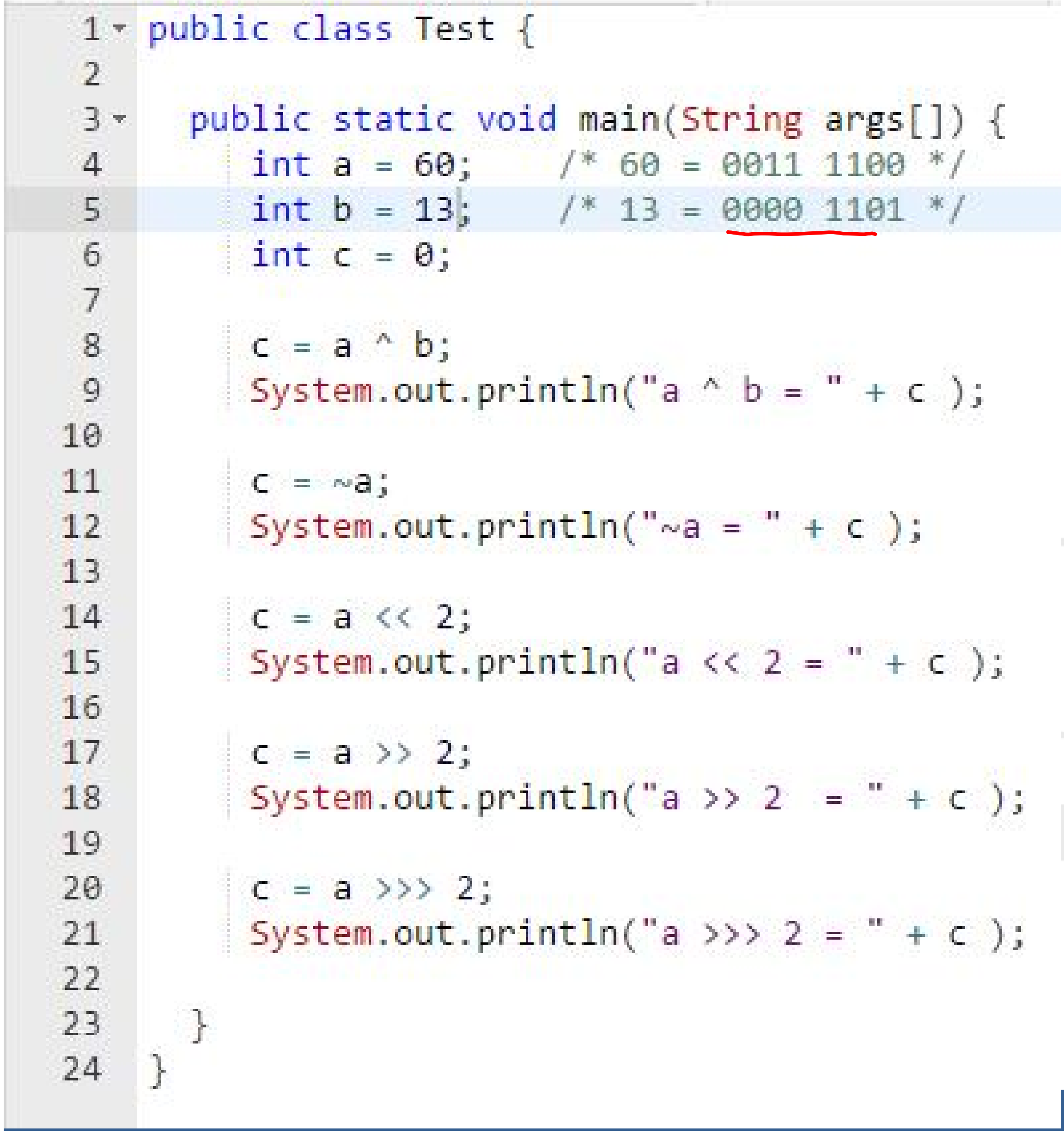
INDIAN INSTITUTE OF TECHNOLOGY ROORKEE

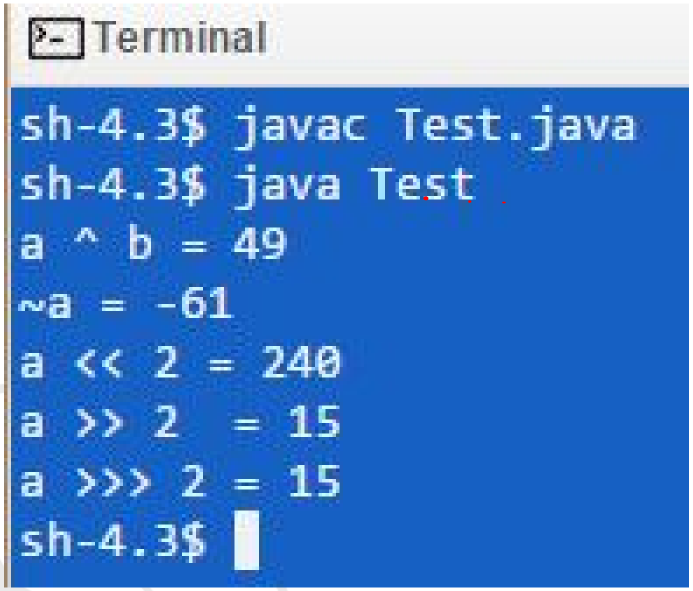
Fundamentals of Object Oriented Programming

CSN- 103

Dr. R. Balasubramanian
Associate Professor
Department of Computer Science and Engineering
Indian Institute of Technology Roorkee
Roorkee 247 667
balarfcs@iitr.ac.in
https://sites.google.com/site/balaiitr/



```java
public class Test {

  public static void main(String args[]) {
     int a = 60;    /* 60 = 0011 1100 */
     int b = 13;    /* 13 = 0000 1101 */
     int c = 0;

     c = a ^ b;
     System.out.println("a ^ b = " + c );

     c = ~a;
     System.out.println("~a = " + c );

     c = a << 2;
     System.out.println("a << 2 = " + c );

     c = a >> 2;
     System.out.println("a >> 2  = " + c );

     c = a >>> 2;
     System.out.println("a >>> 2 = " + c );

  }
}
```



```
sh-4.3$ javac Test.java
sh-4.3$ java Test
a ^ b = 49
~a = -61
a << 2 = 240
a >> 2  = 15
a >>> 2 = 15
sh-4.3$
```

$\text{floor}\ \lfloor 2.7 \rfloor \rightarrow 2$

$\text{ceil}\ \lceil 2.7 \rceil \rightarrow 3$

8th 7th MSB (6th)

$a \rightarrow \underbrace{00\boxed{1}1\ 11\ 00}$

$\underbrace{a << \boxed{2}} \rightarrow 1111\ 00\underline{00}$

$a << b$

$-2 << 2$

$\lceil a * 2^b \rceil$

$\lceil a * 2^2 \rceil$

$60 * 4 = 240$

$a >> b = \lfloor a/2^b \rfloor$

$a >> -2$ ✗

$$\begin{array}{cccccccccc} 10 & 9 & 8 & 7 & 6 & 5 & 4 & 3 & & \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \end{array}$$

