



Programming Concepts

Week 3



Today's Topics

- Data Conversion
 - Chapter 2, Section 2.5
- Classes and Objects
 - Chapter 3, Sections 3.1 - 3.3, 3.5, 3.6



Data and Expressions

Chapter 2

Section 2.5



Objectives

You will be able to:

- Work with representations of numeric values as different *types*.
- Safely convert numbers from one type to another.
- Write expressions with integer values that produce floating point results.



Numeric Data Types

- Every number that a Java program uses is stored as a specific *type*.
- We often need to convert a number from one type to another.
 - Exact conversion
 - Rounding conversion
 - Lossy conversion



Integer Data Types

- The difference between the various integer primitive types is their size, and therefore the range of values that they can store:

| Type | Size | Min Value | Max Value |
|-------|---------|-----------------------|----------------------|
| byte | 1 byte | -128 | 127 |
| short | 2 bytes | -32,768 | 32,767 |
| int | 4 bytes | -2,147,483,648 | 2,147,483,647 |
| long | 8 bytes | $< -9 \times 10^{18}$ | $> 9 \times 10^{18}$ |



Floating Point Types

| Type | Size | Range |
|--------|---------|--|
| float | 4 bytes | +/- 3.4×10^{38} with 7 significant digits |
| double | 8 bytes | +/- 1.7×10^{308} with 15 significant digits |



Kinds of Data Conversions

- Conversions
 - Widening
 - Narrowing
- Widening conversions cannot lose information.
 - Can happen automatically
- Narrowing conversion can lose information.
 - Must be explicitly requested by the program



Conversion Techniques

- Assignment
- Promotion
- Casting



Assignment Conversion

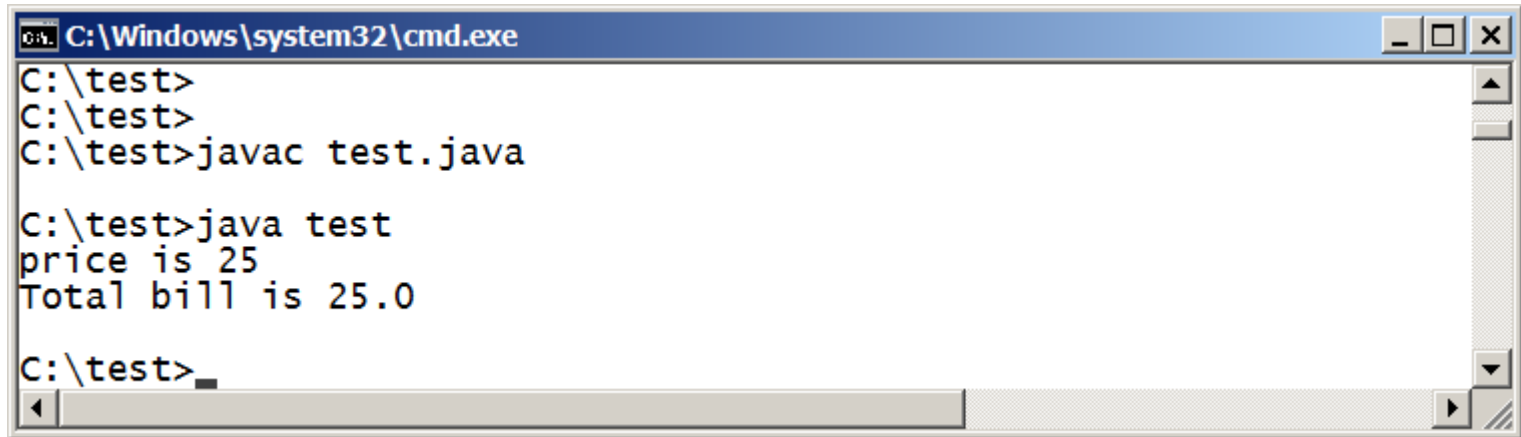
- Happens implicitly when we assign a value to a variable of a different type.
 - Cannot be a narrowing conversion.

```
class test
{
    public static void main (String[] args)
    {
        int price = 25;
        System.out.println("price is " + price);

        float total_bill = price;
        System.out.println("Total bill is " + total_bill);
    }
}
```



Program Running

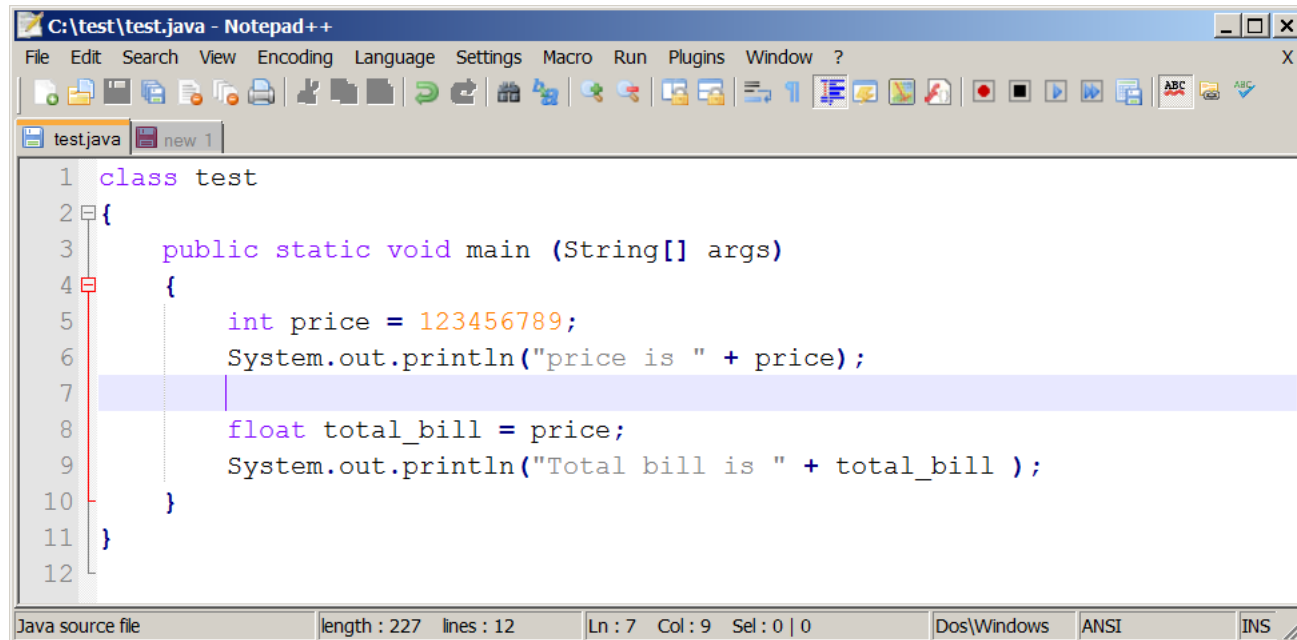


```
C:\Windows\system32\cmd.exe
C:\test>
C:\test>
C:\test>javac test.java

C:\test>java test
price is 25
Total bill is 25.0

C:\test>
```

