java中的输出a+b

import java.io.\*;

import java.util.\*;

public class Main

{

    public static void main(String[] args)

    {

        Scanner in=new Scanner (System.in);

        while(in.hasNext())

        {

            int a,b;

            a=in.nextInt();

            b=in.nextInt();

            System.out.println(a+b);

        }

    }

}

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));

或者、Scanner cin = new Scanner (System.in);

例程：

import java.io.\*;

import java.math.\*;

import java.util.\*;

import java.text.\*;

public class Main

{

    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.

    }

}

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.

例程：

import java.io.\*;

import java.math.\*;

import java.util.\*;

import java.text.\*;

public class Main

{

    public static void main(String[] args)

    {

        Scanner cin = new Scanner (new BufferedInputStream(System.in));

        int a; double b;

        a = 12345; b = 1.234567;

        System.out.println(a + " " + b);

        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.

    }

}

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：

import java.io.\*;

import java.math.\*;

import java.util.\*;

import java.text.\*;

public class Main

{

    public static void main(String[] args)

    {

        int i;

        Scanner cin = new Scanner (new BufferedInputStream(System.in));

        String st = "abcdefg";

        System.out.println(st.charAt(0)); // st.charAt(i)就相当于st[i].

        char [] ch;

        ch = st.toCharArray(); // 字符串转换为字符数组.

        for (i = 0; i < ch.length; i++) ch[i] += 1;

        System.out.println(ch); // 输入为“bcdefgh”.

if (st.startsWith("a")) // 如果字符串以'0'开头.

        {

            st = st.substring(1); // 则从第1位开始copy(开头为第0位).

        }

    }

}

4. 高精度

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

函数：add, subtract, multiply,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：

import java.io.\*;

import java.math.\*;

import java.util.\*;

import java.text.\*;

public class Main

{

    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");

    }

}

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

import java.io.\*;

import java.util.\*;

import java.math.\*;

public class Main

{

    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);

        }

    }

}

//End by abilitytao

6. 排序

函数：Arrays.sort()；

例程：

import java.io.\*;

import java.math.\*;

import java.util.\*;

import java.text.\*;

public class Main

{

    public static void main(String[] args)

    {

        Scanner cin = new Scanner (new BufferedInputStream(System.in));

        int n = cin.nextInt();

        int a[] = new int [n];

        for (int i = 0; i < n; i++) a[i] = cin.nextInt();

        Arrays.sort(a);

        for (int i = 0; i < n; i++) System.out.print(a[i] + " ");

    }

}

7. 结构体排序：

例子：一个结构体有两个元素String x，int y，排序，如果x相等y升序，否者x升序。

一、Comparator

强行对某个对象collection进行整体排序的比较函数，可以将Comparator传递给Collections.sort或Arrays.sort。

接口方法：这里也给出了两种方法，

import java.util.\*;

class structSort{

    String x;

    int y;

}

class cmp implements Comparator<structSort>{

public int compare(structSort o1, structSort o2) {

if(o1.x.compareTo(o2.x) == 0){//这个相当于c/c++中strcmp（o1.x , o2,x）

return o1.y - o2.y;

}

return o1.x.compareTo(o2.x);

}

}

public class Main {

public static void main(String[] args) {

 Comparator<structSort> comparator = new Comparator<structSort>(){

public int compare(structSort o1, structSort o2) {

if(o1.x.compareTo(o2.x) == 0){

return o1.y - o2.y;

}

return o1.x.compareTo(o2.x);

}

 };

 Scanner cin = new Scanner(System.in);

 int n = cin.nextInt();

 structSort a[] = new structSort[10];

 for (int i = 0; i < n; i++) {

a[i] = new structSort();

a[i].x = cin.next();

a[i].y = cin.nextInt();

}

 Arrays.sort(a,0,n,comparator);//这个直接使用Comparator

 Arrays.sort(a,0,n,new cmp());//这个实现Comparator，就就跟c++中的sort函数调用就差不多了

 for (int i = 0; i < n; i++) {

System.out.println(a[i].x+" "+a[i].y);

}

}

}

 二、Comparable

强行对实现它的每个类的对象进行整体排序，实现此接口的对象列表（和数组）可以通过Collections.sort或Arrays.sort进行自动排序。就是输入完了直接就默认排序了，

接口方法：

import java.util.\*;

class structSort implements Comparable<structSort>{

    String x;

    int y;

    public int compareTo(structSort o1) {

if(this.x.compareTo(o1.x) == 0){

return this.y - o1.y;

}

return this.x.compareTo(o1.x);