0
0
0
0
16
32
64
128

240

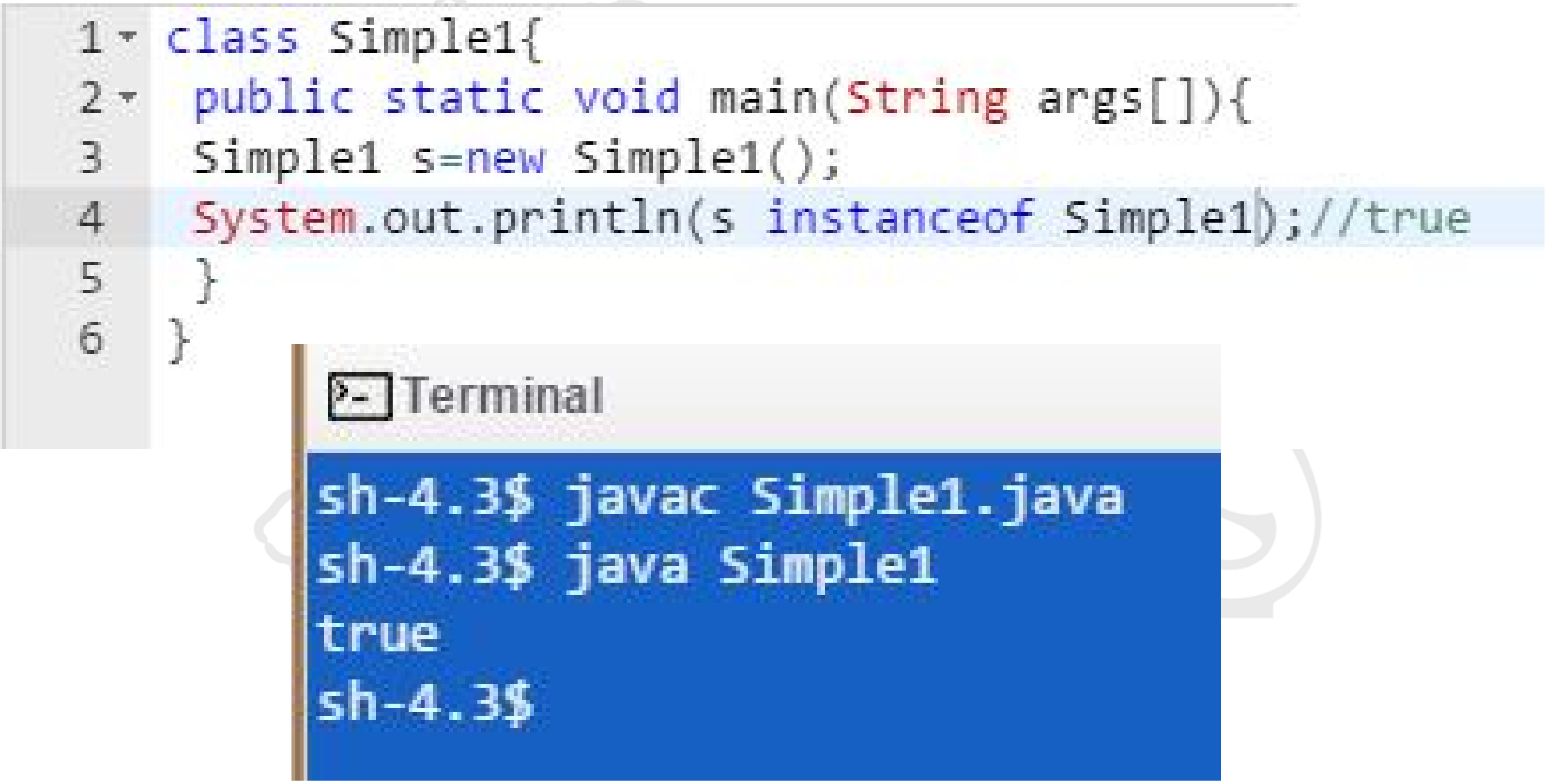
$a >> 2 \longrightarrow$ 0 000 1111 (4th, 3rd, 2, 1) $\longrightarrow \boxed{15}$

$a >> 3 \longrightarrow \left\lfloor \frac{60}{8} \right\rfloor \rightarrow 7$

$00000111 \rightarrow \boxed{7}$

Instanceof Operator

- The **java instanceof operator** is used to test whether the object is an instance of the specified type (class or subclass or interface).



```java
class Simple1{
 public static void main(String args[]){
 Simple1 s=new Simple1();
 System.out.println(s instanceof Simple1);//true
 }
}
```

