Welcome to My Presentation

My Presentation Topic is

Java Operator

Presented By:

Name : Tushar Sarkar

Student ID: 18CSE035

Second Year First Semester

Department of CSE,BSMRSTU,

Gopalganj-8100.

Content

- What is Java Operator?
- Arithmetic Operators
- Arithmetic Operators Example
- Assignment Operators
- Assignment Operators Example
- Relational Operators
- Relational Operators Example
- Logical Operators
- Unary Operators
- Increment and Decrement Operators
- Increment and Decrement Operators Example
- Bitwise Operators
- Ternary Operator

What is Java Operator?

Operators are symbols that perform operations on variables and values.
For example, + is an operator used for addition, while * is also an operator used for multiplication.

Operators in Java can be classified into 6 types:

- Arithmetic Operators
- Assignment Operators
- Relational Operators
- Logical Operators
- Unary Operators
- Bitwise Operators

Arithmetic Operators

Arithmetic operators are used to perform arithmetic operations on variables and data.

For example,

a + b;

- Here, the + operator is used to add two variables a and b.
- Similarly, there are various other arithmetic operators in Java.

Operator	Operation
+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulo Operation

Arithmetic Operators Example

```java
class Main {
  public static void main(String[] args) {

    // declare variables
    int a = 12, b = 5;

    // addition operator
    System.out.println("a + b = " + (a + b));

    // subtraction operator
    System.out.println("a - b = " + (a - b));

    // multiplication operator
    System.out.println("a * b = " + (a * b));

    // division operator
    System.out.println("a / b = " + (a / b));

    // modulo operator
    System.out.println("a % b = " + (a % b));
  }
}
```

Output

```
a + b = 17
a - b = 7
a * b = 60
a / b = 2
a % b = 2
```

In the above we have use **+ - */** and **%** operators to compute addition, subtraction, multiplication, division and modulo operations.

Assignment Operators

Assignment operators are used in Java to assign values to variables.

For example, int age; age = 5;

- Here, = is the assignment operator.
- It assigns the value on its right to the variable on its left.
- That is, **5** is assigned to the variable age.

Let's see some more assignment operators available in Java.

Operator	Example	Equivalent to
=	a = b;	a = b;
+=	a += b	a = a + b;
-=	a -= b;	a = a - b;
*=	a *= b;	a = a * b;
/=	a /= b;	a = a / b;
%=	a %= b;	a = a % b;

Assignment Operators Example

```java
class Main {
  public static void main(String[] args) {

    // create variables
    int a = 4;
    int var;

    // assign value using =
    var = a;
    System.out.println("var using =: " + var);

    // assign value using =+
    var += a;
    System.out.println("var using +=: " + var);

    // assign value using =*
    var *= a;
    System.out.println("var using *=: " + var);
  }
}
```

Output

```
var using =: 4
var using +=: 8
var using *=: 32
```

Relational Operators

Relational operators are used to check the relationship between two operands.

For example,

a < b;

- Here, > operator is the relational operator.
- It checks if a is less than b or not.
- It returns either true or false

Operator	Description	Example
==	Is Equal To	4 == 5 return **false**
!=	Not Equal To	4 != 5 return **true**
>	Greater Than	4 > 5 return **false**
<	Less Than	4 < 5 return **true**
>=	Greater Than or Equal To	4 >= 5 return **false**
<=	Less Than or Equal To	4 <= 5 return **false**

Relational Operators Example

```
class Main {
  public static void main(String[] args) {

    // create variables
    int a = 7, b = 11;

    // value of a and b
    System.out.println("a is " + a + " and b is " + b);

    // == operator
    System.out.println(a == b);  // false

    // != operator
    System.out.println(a != b);  // true

    // > operator
    System.out.println(a > b);  // false

    // < operator
    System.out.println(a < b);  // true

    // >= operator
    System.out.println(a >= b);  // false

    // <= operator
    System.out.println(a <= b);  // true
  }
}
```

Note: Relational operators are used in decision making and loops

Logical Operators

- Logical operators are used to check whether an expression is true or false.
- They are used in decision making.