Conversion May Not Be Exact

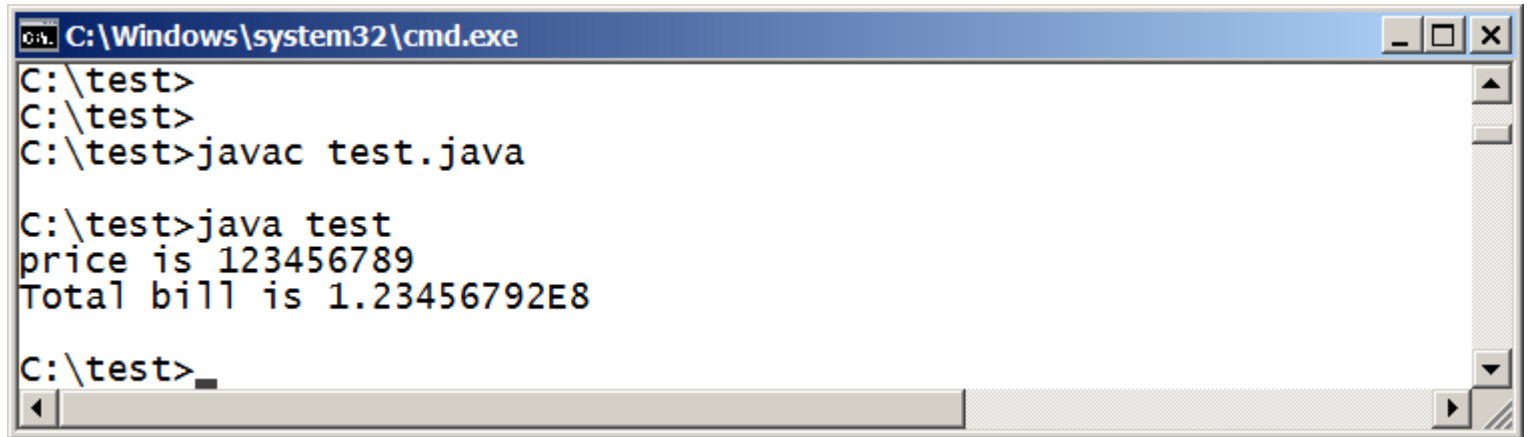


```
1 class test
2 {
3     public static void main (String[] args)
4     {
5         int price = 123456789;
6         System.out.println("price is " + price);
7
8         float total_bill = price;
9         System.out.println("Total bill is " + total_bill );
10    }
11 }
12
```

The screenshot shows a Notepad++ window titled "C:\test\test.java - Notepad++". The code is a Java class named "test" with a "main" method. It declares an integer "price" with the value 123456789 and prints it. Then, it declares a float "total_bill" and assigns it the value of "price". The status bar at the bottom indicates the file is a "Java source file", has a length of 227, 12 lines, and the cursor is at line 7, column 9.



Conversion May Not Be Exact



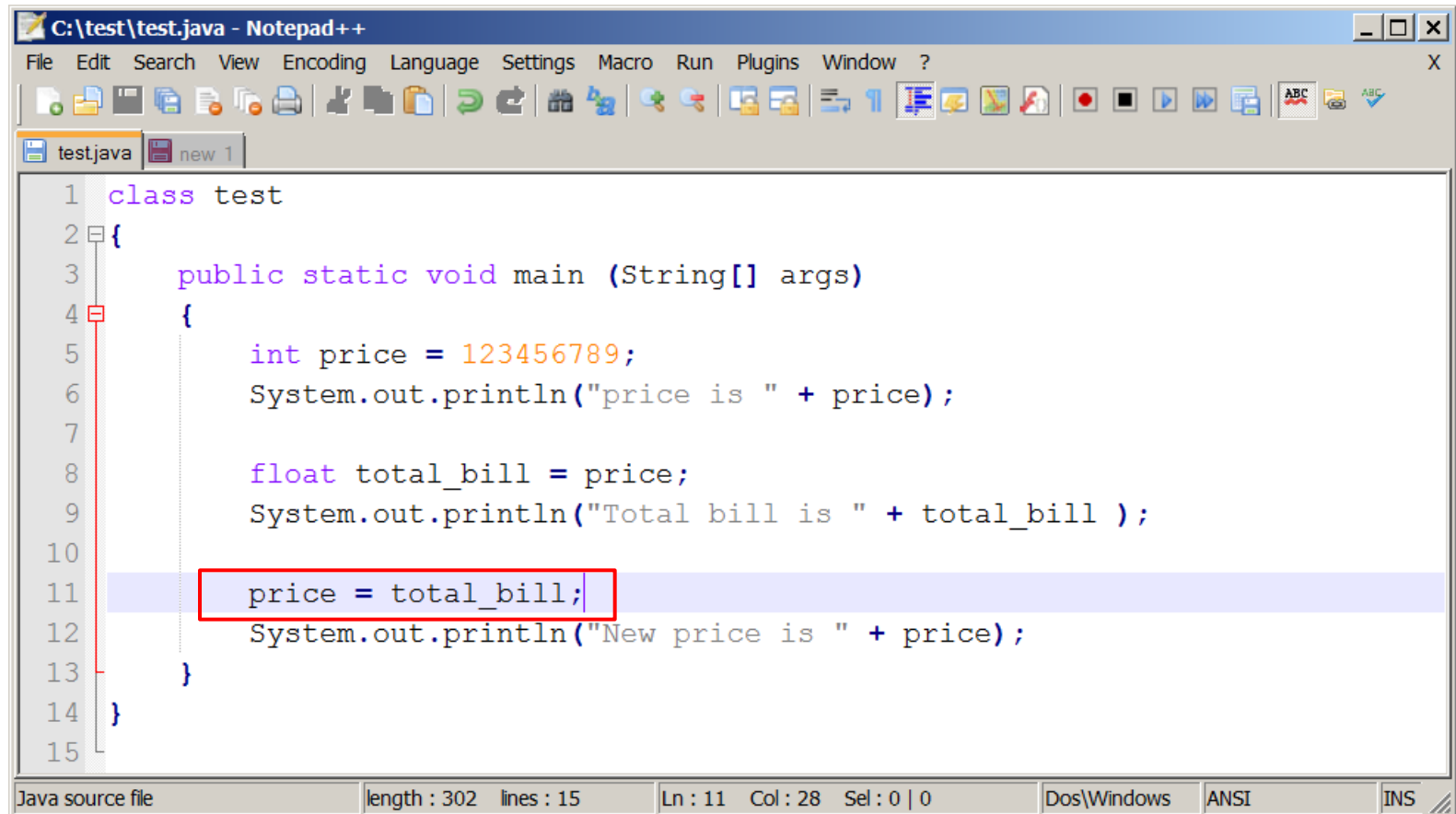
```
C:\Windows\system32\cmd.exe
C:\test>
C:\test>
C:\test>javac test.java

C:\test>java test
price is 123456789
Total bill is 1.23456792E8

C:\test>
```

Even though this was not a narrowing conversion, and the compiler did not object, the result is not exact.

