UNIT 2: OPERATORS AND ASSIGNMENTS

UNIT STRUCTURE

2.1 Learning Objectives

2.2 Introduction

2.3 Introduction to Java Operator

2.4 Assignment Operator

2.5 Comparison Operators

2.6 Instance of Comparison

2.7 Arithmetic Operators

2.8 Shift Operator

2.9 Bitwise Operator

2.10 Logical Operator

2.11 Conditional Operator

2.12 Casting

2.13 Let Us Sum Up

2.14 Further Reading

2.15 Answer to Check Your Progress

2.16 Model Questions

2.1 LEARNING OBJECTIVES

After going through this unit, you will be able to:

- learn about Java operator
- learn the types of operators in Java with examples
- learn about the operator casting

2.2 INTRODUCTION

In the first unit we have already been acquainted with Java environment and various basic features of Java programming language. We have also learnt how to write a simple Java program after installation of Java package. This unit mainly discusses the various operators used in Java programming language. Casting of operators is also necessary to learn which is discussed at the end of this unit.

2.3 INTRODUCTION TO JAVA OPERATOR

Operators are symbols that ask to perform certain specific operations like mathematical or logical manipulation between data items. Java provides many types of operators which can be used according to the need. They are classified based on the functionality they provide. Some of the types are:

- Arithmetic Operators
- Unary Operators
- Assignment Operator
- Relational Operators
- Logical Operators
- Ternary Operator
- Bitwise Operators
- Shift Operators
- Instance of operator

2.4 ASSIGNMENT OPERATOR

An assignment operator assigns the value of an operand to a variable. '=' is an assignment operator.

Syntax: variable=value;

Example: x=7;

Program 2.1:

```
// 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 = 11;
        name = "JavaProgramming";
      // Displaying the assigned values
        System.out.println("num is assigned: " + num);
        System.out.println("name is assigned: " +
         name);
    }
}
```

Output: num is assigned: 11

name is assigned: JavaProgramming



CHECK YOUR PROGRESS

Q 1: What is an Assignment operator?

.....

.....

Q 2: X == 7; is it a correct asignment?

.....

.....

2.5 COMPARISON OPERATORS

A comparison operator compares numbers or strings and gives the result of the comparison.

Example:

=	Equal
==	Strictly Equal
\ =	Not Equal
\ ==	Not strictly equal
>	Greater than
<	Less than
> <	Greater than or less than (same as not equal)
> =	Greater than or equal to
\ <	Not less than

< = Less than or equal to

\ > Not greater than

2.6 INSTANCE OF COMPARISON

To check the specific type of instance whether it is a class, subclass, or interface of a given object instance, and return true or false in Java programs is the instance of the comparison operator.

Example:

The Instance of operation is different in different cases.

- **String data type:**

```
class Simple
{
   public static void main(String args[]){
    String str1 = "Educative!";
    if (str1 instanceof String)
       System.out.println("str1 is instance
       of String");
    else
        System.out.println("str1 is NOT instance
        of String");
     }
    }
```

- **Inherited classes:**

```
class Animal { }
class Mammal extends Animal { }
class instanceofTest
{
      public static void main(String[] args)
      {
            Mammal mobj = new Mammal();
            Animal aobj = new Animal();
        // Is `child` class instance of `child` class?
```

```
 if (mobj instanceof Mammal)
      System.out.println("mobj is instance of
      Mammal");
else
     System.out.println("mobj is NOT instance of
      Mammal");
// Is `child` class instance of `parent` class?
if (mobj instanceof Animal)
  System.out.println("mobj is instance of Animal");
else
 System.out.println("mobj is NOT instance of
 Animal");
// Is `parent` class instance of `child` class?
if (aobj instanceof Mammal)
  System.out.println("mobj is instance of Animal");
else
   System.out.println("mobj is NOT instance of
   Animal");
  }
}
```

2.6 ARITHMETIC OPERATORS

The mathematical operations are performed using arithmetic operators. Some of the arithmetic operators are:

- \+ (addition)

  Syntax:

  num1+num2

  Example:

  ```
  num1 = 10, num2 = 20
  sum = num1 + num2 = 30
  ```

- \- (subtraction)

  Syntax:

num1 - num2

Example:

```
num1 = 20, num2 = 10
 sub = num1 - num2 = 10
```

- * (multiplication)

  Syntax:

  num1 * num2

  Example:

```
num1 = 20, num2 = 10
mult = num1 * num2 = 200
```

- / (division)

  Syntax:

  num1 / num2

  Example:

```
num1 = 20, num2 = 10
div = num1 / num2 = 2
```

- % (modulus).

