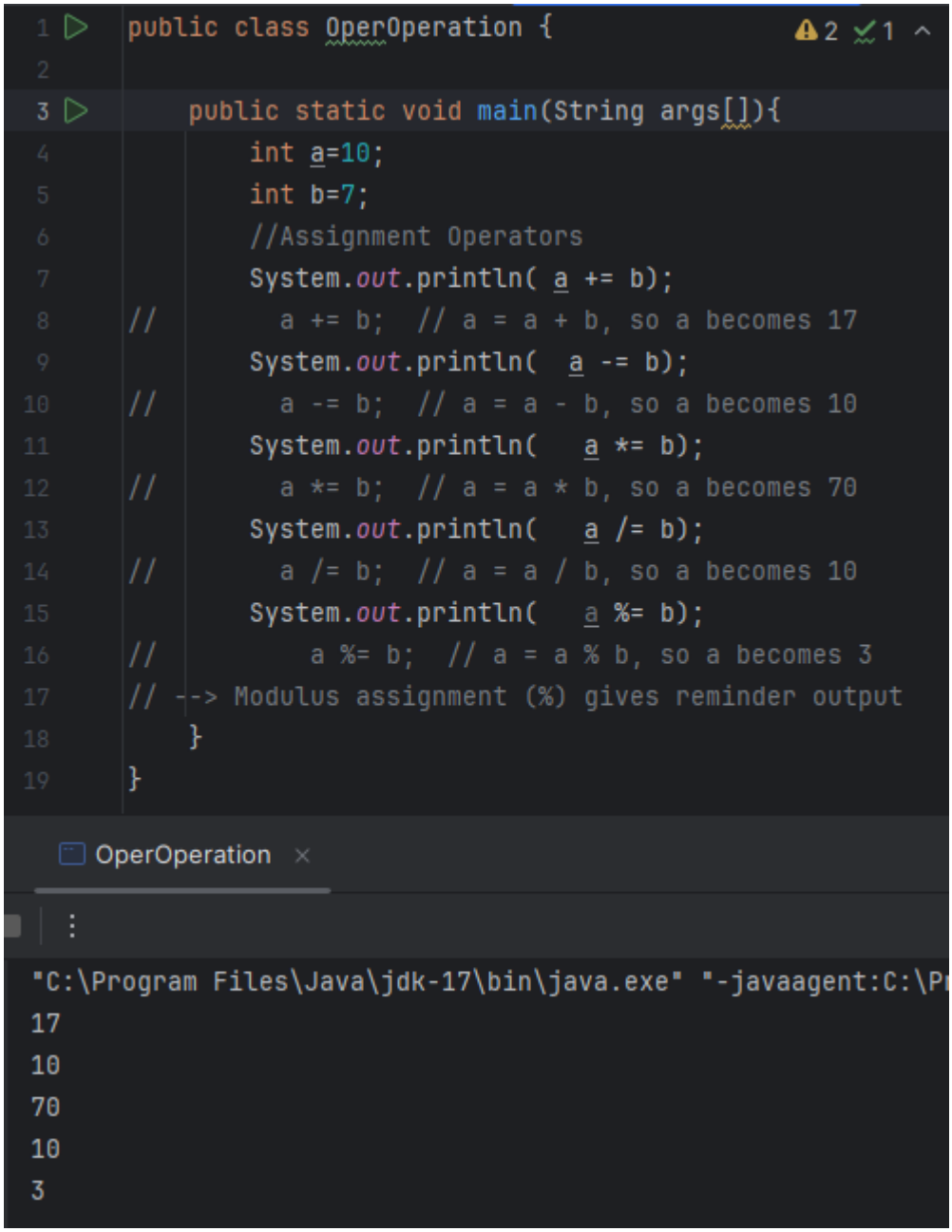
**4. Assignment Operators**

Assignment operators are used to assign values to variables.

| Operator | Description | Example |
|---|---|---|
| = | Simple assignment | a = 5 |
| += | Addition assignment | a += 5 |
| -= | Subtraction assignment | a -= 5 |
| *= | Multiplication assignment | a *= 5 |
| /= | Division assignment | a /= 5 |
| %= | Modulus assignment | a %= 5 |



**Example:**




**5. Unary Operators**

Unary operators operate on a single operand. They are used to perform operations such as incrementing or decrementing the value of a variable.