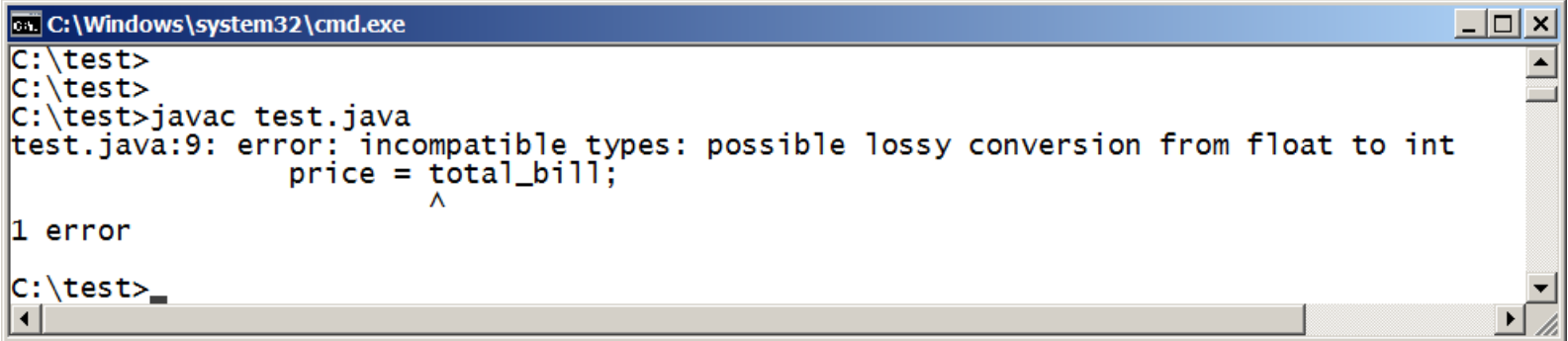
Try Converting Back



```
1 class test
2 {
3     public static void main (String[] args)
4     {
5         int price = 123456789;
6         System.out.println("price is " + price);
7
8         float total_bill = price;
9         System.out.println("Total bill is " + total_bill );
10
11     price = total_bill;
12     System.out.println("New price is " + price);
13 }
14 }
15
```

Java source file | length : 302 | lines : 15 | Ln : 11 | Col : 28 | Sel : 0 | 0 | Dos\Windows | ANSI | INS

Error!

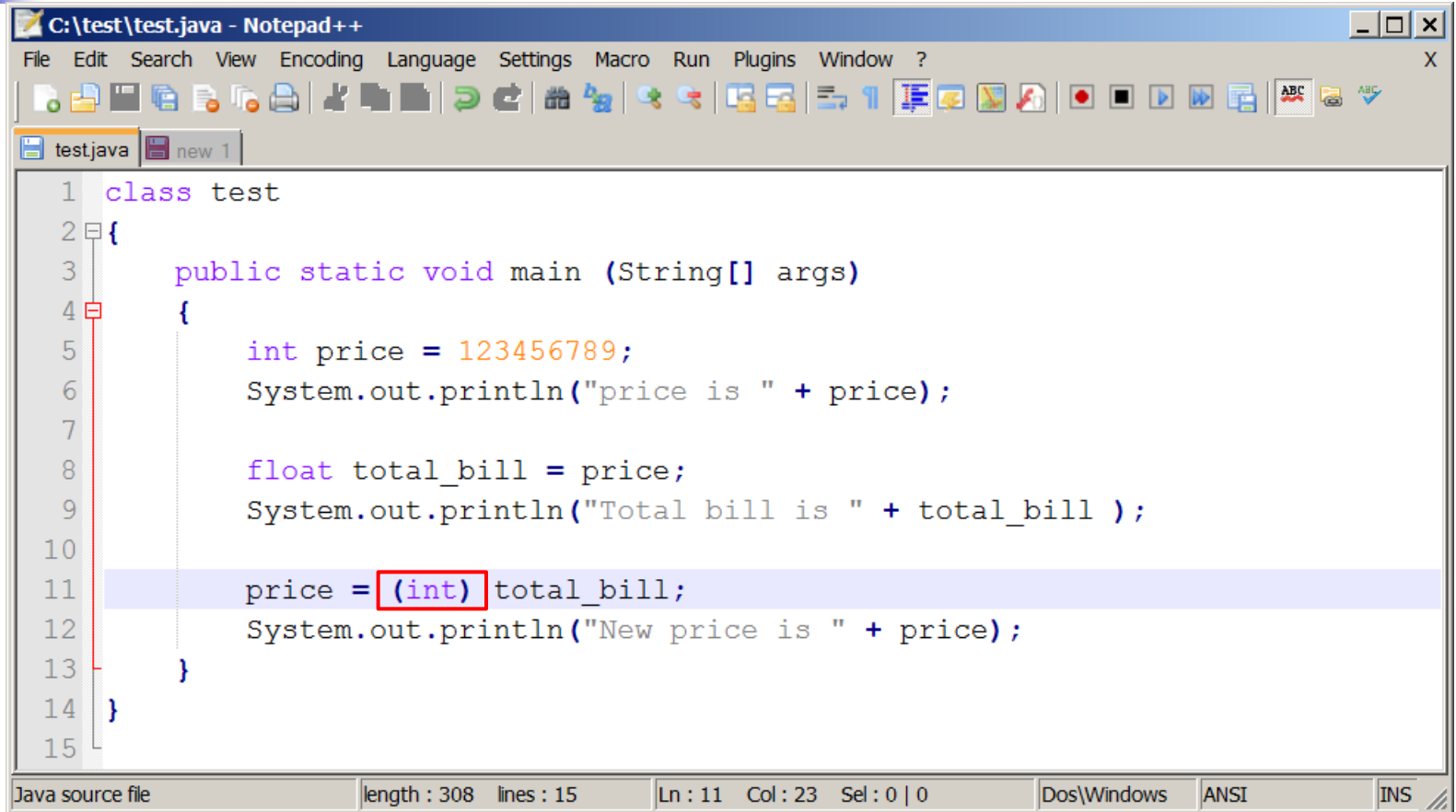


```
C:\Windows\system32\cmd.exe
C:\test>
C:\test>
C:\test>javac test.java
test.java:9: error: incompatible types: possible lossy conversion from float to int
    price = total_bill;
           ^
1 error
C:\test>
```

The compiler will not implicitly do a conversion that could possibly lose information.

We can force it to do the conversion with a *cast*.

