[acm中Java的应用](http://blog.csdn.net/samjustin1/article/details/52087258)

Chapter I.

Java的优缺点各种书上都有，这里只说说用Java做ACM-ICPC的特点：

(1) 最明显的好处是，学会Java，可以参加Java Challenge    :)

(2) 对于熟悉C/C++的程序员来说，Java 并不难学，找本书，一两周业余时间就可以搞定了。当然，这里只是指一般编程，想熟悉所有的Java库还是需要些时间的。

      事实上，Java 只相当于C++的一个改进版，所有的语法都几乎是C++的，很少有变动。

(3) 在一般比赛中，Java程序会有额外的时间和空间，而实际上经过实验，在执行计算密集任务的时候Java并不比C/C++慢多少，只是IO操作较慢而已。

(4) Java 简单而功能强大，有些东西用Java实现起来更为方便，比如高精度。

(5) 用Java不易犯细微的错误，比如C/C++中的指针， “if (n = m) ... ” 等

(6) 目前来看Eclipse已成基本配置，写Java程序反而比C/C++更方便调试。在具体竞赛时也算多一种选择。

(7) 学会Java对以后工作有好处。现在国外很多地方会Java的人比会C/C++的人多。

(8) 会Java可以使你看起来更像偶蹄类动物（牛）      hoho~

Chapter II.

下面说一下ACM-ICPC队员初用Java编程所遇到的一些问题：

1. 基本输入输出：

(1)

JDK1.5.0新增的Scanner类为输入提供了良好的基础，简直就是为ACM-ICPC而设的。

一般用法为：

**[java]** [view plain](http://blog.csdn.net/samjustin1/article/details/52087258) [copy](http://blog.csdn.net/samjustin1/article/details/52087258)

1. **import** java.io.\*
3. **import** java.util.\*
5. **public** classMain
7. {
9. **public** **static** **void** main(String args[])
11. {
13. Scanner cin = **new** Scanner(**new** BufferedInputStream(System.in));
15. ...
17. }
19. }

当然也可以直接 Scanner cin = new Scanner(System.in);

只是加Buffer可能会快一些

(2)

读一个整数：    int n = cin.nextInt();          相当于    scanf("%d", &n);    或 cin >> n;

读一个字符串：String s = cin.next();          相当于    scanf("%s", s);      或 cin >> s;

读一个浮点数：double t = cin.nextDouble();    相当于    scanf("%lf", &t); 或 cin >> t;

读一整行：      String s = cin.nextLine();      相当于    gets(s);            或 cin.getline(...);

判断是否有下一个输入可以用 cin.hasNext() 或 cin.hasNextInt() 或 cin.hasNextDouble() 等，具体见 TOJ 1001 例程。

(3)

输出一般可以直接用 System.out.print() 和 System.out.println()，前者不输出换行，而后者输出。

比如：System.out.println(n);    // n 为 int 型

同一行输出多个整数可以用

      System.out.println(new Integer(n).toString() + " " + new Integer(m).toString());

也可重新定义：

static PrintWriter cout = new PrintWriter(new BufferedOutputStream(System.out));

cout.println(n);

(4)对于输出浮点数保留几位小数的问题，可以使用DecimalFormat类，

**[java]** [view plain](http://blog.csdn.net/samjustin1/article/details/52087258) [copy](http://blog.csdn.net/samjustin1/article/details/52087258)

1. DecimalFormat f = **new** DecimalFormat("#.00#");
3. DecimalFormat g = **new** DecimalFormat("0.000");
5. **double** a = 123.45678, b = 0.12;
7. System.out.println(f.format(a));
9. System.out.println(f.format(b));
11. System.out.println(g.format(b));

这里0指一位数字，#指除0以外的数字。

2. 大数字

BigInteger 和 BigDecimal 是在java.math包中已有的类，前者表示整数，后者表示浮点数

用法：

不能直接用符号如+、-来使用大数字，例如：

(import java.math.\*)    // 需要引入 java.math 包

BigInteger a = BigInteger.valueOf(100);

BigInteger b = BigInteger.valueOf(50);

BigInteger c = a.add(b)    // c = a + b;

主要有以下方法可以使用：

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)

输出大数字时直接使用 System.out.println(a) 即可。

3. 字符串

String 类用来存储字符串，可以用charAt方法来取出其中某一字节，计数从0开始：

String a = "Hello";      // a.charAt(1) = ’e’

用substring方法可得到子串，如上例

System.out.println(a.substring(0, 4))      // output "Hell"

注意第2个参数位置上的字符不包括进来。这样做使得 s.substring(a, b) 总是有 b-a个字符。

字符串连接可以直接用 + 号，如

String a = "Hello";

String b = "world";

System.out.println(a + ", " + b + "!");      // output "Hello, world!"

