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

- Ever 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 explicity 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

C:\test\test.java - Notepad++

```java
class test
{
    public static void main (String[] args)
    {
        int price = 123456789;
        System.out.println("price is " + price);

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

Java source file | length : 227 lines : 12 | Ln : 7 Col : 9 Sel : 0 | 0 | Dos\Windows | ANSI | INS

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

C:\test\test.java - Notepad++

File Edit Search View Encoding Language Settings Macro Run Plugins Window ?

test.java | new 1

```java
class test
{
    public static void main (String[] args)
    {
        int price = 123456789;
        System.out.println("price is " + price);

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

        price = total_bill;
        System.out.println("New price is " + price);
    }
}
```

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

C:\test\test.java - Notepad++

```java
class test
{
    public static void main (String[] args)
    {
        int price = 123456789;
        System.out.println("price is " + price);

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

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

Java source file | length : 308 lines : 15 | Ln : 11 Col : 23 Sel : 0 | 0 | Dos\Windows | ANSI | 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++

```java
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 );

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

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

C:\test\test.java - Notepad++

```java
class test
{
    public static void main (String[] args)
    {
        int price = 123456789;
        System.out.println("price is " + price);

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

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

Java source file | length : 309 lines : 15 | Ln : 10 Col : 9 Sel : 0 | 0 | Dos\Windows | ANSI | INS

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

C:\test\test2.java - Notepad++

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

Java source file | length : 154 lines : 9 | Ln : 6 Col : 57 Sel : 0 | 0 | Dos\Windows | ANSI | INS

C:\Windows\system32\cmd.exe

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

C:\test>java test2
rate is  33

C:\test>_
```

Integer division is done.
Result is truncated.

With a cast

C:\test\test2.java - Notepad++

```java
class test2
{
    public static void main (String[] args)
    {
        int total = 100, count = 3;
        System.out.println("rate is  " + (double) total / count);

    }
}
```

C:\Windows\system32\cmd.exe

```
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.