Casting a float as int



```
1 class test
2 {
3     public static void main (String[] args)
4     {
5         int price = 123456789;
6         System.out.println("price is " + price);
7
8         float total_bill = price;
9         System.out.println("Total bill is " + total_bill );
10
11        price = (int) total_bill;
12        System.out.println("New price is " + price);
13    }
14 }
15
```

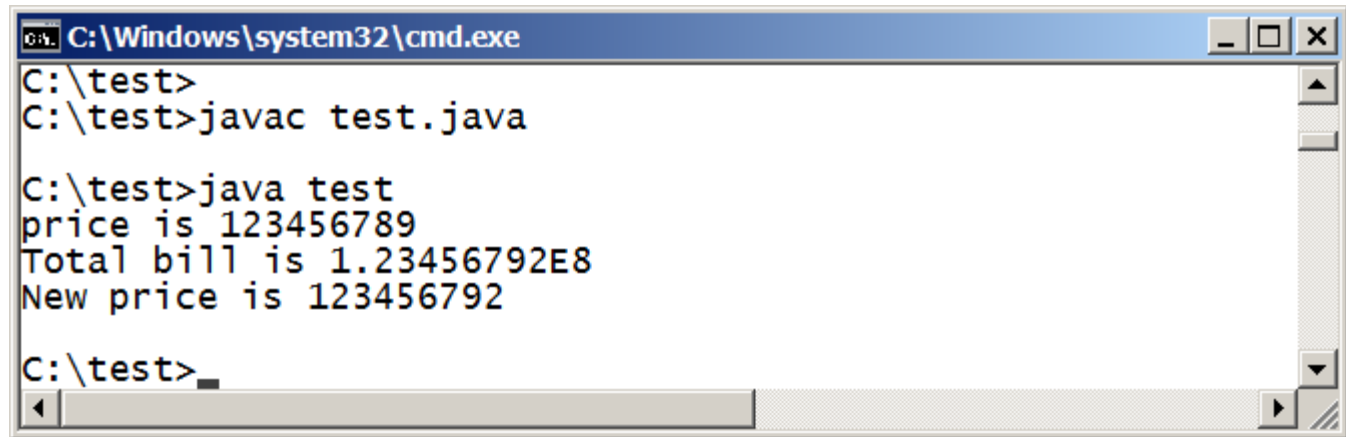
The screenshot shows a Notepad++ window titled "C:\test\test.java - Notepad++". The menu bar includes File, Edit, Search, View, Encoding, Language, Settings, Macro, Run, Plugins, Window, and ?. The toolbar contains various icons for file operations and editing. The editor shows a Java class named 'test' with a 'main' method. The code assigns an integer value to 'price', converts it to a float 'total_bill', and then casts 'total_bill' back to an integer 'price' using '(int)'. The status bar at the bottom indicates 'Java source file', 'length : 308 lines : 15', and cursor position 'Ln : 11 Col : 23 Sel : 0 | 0'. The encoding is set to 'ANSI' and the input language is 'INS'.

The cast says that we know what we are doing and are willing to take responsibility for loss of information.



Round trip conversion

has changed the value of price



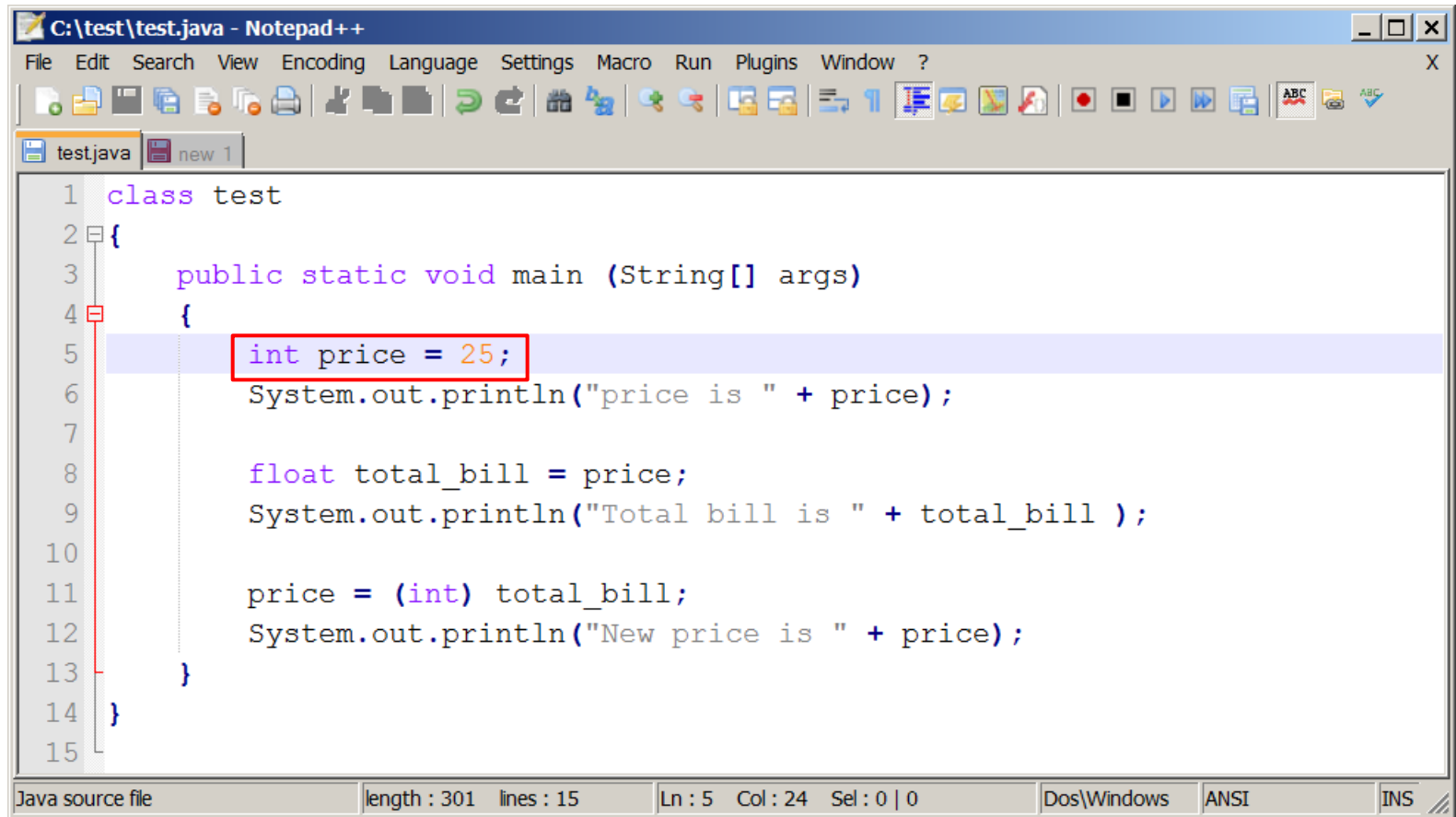
```
C:\Windows\system32\cmd.exe
C:\test>
C:\test>javac test.java

C:\test>java test
price is 123456789
Total bill is 1.23456792E8
New price is 123456792

C:\test>
```

An accountant would not like this!