}

}

public class Main {

public static void main(String[] args) {

 Scanner cin = new Scanner(System.in);

 int n = cin.nextInt();

 structSort a[] = new structSort[10];

 for (int i = 0; i < n; i++) {

a[i] = new structSort();

a[i].x = cin.next();

a[i].y = cin.nextInt();

}

 Arrays.sort(a,0,n);

 for (int i = 0; i < n; i++) {

System.out.println(a[i].x+" "+a[i].y);

}

}

}

acm中Java的应用

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)

JDK 1.5.0 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。

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()

(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，

import java.text.\*;

DecimalFormat f = new DecimalFormat("#.00#");

DecimalFormat g = new DecimalFormat("0.000");

double a = 123.45678, b = 0.12;

System.out.println(f.format(a));

System.out.println(f.format(b));

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：

public class Main

{

      ...

      void dfs(int a)

      {

          if (...) return;

          ...

          dfs(a+1);

      }

      public static void main(String args[])

      {

          ...

          Main e = new Main();

          e.dfs(0);

          ...

      }

}

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()

转自：http://hi.baidu.com/oak\_wesley/blog/item/35839200fd9dc10e1d9583de.html

Java进制转换~集锦

由于Unicode兼容ASCII（0～255），因此，上面得到的Unicode就是ASCII。

java中进行二进制，八进制，十六进制，十进制间进行相互转换

Integer.toHexString(int i)

十进制转成十六进制

Integer.toOctalString(int i)

十进制转成八进制

Integer.toBinaryString(int i)

十进制转成二进制

Integer.valueOf("FFFF",16).toString()

十六进制转成十进制

Integer.valueOf("876",8).toString()

八进制转成十进制

Integer.valueOf("0101",2).toString()

二进制转十进制

至于转换成二进制或其他进制，Java API提供了方便函数，你可以查Java的API手册。

以字符a的ASCII为例：

int i = 'a';

String iBin = Integer.toBinaryString(i);//二进制

String iHex = Integer.toHexString(i);//十六进制

String iOct = Integer.toOctalString(i);//八进制

String iWoKao = Integer.toString(i,3);//三进制或任何你想要的35进制以下的进制

有什么方法可以直接将2,8,16进制直接转换为10进制的吗?

java.lang.Integer类 parseInt(String s, int radix)

使用第二个参数指定的基数，将字符串参数解析为有符号的整数。

examples from jdk:

parseInt("0", 10) returns 0

parseInt("473", 10) returns 473

parseInt("-0", 10) returns 0

parseInt("-FF", 16) returns -255

parseInt("1100110", 2) returns 102

parseInt("2147483647", 10) returns 2147483647

parseInt("-2147483648", 10) returns -2147483648

parseInt("2147483648", 10) throws a NumberFormatException

parseInt("99", 8) throws a NumberFormatException

parseInt("Kona", 10) throws a NumberFormatException

parseInt("Kona", 27) returns 411787

进制转换如何写（二，八，十六）不用算法

Integer.toBinaryString

Integer.toOctalString

Integer.toHexString

例一：

public class Test{

public static void main(String args[]){

   int i=100;

   String binStr=Integer.toBinaryString(i);

   String otcStr=Integer.toOctalString(i);

   String hexStr=Integer.toHexString(i);

   System.out.println(binStr);

例二：

public classTestStringFormat {

public static void main(String[] args) {

   if (args.length == 0) {

      System.out.println("usage: javaTestStringFormat <a number>");

      System.exit(0);

   }

   Integer factor =Integer.valueOf(args[0]);

   String s;

   s =String.format("%d", factor);

   System.out.println(s);

   s = String.format("%x", factor);

   System.out.println(s);

   s = String.format("%o", factor);

   System.out.println(s);

}

}

各种数字类型转换成字符串型：

String s = String.valueOf( value); // 其中 value 为任意一种数字类型。

字符串型转换成各种数字类型：

String s = "169";

byte b = Byte.parseByte( s );

short t = Short.parseShort( s );

int i = Integer.parseInt( s );

long l = Long.parseLong( s );

Float f = Float.parseFloat( s );

Double d = Double.parseDouble( s );

数字类型与数字类对象之间的转换：

byte b = 169;

Byte bo = new Byte( b );

b = bo.byteValue();

short t = 169;

Short to = new Short( t );

t = to.shortValue();

int i = 169;

b = bo.byteValue();

short t = 169;

Short to = new Short( t );

t = to.shortValue();

int i = 169;

Integer io = new Integer( i );

i = io.intValue();

long l = 169;

Long lo = new Long( l );

l = lo.longValue();

float f = 169f;

Float fo = new Float( f );

f = fo.floatValue();

double d = 169f;

Double dObj = new Double( d );

d = dObj.doubleValue();

分类: [ACM](http://www.cnblogs.com/gaojupeng/category/682121.html)