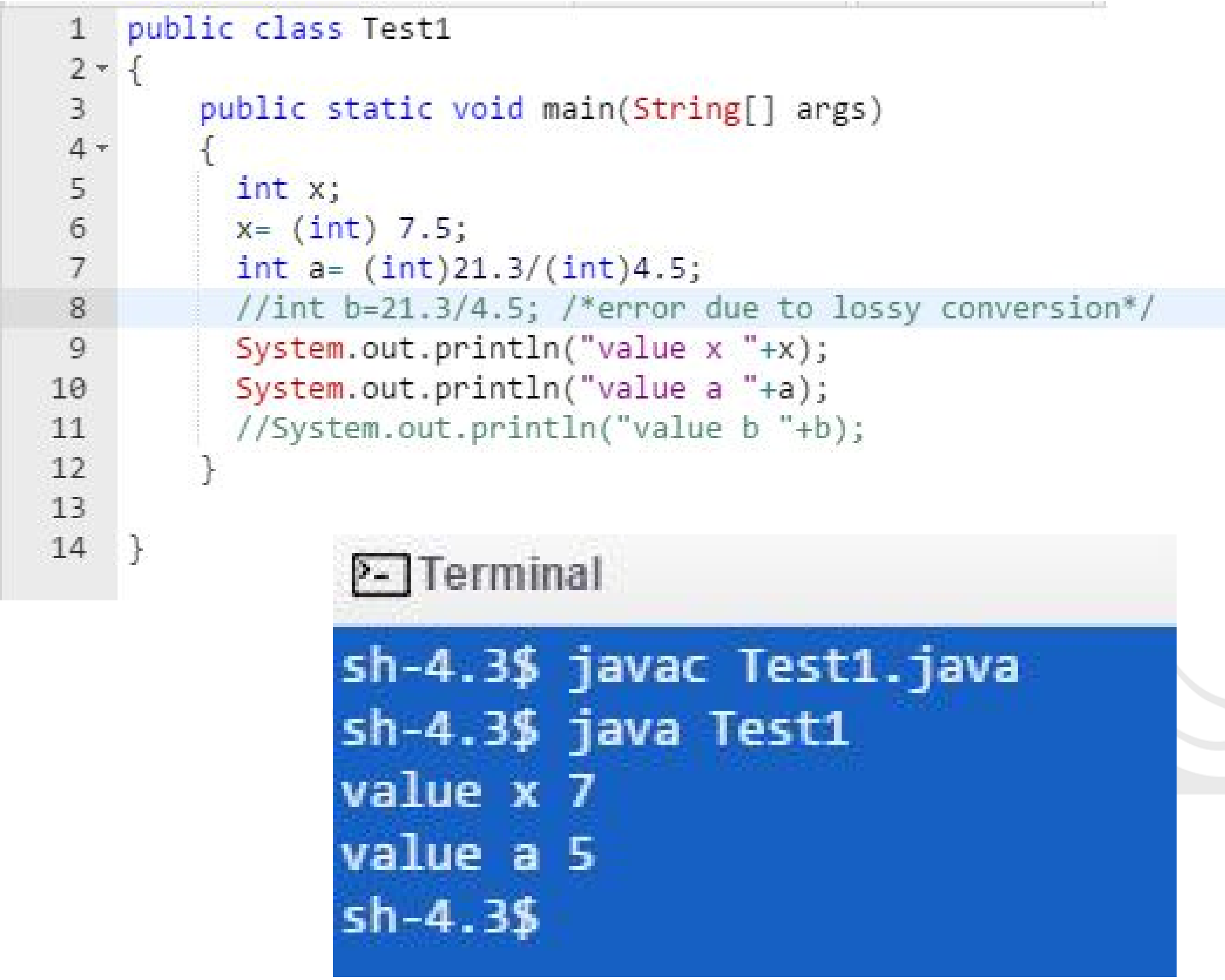
```
sh-4.3$ javac Simple1.java
sh-4.3$ java Simple1
true
sh-4.3$
```