  Syntax:

  num1 % num2

  Example:

```
num1 = 5, num2 = 2
mod = num1 % num2 = 1
```



CHECK YOUR PROGRESS

Q 3: Modulus operator, %, can be applied for which purposes?

.....

Q 4: What is/are the highest order precedence operator(s) in Java?

.....

Q 5: Decrement operator ,--, decreases the value of a variable by what number?

.....

2.7 SHIFT OPERATOR

Shifting of bit patterns to the left or right is done using the shift operator. Various types of shift operators are:

- **Signed left shift operator:**

  It moves all the bits to the left by a given number of bits to the left.

  Syntax:

  left_operand << number

  Program 2.2:

```
// Java Program to demonstrate
// Signed Left-Shift Operator
// Importing required classes
import java.io.*;
// Main class
class GFG
{
      // main driver method
      public static void main(String[] args)
      {
            int number = 2;
       // 2 bit left shift operation
            int Ans = number << 2;
            System.out.println(Ans);
      }
}
```

- **Signed right shift operator:**

  It moves all the bits to the right by a given number of bits to the right.

  Syntax:

  left_operand >> number

  Program 2.3

```
  // Java program to demonstrate
  // the Signed right shift operator
```

```
import java.io.*;
class GFG
{
      public static void main (String[] args)
       {
           {
            int number = 8;
            // 2 bit signed right shift
            int Ans = number >> 2;
            System.out.println(Ans);
           }
      }
}
```

- **Unsigned right shift operator:**

  It moves all the bits to the right and the leftmost bit is filled by 0.

  Syntax:

  left_operand >>> number

  Program 2.4:

```
// Java program to demonstrate
// the Unsigned right shift operator
import java.io.*;
class GFG
      {
      public static void main (String[] args)
      {
            byte num1 = 8;
            byte num2 = -8;
            System.out.println(num1 >>> 2);
            System.out.println(num2 >>> 2);
      }
}
```

2.8 BITWISE OPERATOR

The manipulation of discrete bits of a number is done by using a bitwise operator. Various types of bitwise operators are\:

- **Bitwise OR (|)**

  Example:

  a = 5 = 0101 (In Binary)

  b = 7 = 0111 (In Binary)

  Bitwise OR Operation of 5 and 7

  0101

  | 0111

  ————————

  0111 = 7 (In decimal)

- **Bitwise AND (&)**

  Example:

  a = 5 = 0101 (In Binary)

  b = 7 = 0111 (In Binary)

  Bitwise AND Operation of 5 and 7

  0101

  & 0111

  ————————

  0101 = 5 (In decimal)

- **Bitwise XOR (^)**

  Example:

  a = 5 = 0101 (In Binary)

  b = 7 = 0111 (In Binary)

  Bitwise XOR Operation of 5 and 7

  0101

  ^ 0111

  ————————

  0010 = 2 (In decimal)

- **Bitwise Complement (~)**

  Example:

  a = 5 = 0101 (In Binary)

  Bitwise Complement Operation of 5

  ~ 0101

  ————

  1010 = 10 (In decimal)

2.9 LOGICAL OPERATOR

The operators used to make decisions and perform logical operations are called logical operators. Two or more conditions or relational statements are evaluated using logical operators. Various logical operators are:

- AND Operator (&&) - if(a && b) [if true execute else don't]
- OR Operator (||) - if(a || b) [if one of them is true execute else don't]
- NOT Operator (!) - !(a<b) [returns false if a is smaller than b]

  Example:

  x = 5, y = 15, z = 25

  For AND operator:

  Condition 1: z > x

  Condition 2: z > y

  Output: True [Both Conditions are true]

  For OR Operator:

  Condition 1: z > x

  Condition 2: z > y

  Output: True [One of the Condition if true]

  For NOT Operator:

  Condition 1: z > x

  Condition 2: z > y

  Output: False [Because the result was true and NOT operator did it's opposite]

2.10 CONDITIONAL OPERATOR

The operators used to check the conditions of two or more operands and provide the desired result are called conditional operators. Various conditional operators are:

- Conditional AND
- Conditional OR
- Ternary Operator

Program 2.5:

```
public class ConditionalOperatorExample
{
   public static void main(String args[])
   {
      int a=6, b=3, c=9;
      System.out.println(a>b && a>c || b<c);
      System.out.println((a<c || b>c) && a<b);
   }
}
```

Output:

true

false

2.11 CASTING

Converting a data type to another is called type casting. This is applied in situations where we need to store a value of one type into a variable of another type.

Syntax: () is Cast Operator

RequiredDatatype=(TargetType)Variable

Program 2.6:

```
// Java Program to Implement Type Casting of the
Datatype
// Importing input output classes
```

```
import java.io.*;
// Main class
public class XYZ
{
// Main driver method
      public static void main(String[] args)
      {
            // Declaring an Integer datatype
            int a = 6;
            // Casting to Large datatype
            double db = (double)a;
            // Print and display the casted value
            System.out.println(db);
            // Narrow Casting conversion
            int db1 = (int)db;
            // Print an display narrow casted value
            System.out.println(db1);
      }
}
```

Output:

6.0

6



CHECK YOUR PROGRESS

Q 6: What is casting?

Q 7: int a=6, b=3, c=9;

System.out.println(a>b && a>c || b<c);



2.12 LET US SUM UP

- Operators are symbols that ask to perform certain specific operations like mathematical or logical manipulation between data items.
- An assignment operator assigns the value of an operand to a variable.
- A comparison operator compares numbers or strings and gives the

result of comparison.

- To check the specific type of instance whether it is a class, subclass or an interface of a given object instance and return true or false in Java programs is the instance of comparison operator.
- The mathematical operations are performed using arithmetic operators.
- Shifting of bit patterns to the left or right is done using the shift operator. Various types of shift operators are:
  - Signed left shift operator
  - Signed right shift operator
  - Unsigned right shift operator
- The manipulation of discrete bits of a number is done by using a bitwise operator. Various types of bitwise operators are-
  - Bitwise OR (|)
  - Bitwise AND (&)
  - Bitwise XOR (^)
  - Bitwise Complement (~)
- The operators used to make decisions and perform logical operations are called logical operators. Two or more conditions or relational statements are evaluated using logical operators. Various logical operators are:
  - AND Operator (&&) - if(a && b) [if true execute else don't]
  - OR Operator (||) - if(a || b) [if one of them is true execute else don't]
  - NOT Operator (!) - !(a<b) [returns false if a is smaller than b]
- The operators used to check the conditions of two or more operands and provide the desired result are called conditional operators. Various conditional operators are:
  - Conditional AND
  - Conditional OR
  - Ternary Operator
- 10. Converting a data type to another is called type casting. This is applied in situations where we need to store a value of one type into a variable of another type.



2.13 FURTHER READINGS

1) Schildt, H. (2020). The complete reference Java.



2.14 ANSWERS TO CHECK YOUR PROGRESS

Ans to Q No 1: An assignment operator assigns the value of an operand to a variable. '=' is an assignment operator.

Ans to Q No 2: No

Ans to Q No 3: Both Integers and floating - point numbers

Ans to Q No 4: () and { }

Ans to Q No 5: 1

Ans to Q No 6: Converting a data type to another is called type casting.

Ans to Q No 7: False



2.14 MODEL QUESTIONS

Q 1: What are the various bitwise operators used in java?

Q 2: Explain shift operators with example.

Q 3: How logical operators are used? Explain with examples.

Q 4: Explain the uses of type casting with examples.

Q 5: Explain the uses of conditional operators with examples.

*** ***** ***