第3章 标识符与简单数据类型

主要内容

- 标识符与关键字；
- 简单数据类型；
- 简单数据类型的级别与类型转换运算；
- 从命令行窗口输入、输出数据。

本章学习 Java 中的简单数据类型(基本数据类型),这些简单数据类型和 C 语言中的简单数据类型很相似,但读者务必也要注意和 C 语言的不同之处,特别是 float 常量的格式与 C 语言的区别。

3.1 标识符与关键字

3.1.1 标识符

用来标识类名、变量名、方法名、类型名、数组名、文件名的有效字符序列称为标识符。简单地说,标识符就是一个名字。以下是 Java 关于标识符的语法规则。

- 标识符由字母、下划线、美元符号和数字组成,长度不受限制。
- 标识符的第一个字符不能是数字字符。
- 标识符不能是关键字(关键字见 3.1.2 节)。
- 标识符不能是 true、false 和 null(尽管 true、false 和 null 不是 Java 关键字)。

例如,以下都是标识符。

```
HappyNewYear_java、TigerYear_2010、$98apple、Hello.java.
```

需要特别注意的是,标识符中的字母是区分大小写的,hello 和 Hello 是不同的标识符。

Java 语言使用 Unicode 标准字符集,Unicode 字符集由 UNICODE 协会管理并接受其技术上的修改,最多可以识别 65 536 个字符,Unicode 字符集的前 128 个字符刚好是 ASCII 码表。Unicode 字符集还不能覆盖全部历史上的文字,但大部分国家的“字母表”的字母都是 Unicode 字符集中的一个字符,比如汉字中的“好”字就是 Unicode 字符集中的第 22 909 个字符。Java 所谓的字母包括了世界上大部分语言中的“字母表”,因此,Java 使用的字母不仅包括通常的拉丁字母 a、b、c 等,也包括汉语中的汉字、日文的片假名和平假名、朝鲜文、俄文、希腊字母以及其他许多语言中的文字。

3.1.2 关键字

关键字就是 Java 语言中已经被赋予特定意义的一些单词。不可以把关键字作为标识

符来用。以下是 Java 的 50 个关键字。

abstract assert boolean break byte case catch char class const continue default do double else enum extends final finally float for goto if implements import instanceof int interface long native new package private protected public return short static strictfp super switch synchronized this throw throws transient try void volatile while。

3.2 简单数据类型

简单数据类型也称作基本数据类型。Java 语言有 8 种简单数据类型,分别是:boolean、byte、short、int、long、float、double、char。

这 8 种简单数据类型习惯上可分为以下四大类型。

逻辑类型:boolean。

整数类型:byte、short、int、long。

字符类型:char。

浮点类型:float、double。

3.2.1 逻辑类型

- 常量:true,false。
- 变量:使用关键字 boolean 来声明逻辑变量,声明时也可以赋给初值,例如:

```
boolean x,ok = true,关闭 = false;
```

3.2.2 整数类型

整型数据分为 4 种。

1. int 型

- 常量:123,6000(十进制),077(八进制),0x3ABC(十六进制)。
- 变量:使用关键字 int 来声明 int 型变量,声明时也可以赋给初值,例如:

```
int x = 12,y = 9898,z;
```

对于 int 型变量,内存分配给 4 个字节(4B),因此,int 型变量的取值范围是:$-2^{31}\sim2^{31}-1$。

2. byte 型

- 变量:使用关键字 byte 来声明 byte 型变量,例如:

```
byte x =- 12,tom = 28,漂亮 = 98;
```

- 常量:Java 中不存在 byte 型常量的表示法,但可以把一定范围内的 int 型常量赋值给 byte 型变量。对于 byte 型变量,内存分配给 1 个字节,占 8 位,因此 byte 型变量的取值范围是$-2^{7}\sim2^{7}-1$。如果需要强调一个整数是 byte 型数据时,可以使用强制转换运算的结果来表示,例如:(byte)-12,(byte)28。

