BigDecimal使用：import java.math.BigDecimal;

官网api，

<http://docs.oracle.com/javase/6/docs/api/java/math/BigDecimal.html>

Multiply 乘法；

Divide 除法；

Add 加法；

Subtract 减法；

//除以divisor后的结果，保留有scale个小数位，保留模式为roundingMode；

pubilc BigDecimal divide(BigDecimal divisor, int scale, int roundingMode)

//格式化处理：2位小数，四舍五入；

BigDecimal mData = new BigDecimal("9.655").setScale(2, BigDecimal.ROUND\_HALF\_UP);

~

BigDecimal.ROUND\_HALF\_UP 四舍五入；

BigDecimal.ROUND\_up 进一(有多余位数，去掉并进一)

BigDecimal.ROUND\_DOWN 去尾(有多余位数，直接去除)

示例：借款偿还计算；

本金，年率，借款期(月单位)；

~

偿还利率：本金\*年率/12=月利率；

偿还本金：本金/借款期=月本金；

偿还本金+利率：月利率+月本金；

BigDecimal mon = **new** BigDecimal(50000); //本金5万

BigDecimal apr = **new** BigDecimal(7); //利率7%

BigDecimal yue = **new** BigDecimal(7); //借款期

//利率；100指一月，1200指12月；---因为前面乘以7，后面的除数扩大100倍；

BigDecimal planInterest = mon.multiply(apr).divide(**new** BigDecimal(100),2, BigDecimal.***ROUND\_HALF\_UP***);

//本金；

BigDecimal planPrinciple = mon.divide(**new** BigDecimal(yue), 2,BigDecimal.***ROUND\_HALF\_UP***);

//本金+利率：

planInterest.add(planPrinciple);

//减法：

planInterest.subtract(planPrinciple);

int r=big\_decimal.compareTo(BigDecimal.Zero); //和0，Zero比较

if(r==0) //等于

if(r==1) //大于

if(r==-1) //小于

3位加,号：如，112,223.45元；

BigDecimal TotalMoney =

DecimalFormat d = **new** DecimalFormat(",##0.00");

d.format(TotalMoney); //返回string格式；

string的转换为资金类型：

String nowApr = ...; //string类型的资金格式的字符串

**new** BigDecimal(nowApr);

----------------------------------

float,double和decimal类型

float:浮点型，含字节数为4，32bit，数值范围为-3.4E38~3.4E38（7个有效位）

double:双精度实型，含字节数为8，64bit数值范围-1.7E308~1.7E308（15个有效位）

decimal:数字型，128bit，不存在精度损失，常用于银行帐目计算。（28个有效位）

float f = 345.98756f;//结果显示为345.9876，只显示7个有效位，对最后一位数四舍五入。

double d=345.975423578631442d;//结果显示为345.975423578631，只显示15个有效位，对最后一位四舍五入。

注：float和double的相乘操作，数字溢出不会报错，会有精度的损失。

decimal dd=345.545454879.....//可以支持28位，对最后一位四舍五入。

注：当对decimal类型进行操作时，数值会因溢出而报错。

具体使用：

7位够的(包括小数7位够的)，使用float；

15位够的，使用double； ---小数一般默认使用double；

高精度，涉及计算的，使用decimal； | money计算的，使用此；

///////////////////////////////////////////////////////////////////

integer类型：

-128-127保存在缓冲池中，可以用==比较；

>127/<-128的，保存在堆中，不能用==比较；

~解决：

equals比较，

num1.equals(num2)

金额数据的，页面显示时，转换为有逗号的标准格式显示：

转换方法1：页面中转换，

<fmt:formatNumber value=*"*${transfer.accountReal}*"* pattern=*"#,##0.##"*/>

转换方法2：后台转换，

实体类或实体类继承类中，新声明一个变量，

**import** java.text.DecimalFormat;

**private** **static** DecimalFormat *df* = **new** DecimalFormat("#,##0.##");

//金额转换显示accountReal

**private** String accountRealStr;

**public** String getAccountRealStr() {

**return** *df*.format(**super**.getAccountReal());

}

页面显示，${transfer.accountRealStr}元

--------------

Math:

java.lang.Math

/\*\*

\* abs求绝对值

\*/

System.out.println(Math.abs(-10.4)); //10.4

System.out.println(Math.abs(10.1)); //10.1

//x的y次方，

Math.*pow*(x,y)