如想直接将字符串中的某字节改变，可以使用另外的StringBuffer类。

4. 调用递归（或其他动态方法）

在主类中 main 方法必须是 public static void 的，在 main 中调用非static类时会有警告信息，

可以先建立对象，然后通过对象调用方法：

5. 其他注意的事项

(1) Java 是面向对象的语言，思考方法需要变换一下，里面的函数统称为方法，不要搞错。

(2) Java 里的数组有些变动，多维数组的内部其实都是指针，所以Java不支持fill多维数组。

      数组定义后必须初始化，如 int[] a = new int[100];

(3) 布尔类型为 boolean，只有true和false二值，在 if (...) / while (...) 等语句的条件中必须为boolean类型。

      在C/C++中的 if (n % 2) ... 在Java中无法编译通过。

(4) 下面在java.util包里Arrays类的几个方法可替代C/C++里的memset、qsort/sort 和 bsearch:

Arrays.fill()

Arrays.sort()

Arrays.binarySearch()

[ACM中java的使用](http://blog.csdn.net/samjustin1/article/details/52087185)

这里指的java速成，只限于java语法，包括输入输出，运算处理，字符串和高精度的处理，进制之间的转换等，能解决OJ上的一些高精度题目。

1. 输入：

格式为：Scanner cin = new Scanner (new BufferedInputStream(System.in));

例：

**[java]** [view plain](http://blog.csdn.net/samjustin1/article/details/52087185) [copy](http://blog.csdn.net/samjustin1/article/details/52087185)

1. **import** java.io.\*;
3. **import** java.math.\*;
5. **import** java.util.\*;
7. **import** java.text.\*;
9. **public** classMain
11. {
13. **public** **static** **void** main(String[] args)
15. {
17. Scanner cin = **new** Scanner (**new** BufferedInputStream(System.in));
19. **int** a; **double** b; BigInteger c; String st;
21. 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.
23. }
25. }

2. 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.

例：

**[java]** [view plain](http://blog.csdn.net/samjustin1/article/details/52087185) [copy](http://blog.csdn.net/samjustin1/article/details/52087185)

1. **import** java.io.\*;
3. **import** java.math.\*;
5. **import** java.util.\*;
7. **import** java.text.\*;
9. **public** classMain
11. {
13. **public** **static** **void** main(String[] args)
15. {
17. Scanner cin = **new** Scanner (**new** BufferedInputStream(System.in));
19. **int** a; **double** b;
21. a = 12345; b = 1.234567;
23. System.out.println(a + " " + b);
25. System.out.printf("%d %10.5f\n", a, b); // 输入b为字宽为10，右对齐，保留小数点后5位，四舍五入.
27. }
29. }

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));

3. 字符串处理

java中字符串String是不可以修改的，要修改只能转换为字符数组.

例程：

**[java]** [view plain](http://blog.csdn.net/samjustin1/article/details/52087185) [copy](http://blog.csdn.net/samjustin1/article/details/52087185)

1. **import** java.io.\*;
3. **import** java.math.\*;
5. **import** java.util.\*;
7. **import** java.text.\*;
9. **public** classMain
11. {
13. **public** **static** **void** main(String[] args)
15. {
17. **int** i;
19. Scanner cin = **new** Scanner (**new** BufferedInputStream(System.in));
21. String st = "abcdefg";
23. System.out.println(st.charAt(0)); // st.charAt(i)就相当于st[i].
25. **char** [] ch;
27. ch = st.toCharArray(); // 字符串转换为字符数组.
29. **for** (i = 0; i < ch.length; i++) ch[i] += 1;
31. System.out.println(ch); // 输入为“bcdefgh”.
33. **if** (st.startsWith("a")) // 如果字符串以'0'开头.
35. {
37. st = st.substring(1); // 则从第1位开始copy(开头为第0位).
39. }
41. }
43. }

4. 高精度

BigInteger和BigDecimal可以说是acmer选择java的首要原因。

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().