3. short 型

- 变量:使用关键字 short 来声明 short 型变量,例如:

```
short x = 12, y = 1234;
```

- 常量：和 byte 型类似，Java 中也不存在 short 型常量的表示法，但可以把一定范围内的 int 型常量赋值给 short 型变量。对于 short 型变量，内存分配给 2 个字节，占 16 位，因此 short 型变量的取值范围是 -2^{15}～$2^{15}-1$。如果需要强调一个整数是 short 型数据时，可以使用强制转换运算的结果来表示，例如：(short)－12，(short)28。

4. long 型

- 常量：long 型常量用后缀 L 来表示，例如 108L(十进制)、07123L(八进制)、0x3ABCL(十六进制)。
- 变量：使用关键字 long 来声明 long 型变量，例如：

```
long width = 12L, height = 2005L, length;
```

对于 long 型变量，内存分配给 8 个字节，占 64 位，因此 long 型变量的取值范围是 -2^{63}～$2^{63}-1$。

3.2.3 字符类型

- 常量：'A'，'b'，'?'，'!'，'9'，'好'，'\t'，'き'，'モ'等，即用单引号扩起的 Unicode 表中的一个字符。
- 变量：使用关键字 char 来声明 char 型变量，例如：

```
char ch = 'A', home = '家', handsome = '酷';
```

对于 char 型变量，内存分配给 2 个字节，占 16 位，最高位不是符号位，没有负数的 char。char 型变量的取值范围是 0～65 535。对于

```
char x = 'a';
```

那么内存 x 中存储的是 97，97 是字符 a 在 Unicode 表中的排序位置。因此，允许将上面的语句写成

```
char x = 97;
```

有些字符(如回车符)不能通过键盘输入到字符串或程序中，这时就需要使用转意字符常量，例如：

\n(换行)，\b(退格)，\t(水平制表)，\'(单引号)，\"(双引号)，\\(反斜线)等。

例如：

```
char ch1 = '\n', ch2 = '\"', ch3 = '\\';
```

再比如，字符串："我喜欢使用双引号\""中含有双引号字符，但是，如果写成："我喜欢使用双引号""，就是一个非法字符串。

要观察一个字符在 Unicode 表中的顺序位置，可以使用 int 型显示转换，如(int)'a'或 int p='a'。如果要得到一个 0～65 536 之间的数代表的 Unicode 表中相应位置上的字符必须使用 char 型显示转换。

在例 3.1 中,分别用显示转换来显示一些字符在 Unicode 表中的位置,以及 Unicode 表中某些位置上的字符,运行效果如图 3.1 所示。

```
C:\ch3>java Example3_1
汉字:好的位置:22909
日文:あ的位置:12353
20320位置上的字符是:你
21319位置上的字符是:升
```

图 3.1 显示 Unicode 表中的字符

【例 3.1】

Example3_1.java

```
public class  Example3_1 {
   public static void main (String args[ ]) {
      char chinaWord = '好', japanWord = 'あ';
      int  position = 20320;
      System.out.println("汉字:" + chinaWord + "的位置:" + (int)chinaWord);
      System.out.println("日文:" + japanWord + "的位置:" + (int)japanWord);
      System.out.println(position + "位置上的字符是:" + (char)position);
      position = 21319;
      System.out.println(position + "位置上的字符是:" + (char)position);
   }
}
```

3.2.4 浮点类型

浮点型分为 float 和 double 型。

1. float 型

- 常量:453.5439f,21379.987F,231.0f(小数表示法),2e40f(2 乘 10 的 40 次方,指数表示法)。需要特别注意的是:常量后面必须要有后缀 f 或 F。
- 变量:使用关键字 float 来声明 float 型变量,例如:

```
float x = 22.76f, tom = 1234.987f, weight = 1e - 12F;
```

float 变量在存储 float 型数据时保留 8 位有效数字,实际精度取决于具体数值。例如,如果将常量 12345.123456789f 赋值给 float 型变量 x:

```
x = 12345.123456789f
```

那么,x 存储的实际值是 12345.123046875(保留 8 位有效数字)。

对于 float 型变量,内存分配给 4 个字节,占 32 位,float 型变量的取值范围大约是 10^{-38}～10^{38} 和 -10^{38}～10^{-38}。