| Operator | Description | Example |
|---|---|---|
| + | Unary plus (indicates a positive value) | +a |
| - | Unary minus (negates the value) | -a |
| ++ | Increment (increase by 1) | a++ or ++a |
| -- | Decrement (decrease by 1) | a-- or --a |

| Operator | Description | Example |
|---|---|---|
| ! | Logical NOT (negates boolean value) | !a |

Example:

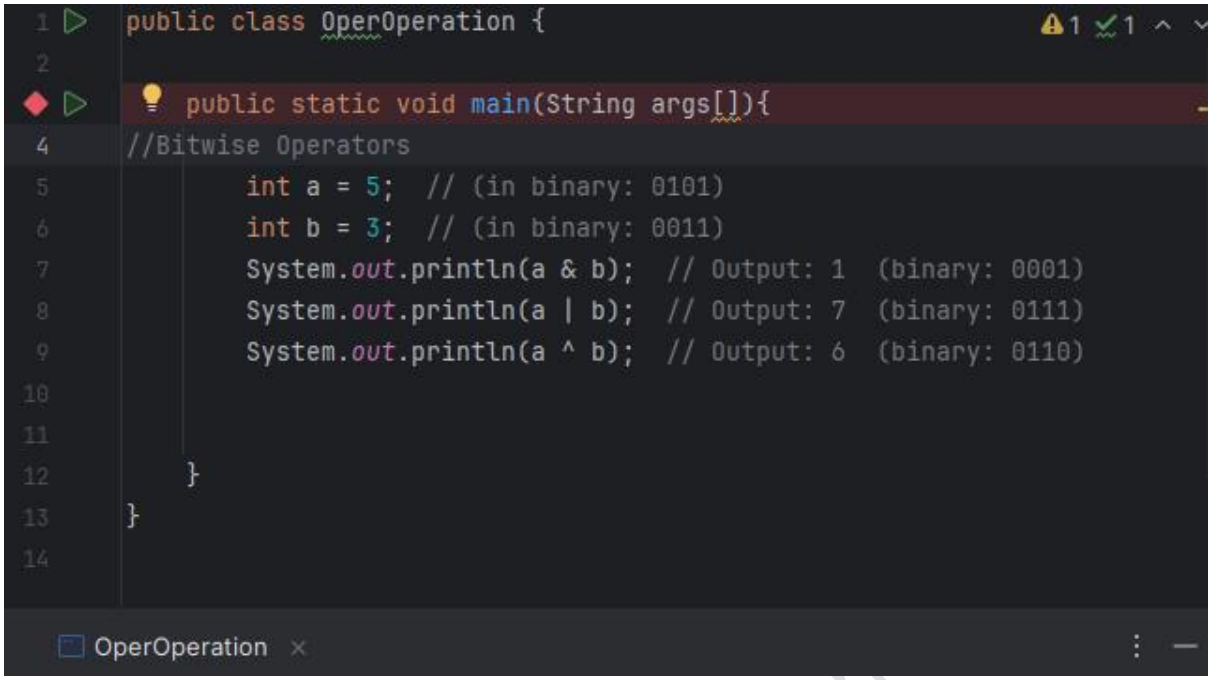
```java
public class OperOperation {

    public static void main(String args[]){

        int a=5;

        System.out.println(++a);  // Output: 6 (pre-increment)
        System.out.println(a++);  // Output: 6 (post-increment) 7
        System.out.println(--a);  // Output: 6 (pre-decrement) 7
        System.out.println(a--);  // Output: 6 (post-decrement) 6
        System.out.println(a) ;  // Output: 5

    }
}
```

**6. Bitwise Operators**

Bitwise operators work on bits and perform bit-by-bit operations.

| Operator | Description | Example |
|---|---|---|
| & | Bitwise AND | a & b |
| \| | a\|b | Bitwise OR |
| ^ | Bitwise XOR | a ^ b |
| ~ | Bitwise NOT | ~a |
| << | Left shift | a << b |
| >> | Right shift | a >> b |
| >>> | Unsigned right shift | a >>> b |