例程：

**[java]** [view plain](http://blog.csdn.net/samjustin1/article/details/52087185) [copy](http://blog.csdn.net/samjustin1/article/details/52087185)

1. **import** java.io.\*;
3. **import** java.math.\*;
5. **import** java.util.\*;
7. **import** java.text.\*;
9. **public** classMain
11. {
13. **public** **static** **void** main(String[] args)
15. {
17. Scanner cin = **new** Scanner (**new** BufferedInputStream(System.in));
19. **int** a = 123, b = 456, c = 7890;
21. BigInteger x, y, z, ans;
23. x = BigInteger.valueOf(a); y = BigInteger.valueOf(b); z = BigInteger.valueOf(c);
25. ans = x.add(y); System.out.println(ans);
27. ans = z.divide(y); System.out.println(ans);
29. ans = x.mod(z); System.out.println(ans);
31. **if** (ans.compareTo(x) == 0) System.out.println("1");
33. }
35. }

5. 进制转换

java很强大的一个功能。

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.

//Added by abilitytao

1.如果要将一个大数以2进制形式读入 可以使用cin.nextBigInteger(2);

当然也可以使用其他进制方式读入；

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

例程：POJ 2305

**[java]** [view plain](http://blog.csdn.net/samjustin1/article/details/52087185) [copy](http://blog.csdn.net/samjustin1/article/details/52087185)

1. **import** java.io.\*;
3. **import** java.util.\*;
5. **import** java.math.\*;
7. **public** classMain
9. {
11. **public** **static** **void** main(String[] args)
13. {
15. **int** b;
17. BigInteger p,m,ans;
19. String str ;
21. Scanner cin = **new** Scanner (**new** BufferedInputStream(System.in));
23. **while**(cin.hasNext())
25. {
27. b=cin.nextInt();
29. **if**(b==0)
31. **break**;
33. p=cin.nextBigInteger(b);
35. m=cin.nextBigInteger(b);
37. ans=p.mod(m);
39. str=ans.toString(b);
41. System.out.println(str);
43. }
45. }
47. }

6. 排序

函数：Arrays.sort();至于怎么排序结构体，像C++里写个cmp的方法，在java还不太清楚，希望有人指点下～～

例程：

**[java]** [view plain](http://blog.csdn.net/samjustin1/article/details/52087185) [copy](http://blog.csdn.net/samjustin1/article/details/52087185)

1. **import** java.io.\*;
3. **import** java.math.\*;
5. **import** java.util.\*;
7. **import** java.text.\*;
9. **public** classMain
11. {
13. **public** **static** **void** main(String[] args)
15. {
17. Scanner cin = **new** Scanner (**new** BufferedInputStream(System.in));
19. **int** n = cin.nextInt();
21. **int** a[] = **new** **int** [n];
23. **for** (**int** i = 0; i < n; i++) a[i] = cin.nextInt();
25. Arrays.sort(a);
27. **for** (**int** i = 0; i < n; i++) System.out.print(a[i] + " ");
29. }
31. }

7. POJ高精度题目汇总：

POJ 1131 1205 1220 1405 1503 1604 1894 2084 2305 2325 2389 2413 3101 3199

[ACM之Java输入输出](http://blog.csdn.net/samjustin1/article/details/52087061)

**一、Java之ACM注意点**

1. 类名称必须采用public class Main方式命名

2. 在有些OJ系统上，即便是输出的末尾多了一个“ ”，程序可能会输出错误，所以在我看来好多OJ系统做的是非常之垃圾

3. 有些OJ上的题目会直接将OI上的题目拷贝过来，所以即便是题目中有输入和输出文件，可能也不需要，因为在OJ系统中一般是采用标准输入输出，不需要文件

4. 在有多行数据输入的情况下，一般这样处理，

**[java]** [view plain](http://blog.csdn.net/samjustin1/article/details/52087061) [copy](http://blog.csdn.net/samjustin1/article/details/52087061)

