CORE JAVA PROGRAMMING OPERATOR (1Z0-808)

By www.HadoopExam.com

Note: These instructions should be used with the HadoopExam Apache OOzie: Professional Trainings. Where it is executed and you can do hands on with trainer.

Java Operators:

Operator: unary, binary, and ternary. It can be applied to variables, literal or values.

Operand: variables, literal or values on which operator is applied is known as operand.

- Java operators are not necessarily evaluated from left-to-right order

```
public class Welcome {

   public static void main(String[] args) {
         int val1 = 10;
         double val2 = 10 + 2 * --val1;
         System.out.println("Value of val1 " + val1);
         System.out.println("Value of val2 " + val2);
   }
}
```

- **Order of Precedence**

Order of operator precedence

Post-unary operators	expression++, expression--
Pre-unary operators	++expression, --expression
Other unary operators	+, -, !
Multiplication/Division/Modulus	*, /, %
Addition/Subtraction	+, -
Shift operators	<<, >>, >>>
Relational operators	<, >, <=, >=, instanceof
Equal to/not equal to	==, !=
Logical operators	&, ^, |
Short-circuit logical operators	&&, ||
Ternary operators	boolean expression ? expression1 : expression2
Assignment operators	=, +=, -=, *=, /=, %=, &=, ^=, !=, <<=, >>=, >>>=

Arithmetic Operator:

- Addition (+), subtraction (-), multiplication (*), division (/), and modulus (%),++ and –

Calculate the result for below expression

```
int val1 = 3 * 9 + 10 * 4 - 18;
```

Ans: 49

Evaluate below expression

```
int val1 = 3 * (9 + 10 * 4) - 18;
```

Ans: 129

Note: operators + and += may be applied to String values, which results in String concatenation.

Division and Modulo Operator:

```
System.out.println(100%10);
System.out.println(11%3);
System.out.println(9%10);
System.out.println(11%2);
System.out.println(100/10);
System.out.println(12%10);
System.out.println(12%3);
```

With negative value

```
System.out.println(-100%10);
System.out.println(-11%3);
System.out.println(-9%10);
System.out.println(-11%2);
System.out.println(-100/10);
System.out.println(-12%10);
System.out.println(-12%3);
```

Numeric Promotion:

- A long value takes up more space than an int value.

- If two values have different data types, Java will automatically promote one of the values to the larger of the two data types
- If one of the values is integral and the other is floating-point, Java will automatically promote the integral value to the floating-point value's data type.
- Smaller data types, namely byte, short, and char, are first promoted to int any time they're used with a Java binary arithmetic operator, even if neither of the operands is int. (This rule does not apply on unary operator e.g. ++ , applying ++ to a short value results in a short value.)
- After all promotion has occurred and the operands have the same data type, the resulting value will have the same data type as its promoted operands.

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

         //Rule 1
         int val1 = 10;
         long val2 = 12;
         System.out.println(val1+val2); //22 but type of value will be long.

         //Rule 2
         double val11 = 10.1;
         float val12 = 12.0f; //f is mandatory here. or you can have (float)12.0
         double result = val11+val12; //both operands being promoted to a double
         System.out.println(result); //22.1 but type of value will be double.

         //Rule 3
         short v1=10;
         short v2=2;
         System.out.println(v1/v2); //5 but type of value will be int. resulting output is not a short

         //All rules
         short x1 = 120;
         float y1 = 12.0f;
         double z1 = 12;

         result = x1*y1/z1;
         //x1 will automatically be promoted to int  because it is a short
         //The promoted x1 value will then be automatically promoted to a float to multiply with y1.
         //The result of x1 * y1 will then be automatically promoted to a double
         System.out.println(result); //120.0

   }
}
```

Unary Operator

- a *unary* operator is one that requires exactly one operand, or variable, to function.

```
I++, j--
```

Negation Operator (!)

- Also known as logical complement operator.

