Royal University of Bhutan

Unit II

Java Programming Fundamentals

Tutor: Pema Galey

Learning Outcomes

In this session, you will learn about:

- Java Operators
- Java Input and Output
- Java Comments

Type of Operators

- Operators are symbols that perform operations on variables and values.
- Operators in Java can be classified into 6 types:
  1. Arithmetic Operators
  2. Assignment Operators
  3. Relational Operators
  4. Logical Operators
  5. Unary Operators
  6. Bitwise Operators

1. Java Arithmetic Operators

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

Operator	Operation
+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulo Operation (Remainder after division)

2. Java Assignment Operators

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

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;

3. Java Relational Operators

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

Operator	Description	Example
==	Is Equal To	3 == 5 returns **false**
!=	Not Equal To	3 != 5 returns **true**
>	Greater Than	3 > 5 returns **false**
<	Less Than	3 < 5 returns **true**
>=	Greater Than or Equal To	3 >= 5 returns **false**
<=	Less Than or Equal To	3 <= 5 returns **true**

4. Java 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

5. Java Unary Operators

- Unary operators are used with only one operand.

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

5. Java Unary Operators

Increment and Decrement Operators

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

- Example:

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

6. Java Bitwise Operators

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

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

Other Operators

Other operators:

1. Java *instanceof* Operator
2. Java Ternary Operator

Java instanceof Operator

- The instanceof operator checks whether an object is an instanceof a particular class.

- ***Note**: we will study after implementation of objects*

Java 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.

Java Output

- In Java, you can simply use to send output to standard output (screen).
  - `System.out.println();` or
  - `System.out.print();` or
  - `System.out.printf();`

- **`System`** is a class
- **`out`** is a **`public static`** field: it accepts output data.

Java Output

- **print()** - It prints string inside the quotes.
- **println()** - It prints string inside the quotes similar like print() method. Then the cursor moves to the beginning of the next line.
- **printf()** - It provides string formatting

```
class PrintVariables {
   public static void main(String[] args) {
       Double number = 10.5;
       System.out.print("I am " + "awesome.");
       System.out.println("Number = " + number);
       System.out.println(123);
} }
```

Java Input

- Java provides different ways to get input from the user. However, you will learn to get input from user using the object of **`Scanner`** class.
- In order to use the object of Scanner, we need to import **`java.util.Scanner`** package.

```
import java.util.Scanner;
```

Java Input

- We need to create an object of the Scanner class. We can use the object to take input from the user.

```
// create an object of Scanner
Scanner input = new Scanner(System.in);

// take input from the user
int number = input.nextInt();
```

Java Input

- Some methods available through Scanner class to read different kinds of input:

nextByte( )	Read a byte
nextShort( )	Read a short
nextInt( )	Read an integer
nextLong( )	Read a long
nextFloat( )	Read a floating point
nextDouble( )	Read a double
nextBoolean( )	Read a boolean
nextLine( )	Read a complete line
next( )	Read a word

Example: Java Input

```
// User Input to get Integer value
import java.util.Scanner;
class Input {
        public static void main(String[] args) {
                Scanner input = new Scanner(System.in);
                System.out.print("Enter an integer: ");
                int number = input.nextInt();
                System.out.println("You entered " + number);
                // closing the scanner object
                input.close();
        }
    }
```

Activity: Java Input

- Write a program to read the user input for different data types.
  - fong
  - float
  - double
  - String

Java Comments

- In computer programming, comments are a portion of the program that are completely ignored by Java compilers.
- They are mainly used to help programmers to understand the code.
- For example,

```
// declare and initialize two variables
int a =1;
int b = 3;
// print the output
System.out.println("This is output");
```

Types of Comments in Java

- In Java, there are two types of comments:

1. *single-line comment*
2. *multi-line comment*

1. Single Line Comment

A single-line comment starts and ends in the same line. To write a single-line comment, we can use the // symbol. Also, known as **End of Line** comment.

```
// print the output
System.out.println("This is output");
```

Types of Comments in Java

2. **Multi-Line Comment**

- When we want to write comments in multiple lines, we can use the multi-line comment.
- To write multi-line comments, we can use the /*....*/ symbol.
- Also, known as **Traditional Comment**

```
/* This is an example of multi-line comment.
* The program prints "Hello, World!" to the standard output.
*/
class HelloWorld {
        public static void main(String[] args) {
                System.out.println("Hello, World!");
} }
```

Summary

- Different types of Java Operators
- Java User Input
- Types of Java Output
- Java Comments

Thank you!