1. **static** Scanner in = **new** Scanner(System.in);
2. **while**(in.hasNextInt())
3. 或者是
4. **while**(in.hasNext())

5. 有关System.nanoTime()函数的使用，该函数用来返回最准确的可用系统计时器的当前值，以毫微秒为单位。

**[java]** [view plain](http://blog.csdn.net/samjustin1/article/details/52087061) [copy](http://blog.csdn.net/samjustin1/article/details/52087061)

1. **long** startTime = System.nanoTime();
2. // ... the code being measured ...
3. **long** estimatedTime = System.nanoTime() - startTime;

**二、Java之输入输出处理**

由于ACM竞赛题目的输入数据和输出数据一般有多组（不定），并且格式多种多样，所以，如何处理题目的输入输出是对大家的一项最基本的要求。这也是困扰初学者的一大问题。

1. 输入：

格式1：Scanner sc = new Scanner (new BufferedInputStream(System.in));

格式2：Scanner sc = new Scanner (System.in);

在读入数据量大的情况下，格式1的速度会快些。

读一个整数： int n = sc.nextInt(); 相当于 scanf("%d", &n); 或 cin >> n;

读一个字符串：String s = sc.next(); 相当于 scanf("%s", s); 或 cin >> s;

读一个浮点数：double t = sc.nextDouble(); 相当于 scanf("%lf", &t); 或 cin >> t;

读一整行： String s = sc.nextLine(); 相当于 gets(s); 或 cin.getline(...);

判断是否有下一个输入可以用sc.hasNext()或sc.hasNextInt()或sc.hasNextDouble()或sc.hasNextLine()

例1：读入整数

Input  输入数据有多组，每组占一行，由一个整数组成。   
Sample Input   
56  
67  
100  
123

**[java]** [view plain](http://blog.csdn.net/samjustin1/article/details/52087061) [copy](http://blog.csdn.net/samjustin1/article/details/52087061)

1. **import** java.util.Scanner;
3. **public** **class** Main {
4. **public** **static** **void** main(String[] args) {
5. Scanner sc = **new** Scanner(System.in);
6. **while** (sc.hasNext()) { // 判断是否结束
7. **int** score = sc.nextInt();// 读入整数
8. <span style="white-space:pre">            </span>。。。
9. }
10. }
11. }

例2：读入实数

输入数据有多组，每组占2行，第一行为一个整数N，指示第二行包含N个实数。

Sample Input  
4   
56.9  67.7  90.5  12.8   
5   
56.9  67.7  90.5  12.8 

**[java]** [view plain](http://blog.csdn.net/samjustin1/article/details/52087061) [copy](http://blog.csdn.net/samjustin1/article/details/52087061)

1. **import** java.util.Scanner;
3. **public** **class** Main {
4. **public** **static** **void** main(String[] args) {
5. Scanner sc = **new** Scanner(System.in);
6. **while** (sc.hasNext()) {
7. **int** n = sc.nextInt();
8. **for** (**int** i = 0; i < n; i++) {
9. **double** a = sc.nextDouble();

**[java]** [view plain](http://blog.csdn.net/samjustin1/article/details/52087061) [copy](http://blog.csdn.net/samjustin1/article/details/52087061)

1. <span style="white-space:pre">                </span>。。。
2. }
3. }
4. }
5. }

例3：读入字符串【杭电2017 字符串统计】

输入数据有多行，第一行是一个整数n，表示测试实例的个数，后面跟着n行，每行包括一个由字母和数字组成的字符串。

Sample Input    
2  
asdfasdf123123asdfasdf  
asdf111111111asdfasdfasdf

**[java]** [view plain](http://blog.csdn.net/samjustin1/article/details/52087061) [copy](http://blog.csdn.net/samjustin1/article/details/52087061)

1. **import** java.util.Scanner;
3. **public** **class** Main {
4. **public** **static** **void** main(String[] args) {
5. Scanner sc = **new** Scanner(System.in);
6. **int** n = sc.nextInt();
7. **for** (**int** i = 0; i < n; i++) {
8. String str = sc.next();
10. }
11. }
12. }

**[java]** [view plain](http://blog.csdn.net/samjustin1/article/details/52087061) [copy](http://blog.csdn.net/samjustin1/article/details/52087061)