```
public class Welcome {
   public static void main(String[] args) {
         boolean x=false;
         System.out.println(!x);
         System.out.println(!!x);

         int y=100;
         System.out.println(-y);

   }
}
```

Increment and decrement operator:

```
public class Welcome {
      public static void main(String[] args) {
            int counter = 0;
            System.out.println(counter); // Outputs 0
            System.out.println(++counter); // Outputs 1
            System.out.println(counter); // Outputs 1
            System.out.println(counter--); // Outputs 1
            System.out.println(counter); // Outputs 0

            int val1 = 7;
            int val2 = ++val1 * 20 / val1-- + --val1;
            //How this evaluated
            // 8 * 20 /val1-- + --val1
            // 8 * 20 /8 + --val1
            //// 8 * 20 /8 + 6
            // Now evaluate from left to right
            //160/8 + 6
```

```
            //20+6
            System.out.println("val1 is " + val1);
            System.out.println("val2 is " + val2);
      }
}
```

Assignment Operator:

```
public class Welcome {
      public static void main(String[] args) {
            //int counter = 0.0;
            int counter = (int) 0.0;
            short y = (short)9999999;

            short val1 = 10;
            short val2 = 3;
            short result = (short) (val1 * val2); // DOES NOT COMPILE if not casted

            System.out.println(counter);
            System.out.println(y); //:) It is a different value.
            System.out.println(result); //:) It is a different value.
      }
}
```

- **Integer overflow:** The expressions in the previous example now compile, although there's a cost. The second value, **9999999**, is too large to be stored as a short, so numeric overflow occurs and it becomes −27009. Overflow is when a number is so large that it will no longer fit within the data type, so the system "wraps around" to the next lowest value and counts up from there. There's also an analogous underflow, when the number is too low to fi t in the data type.

Compounding Assignment:

```
public class Welcome {
      public static void main(String[] args) {
            int v1= 10;
            int v2 = 20;
            v2*=v1; //Compounding operation
```

```
            System.out.println(v2);

            long x = 23;
            long y = (x=22);
            System.out.println(x); // Outputs 22
            System.out.println(y); // Also, outputs 22
      }
}
```

Relational Operator:

```
public class Welcome {
      public static void main(String[] args) {
            int val1 = 10, val2 = 20, val3 = 10;
            System.out.println(val1 < val2); // Outputs true
            System.out.println(val1 <= val2); // Outputs true
            System.out.println(val1 >= val3); // Outputs true
            System.out.println(val1 > val3); // Outputs false
      }
}
```

- ***a instanceof b*** : True if the reference that a points to is an instance of a class, subclass, or class that implements a particular interface, as named in b

Logical Operators: (&), (|), and (^)

X & Y : AND		
	Y=true	y=false
x=true	TRUE	FALSE
X=false	FALSE	FALSE

X | Y : Inclusive Or		
	Y=true	y=false
x=true	TRUE	TRUE
X=false	TRUE	FALSE

X ^ Y : Exclusive Or		
	Y=true	y=false
x=true	FALSE	TRUE
X=false	TRUE	FALSE

Here are some tips to help remember this table:

- AND is only true if both operands are true.
- Inclusive OR is only false if both operands are false.
- Exclusive OR is only true if the operands are different.

Short circuit operator: (|| and &&) :

- The short-circuit operators are nearly identical to the logical operators, & and |, respectively, except that the right-hand side of the expression may never be evaluated if the final result can be determined by the left-hand side of the expression.

```
public class Welcome {
	public static void main(String[] args) {
		boolean y=false;
		boolean x = true || (y=true);
		System.out.println("X = " + x + " AND Y = " + y);

		boolean z = true && (y=true);
		System.out.println("Z = " + z + " AND Y = " + y);

		int val1 = 6;
		boolean val2 = (val1 >= 6) || (++val1 <= 7);
		System.out.println("Val1 = " + val1 + " AND Val2 = " + val2);
	}
}
```

Equality Operators: There are three important equality operator needs to be learned. This helps us to determine whether two things are equal or not. Bothe require two operands and return Boolean value.

- equals() : It applies only for objects.
- == and != : This applies for both Objects (but there is a catch) as well as primitive data types.

Let's take case by case scenario

1. Comparing primitive datatypes using == and !=
   i. 10 == 10.0 (will return true)
   ii. 10 !=5 (will return true)