Dot Operator

- The dot (.) operator is used to access the instance variables and methods of class objects.

```
Person1.age
Person1.salary()
```

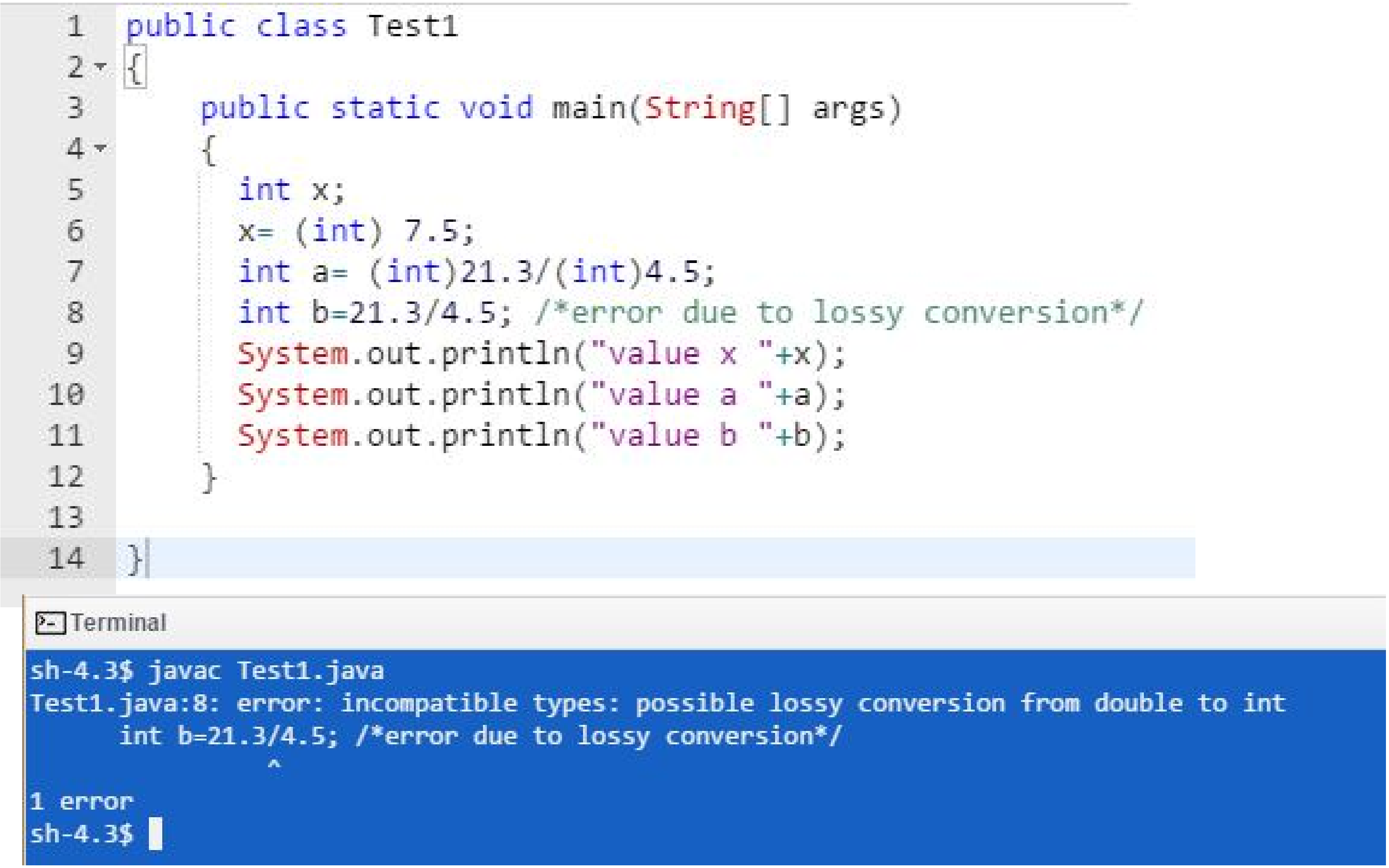
Casting a value



```java
public class Test1
{
    public static void main(String[] args)
    {
      int x;
      x= (int) 7.5;
      int a= (int)21.3/(int)4.5;
      //int b=21.3/4.5; /*error due to lossy conversion*/
      System.out.println("value x "+x);
      System.out.println("value a "+a);
      //System.out.println("value b "+b);
    }

}
```

```
sh-4.3$ javac Test1.java
sh-4.3$ java Test1
value x 7
value a 5
sh-4.3$
```