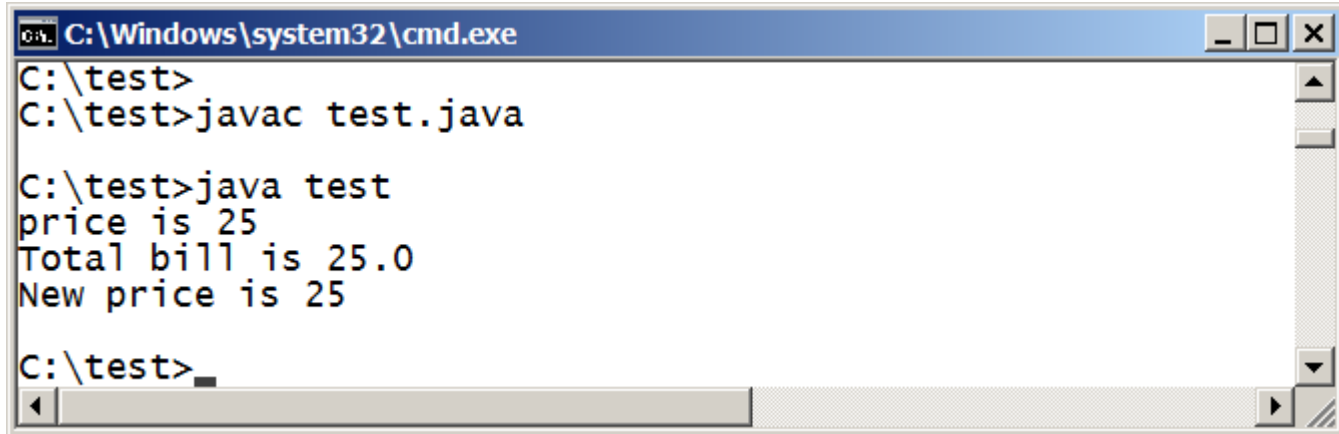
What if the price had been lower?



```
C:\test\test.java - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
test.java new 1
1 class test
2 {
3     public static void main (String[] args)
4     {
5         int price = 25;
6         System.out.println("price is " + price);
7
8         float total_bill = price;
9         System.out.println("Total bill is " + total_bill );
10
11        price = (int) total_bill;
12        System.out.println("New price is " + price);
13    }
14 }
15
Java source file    length : 301    lines : 15    Ln : 5    Col : 24    Sel : 0 | 0    Dos\Windows    ANSI    INS
```



Round trip conversion is exact



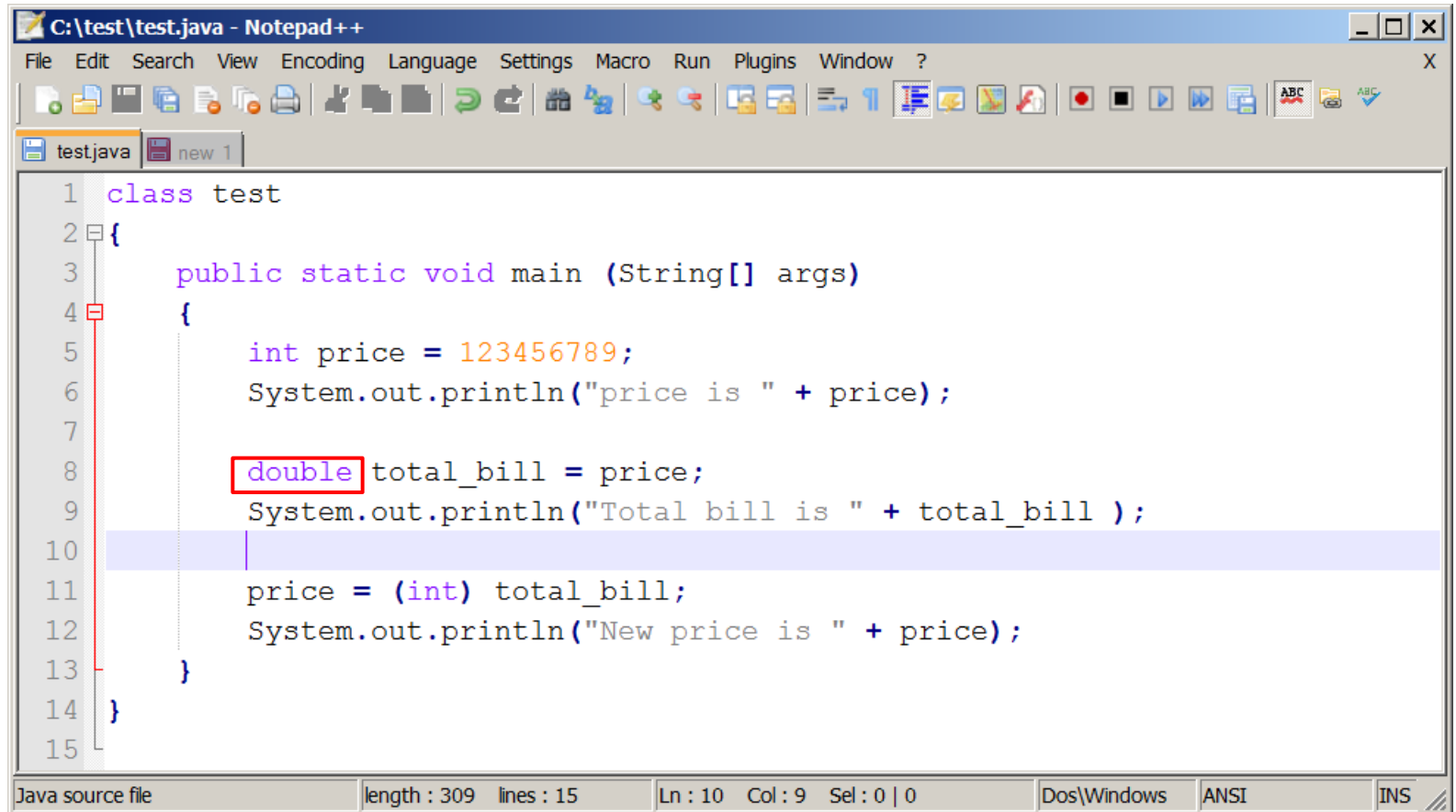
```
C:\Windows\system32\cmd.exe
C:\test>
C:\test>javac test.java

C:\test>java test
price is 25
Total bill is 25.0
New price is 25

C:\test>
```

The float type is only good for seven significant digits.

