CSE3040 Java Language

Lecture #03

Dept. of Computer Engineering,
Sogang University

1.4. Arithmetic Operators

- Arithmetic operations of Java is also similar to that of C and C++.

- Assignment (=)
  - x = *expression*

- Basic arithmetic (+, -, *, /, %)
  - When dividing numbers, if the two numbers are integers, the result also becomes an integer. If you want a floating-point number as a result, at least one of the operands should be a floating-point type.

  - 17.0 / 5 → 3.4
  - 17 / 5 → 3

1.4. Arithmetic Operators

- Unary operators
  - n++; // increment n
  - n--; // decrement n

- Difference between n++ and ++n;
  - n++: evaluate the line, and then increment n
  - ++n: increment n, and evaluate the line.

  - String arg = args[n++]
    - arg becomes args[n], and then n is incremented.

  - String arg = args[++n]
    - n is incremented, and the assignment is done.

  - The resulting value of arg is different!

1.4. Arithmetic Operators

```
class opPlus {
   public static void main(String[] args)
   {
     int  x=1;
     System.out.println("x:" + x);
     for (int i=1 ; i<=3 ; i++)
          System.out.print("x:" + x++ + "  "); // x++ : x incremented after print
     System.out.println("\nx:" + x);                    // x  ← 4
     x = x*10;
     System.out.println("\nx:" + x);                    // x  ← 40
     for (int i=1 ; i<=3 ; i++)
          System.out.print("x:" + ++x + "  "); // x++ :  x printed after increment
     System.out.println("\nx:" + x);                    // x ← 43
   }
}
```

1.4. Arithmetic Operators

- Mathematics methods
  - Math.pow(x, y): returns x^y
  - Math.sqrt(x): square root of x.
  - Math.min
  - Math.max
  - Math.PI
  - Math.E
  - Math.random(): returns a random number in the range [0, 1).

- These methods are called static methods.
  - opposite: instance methods
  - They are declared using keyword static.
  - These methods can be used without creating an instance.
  - Calling format: *class_name.method_name*.

1.4. Arithmetic Operators

- If two operands in an arithmetic operation are of different types, an automatic type conversion occurs to match the type of the two variables.

```
3.14 + 42
```

- Type conversion occurs in the following order:
  - If one of the operands is double, then the other operand becomes double.
  - Else if one of the operands is float, then the other operand becomes float.
  - Else if one of the operands is long, then the other operand becomes long.
  - Otherwise, the two operands become int types.

1.4. Arithmetic Operators

- Typecasting
  - The programmer can explicitly change the type of a variable

```
double x = 3.75
int n = (int) x;
```

```
int n = 1;
char next = (char)('J' + n);
```

```
int n = (int) 3000000000L;
```

1.4. Arithmetic Operators

- Type conversion using Casting
  - automatic conversion

    byte → short → int → long → float → double

    When you assign right data type to left, you must use cast operator.(to left-hand side data type)

```
double d = 100;          // Okay!
short s = 3.14;          // compile error!  Must be    short s = (short) 3.14;

double d = 100;
long l = d;   // compile error!  Must be    long l = (long) d;
```

    byte : -128 ~ 127

    short: -32768 ~ 32767

    short ← byte (OK) byte ← short (may cause overflow)

1.4. Arithmetic Operators

```
 class dtCastJungSoo {
   public static void main(String[] args)
      {
      byte   b = (byte)100000;
      short  s = (short)100000;

      System.out.println("b : " + b);
      System.out.println("s : " + s);
     }
}
```

1.4. Arithmetic Operators

- Arithmetic operation and data type
  - int → long → float → double
  - Division between integer type(int , long) is integer
  - short s = 100 + 1;
    - 100+1 is integer type. It is impossible to assign int type to short typed. -> compile error
    - short s = (short)(100+1);
  - int n = 1; short s = 100 - n ;
    - 100 - n is integer type. -> compile error
    - short s = (short)(100-n);
  - double d = 5/4;
    - 5,4 is integer. 5/4 is 1. Therefore, d is 1.
  - double d = 5/4.0;
    - 4.0 is double type. So 5/4.0 is double type. d is 1.25.
    - 4.0 is equal to 4.
  - double d = 5/4f;
    - 4f is float type. So 5/4.0 is float type. d is 1.25.

1.4. Arithmetic Operators

- Relational operators and logical operators
  - == (equal)
  - != (not equal)
  - < (less than)
  - > (greater than)
  - <= (less than or equal)
  - >= (greater than or equal)
  - These operators return either true or false.
  - Relational operators can be used with,
    - && (logical AND)
    - || (logical OR)
    - ! (logical NOT)
  - If multiple propositions are combined using '&&' and the first proposition is false, the second proposition is not evaluated.
  - If multiple propositions are combined using '||' and the first proposition it true, the second proposition is not evaluated.

