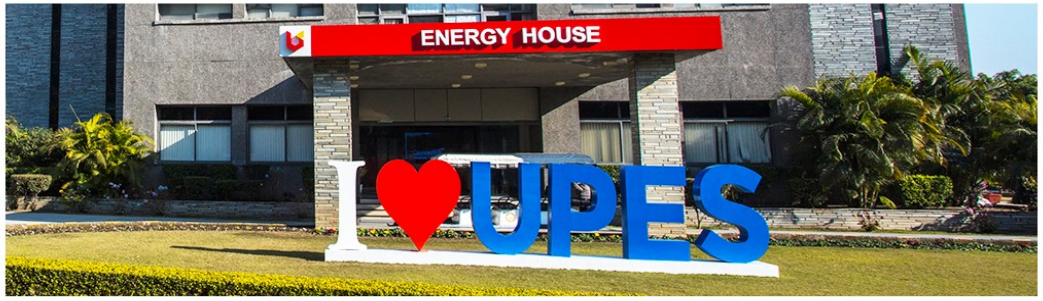

UPES

UNIVERSITY WITH A PURPOSE

Object Oriented Programming

UPES
UNIVERSITY WITH A PURPOSE

Operators and Expressions

Arithmetic Operations

There are several binary arithmetic operators that operate on any of the primitive numerical types:

```
+   addition
-   subtraction (also used for negation)
*   multiplication
/   division
%   remainder
```

1. Integer Arithmetic: Integer arithmetic is two's-complement arithmetic.

Exp: 7/2 = 3, and -7/2 = -3 and 7%2 = 1, and -7%2 = 1.

Exp: Dividing by zero or remainder by zero is invalid for integer arithmetic and throws ArithmeticException.

UPES
UNIVERSITY WITH A PURPOSE

Operators and Expressions contd..

2. Floating-Point Arithmetic:

- Arithmetic with finite operands performs as expected, within the limits of precision of double or float.
- The result of an invalid expression, such as dividing infinity by infinity, is a NaN

- We can get an infinity value from the constants POSITIVE_INFINITY and NEGATIVE_INFINITY in the wrapper classes Float and Double.

Operators and Expressions contd..

x	y	x/y	x%y
Finite	±0.0	±∞	NaN
Finite	±∞	±0.0	x
±0.0	±0.0	NaN	NaN
±∞	Finite	±∞	NaN
±∞	±∞	NaN	NaN

3. Strict and Non-Strict Floating-Point Arithmetic:

- Floating-point arithmetic can be executed in one of two modes: FP-strict or not FP-strict.
- strictfp: you will always get exactly equivalent results on all JVM implementations.
- Non-strictfp: executed with somewhat relaxed rules.
- When you declare a method `strictfp`, all the code in the method will be executed according to strict constraints.

UPES
UNIVERSITY WITH A PURPOSE

Operators and Expressions contd..

General Operators:

Increment and Decrement Operators: The ++ and -- operators are the increment and decrement operators and can only be applied to numeric variables or numeric array elements. The expression i++ is equivalent to i= i+ 1 except that i is evaluated only once.

Exp: ++arr[where()];

Statement invokes where only once and uses the result as an index into the array only once. On the other hand, in the Statement arr[where()]=arr[where()]+1;

the where method is called twice: once to determine the index on the right-hand side, and a second time to determine the index on the left-hand side. If where returns a different value each time it is invoked, the results will be quite different from those of the ++ expression. To avoid the second invocation of where you would have to store its result in a temporary.

Operators and Expressions contd..

Exp:

```
class IncOrder {
public static void main(String[] args) {
int i = 16;
System.out.println(++i + " " + i++ + " " + i);
}
}
```

The output is

17 17 18

Note: The increment and decrement operators ++ and -- can also be applied to char variables to get to the next or previous Unicode character.

Operators and Expressions contd..

Relational and Equality Operators: (>, <, >=, <=, ==, !=)

- All relational and equality operators in java yield boolean values.
- All relational and equality operators that test a number against NaN return false, except !=, which always returns true,

Exp: Double.NaN == Double.NaN

Is always **false**.

UPES
UNIVERSITY WITH A PURPOSE

Operators and Expressions contd..

Logical Operators:

```
&   logical AND
|   logical inclusive OR
^   logical exclusive or (XOR)
!   logical negation
&&      conditional AND
||      conditional OR
```

UPES
UNIVERSITY WITH A PURPOSE

Operators and Expressions contd..

Instanceof: The instanceof operator evaluates whether a reference refers to an object that is an instance of a particular class or interface. The left-hand side is a reference to an object, and the right-hand side is a class or interface name.