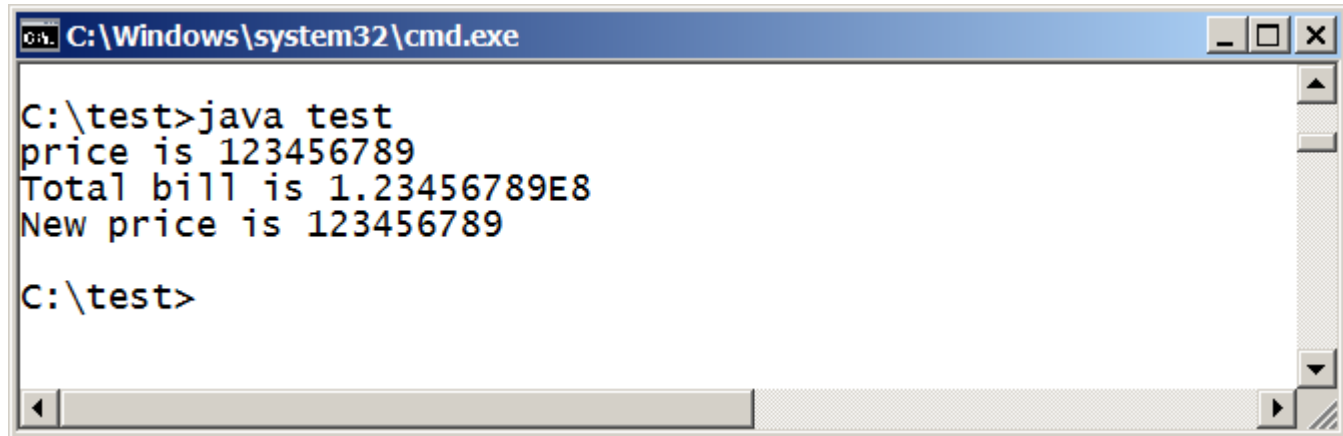
Try it with double

A screenshot of a Notepad++ window titled "C:\test\test.java - Notepad++". The window contains Java code for a class named "test". The code defines a "main" method that declares an integer "price" with the value 123456789, prints it, then declares a "double" variable "total_bill" and assigns it the value of "price". The "double" keyword is highlighted with a red box. Finally, it casts "total_bill" back to an integer and prints the new value. The status bar at the bottom shows "Java source file", "length : 309 lines : 15", and cursor position "Ln : 10 Col : 9 Sel : 0 | 0".

```
1 class test
2 {
3     public static void main (String[] args)
4     {
5         int price = 123456789;
6         System.out.println("price is " + price);
7
8         double total_bill = price;
9         System.out.println("Total bill is " + total_bill );
10
11         price = (int) total_bill;
12         System.out.println("New price is " + price);
13     }
14 }
15
```



Both conversions are exact



```
C:\Windows\system32\cmd.exe
C:\test>java test
price is 123456789
Total bill is 1.23456789E8
New price is 123456789
C:\test>
```



Casting floating point as integer

- If we cast a floating point value as an integer, the fractional part is dropped.
 - Not rounded!

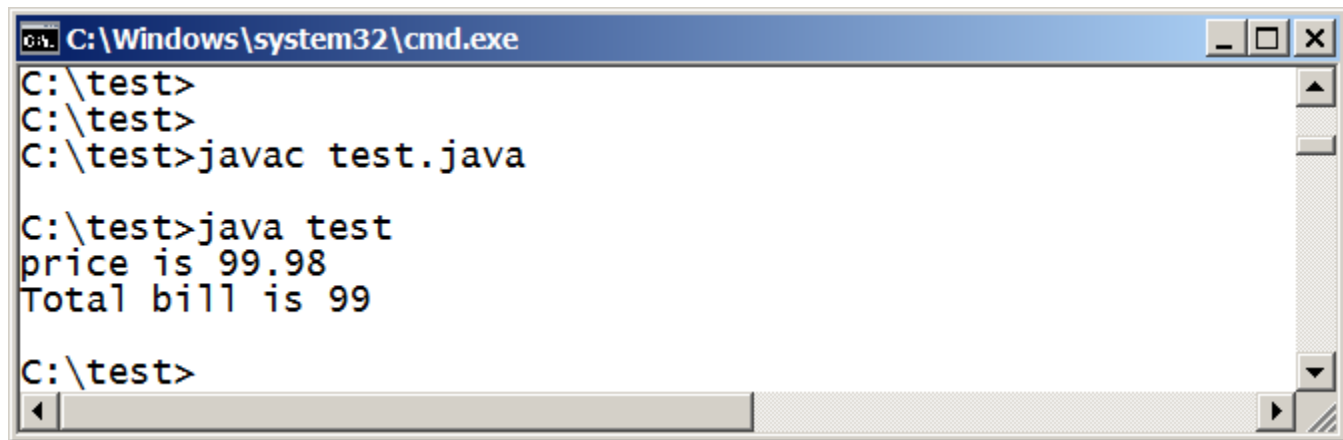


Casting floating point as integer

```
class test
{
    public static void main (String[] args)
    {
        double price = 99.98;
        System.out.println("price is " + price);

        int total_bill = (int) price;
        System.out.println("Total bill is " + total_bill);
    }
}
```

Casting floating point as integer



```
C:\Windows\system32\cmd.exe
C:\test>
C:\test>
C:\test>javac test.java

C:\test>java test
price is 99.98
Total bill is 99

C:\test>
```

The floating point value is truncated.
Only the integer part is retained.



Conversion by Promotion

- When we write an expression including both integer and floating point types the compiler will automatically convert the integer values to floating point.
 - Widening conversion.
- Conversion is only for evaluation of the expression.
 - The variables in the expression are not changed.



Conversion by Promotion

Suppose a purchase is subject to 7% sales tax.

```
class test
{
    public static void main (String[] args)
    {
        final double tax_rate = 0.07;
        int price = 100;
        System.out.println("price is " + price);
        double sales_tax = price * tax_rate;

        double total_bill = price + sales_tax;
        System.out.println("Total bill is " + total_bill);
    }
}
```



Conversion by Promotion

When the expression

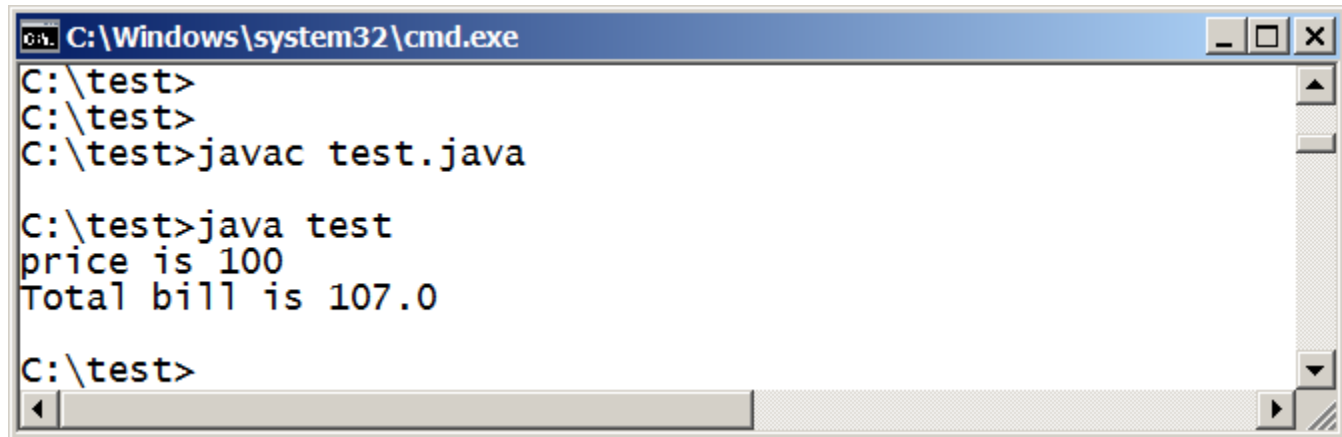
`price * tax_rate`

is evaluated, the value of `price` is converted to `double` before being multiplied by `tax_rate`.