1. **import** java.util.Scanner;
3. **public** **class** Main {
4. **public** **static** **void** main(String[] args) {
5. Scanner sc = **new** Scanner(System.in);
6. **int** n = Integer.parseInt(sc.nextLine());
7. **for** (**int** i = 0; i < n; i++) {
8. String str = sc.nextLine();
10. }
11. }
12. }

例3：读入字符串【杭电2005 第几天？】  
给定一个日期，输出这个日期是该年的第几天。   
Input  输入数据有多组，每组占一行，数据格式为YYYY/MM/DD组成  
1985/1/20  
2006/3/12

**[java]** [view plain](http://blog.csdn.net/samjustin1/article/details/52087061) [copy](http://blog.csdn.net/samjustin1/article/details/52087061)

1. **import** java.util.Scanner;
3. **public** **class** Main {
4. **public** **static** **void** main(String[] args) {
5. Scanner sc = **new** Scanner(System.in);
6. **int**[] dd = { 0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };
7. **while** (sc.hasNext()) {
8. **int** days = 0;
9. String str = sc.nextLine();
10. String[] date = str.split("/");
11. **int** y = Integer.parseInt(date[0]);
12. **int** m = Integer.parseInt(date[1]);
13. **int** d = Integer.parseInt(date[2]);
14. **if** ((y % 400 == 0 || (y % 4 == 0 && y % 100 != 0)) && m > 2)
15. days++;
16. days += d;
17. **for** (**int** i = 0; i < m; i++) {
18. days += dd[i];
19. }
20. System.out.println(days);
21. }
22. }
23. }

2. 输出

函数：

System.out.print();

System.out.println();

System.out.format();

System.out.printf();

例4 杭电1170Balloon Comes!

Give you an operator (+,-,\*, / --denoting addition, subtraction, multiplication, division respectively) and two positive integers, your task is to output the result.

Input

Input contains multiple test cases. The first line of the input is a single integer T (0<T<1000) which is the number of test cases. T test cases follow. Each test case contains a char C (+,-,\*, /) and two integers A and B(0<A,B<10000).Of course, we all know that A and B are operands and C is an operator.

Output

For each case, print the operation result. The result should be rounded to 2 decimal places If and only if it is not an integer.

Sample Input

4

+ 1 2

- 1 2

\* 1 2

/ 1 2

Sample Output

3

-1

2

0.50

**[java]** [view plain](http://blog.csdn.net/samjustin1/article/details/52087061) [copy](http://blog.csdn.net/samjustin1/article/details/52087061)

1. **import** java.util.Scanner;
3. **public** **class** Main {
4. **public** **static** **void** main(String[] args) {
5. Scanner sc = **new** Scanner(System.in);
6. **int** n = sc.nextInt();
7. **for** (**int** i = 0; i < n; i++) {
8. String op = sc.next();
9. **int** a = sc.nextInt();
10. **int** b = sc.nextInt();
11. **if** (op.charAt(0) == '+') {
12. System.out.println(a + b);
13. } **else** **if** (op.charAt(0) == '-') {
14. System.out.println(a - b);
15. } **else** **if** (op.charAt(0) == '\*') {
16. System.out.println(a \* b);
17. } **else** **if** (op.charAt(0) == '/') {
18. **if** (a % b == 0)
19. System.out.println(a / b);
20. **else**
21. System.out.format("%.2f", (a / (1.0 \* b))).Println();
22. }
23. }
24. }
25. }

3. 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));

**[java]** [view plain](http://blog.csdn.net/samjustin1/article/details/52087061) [copy](http://blog.csdn.net/samjustin1/article/details/52087061)

