JAVA 在 ACM中使用模板

> by gemengmeng

输入

```
import java.io.*;
import java.math.*;
import java.util.*;
import java.text.*;
public classMain
{
    public static void main(String[] args)
    {
        Scanner cin = new Scanner (new BufferedInputStream(System.in));
        int a; double b; BigInteger c; String st;
        a = cin.nextInt(); b = cin.nextDouble(); c = cin.nextBigInteger();
        d = cin.nextLine(); // 每种类型都有相应的输入函数.
    }
}
```

输出

```
//函数: System.out.print(); System.out.println(); System.out.printf();
System.out.print(); // cout << …;
System.out.println(); // cout << … << endl;
System.out.printf(); // 与C中的printf用法类似.
//规格化的输出:
//函数:
// 这里0指一位数字, #指除0以外的数字(如果是0, 则不显示),四舍五入.
DecimalFormat fd = new DecimalFormat("#.00#");
DecimalFormat gd = new DecimalFormat("0.000");
System.out.println("x =" + fd.format(x));
System.out.println("x =" + gd.format(x));
```

高精度

```
/*
```

```
函数: add, subtract, divide, mod, compareTo等, 其中加减乘除模都要求是
BigInteger(BigDecimal)和BigInteger(BigDecimal)之间的运算, 所以需要把int(double)类型
转换为BigInteger(BigDecimal), 用函数BigInteger.valueOf().
*/
import java.io.*;
import java.math.*;
import java.util.*;
import java.text.*;
public classMain
{
    public static void main(String[] args)
    {
        Scanner cin = new Scanner (new BufferedInputStream(System.in));
        int a = 123, b = 456, c = 7890;
        BigInteger x, y, z, ans;
        x = BigInteger.valueOf(a); y = BigInteger.valueOf(b); z =
BigInteger.valueOf(c);
        ans = x.add(y); System.out.println(ans);
        ans = z.divide(y); System.out.println(ans);
        ans = x.mod(z); System.out.println(ans);
        if (ans.compareTo(x) == 0) System.out.println("1");
    }
}
BigInteger add(BigInteger other)
BigInteger subtract(BigInteger other)
BigInteger multiply(BigInteger other)
BigInteger divide(BigInteger other)
BigInteger mod(BigInteger other)
int compareTo(BigInteger other)
static BigInteger valueOf(long x)
x.equals(BigInteger.ZERO)
a.gcd(b) //常用函数

stripTrailingZeros //清除不影响大小的零
toPlainString //以朴素的方式（而非科学计数法或工程计数法）将BigDecimal转换成String
hasNext //判断是否读入完成
next //读入字符串, 直到空格为止
nextLine //读入一整行字符串
public String substring(int beginIndex)
//返回该字符串从beginIndex开始到结尾的子字符串;
public String substring(int beginIndex,int endIndex)
//返回该字符串从beginIndex开始到endsIndex结尾的子字符串

import java.util.*;
import java.io.*;
import java.math.*;
```

```
public class Main {
    public static void main(String[] argc){
        Scanner sc=new Scanner(new BufferedInputStream(System.in));
        while(sc.hasNext()){
            String s=sc.next();
            int n=sc.nextInt();
            BigDecimal x=new BigDecimal(s);
            s=x.pow(n).stripTrailingZeros().toPlainString();
            if(s.charAt(0)=='0'){
                s=s.substring(1);
            }
            System.out.println(s);
        }
        sc.close();
    }
}
```