Use of Casts



```java
public class Test1
{
    public static void main(String[] args)
    {
      int x;
      x= (int) 7.5;
      int a= (int)21.3/(int)4.5;
      int b=21.3/4.5; /*error due to lossy conversion*/
      System.out.println("value x "+x);
      System.out.println("value a "+a);
      System.out.println("value b "+b);
    }

}
```

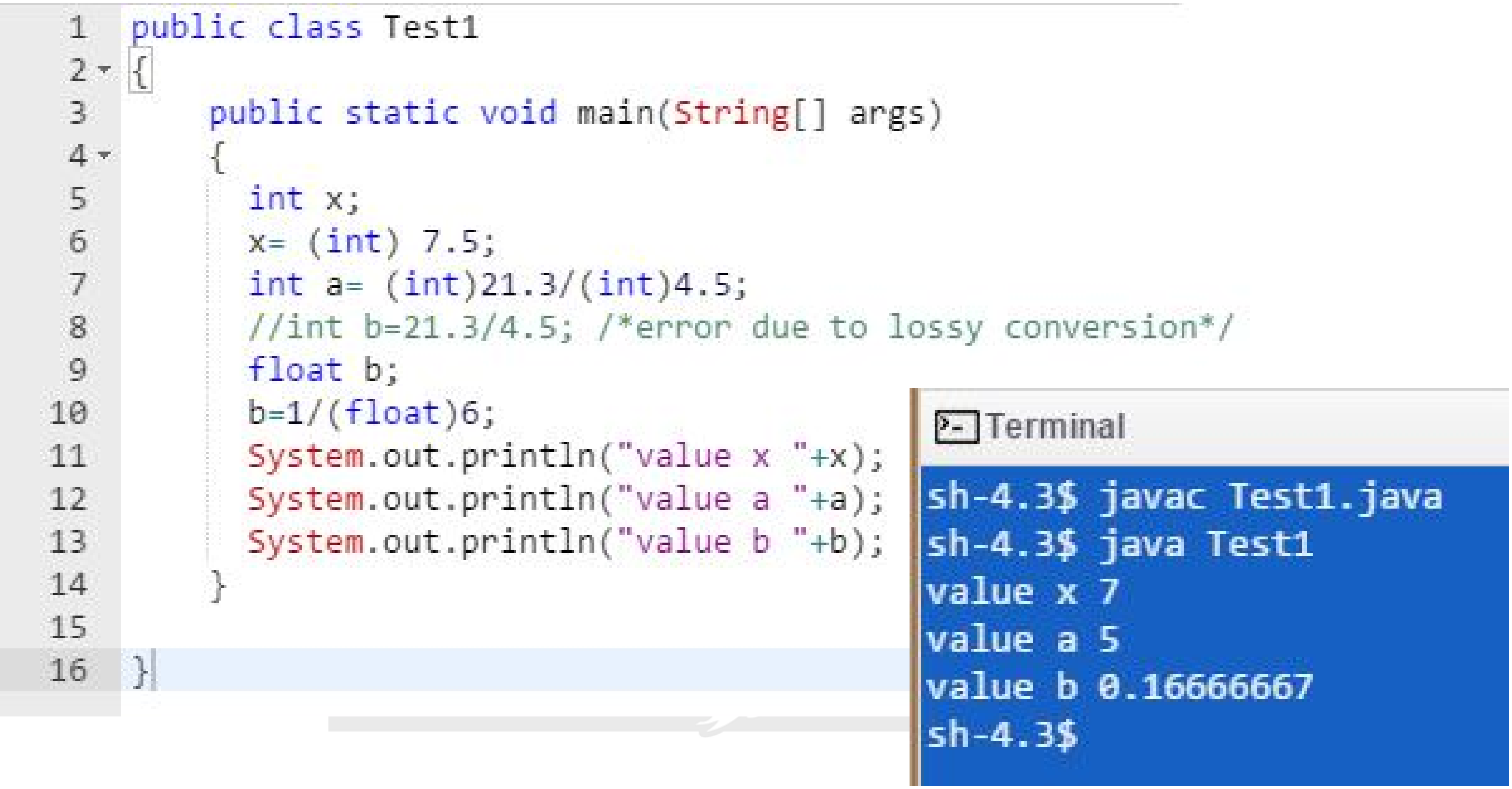
```
sh-4.3$ javac Test1.java
Test1.java:8: error: incompatible types: possible lossy conversion from double to int
      int b=21.3/4.5; /*error due to lossy conversion*/
                ^
1 error
sh-4.3$
```