Operator	Example	Meaning
&& (Logical AND)	Expression1 **&&** Expression2	**True** only if both **expression1** and **expression2** are **true**
|| (Logical OR)	Expression1 || Expression2	**True** if either **expression1** or **expression2** is **true**
! (Logical NOT)	**!**Expression	**True** if **expression** is **false** and vice versa

Logical Operators Example

Working of Program :

- (5>3) && (8>5) returns **true** because (5>3) and (8>5) are **true**.
- (5>3) && (8<5) returns **false** because (8<5) are **false**.
- (5<3 || 8>5) returns **true** because **(8>5)** are true.
- (5<3 || 8<5) returns **false** because **(5<3)** and **(8<5**) are false.
- !(5==3) returns **false** because **5** is not equal to **3**.

```
class Main {
  public static void main(String[] args) {

    // && operator
    System.out.println((5 > 3) && (8 > 5));  // true
    System.out.println((5 > 3) && (8 < 5));  // false

    // || operator
    System.out.println((5 < 3) || (8 > 5));  // true
    System.out.println((5 > 3) || (8 < 5));  // true
    System.out.println((5 < 3) || (8 < 5));  // false

    // ! operator
    System.out.println(!(5 == 3));  // true
    System.out.println(!(5 > 3));  // false
  }
}
```

Unary Operators

- Unary operators are used with only one operand.
- For example, ++ is a unary operator that increases the value of a variable by **1**.
- That is, ++5 will return **6**.

Different types of unary operators are:

Operator	Meaning
+	**Unary plus** : not necessary to use since numbers are positive without using it
-	**Unary minus** : inverts the sign of an expression
++	**Increment operator** : increments value by 1
--	**Decrement operator** : decrements value by 1
!	**Logical complement operator** : inverts the value of a boolean

Increment and Decrement Operators

- Java also provides increment and decrement operators : **++** and **--** respectively.
- **++** increase the value of the operand by 1, while **--** decrease it by 1.
- For Example,

```
int num = 5;
// increase num by 1
++num;
```

- Here, the value of num gets increased to **6** from its initial value of **5**.

Increment and Decrement Operators Example

```java
class Main {
  public static void main(String[] args) {

    // declare variables
    int a = 12, b = 12;
    int result1, result2;

    // original value
    System.out.println("Value of a: " + a);

    // increment operator
    result1 = ++a;
    System.out.println("After increment: " + result1);

    System.out.println("Value of b: " + b);

    // decrement operator
    result2 = --b;
    System.out.println("After decrement: " + result2);
  }
}
```

Output

```
Value of a: 12
After increment: 13
Value of b: 12
After decrement: 11
```

- In the above program, we have used the ++ and -- operator as **prefixes (++a, --b)**. We can also use these operators as **postfix (a++, b++)**.
- There is a slight difference when these operators are used as prefix versus when they are used as a postfix.

Bitwise Operators

- Bitwise operators in Java are used to perform operations on individual bits.
- For example,

```
Bitwise complement Operation of 35

35 = 00100011 (In Binary)

~ 00100011
  ----------
   11011100   = 220 (In decimal)
```

Here, ~ is a bitwise operator.

It inverts the value of each bit (**0** to **1** and **1** to **0**).

Various Bitwise Operators

Operator	Description
~	Bitwise Complement
<<	Left Shift
>>	Right Shift
>>>	Unsigned Right Shift
&	Bitwise AND
^	Bitwise Exclusdive OR

Ternary Operator

The ternary operator (conditional operator) is shorthand for the if-then-else statement.
For example,

variable = Expression ? expression1 : expression2

Here's how it works.

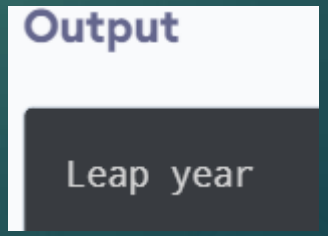
- If the Expression is true, expression1 is assigned to the variable.
- If the Expression is false, expression2 is assigned to the variable.

Let's see an example of a ternary operator.

```
class Java {
  public static void main(String[] args) {

    int februaryDays = 29;
    String result;

    // ternary operator
    result = (februaryDays == 28) ? "Not a leap year" : "Leap year";
    System.out.println(result);
  }
}
```



Thank you