1.4. Arithmetic Operators

```
class opBiGyo {
        public static void main(String[] args)
        {
                System.out.println("3>2 : " + (3>2));  // true
                System.out.println("1>2 : " + (1>2));  // false
        }
}
```

1.4. Arithmetic Operators

- Multiple propositions combined

```
n != 0 && s + (100 - s) / n < 50
```

  - If n == 0, the second condition will cause division by zero.
  - However, if n==0, the second condition is not evaluated, since it is false anyway.
  - So this statement does not produce an error.

```
n == 0 || s + (100 - s) / n >= 50
```

  - Similarly, this statement does not produce an error either.

1.4. Arithmetic Operators

- Assignment operators

  - Arithmetic Assignment: + = – = * = / = % =
  - Bit Assignment: << = (shift a bit pattern to the left)
    >> = (shift a bit pattern to the right)
    | = (bit OR) & = (bit AND) ^ = (bit XOR)

```
j + = 5;          // j ← j + 5
j – = (a – 3)     // j ← j – (a – 3)
j * = 10          // j ← j * 10
j / = 10          // j ← j / 10
j % = 10          // j ← j % 10
```

1.4. Arithmetic Operators

```
class opDaeIb {
   public static void main(String[] args)
   {
      int  a=3, b, c, d;

        System.out.println(" a+=5 Gab : " + (a += 5) + '\n');
                                          // (a = a + 5)

      a=b=c=d=10;                  // a,b,c,d <- 10

      a += b -= c *= d /= 5;

      System.out.println(" a : " + a);
      System.out.println(" b : " + b);
      System.out.println(" c : " + c);
      System.out.println(" d : " + d);
      }
}
```

Programming Lab #03

03-1. Unary Operators

- What will be printed on the display when you execute this program?
- Guess first, and then run this program and see the result for yourself.

```
public class Ex03_1 {
  public static void main(String[] args) {
    int x = 1;
    System.out.println("x: " + x);

    for(int i=1; i<=3; i++)
      System.out.print("x: " + x++ + " ");
    System.out.println("\nx: " + x);

    x = x * 10;
    System.out.println("\nx: " + x);

    for(int i=1; i<=3; i++)
      System.out.print("x: " + ++x + " ");

    System.out.println("\nx: " + x);
  }
}
```

03-2. Typecasting

- What will be printed on the display when you execute this program?
- Guess first, and then run this program and see the result for yourself.
- What happens if you remove `(byte)` or `(short)`? Can you explain why?
- What happens if you remove `(long)`? Can you explain why?

```
public class Ex03_2 {
  public static void main(String[] args) {
    byte b = (byte)100000;
    short s = (short)100000;
    int i = (int)100000;
    long l = (long)100000;
    System.out.println("b: " + b);
    System.out.println("s: " + s);
    System.out.println("i: " + i);
    System.out.println("l: " + l);
  }
}
```

03-3. Divisions

- What will be printed on the display when you execute this program?
- Guess first, and then run this program and see the result for yourself.

```
public class Ex03_3 {
  public static void main(String[] args) {
    double a = 5 / 4;
    double b = 5 / 4.0;
    double c = 5 / 4f;
    System.out.println("a: " + a);
    System.out.println("b: " + b);
    System.out.println("c: " + c);
  }
}
```

03-4. Relational Operators

- What will be printed on the display when you execute this program?
- Guess first, and then run this program and see the result for yourself.
- What happens if you remove the part "n != 0" or "n==0"? Try it.

```
public class Ex03_4 {
  public static void main(String[] args) {
    System.out.println("3>2: " + (3>2));
    System.out.println("1>2: " + (1>2));

    int n = 0, s = 0;
    System.out.println("n != 0 && s + (100 - s) / n < 50: " +
                       (n != 0 && s + (100 - s) / n < 50));
    System.out.println("n == 0 || s + (100 – s) / n >= 50: " +
                       (n == 0 || s + (100 – s) / n >= 50));
  }
}
```

03-5. Assignment Operators

- What will be printed on the display when you execute this program?
- Guess first, and then run this program and see the result for yourself.

```
public class Ex03_5 {
  public static void main(String[] args) {
    int a = 3, b, c, d;
    System.out.println("a += 5: " + (a += 5));

    a = b = c = d = 10;
    a += b -= c *= d /= 5;

    System.out.println("a: " + a);
    System.out.println("b: " + b);
    System.out.println("c: " + c);
    System.out.println("d: " + d);
  }
}
```

End of Class

Instructor office: AS818A

Email: jso1@sogang.ac.kr