Use of Casts

- Can be used for round-off a given value

```
x=(int) (y+0.5);
```

$\lfloor y+0.5 \rfloor$



```java
public class Test1
{
    public static void main(String[] args)
    {
      int x;
      x= (int) 7.5;
      int a= (int)21.3/(int)4.5;
      //int b=21.3/4.5; /*error due to lossy conversion*/
      float b;
      b=1/(float)6;
      System.out.println("value x "+x);
      System.out.println("value a "+a);
      System.out.println("value b "+b);
    }

}
```

```
sh-4.3$ javac Test1.java
sh-4.3$ java Test1
value x 7
value a 5
value b 0.16666667
sh-4.3$
```

Precedence of Arithmetic Operators

```
x=a-b/3+c*2-1;
a=9, b=12, c=3;
```

$12/3 \to 4$

$c * \overset{2}{\cancel{2}} \to 6$

$x = 9 - 4 + 6 - 1$

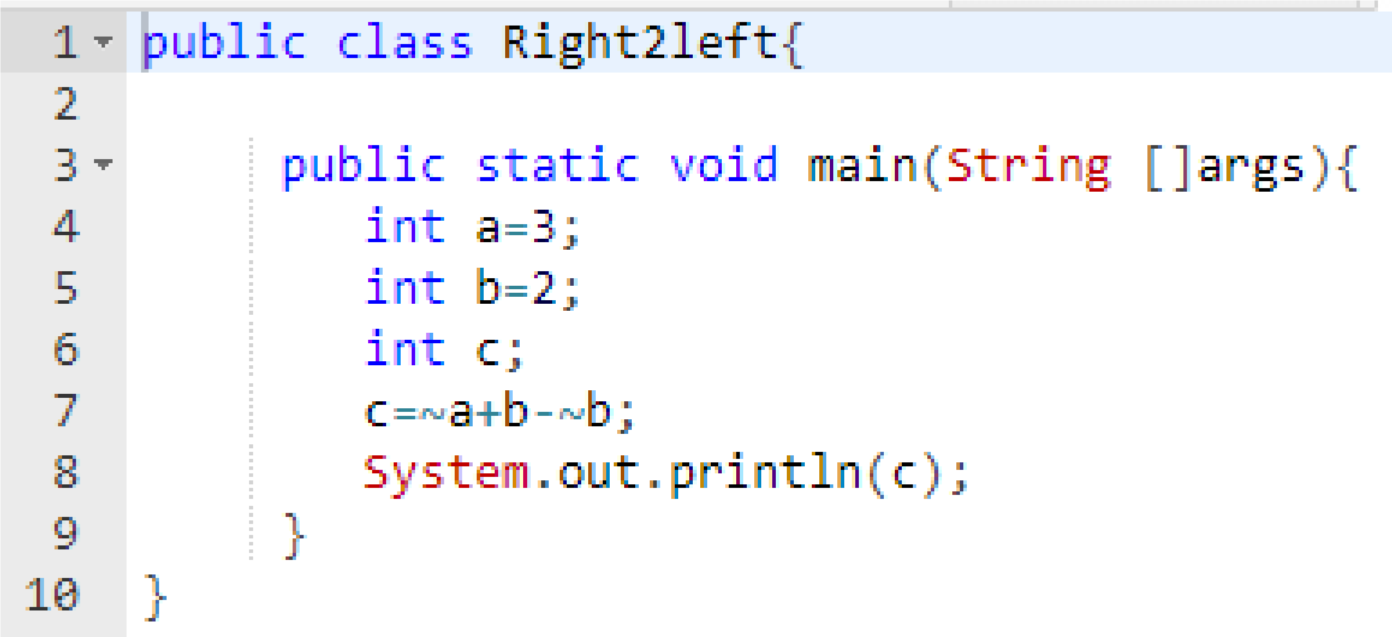
Operator Precedence and Associativity

Operator	Description	Level	Associativity
[] . () ++ --	access array element access object member invoke a method post-increment post-decrement	1	left to right
