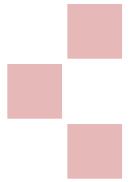
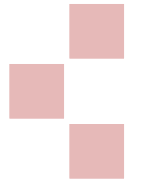


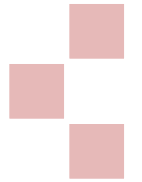
Java Operators

Nurochman



Java Operators

- Unary operator
- Arithmetic operator
- Shift operator: <<, >>, dan >>>
- Comparison operator
- Bitwise operators : &, ^, dan |.
- Binary operators : &, ^, dan |.
- Short–Circuit Logical Operator
- Conditional operator : ?
- Assignment operator
- Operator lain : new, instanceof
- Urutan pemrosesan

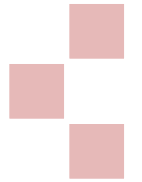


Unary Operators

Operator yang membutuhkan hanya satu operan.

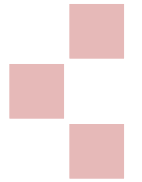
Macam-macamnya:

- Operator increment dan decrement : ++ dan - -
- Operator unary plus dan minus : + dan –
- Operator bitwise inversion : ~
- Operator boolean complement : !
- Cast : ()



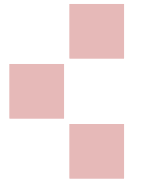
Operator Increment dan Decrement

Operator	*Penggunaan*	*Keterangan*
++	op++	Menambahkan nilai 1 pada op; mengevaluasi nilai op sebelum diincrementasi/ditambahkan
++	++op	Menambahkan nilai 1 pada op; mengevaluasi nilai op setelah diincrementasi/ditambahkan
--	op--	Mengurangkan nilai 1 pada op; mengevaluasi nilai op sebelum didecrementasi/dikurangkan
--	--op	Mengurangkan nilai 1 pada op; mengevaluasi nilai op setelah didecrementasi/dikurangkan



Unary Operator + dan -

1. X = -3;
2. Y = +3;
3. Z = -(Y+6);



The Unary Operators

- The Bitwise Inversion Operator: ~

  – converting all the 1 bits in a binary value to 0s and all the 0 bits to 1s.

- Example:

  00001111 -> 11110000

- The Boolean Complement Operator: !

  – inverts the value of a boolean expression.

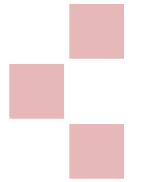
- Example:

  !true-> false

  !false-> true

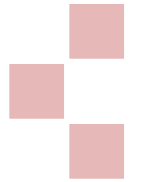
Operator Aritmatika

Operator	*Penggunaan*	*Keterangan*
+	op1 + op2	Menambahkan op1 dengan op2
*	op1 * op2	Mengalikan op1 dengan op2
/	op1 / op2	Membagi op1 dengan op2
%	op1 % op2	Menghitung sisa dari pembagian op1 dengan op2
-	op1 - op2	Mengurangkan op2 dari op1



Shift Operators

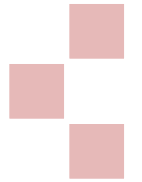
- << : left shift
- >> : signed right shift
- >>> : unsigned right shift



Right shift

```
class RightShift {
  public static void main(String[] args) {
      int i = 7;
      i = i >> 2;
      System.out.println(i);
  }
}
```

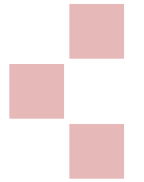
Output ???



Unsigned right shift

```
class UnsignedRightShift {
  public static void main(String[] args) {
      int i = -1;
      i = i >>> 30;
      System.out.println(i);
  }
}
```

Output ???



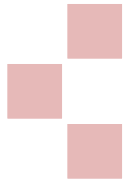
Comparison Operators

- Menghasilkan nilai boolean
- Yang termasuk comparison operator:

  - Ordinal comparison: <, <=, >, >=

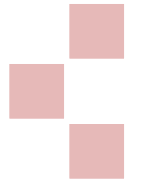
  - The **instanceof** Operator : Tests the class of an object at runtime.

  - The Equality Comparison Operators: == and !=



Ordinal and Equality Comparison Operators

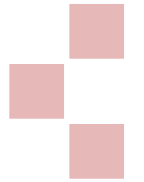
Operator	*Penggunaan*	*Keterangan*
>	op1 > op2	op1 lebih besar dari op2
>=	op1 >= op2	op1 lebih besar dari atau sama dengan op2
<	op1 < op2	op1 kurang dari op2
<=	op1 <= op2	op1 kurang dari atau sama dengan op2
==	op1 == op2	op1 sama dengan op2
!=	op1 != op2	op1 tidak sama dengan op2



The “instanceof” operator

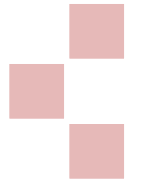
- op1 **instanceof** op2
- op1 merupakan object
- op2 merupakan class, interface, array
- Contoh:

```
String s = new String(“Java Languange”);
boolean compare1 = s instanceof String;
```



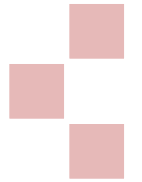
Equality for primitives

```
System.out.println('a' =='a');
System.out.println('a' =='b');
System.out.println(5 != 6);
System.out.println(5.0 == 5L);
System.out.println(true == false);
```



