Java Assignment Operators with Examples

Difficulty Level : Basic Last Updated : 09 Dec, 2021

Operators constitute the basic building block to any programming language. Java too provides many types of operators which can be used according to the need to perform various calculations and functions, be it logical, arithmetic, relational, etc. They are classified based on the functionality they provide.

Types of Operators:

1. Arithmetic Operators
2. Unary Operators
3. Assignment Operator
4. Relational Operators
5. Logical Operators
6. Ternary Operator
7. Bitwise Operators
8. Shift Operators

This article explains all that one needs to know regarding the Assignment Operators.

Assignment Operators

These operators are used to assign values to a variable. The left side operand of the assignment operator is a variable, and the right side operand of the assignment operator is a value. The value on the right side must be of the same data type of the operand on the left side. Otherwise, the compiler will raise an error. This means that the assignment operators have right to left associativity, i.e., the value given on the right-hand side of the operator is assigned to the variable on the left. Therefore, the right-hand side value must be declared before using it or should be a constant. The general format of the assignment operator is,

```
variable operator value;
```

Types of Assignment Operators in Java

The Assignment Operator is generally of two types. They are:

1. Simple Assignment Operator: The Simple Assignment Operator is used with the "=" sign where the left side consists of the operand and the right side consists of a value. The value of the right side must be of the same data type that has been defined on the left side.

2. Compound Assignment Operator: The Compound Operator is used where +,-,*, and / is used along with the = operator.

Let's look at each of the assignment operators and how they operate:

1. (=) operator:

This is the most straightforward assignment operator, which is used to assign the value on the right to the variable on the left. This is the basic definition of an assignment operator and how it functions.

Syntax:

```
num1 = num2;
```

Example:

```
a = 10;
ch = 'y';
```

```java
// Java code to illustrate "=" operator

import java.io.*;

class Assignment {
    public static void main(String[] args)
    {
        // Declaring variables
        int num;
        String name;

        // Assigning values
        num = 10;
        name = "GeeksforGeeks";

        // Displaying the assigned values
        System.out.println("num is assigned: " + num);
        System.out.println("name is assigned: " + name);
    }
}
```

Output

```
num is assigned: 10
name is assigned: GeeksforGeeks
```

2. (+=) operator:

This operator is a compound of '+' and '=' operators. It operates by adding the current value of the variable on the left to the value on the right and then assigning the result to the operand on the left.

Syntax:

```
num1 += num2;
```

Example:

```
a += 10

This means,
a = a + 10
```

```
// Java code to illustrate "+="

import java.io.*;

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

        // Declaring variables
        int num1 = 10, num2 = 20;

        System.out.println("num1 = " + num1);
        System.out.println("num2 = " + num2);
        // Adding & Assigning values
        num1 += num2;

        // Displaying the assigned values
        System.out.println("num1 = " + num1);
    }
}
```

Output

```
num1 = 10
num2 = 20
num1 = 30
```

3. (-=) operator:

This operator is a compound of ‘-‘ and ‘=’ operators. It operates by subtracting the variable’s value on the right from the current value of the variable on the left and then assigning the result to the operand on the left.

Syntax:

```
num1 -= num2;
```

Example:

```
a -= 10

This means,
a = a - 10
```

```
// Java code to illustrate "-="

import java.io.*;

class Assignment {
    public static void main(String[] args)
    {
        // Declaring variables
        int num1 = 10, num2 = 20;

        System.out.println("num1 = " + num1);
        System.out.println("num2 = " + num2);

        // Subtracting & Assigning values
        num1 -= num2;

        // Displaying the assigned values
        System.out.println("num1 = " + num1);
    }
}
```

Output

```
num1 = 10
num2 = 20
num1 = -10
```

4. (*=) operator:

This operator is a compound of ‘*’ and ‘=’ operators. It operates by multiplying the current value of the variable on the left to the value on the right and then assigning the result to the operand on the left.

Syntax:

```
num1 *= num2;
```

Example:

```
a *= 10
This means,
a = a * 10
```

```
// Java code to illustrate "*="

import java.io.*;
```

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

        // Declaring variables
        int num1 = 10, num2 = 20;

        System.out.println("num1 = " + num1);
        System.out.println("num2 = " + num2);

        // Multiplying & Assigning values
        num1 *= num2;

        // Displaying the assigned values
        System.out.println("num1 = " + num1);
    }
}
```

Output

```
num1 = 10
num2 = 20
num1 = 200
```

5. (/=) operator:

This operator is a compound of '/' and '=' operators. It operates by dividing the current value of the variable on the left by the value on the right and then assigning the quotient to the operand on the left.

Syntax:

```
num1 /= num2;
```

Example:

```
a /= 10
This means,
a = a / 10
```

```
// Java code to illustrate "/="

import java.io.*;

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

        // Declaring variables
        int num1 = 20, num2 = 10;

        System.out.println("num1 = " + num1);
        System.out.println("num2 = " + num2);

        // Dividing & Assigning values
        num1 /= num2;

        // Displaying the assigned values
        System.out.println("num1 = " + num1);
    }
}
```

Output

```
num1 = 20
num2 = 10
num1 = 2
```

6. (%=) operator:

This operator is a compound of ‘%’ and ‘=’ operators. It operates by dividing the current value of the variable on the left by the value on the right and then assigning the remainder to the operand on the left.

Syntax:

```
num1 %= num2;
```

Example:

```
a %= 3

This means,
a = a % 3
```

```java
// Java code to illustrate "%="

import java.io.*;

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

        // Declaring variables
        int num1 = 5, num2 = 3;

        System.out.println("num1 = " + num1);
        System.out.println("num2 = " + num2);

        // Moduling & Assigning values
        num1 %= num2;

        // Displaying the assigned values
        System.out.println("num1 = " + num1);
    }
}
```

Output

```
num1 = 5
num2 = 3
num1 = 2
```