2. double 型

- 常量:2389.539d,2318908.987,0.05(小数表示法),1e－90(1 乘 10 的－90 次方,指数表示法)。对于 double 常量,后面可以有后缀 d 或 D,但允许省略该后缀。
- 变量:使用关键字 double 来声明 double 型变量,例如:

```
double height = 23.345, width = 34.56D, length = 1e12;
```

double 变量在存储 double 型数据时保留 16 位有效数字,实际精度取决于具体数值。

对于 double 型变量,内存分配给 8 个字节,占 64 位,double 型变量的取值范围大约是 10^{-308}～10^{308} 和 -10^{-308}～10^{308}。

在下面的例 3.2 中有两个类,其中 People 类具有刻画人的年龄和体重的简单类型变

量，主类 Example2_5 负责用 People 类创建两个对象。程序运行效果如图 3.2 所示。

```
C:\ch3>java Example3_2
我的年龄:21岁
我的体重:66.5公斤
我的年龄:19岁
我的体重:56.9公斤
```

图 3.2　体重和年龄

【例 3.2】

People.java

```
public class People {
    int age;
    float weight;
    void speak() {
        System.out.println("我的年龄:" + age + "岁");
        System.out.println("我的体重:" + weight + "公斤");
    }
}
```

Example3_2.java

```
public class Example3_2 {
    public static void main (String args[ ]) {
        People zhang,zhou;
        zhang = new People();
        zhang.weight = 66.5F;
        zhang.age = 21;
        zhang.speak();
        zhou = new People();
        zhou.weight = 56.9F;
        zhou.age = 19;
        zhou.speak();
   }
}
```

3.3　简单数据类型的级别与类型转换运算

当我们把一种基本数据类型变量的值赋给另一种基本类型变量时，就涉及数据转换。下列基本类型会涉及数据转换（不包括逻辑类型）。将这些类型按精度从低到高排列。

```
byte  short  char  int  long  float  double
```

当把级别低的变量的值赋给级别高的变量时，系统自动完成数据类型的转换。例如：

```
float x = 100;
```

如果输出 x 的值，结果将是 100.0。

例如：

```
int x = 50;
float y;
y = x;
```

如果输出 y 的值，结果将是 50.0。

当把级别高的变量的值赋给级别低的变量时,必须使用显示类型转换运算。显示转换的格式:

```
(类型名) 要转换的值;
```

例如

```
int x = (int)34.89;
long y = (long)56.98F;
int z = (int)1999L;
```

输出 x,y 和 z 的值将是 34,56 和 1999,强制转换运算可能导致精度的损失。

当把一个 int 型常量赋值给一个 byte 和 short 型变量时,不可超出这些变量的取值范围,否则必须进行类型转换运算。例如,常量 128 属于 int 型常量,超出 byte 变量的取值范围,如果赋值给 byte 型变量,必须进行 byte 类型转换运算(将导致精度的损失),如下所示。

```
byte a = (byte)128;
byte b = (byte)( - 129);
```

那么 a 和 b 得到的值分别是－128 和 127。

另外,一个常见的错误是把一个 double 型常量赋值给 float 型变量时没有进行强制转换运算,例如

```
float x = 12.4;
```

将导致语法错误,编译器将提示“possible loss of precision”。正确的做法是

```
float x = 12.4F
```

或

```
float x = (float)12.4;
```

下面的例 3.3 使用了类型转换运算,运行效果如图 3.3 所示。

【例 3.3】

Example3_3.java

```
public class Example3_3 {
   public static void main (String args[]) {
      byte b = 22;
      int  n = 129;
      float f = 123456.6789f ;
      double d = 123456789.123456789;
      System.out.println("b =   " + b);
      System.out.println("n =   " + n);
      System.out.println("f =   " + f);
      System.out.println("d =   " + d);
      b = (byte)n;                  //导致精度的损失
      f = (float)d;                 //导致精度的损失
      System.out.println("b =   " + b);
      System.out.println("f =   " + f);
   }
}
```

```
C:\ch3>java Example3_3
b=  22
n=  129
f=  123456.68
d=  1.23456789123456792E8
b=  -127
f=  1.23456792E8
```