**Example:**

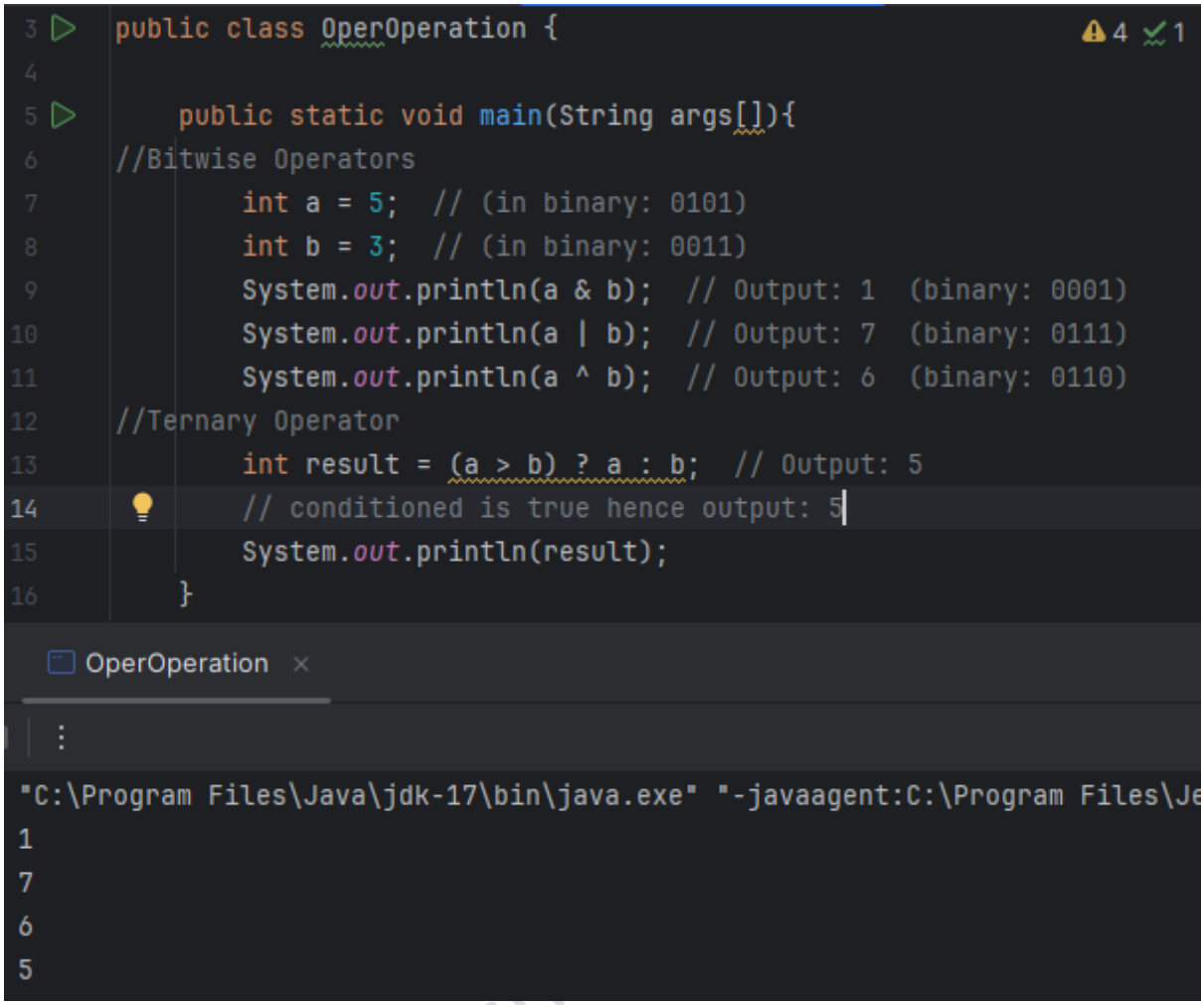



**7. Ternary Operator**

The ternary operator is a shorthand for the if-else statement. It takes three operands and returns one of two values based on a condition.

| Operator | Description | Example |
|---|---|---|
| ?: | Ternary (conditional) operator | condition? expr1: expr2 |

Example:


public class OperOperation {
4 1
public static void main(String args[]){
//Bitwise Operators
int a = 5; // (in binary: 0101)
int b = 3; // (in binary: 0011)
System.out.println(a & b); // Output: 1 (binary: 0001)
System.out.println(a | b); // Output: 7 (binary: 0111)
System.out.println(a ^ b); // Output: 6 (binary: 0110)
//Ternary Operator
int result = (a > b) ? a : b; // Output: 5
// conditioned is true hence output: 5
System.out.println(result);
}
OperOperation
"C:\Program Files\Java\jdk-17\bin\java.exe" "-javaagent:C:\Program Files\Je
1
7
6
5