Java 核心技术

第八章 Java常用类

第二节 数字相关类

华东师范大学 陈良育

数字类

- Java 数字类
  - 整数　Short, Int, Long
  - 浮点数 Float, Double
  - 大数类 BigInteger(大整数), BigDecimal(大浮点数)
  - 随机数类 Random
  - 工具类 Math
- java.math包

整数类型(1)

- short，16位，2个字节，有符号的以二进制补码表示的整数
  - (-32768-32767, -2^15-2^15-1)，默认值0
- int, 32位，4个字节,有符号的以二进制补码表示的整数
  - (-2147483648--2147483647, -2^31-2^31-1)，默认值0
- long, 64位，8个字节,有符号的以二进制补码表示的整数
  - -9,223,372,036,854,775,808（-2^63）--
    9,223,372,036,854,775,807（2^63 -1），默认值0L

整数类型(2)


```
public class IntegerTest {
    public static void main(String[] args) {
        short a1 = 32767;
        System.out.println(a1);
        //short a2 = 32768;  error 越界

        int b1 = 2147483647;
        System.out.println(b1);
        //int b2 = 2147483648; error 越界

        long c1 = 10000000000L;
        System.out.println(c1);

        long c2 = 2147483647;  //隐式做了从int变成long的操作
        System.out.println(c2);

        long c3 = 2147483648L; //去掉L将报错
        System.out.println(c3);
    }
}
```

浮点数类型(1)

- float，单精度，32位，4个字节，符合IEEE 754标准的浮点数，默认值0.0f。float的范围为1.40129846432481707e-45 to 3.40282346638528860e+38 (无论正负)。
- double，双精度，32位，4个字节，符合IEEE 754标准的浮点数，默认值0.0d。double的范围为4.94065645841246544e-324d to 1.79769313486231570e+308d (无论正负) 。
- float和double都不能用来表示很精确的数字。

浮点数类型(2)


```
public class FloatingTest {
    public static void main(String[] args) {
        float f1 = 1.23f;
        // float f2 = 1.23;  error, float赋值必须带f
        double d1 = 4.56d;
        double d2 = 4.56;  //double 可以省略末尾d

        System.out.println(f1);
        System.out.println((double)f1); //转换到double
        System.out.println(d1);
        System.out.println((float)d2);

        System.out.println(f1==1.229999999f); //true
        System.out.println(f1-1.229999999f); //0.0
        System.out.println(d2==4.559999999999999d); //true
        System.out.println(d2-4.559999999999999d); //0.0
    }
}
```

大数字类

- 大整数类BigInteger
  - 支持无限大的整数运算
  - 查看BigIntegerTest.java

- 大浮点数BigDecimal
  - 支持无限大的小数运算
  - 注意精度和截断
  - 查看BigDecimalTest.java

随机数类

- Random 随机数
  - nextInt() 返回一个随机int
  - nextInt(int a) 返回一个[0,a)之间的随机int
  - nextDouble()返回一个[0.0,1.0]之间double
  - ints 方法批量返回随机数数组
- Math.random() 返回一个[0.0,1.0]之间double
- 查看RandomTest.java

数字工具类

- java.lang.Math
  - 绝对值函数abs
  - 对数函数log
  - 比较函数max、min
  - 幂函数pow
  - 四舍五入函数round等
  - 向下取整floor
  - 向上取整ceil
- 查看MathTest.java

总结

- 根据数字特点选择合适的类
- 尽量使用类库自带的方法
- 浮点数需要注意精度

代码(1) IntegerTest.java


```java
public class IntegerTest {
    public static void main(String[] args) {
        short a1 = 32767;
        System.out.println(a1);
        //short a2 = 32768;  error 越界

        int b1 = 2147483647;
        System.out.println(b1);
        //int b2 = 2147483648; error 越界

        long c1 = 10000000000L;
        System.out.println(c1);

        long c2 = 2147483647;  //隐式做了从int变成long的操作
        System.out.println(c2);

        long c3 = 2147483648L; //去掉L将报错
        System.out.println(c3);
    }
}
```

代码(2) FloatingTest.java


```java
public class FloatingTest {
    public static void main(String[] args) {
        float f1 = 1.23f;
        // float f2 = 1.23;  error, float赋值必须带f
        double d1 = 4.56d;
        double d2 = 4.56;  //double 可以省略末尾d

        System.out.println(f1);
        System.out.println((double)f1); //转换到double
        System.out.println(d1);
        System.out.println((float)d2);

        System.out.println(f1==1.229999999f); //true
        System.out.println(f1-1.229999999f); //0.0
        System.out.println(d2==4.559999999999999d); //true
        System.out.println(d2-4.559999999999999d); //0.0
    }
}
```