> 高精度数字本身不可变 每次调用产生新的对象

```
            String temp1 = "-10000000000000000000000000000000000000";
            BigInteger bg1 = new BigInteger(temp1);          //注意初始化的方式,使用字符串来初始化
            System.out.println(bg1.abs());                   //绝对值方法 object.abs()

            String temp2 = "100000000000000000000000000";
            BigInteger bg2 = new BigInteger(temp2);
            System.out.println(bg1.add(bg2));
//加法    object.add(BigInteger b)
            System.out.println(bg1.subtract(bg2));
//减法   返回为 bg1 - bg2  (this - param)
            System.out.println(bg1.multiply(bg2));           //乘法  返回 bg1 * bg2
            System.out.println(bg1.divide(bg2));             //除法  返回bg1 / bg2
            System.out.println(bg1.mod(bg2));
//取模运算 返回的是   bg1%bg2 (this mod param)
            System.out.println(bg1.gcd(bg2));
//直接封装好了 求解bg1,bg2 的最大公约数
            int temp5 = 5;
            System.out.println(bg2.pow(temp5));
//乘方运算  注意这个方法的参数是基本类型int
```

```
            System.out.println(bg2.compareTo(bg1));          // 比较方法 结果为1
bg2大
            System.out.println(bg1.compareTo(bg2));          // 结果为-1  bg2大
           //这个地方注意比较的方法，还有一个方法是equal()
            String temp3 = "1000";
            String temp4 = "001000";
            BigInteger bg3  = new BigInteger(temp3);
            BigInteger bg4 = new BigInteger(temp4);
            System.out.println(bg3.compareTo(bg4));          //结果为0 表示相等
            System.out.println(bg3.equals(bg4));
//返回结果为true      这样看是没有区别，但是更推荐比较的时候使用compareTo()方法，
//在BigDecimal更直观，例如0.1 与0.10 ，equal返回false 而compareTo则是正确的结果。



            String temp1 = "1.22222222222222222222222";
            BigDecimal bd1 = new BigDecimal(temp1);
            String temp2 = "2.33333333333333333333333";
            BigDecimal bd2 = new BigDecimal(temp2);
            System.out.println(bd1.add(bd2));
// 加法   输出    3.55555555555555555555552
            System.out.println(bd1.add(bd2).doubleValue());
// 输出     3.5555555555555554 这里用了一个方法将结果转化为double类型了

            System.out.println(bd2.subtract(bd1));
//减法     输出 1.11111111111111111111108
            System.out.println(bd2.subtract(bd1).doubleValue());
//输出    1.1111111111111112

            System.out.println(bd2.multiply(bd1));
//乘法 输出     2.85185185185185185185185139259259259259259259259926
            System.out.println(bd2.multiply(bd1).doubleValue());
//乘法     2.8518518518518516

            System.out.println(bd2.divide(bd1, 5, RoundingMode.HALF_UP));

//除法应该注意很有可能会有除不尽的情况，这时候会有异常抛出，所以要传入控制参数
            System.out.println(bd2.divide(bd1, 5,
RoundingMode.HALF_UP).doubleValue());
//输出都是   1.90909

            System.out.println(bd1.compareTo(bd2));                  //比较方法
            BigDecimal bd3 = new BigDecimal("1.20");
            BigDecimal bd4 = new BigDecimal("1.2");
            System.out.println(bd3.compareTo(bd4));           //返回0表示相等
```

```
            System.out.println(bd3.equals(bd4));              //返回的是false  是
错误的
                                                              //所以比较的时候使用
compareTo()方法
```

进制转换

```
//函数:
String st = Integer.toString(num, base); // 把num当做10进制的数转成base进制的
st(base <= 35).
int num = Integer.parseInt(st, base); // 把st当做base进制，转成10进制的
int(parseInt有两个参数,第一个为要转的字符串,第二个为说明是什么进制).
BigInter m = new BigInteger(st, base); // st是字符串，base是st的进制.
//1.如果要将一个大数以2进制形式读入 可以使用
cin.nextBigInteger(2);
//2.如果要将一个大数转换成其他进制形式的字符串 使用
cin.toString(2);//将它转换成2进制表示的字符串
```

```
import java.io.*;
import java.util.*;
import java.math.*;
public classMain
{
    public static void main(String[] args)
    {
        int b;
        BigInteger p,m,ans;
        String str ;
        Scanner cin = new Scanner (new BufferedInputStream(System.in));
        while(cin.hasNext())
        {
            b=cin.nextInt();
            if(b==0)
                break;
            p=cin.nextBigInteger(b);
            m=cin.nextBigInteger(b);
            ans=p.mod(m);
            str=ans.toString(b);
            System.out.println(str);
        }
    }
```

```
}
```