图 3.3　类型转换运算

3.4 从命令行窗口输入、输出数据

3.4.1 输入基本型数据

Scanner 是 JDK1.5 新增的一个类，可以使用该类创建一个对象。

```
Scanner reader = new Scanner(System.in);
```

然后 reader 对象调用下列方法，读取用户在命令行（例如，MS-DOS 窗口）输入的各种基本类型数据。

```
nextBoolean();nextByte(),nextShort(),nextInt(),nextLong(),nextFloat(),nextDouble()。
```

上述方法执行时都会堵塞，程序等待用户在命令行输入数据回车确认。

在下面的例 3.4 中用到了例 3.2 中的 People 类。例 3.4 中的主类中用 People 类创建 zhangSan 对象，并要求户在键盘依次输入 zhangSan 对象的年龄和体重，每输入一个数字都需要按回车键确认。运行效果如图 3.4 所示。

```
C:\ch3>java Example3_4
输入年龄，回车确认
25
输入体重，回车确认
78.89
我的年龄:25岁
我的体重:78.89公斤
```

图 3.4 从命令行输入数据

【例 3.4】

Example2_6.java

```
import java.util.Scanner;
public class Example3_4 {
   public static void main(String args[]) {
      People zhangSan = new People();
      Scanner reader = new Scanner(System.in);
      System.out.println("输入年龄,回车确认");
      zhangSan.age = reader.nextInt();
      System.out.println("输入体重,回车确认");
      zhangSan.weight = reader.nextFloat();
      zhangSan.speak();
   }
}
```

3.4.2 输出基本型数据

System.out.println()或 System.out.print()可输出串值、表达式的值，二者的区别是前者输出数据后换行，后者不换行。允许使用并置符号＋将变量、表达式或一个常数值与一个字符串并置一起输出，如：

```
System.out.println(m + "个数的和为" + sum);
System.out.println(":" + 123 + "大于" + 122)。
```

需要特别注意的是，在使用 System.out.println()或 System.out.print()输出字符串常量时，不可以出现“回车”，例如，下面的写法无法通过编译：

```
System.out.println("你好,
                    很高兴认识你" );
```

如果需要输出的字符串的长度较长,可以将字符串分解成几部分,然后使用并置符号“+”将它们首尾相接,例如,以下是正确的写法:

```
System.out.println("你好," +
                   "很高兴认识你" );
```

另外,JDK1.5 新增了和 C 语言中 printf 函数类似的数据输出方法,该方法使用格式如下。

```
System.out.printf("格式控制部分",表达式 1,表达式 2,…,表达式 n)
```

格式控制部分由格式控制符号%d、%c、%f、%s 和普通的字符组成,普通字符原样输出。格式符号用来输出表达式的值。

%d:输出 int 类型数据值。

%c:输出 char 型数据。

%f:输出浮点型数据,小数部分最多保留 6 位。

%s:输出字符串数据。

输出数据时也可以控制数据在命令行的位置,例如:

%md:输出的 int 型数据占 m 列。

%m.nf:输出的浮点型数据占 m 列,小数点保留 n 位。

例如:

```
System.out.printf("%d,%f",12,23.78);
```

3.5 上机实践

1. 实验目的

掌握从键盘为简单型变量输入数据。掌握使用 Scanner 类创建一个对象,例如:

```
Scanner reader = new Scanner(System.in);
```

学习让 reader 对象调用下列方法读取用户在命令行(例如,MS-DOS 窗口)输入的各种简单类型数据:

```
nextBoolean();nextByte(),nextShort(),nextInt(),nextLong(),nextFloat(),nextDouble()。
```

在调试程序时,体会上述方法都会堵塞,即程序等待用户在命令行输入数据回车确认。

2. 实验要求

编写一个 Java 应用程序,在主类的 main 方法中声明用于存放产品数量的 int 型变量 amount 和产品单价的 float 型变量 price,以及存放全部产品总价值 float 型变量 sum。