```
import java.io.File;

public class Welcome {
	public static void main(String[] args) {
		int v1=10;

		System.out.println("Print 1 = " + (v1==10.0));

		boolean x=false;
```

```
            boolean y=false;
            System.out.println("Print 1.A = " + (x==y));
            System.out.println("Print 1.B = " + (x!=y));

            String str = "HadoopExam.com";
            String str1 = "QuickTechie.com";
            String str2 = "HadoopExam.com";
            System.out.println("Print 2 = " + str.equals(null));
            System.out.println("Print 3 = " + str.equals(str1));
            System.out.println("Print 4 = " + str.equals(str2));

            Integer i1 = new Integer(10);
            Integer i2 = new Integer(10);
            System.out.println("Print 5 = " + i1.equals(i2));
            System.out.println("Print 6 = " + (i1==i2)); //check here

            File f1 = new File("HadoopExam.txt");
            File f2 = new File("HadoopExam.txt");
            File f3 = f1;
            System.out.println("Print 7 = " + (f1 == f2)); // Outputs false
            System.out.println("Print 8 = " + (f1 == f3)); // Outputs true

      }
}
```

All Products List of www.HadoopExam.com

TRAINING'S

- Hadoop BigData Professional Training (3500INR/$79)
- HBase (NoSQL) Professional Training (3500INR/$79)
- Apache Spark Professional Training (~~3900INR/$89~~ for a week 3500INR/$75)
- Apache OOZie (Hadoop workflow) Professional Training
- Beginner AWS Training Course- **(HETRNAWS101)**
- Core Java 1z0-808 Exam training

MAPR HADOOP AND NOSQL CERTIFICATION

- **MapR Hadoop Developer Certification**
- **MapR HBase NoSQL Certification**
- **MapR Spark Developer Certification (In Progress)**

CLOUDERA HADOOP AND NOSQL CERTIFICATION

- CCA50X : Hadoop Administrator
- CCA-175 Cloudera® (Hadoop and Spark Developer)
- CCP:DE575 : Cloudera® Data Engineer Certification

DATABRICKSA OREILLY SPARK CERTIFICATION

- Apache Spark Developer

AWS: AMAZON WEBSERVICE CERTIFICATION

- AWS Solution Architect : Associate
- AWS Solution Architect: Professional
- AWS Developer : Associate
- AWS Sysops Admin : Associate

MICROSOFT AZURE CERTIFICATION

- Azure 70-532
- Azure 70-533

DATA SCIENCE CERTIFICATION

- EMC E20-007

EMC CERTIFICATIONS

- EMC E20-007

SAS ANALYTICS CERTIFICATION

- SAS Base A00-211
- SAS Advanced A00-212
- SAS Analytics : A00-240
- SAS Administrator : A00-250

ORACLE JAVA CERTIFICATION

- Java 1z0-808
- Java 1z0-809

ORACLE DATABASE CLOUD CERTIFICATION

- 1z0-060 (Oracle 12c)
- 1z0-061 (Oracle 12c)

Subscribe Here for Regular Updates: Like New Training Module launched

Become Author and Trainer: We are looking for Author (Writing Technical Books) and Trainer (Creating Training Material): **No Compromise on Quality.**

Benefit: You will get very good revenue sharing. Please drop us an email to hadoopexam@gmail.com (For the skills, you feel you are master)

We are sure, you are good at least one technology. Don't limit your potential, contact us immediately with your skill. Our expert team will contact you with more detail. You training and Books will reach to all our existing network and with our expert marketing team we will help you to reach as much as technical professional, with our Smart Advertising network. Contact us with sending an email hadoopexam@gmail.com

Opportunity to share your knowledge with all learners who are in need. We are helping 1000's of learners since last 4 years and established ourselves with Quality low cost material.