代码(3) BigIntegerTest.java


```
import java.math.BigInteger;

public class BigIntegerTest {

    public static void main(String[] args) {
        BigInteger b1 = new BigInteger("123456789"); // 声明BigInteger对象
        BigInteger b2 = new BigInteger("987654321"); // 声明BigInteger对象
        System.out.println("b1: " + b1 +  ", b2:" + b2);
        System.out.println("加法操作：" + b2.add(b1)); // 加法操作
        System.out.println("减法操作：" + b2.subtract(b1)); // 减法操作
        System.out.println("乘法操作：" + b2.multiply(b1)); // 乘法操作
        System.out.println("除法操作：" + b2.divide(b1)); // 除法操作
        System.out.println("最大数：" + b2.max(b1)); // 求出最大数
        System.out.println("最小数：" + b2.min(b1)); // 求出最小数
        BigInteger result[] = b2.divideAndRemainder(b1); // 求出余数的除法操作
        System.out.println("商是：" + result[0] + "；余数是：" + result[1]);
        System.out.println("等价性是：" + b1.equals(b2));
        int flag = b1.compareTo(b2);
        if (flag == -1)
            System.out.println("比较操作:b1<b2");
        else if (flag == 0)
            System.out.println("比较操作:b1==b2");
        else
            System.out.println("比较操作:b1>b2");

    }
}
```

代码(4) BigDecimalTest.java


```java
import java.math.BigDecimal;

public class BigDecimalTest {
    public static void main(String[] args) {
        BigDecimal b1 = new BigDecimal("123456789.987654321"); // 声明BigDecimal对象
        BigDecimal b2 = new BigDecimal("987654321.123456789"); // 声明BigDecimal对象
        System.out.println("b1: " + b1 +  ", b2:" + b2);
        System.out.println("加法操作：" + b2.add(b1)); // 加法操作
        System.out.println("减法操作：" + b2.subtract(b1)); // 减法操作
        System.out.println("乘法操作：" + b2.multiply(b1)); // 乘法操作
        //需要指定位数，防止无限循环，或者包含在try-catch中
        System.out.println("除法操作：" + b2.divide(b1,10,BigDecimal.ROUND_HALF_UP)); // 除法操作

        System.out.println("最大数：" + b2.max(b1)); // 求出最大数
        System.out.println("最小数：" + b2.min(b1)); // 求出最小数

        int flag = b1.compareTo(b2);
        if (flag == -1)
            System.out.println("比较操作: b1<b2");
        else if (flag == 0)
            System.out.println("比较操作: b1==b2");
        else
            System.out.println("比较操作: b1>b2");

        System.out.println("====================");

        //尽量采用字符串赋值
        System.out.println(new BigDecimal("2.3"));
        System.out.println(new BigDecimal(2.3));

        System.out.println("====================");

        BigDecimal num1 = new BigDecimal("10");
        BigDecimal num2 = new BigDecimal("3");
        //需要指定位数，防止无限循环，或者包含在try-catch中
        BigDecimal num3 = num1.divide(num2, 3, BigDecimal.ROUND_HALF_UP);
        System.out.println(num3);
    }
}
```

代码(5) RandomTest.java


```java
import java.util.Random;

public class RandomTest {

    public static void main(String[] args)
    {
        //第一种办法，采用Random类 随机生成在int范围内的随机数
        Random rd = new Random();
        System.out.println(rd.nextInt());
        System.out.println(rd.nextInt(100)); //0--100的随机数
        System.out.println(rd.nextLong());
        System.out.println(rd.nextDouble());
        System.out.println("==========================");

        //第二种，生成一个范围内的随机数 例如0到时10之间的随机数
        //Math.random[0,1)
        System.out.println(Math.round(Math.random()*10));
        System.out.println("==========================");

        //JDK 8 新增方法
        rd.ints();  //返回无限个int类型范围内的数据
        int[] arr = rd.ints(10).toArray();  //生成10个int范围类的个数。
        for (int i = 0; i < arr.length; i++) {
            System.out.println(arr[i]);
        }
        System.out.println("==========================");

        arr = rd.ints(5, 10, 100).toArray();
        for (int i = 0; i < arr.length; i++) {
            System.out.println(arr[i]);
        }

        System.out.println("==========================");

        arr = rd.ints(10).limit(5).toArray();
        for (int i = 0; i < arr.length; i++) {
            System.out.println(arr[i]);
        }
    }

}
```

代码(6) MathTest.java


```java
public class MathTest {

    public static void main(String[] args) {

        System.out.println(Math.abs(-5));     //绝对值
        System.out.println(Math.max(-5,-8)); //最大值
        System.out.println(Math.pow(-5,2));  //求幂
        System.out.println(Math.round(3.5)); //四舍五入
        System.out.println(Math.ceil(3.5));  //向上取整
        System.out.println(Math.floor(3.5)); //向下取整
    }

}
```

谢 谢！