使用 Scanner 对象调用方法让用户从键盘为 amount,price 变量输入值,然后程序计算出全部产品总价值,并输出 amount,prince,sum 的值。程序运行参考效果如图 3.5 所示。

```
D:\1000>java  InputData
输入产品数量(回车确认):
46
输入产品单价(回车确认):
25.67
数量:46,单价:25.67,总价值:1180.82
```

图 3.5 从键盘输入数据

3. 程序模板

请按模板要求,将【代码】替换为 Java 程序代码。

InputData.java

```
import java.util.Scanner;
public class InputData {
   public static void main(String args[]) {
      Scanner reader = new Scanner(System.in);
      int amount = 0 ;
      float price = 0,sum = 0;
      System.out.println("输入产品数量(回车确认):");
      【代码 1】                    //从键盘为 amount 赋值
      System.out.println("输入产品单价(回车确认):");
      【代码 2】                    //从键盘为 price 赋值
      sum = price * amount;
      System.out.printf("数量:%d,单价:%5.2f,总价值:%5.2f",amount,price,sum);
   }
}
```

4. 实验指导

由于 amount 是 int 型,因此代码 1 应该是 amount = reader.nextInt();而 price 是 float 型,因此代码 2 应该是 price = reader.nextFloat();不可以是 price = reader.nextDouble();。

另外,Scanner 对象可以调用 hasNextXXX()方法判断用户输入的数据的类型,例如,如果用户在键盘输入带小数点的数字:12.34(回车),那么 reader 对象调用 hasNextDouble()返回的值是 true,而调用 hasNextByte()、hasNextInt()以及 hasNextLong()返回的值都是 false;如果用户在键盘输入一个 byte 取值范围内的整数:89(回车),那么 reader 对象调用 hasNextByte()、hasNextInt()、hasNextLong()以及 hasNextDouble()返回的值都是 true。nextLine()等待用户在命令行输入一行文本回车,该方法得到一个 String 类型的数据,String 类型将在本书第 9 章讲述。

在从键盘输入数据时,我们经常让 reader 对象先调用 hasNextXXX()方法等待用户在键盘输入数据,然后再调用 nextXXX()方法读取数据。

5. 实验后的练习

上机调试下列程序。该程序可以让用户在键盘依次输入若干个数字,每输入一个数字都需要按回车键确认,最后用户在键盘输入一个非数字字符串结束整个输入操作过程(用户输入非数字数字后 reader.hasNextDouble()的值将是 false)。程序将计算出这些数的和以及平均值。

```
import java.util.*;
public class LianXi{
    public static void main (String args[ ]){
       Scanner reader = new Scanner(System.in);
       double sum = 0;
       int m = 0;
       while(reader.hasNextDouble()){
            double x = reader.nextDouble();
            m = m + 1;
            sum = sum + x;
```

```
            }
            System.out.printf("%d个数的和为%f\n",m,sum);
            System.out.printf("%d个数的平均值是%f\n",m,sum/m);
        }
    }
```


习　　题

1. 什么叫标识符？标识符的规则是什么？true 是否可以作为标识符？
2. 什么叫关键字？true 和 false 是否是关键字？请说出 6 个关键字。
3. Java 的基本数据类型都是什么？
4. 上机运行下列程序，注意观察输出的结果。

```
public class E {
    public static void main (String args[ ]) {
      for(int i = 20302;i <= 20322;i++) {
          System.out.println((char)i);
      }
    }
}
```

5. 上机调试下列程序，注意 System.out.print()和 System.out.println()的区别。

```
public class OutputData {
  public static void main(String args[]) {
    int x = 234,y = 432;
    System.out.println(x + "<" + (2 * x));
    System.out.print("我输出结果后不回车");
    System.out.println("我输出结果后自动回车到下一行");
    System.out.println("x + y = " + (x + y));
  }
}
```

6. 编写一个 Java 应用程序，输出全部的大写英文字母。
7. 是否可以将例 3.4 中的

```
zhangSan.weight = reader.nextFloat();
```

更改为

```
zhangSan.weight = reader.nexDouble();
```