Bit Manipulation Operators: The binary bitwise operators are:

```
 &     bitwise AND
 |     bitwise inclusive OR
 ^     bitwise exclusive or (XOR)
```

- The bitwise operators perform operation on each pair of bits in the two operands. The AND of two bits yields a 1 if both bits are 1, the OR of two bits yields a 1 if either bit is 1, and the XOR of two bits yields a 1 only if the two bits have different values.

Exp: 0xf00f & 0x0ff0
= 0x0000
Exp: 0xf00f | 0x0ff0

UPES
UNIVERSITY WITH A PURPOSE

Operators and Expressions contd..

The Conditional Operator ?:

The conditional operator provides a single expression that yields one of two values based on a boolean expression. The statement

```
value = (userSetIt ? usersValue : defaultValue);
```

is equivalent to

```
if (userSetIt)
value = usersValue;
else
value = defaultValue;
```

Operators and Expressions contd..

Assignment Operators:

```
z = 3;
x = y = z = 3;
a *= b + 1 is analogous to a = a * (b + 1)
```

String Concatenation Operator:

```
You can use + to concatenate two strings.
String boo = "boo";
String cry = boo + "hoo";
cry += "!";
System.out.println(cry);
```

new: The new operator is a unary prefix operator. It has one operand that follows the operator. Technically, the use of new is known as an instance creation expression because it creates an instance of a class or array. The value of the expression is a reference to the object created.

UPES
UNIVERSITY WITH A PURPOSE

Operators and Expressions contd..

Order of Evaluation:

- given `x+y+z`, the compiler evaluates x, evaluates y, adds the values together, evaluates z, and adds that to the previous result.

Expression Type:

- Every expression has a type. The type of an expression is determined by the types of its component parts and the semantics of operators.
- If an arithmetic or bit manipulation operator is applied to integer values, the result of the expression is of type int unless one or both sides are long, in which case the result is long.
- If either operand of an arithmetic operator is floating point, the operation is performed in floating-point arithmetic. Such operations are done in float unless at least one operand is a double, in which case double is used for the calculation and result.

Operators and Expressions contd..

Type Conversions

Implicit Type Conversions:

- Any numeric value can be assigned to any numeric variable whose type supports a larger range of values widening primitive conversion.
- A char can be used wherever an int is valid. A floating-point value can be assigned to any floating-point variable of equal or greater precision.

```
long orig = 0x7effffff00000000L;
float fval = orig;
long lose = (long) fval;
System.out.println("orig = " + orig);
System.out.println("fval = " + fval);
System.out.println("lose = " + lose);

orig = 9151314438521880576
fval = 9.1513144E18
lose = 9151314442816847872
```

UPES
UNIVERSITY WITH A PURPOSE

Operators and Expressions contd..

```
short s1 = 27; // implicit int to short
byte b1 = 27; // implicit int to byte
short s3 = 0x1ffff; // INVALID: int value too big for short
```

Explicit Type Casts:

- A boolean cannot be cast to an intbut explicit casting can be used to assign a double to a long. Exp: double d = 7.99;
  long l = (long) d;
- When a floating-point value is cast to an integer, the fractional part is lost by rounding toward zero; for instance, (int)-72.3 is -72.
- A char can be cast to any integer type and vice versa. When an integer is cast to a char, only the bottom 16 bits of data are used; the rest are discarded. When a char is cast to an integer type, any additional upper bits are filled with zeros.

String Conversions:

- Whenever a + operator has at least one String operand, it is interpreted as the string concatenation operator and the other operand, if not a String, is implicitly converted into

Operators and Expressions contd..

Operator Precedence and Associativity

postfix operators	[] . (params) expr++ expr--
unary operators	++expr --expr +expr -expr ~ !
creation or cast	new (type)expr
multiplicative	* / %
additive	+ -
shift	<< >> >>>
relational	< > >= <= instanceof
equality	== !=
AND	&
exclusive OR	^
inclusive OR	|
conditional AND	&&
conditional OR	||
conditional	?:
assignment	= += -= *= /= %= >>= <<= >>>= &= ^= |=

UPES
UNIVERSITY WITH A PURPOSE

Operators and Expressions contd..

- All binary operators except assignment operators are left-associative. Assignment is right-associative. In other words, `a=b=c is equivalent to a=(b=c), so it is convenient to chain assignments together. The conditional operator ?: is right-associative.`

References

Gosling, J., Holmes, D. C., & Arnold, K. (2005). The Java programming language.