++ -- + - ! ~	pre-increment pre-decrement unary plus unary minus logical NOT bitwise NOT	2	right to left
() new	cast object creation	3	right to left
* / %	multiplicative	4	left to right

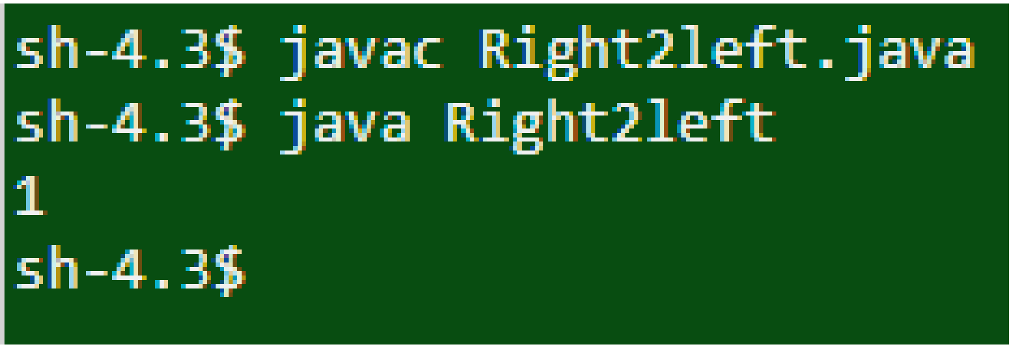
+ − +	Additive string concatenation	5	left to right
<< >> >>>	shift	6	left to right
< <= > >= instanceof	relational type comparison	7	left to right
== !=	equality	8	left to right

&	bitwise AND	9	left to right
^	bitwise XOR	10	left to right
|	bitwise OR	11	left to right
&&	conditional AND	12	left to right
||	conditional OR	13	left to right
?:	conditional	14	right to left
= += -= *= /= %= &= ^= |= <<= >>= >>>=	assignment	15	right to left

Right to Left



```java
public class Right2left{

	public static void main(String []args){
		int a=3;
		int b=2;
		int c;
		c=~a+b-~b;
		System.out.println(c);
	}
}

```



```
sh-4.3$ javac Right2left.java
sh-4.3$ java Right2left
1
sh-4.3$
```

https://goo.gl/4ziHht