- The conversion is lossless (exact).
- The result is a `double`.
- The variable `price` is unchanged.



Program Running



```
C:\Windows\system32\cmd.exe
C:\test>
C:\test>
C:\test>javac test.java

C:\test>java test
price is 100
Total bill is 107.0

C:\test>
```

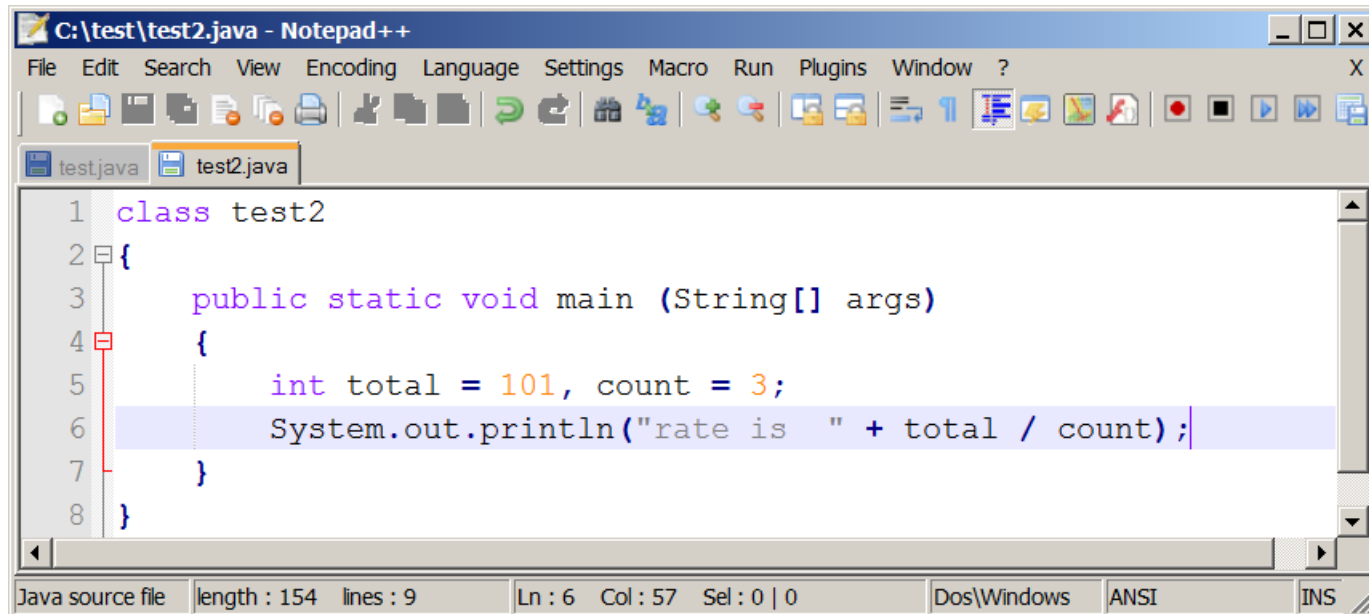


Casting integer as floating point

Sometimes we want a floating point result from an operation on integer values.

- Especially with division
- Cast one of the operands as double
 - The other will be converted automatically by promotion.
 - Floating point division will be done.
 - Result will be a double.

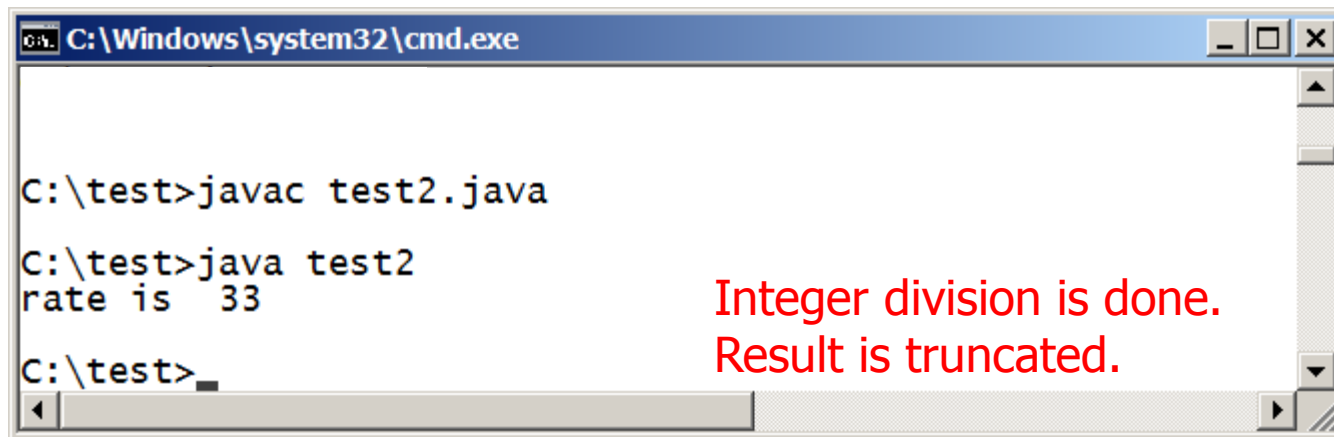
Without a cast



A screenshot of the Notepad++ editor window titled "C:\test\test2.java - Notepad++". The window shows a Java class named "test2" with a "main" method. Inside the "main" method, there is a line of code: "System.out.println("rate is " + total / count);". The code is as follows:

```
1 class test2
2 {
3     public static void main (String[] args)
4     {
5         int total = 101, count = 3;
6         System.out.println("rate is " + total / count);
7     }
8 }
```

The status bar at the bottom indicates "Java source file", "length : 154", "lines : 9", "Ln : 6", "Col : 57", "Sel : 0 | 0", "Dos\Windows", "ANSI", and "INS".