1. **import** java.util.Scanner;
3. **public** **class** Main {
4. **public** **static** **void** main(String[] args) {
5. NumberFormat formatter = **new** DecimalFormat("000000");
6. String s = formatter.format(-1234.567); // -001235
7. System.out.println(s);
8. formatter = **new** DecimalFormat("##");
9. s = formatter.format(-1234.567); // -1235
10. System.out.println(s);
11. s = formatter.format(0); // 0
12. System.out.println(s);
13. formatter = **new** DecimalFormat("##00");
14. s = formatter.format(0); // 00
15. System.out.println(s);
17. formatter = **new** DecimalFormat(".00");
18. s = formatter.format(-.567); // -.57
19. System.out.println(s);
20. formatter = **new** DecimalFormat("0.00");
21. s = formatter.format(-.567); // -0.57
22. System.out.println(s);
23. formatter = **new** DecimalFormat("#.#");
24. s = formatter.format(-1234.567); // -1234.6
25. System.out.println(s);
26. formatter = **new** DecimalFormat("#.######");
27. s = formatter.format(-1234.567); // -1234.567
28. System.out.println(s);
29. formatter = **new** DecimalFormat(".######");
30. s = formatter.format(-1234.567); // -1234.567
31. System.out.println(s);
32. formatter = **new** DecimalFormat("#.000000");
33. s = formatter.format(-1234.567); // -1234.567000
34. System.out.println(s);
36. formatter = **new** DecimalFormat("#,###,###");
37. s = formatter.format(-1234.567); // -1,235
38. System.out.println(s);
39. s = formatter.format(-1234567.890); // -1,234,568
40. System.out.println(s);
42. // The ; symbol is used to specify an alternate pattern for negative
43. // values
44. formatter = **new** DecimalFormat("#;(#) ");
45. s = formatter.format(-1234.567); // (1235)
46. System.out.println(s);
48. // The ' symbol is used to quote literal symbols
49. formatter = **new** DecimalFormat(" '# '# ");
50. s = formatter.format(-1234.567); // -#1235
51. System.out.println(s);
52. formatter = **new** DecimalFormat(" 'abc '# ");
53. s = formatter.format(-1234.567); // - abc 1235
54. System.out.println(s);
56. formatter = **new** DecimalFormat("#.##%");
57. s = formatter.format(-12.5678987);
58. System.out.println(s);
59. }
60. }

4. 字符串处理 String

String 类用来存储字符串，可以用charAt方法来取出其中某一字节，计数从0开始：

String a = "Hello"; // a.charAt(1) = 'e'

用substring方法可得到子串，如上例

System.out.println(a.substring(0, 4)) // output "Hell"

注意第2个参数位置上的字符不包括进来。这样做使得 s.substring(a, b) 总是有 b-a个字符。

字符串连接可以直接用 + 号，如

String a = "Hello";

String b = "world";

System.out.println(a + ", " + b + "!"); // output "Hello, world!"

如想直接将字符串中的某字节改变，可以使用另外的StringBuffer类。

5. 高精度  
BigInteger和BigDecimal可以说是acmer选择java的首要原因。  
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().

**[java]** [view plain](http://blog.csdn.net/samjustin1/article/details/52087061) [copy](http://blog.csdn.net/samjustin1/article/details/52087061)

1. **import** java.io.BufferedInputStream;
2. **import** java.math.BigInteger;
3. **import** java.util.Scanner;
4. **public** **class** Main {
5. **public** **static** **void** main(String[] args)   {
6. Scanner cin = **new** Scanner (**new** BufferedInputStream(System.in));
7. **int** a = 123, b = 456, c = 7890;
8. BigInteger x, y, z, ans;
9. x = BigInteger.valueOf(a);
10. y = BigInteger.valueOf(b);
11. z = BigInteger.valueOf(c);
12. ans = x.add(y); System.out.println(ans);
13. ans = z.divide(y); System.out.println(ans);
14. ans = x.mod(z); System.out.println(ans);
15. **if** (ans.compareTo(x) == 0) System.out.println("1");
16. }
17. }

6. 进制转换  
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.

7. 数组排序  
函数：Arrays.sort();

**[java]** [view plain](http://blog.csdn.net/samjustin1/article/details/52087061) [copy](http://blog.csdn.net/samjustin1/article/details/52087061)

1. **public** **class** Main {
2. **public** **static** **void** main(String[] args)    {
3. Scanner cin = **new** Scanner (**new** BufferedInputStream(System.in));
4. **int** n = cin.nextInt();
5. **int** a[] = **new** **int** [n];
6. **for** (**int** i = 0; i < n; i++) a[i] = cin.nextInt();
7. Arrays.sort(a);
8. **for** (**int** i = 0; i < n; i++) System.out.print(a[i] + " ");
9. }
10. }