Equality for Reference Variables

```
String a = new String(“Java”);
String b = new String (“Java”);
String c = a;
System.out.println(a==b);
System.out.println(a==c);
```



Bitwise Operators : &, ^, |

- Provide logical AND, OR and XOR operations on integral data types.

```
public class And {
   public static void main(String args[]) {
      int i;
      i = 6 & 13;
      System.out.println("Hasil operasi & = " + i);
   }
}
```

Output ???

```
public class Or {
   public static void main(String args[]) {
      int i;
      i = 5 | 9;
      System.out.println("Hasil operasi & = " + i);
   }
}
```

Binary Operators : &, ^, |

- AND, OR and XOR operations on logical data types.
- Semua operan akan dieksekusi.

x1	*x2*	*Hasil*
TRUE	TRUE	TRUE
TRUE	FALSE	FALSE
FALSE	TRUE	FALSE
FALSE	FALSE	FALSE

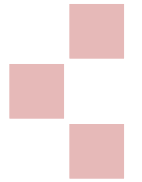
Operator &

x1	*x2*	*Hasil*
TRUE	TRUE	TRUE
TRUE	FALSE	TRUE
FALSE	TRUE	TRUE
FALSE	FALSE	FALSE

Operator |

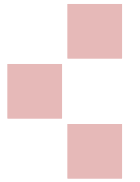
x1	*x2*	*Hasil*
TRUE	TRUE	FALSE
TRUE	FALSE	TRUE
FALSE	TRUE	TRUE
FALSE	FALSE	FALSE

Operator ^



Short Circuit Logical Operators : &&, ||

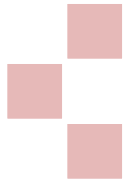
- Applicable **only to boolean** values and not integral types.
- For an AND operation, if one operand is false, the result is false, without regard to the other operand.
- For an OR operation, if one operand is true, the result is true, without regard to the other operand.
- For boolean expression X :
  - false && X = false
  - true || X = true



Short Circuit &&

```
public class ShortCircuitBooleanAnd {
    public static void main(String args[]) {
        int a=5, b=7;
        if ((a<2) && (b++<10)) b+=2;
        System.out.println(b);
    }
}
```

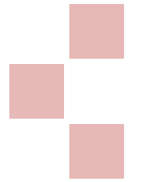
Output ???



Short Circuit ||

```
public class ShortCircuitBooleanOr {
    public static void main(String args[]) {
        int a=5, b=7;
        if ((a>2) || (b++<10)) b+=2;
        System.out.println(b);
    }
}
```

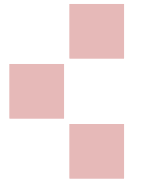
Output ???



Conditional Operators “?:”

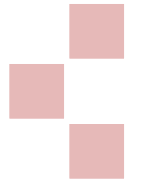
- Dikenal dengan ternary operator
- Butuh 3 operand
- Menyederhanakan if-else
- Sintaks :

  exp1?exp2:exp3 Dimana nilai exp1 adalah suatu pernyataan boolean
- Jika exp1 bernilai true, exp2 merupakan hasil operasi. Jika bernilai false, kemudian exp3 merupakan hasil operasinya.



Conditional Operators “?:”

```
String status = "";
int grade = 80;
//mendapatkan status pelajar
status = (grade >= 60)?"Passed":"Fail";
//print status
System.out.println( status );
```

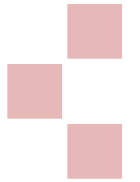


Assignment Operators

- `=`
- `+= -= *= /= %= <<=` ...
  - You can use any of the arith/logic operators with assignment.

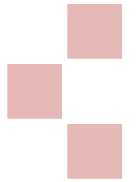
- An assignment expression has a value (just like in C/C++):

```
a = (b = 3);
x += (y -=2) +2;
```



Operator Precedence

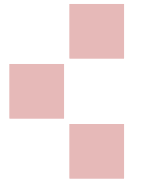
Category	Operators
Unary	++ -- + - ! ~ ()
Arithmetic	* / %
	+ -
Shift	<< >> >>>
Comparison	< <= > >= instanceof
	== !=
Bitwise	& ^ |
Short-circuit	&& ||
Conditional	?:
Assignment	= "op="



Kasus Operator Precedence

- int a = 6%2*5+4/2+88-10
- Berapa nilai a ?
- int [] a = {4, 4};
- int b = 1;
- a[b] = b = 0;

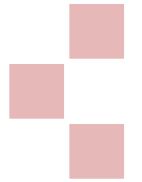
  untuk assignment berlaku aturan asosiatif ?? dari kanan ke kiri.
- a[b] -> a[1]
- b = 0
- a[1] = 0



The + String Operator

- Strings in Java are objects (just like any other object), except that the language includes special support:
  - String literals are part of the language.
  - The string concatenation operator **+** is part of the language.

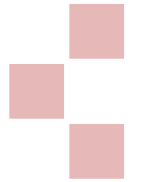
```
System.out.println("Hello " + "World");
```



Operator Syntax

- **Binary operators**
  - *value1 operator value2*
    - a + 2
    - b * b
    - c / y
- **Unary operators**
  - single value
    - a++
    - -b
- **Ternary operators**
  - exp1 ? exp2 : exp3

    (a<b) ? c : d



Referensi lain

Lihat Modul JENI-Intro1-Bab-04-Dasar2 Pemrograman Java hal 16

Pertanyaan ???