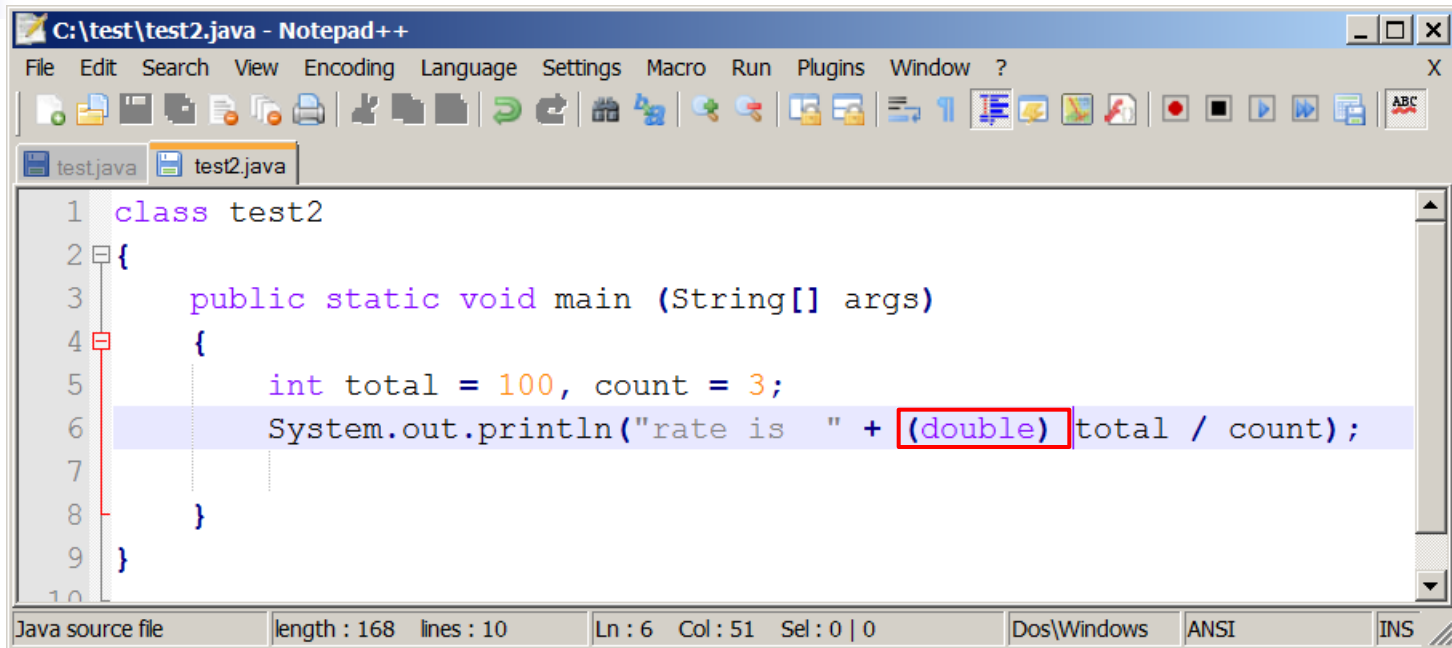


A screenshot of a Windows command prompt window titled "C:\Windows\system32\cmd.exe". The prompt shows the execution of the Java code from the previous screenshot. The output is "rate is 33".

```
C:\test>javac test2.java
C:\test>java test2
rate is 33
C:\test>
```

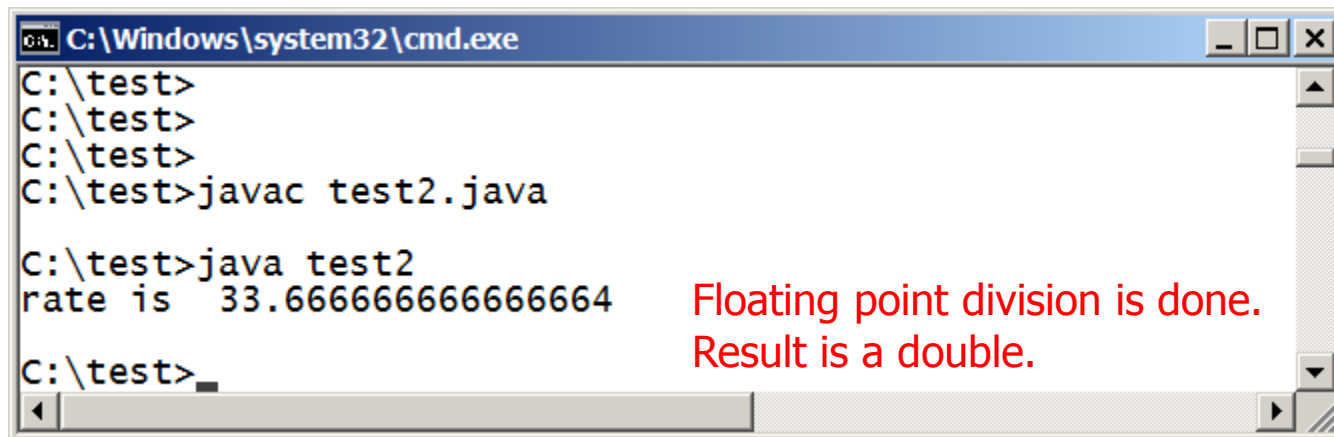
Integer division is done.
Result is truncated.

With a cast



A screenshot of the Notepad++ text editor. The title bar reads 'C:\test\test2.java - Notepad++'. The menu bar includes File, Edit, Search, View, Encoding, Language, Settings, Macro, Run, Plugins, Window, and ?. The toolbar contains various icons for file operations and editing. The editor shows a Java class named 'test2' with a 'main' method. Line 6 contains the statement 'System.out.println("rate is " + (double)total / count);', where '(double)' is highlighted with a red rectangle. The status bar at the bottom indicates 'Java source file', 'length : 168 lines : 10', 'Ln : 6 Col : 51 Sel : 0 | 0', 'Dos\Windows', 'ANSI', and 'INS'.

```
1 class test2
2 {
3     public static void main (String[] args)
4     {
5         int total = 100, count = 3;
6         System.out.println("rate is " + (double)total / count);
7     }
8 }
9
10
```



A screenshot of a Windows command prompt window titled 'C:\Windows\system32\cmd.exe'. The prompt is at 'C:\test>'. The user has entered 'javac test2.java' and 'java test2'. The output of the second command is 'rate is 33.666666666666664'. To the right of the output, red text states: 'Floating point division is done. Result is a double.' The command prompt shows the history of previous commands.

```
C:\test>
C:\test>
C:\test>
C:\test>javac test2.java

C:\test>java test2
rate is 33.666666666666664
C:\test>
```

Floating point division is done.
Result is a double.



Exercise

What is the value of the variable unitPrice?

```
int totalPrice = 12;  
int quantity = 5;  
double unitPrice = 0;  
unitPrice = (double) totalPrice / quantity;
```




Readings and Assignments

- Reading: Chapter 2.5
- Self-Assessment Exercises:
 - Self-Review Questions
SR 2.33, 2.34, 2.35
 - After Chapter Exercises
EX 2.7, 2.8, 2.9
- These are not to be submitted in Canvas.
- Check your own answers
 - SR Answers in back of the book
 - EX Write a program if you are not sure.