

# Java-Practice

---

关于java一些的日常练习

## 目录

原码补码反码

---

### 循环练习题

睡眠排序

冒泡排序

使用Math方法

二维数组遍历

学学人家作弊

数值反转

数组从新排列

---

判定年份是否为闰年

求大于200的最小质数

求1-100以内奇数和值

个人所得税

打印99表

求水仙花数

输出正三角形

数组排序找出最大值及其下标

清除数组中重复元素

数组拷贝

数组查重拷贝

计算器

---

### 对象与方法练习题

使用封装方法对三角形进行判断

## 文件管理类的使用方法

### 查找文件夹及文件名

---

---

## 原码补码反码

### 唯一运算

数字与数字 是相加 6+6 加号为+ 数字与字符串或字符串与数字 是连字符 +"6" 加号为连字符

switch语句 用char int 类型的jdk7以前的版本 可以用String类型，不过只有jdk7版本才可以用String ps:break 终止循环 变量的作用域与代码块{}有关 在当前代码块 或当前代码块内部的代码块中有效

---

## 循环练习题

### 睡眠排序

```
package test;

public class TestRunnable implements Runnable{

    private String num;

    public TestRunnable(int num) {
        this.num = num + "";
    }

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        int[] nums = {0,1,22,2,2,2,9,7,66,99,88,22};
        for(int i=0;i<nums.length;i++) {
            new Thread(new TestRunnable(nums[i])).start();
        }
    }

    @Override
    public void run() {
        // TODO Auto-generated method stub
        try {
            Thread.sleep(Integer.parseInt(num));
            System.out.println(num);

        } catch (NumberFormatException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        } catch (InterruptedException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
}
```

```
}

```

## 冒泡排序

使用冒牌排序，对A数组内数字进行由小到大排序

```
public static void main(String[] args) {
    //由小到大进行排序
    int[] a = {23,13,45,6,78,12,4,1};
    //冒泡排序
    //循环a.length次
    //循环次数 可以控制时间
    for(int j=0;j<a.length;j++){
        //1\最大值放在提取出来13 23 6 45 12 4 1 78
        //1\第二大放在提取出来13 6 23 12 4 1 45 78
        //每次排序进行比较 加j与不加j区别 需要优化
        for(int i=0;i<a.length-1-j;i++){
            if(a[i]>a[i+1]){
                int temp = 0;
                //a[i] a[i+1] 进行位置互换
                temp =a[i+1];
                a[i+1]=a[i];
                a[i]=temp;
            }
        }
    }
    //输出
    for(int n:a){
        System.out.print(n+" ");
    }

    //arrays.sort排序
    Arrays.sort(a);
    for(int i=0;i<a.length;i++){
        System.out.print(a[i]+" ");
    }
}
```

- 思想：冒泡排序，主要使用双重for循环，第一层for循环，为了控制循环次数， $j < a.length$  循环次数小于数组长度，否则会出现数组越界情况；for循环，为了比较数组内数组的大小，共对数组内数字比较 $a.length-1-j$ 次。if语句循环， $a[i]$  $a[i+1]$ 两两数进行比较，并互换位置。

## 使用Math方法

```
public static void main(String[] args) {
    //从大到小

```

```

int[] a = {34,45,67,49,97,43,89,3,113};
for(int j=0;j<a.length;j++){
    for(int i=0;i<a.length-j-1;i++){
        int temp=a[i];
        a[i] = Math.max(a[i], a[i+1]);
        a[i+1] = Math.min(temp, a[i+1]);
    }
}
for(int n:a){
    System.out.println(n);
}
}

```

## 二维数组遍历

### 使用三种循环方式对数组进行遍历

```

public static void main(String[] args) {
    //a[] 代表第几位数组 [] 代表第一个数组中的第几位数 相当于x,y轴
    int[][] a = { {1,2,3},{2,3,6} };
    System.out.println(a[0][0]);
    System.out.println("-----二维数组for循环的遍历-----");

    System.out.println("while循环遍历");
    int i=0;
    int n=0;
    while(n<a.length){
        while(i<a[n].length){
            System.out.print(a[n][i]+",");
            i++;
        }
        i=0;
        n++;
        System.out.println();
    }

    System.out.println("while for结合循环遍历");
    int i=0;
    for(int n=0;n<a.length;n++){
        while(i<a[n].length){
            System.out.print(a[n][i]+",");
            i++;
        }
        i=0;
        System.out.println();
    }

    System.out.println("foreach结合循环遍历");
    for(int[] b:a){
        for(int n:b){

```

```

        System.out.println(n+",");
    }
}
}

```

## 学学人家作弊

### 输出空心正方形

```

public static void main(String[] args) {
    //开始作弊了昂
    //直接打印64个*   如果对8取余   换行
    //使用数组
    int[] a = {
        1,1,1,1,1,1,1,1,
        1,0,0,0,0,0,0,1,
        1,0,0,0,0,0,0,1,
        1,0,0,0,0,0,0,1,
        1,0,0,0,0,0,0,1,
        1,0,0,0,0,0,0,1,
        1,0,0,0,0,0,0,1,
        1,0,0,0,0,0,0,1,
        1,1,1,1,1,1,1,1
    };
    for(int i=0;i<a.length;i++){
        if(a[i]==1){
            System.out.print("*");
        }else if(a[i]==0){
            System.out.print(" ");
        }
        if((i+1)%8==0){
            System.out.println();
        }
    }
}

```

- 思想：使用数组（可以打印任何形状，缺点：太麻烦，大数值键盘会碎=。=，无脑操作！），对数组进行定位 $a[i]$ ，若 $a[i]==1$ ，则输出 $*$ ，则输出其他内容（自己定）使用 $(i+1) \% \text{行数}$  控制来行数。

## 数值反转

### 输入四位整数1234 进行反转 4321

```

//第一种方法
public static void main(String[] args) {
    // 输入四位整数1234 进行反转 4321
}

```

```

Scanner s = new Scanner(System.in);
int a = s.nextInt();
int sum=0;

while(a!=0){
    /*
    * 1234
    * 4 321
    */
    sum = a%10+sum*10;
    a = a/10;
}
System.out.println(sum);
}
//第二种方法
public static void main(String[] args) {
// 控制台输入值
int a = 0;
int b = 0;
int temp = 0;
Scanner sc = new Scanner(System.in);
System.out.println("请输入a的值:");
a = sc.nextInt();
System.out.println("请输入b的值:");
b = sc.nextInt();
System.out.println("输入的a的值为"+a+"输入的b值为"+b);

temp = a;
a = b;
b = temp;
System.out.println("交换后:");
System.out.println("输入的a的值为"+a+"输入的b值为"+b);
}

```

思想：大部分第一时间都会想到：我可以将这些数按个位十位百位千位提炼出来，然后再依次放入对应的位置。我想说：方法很好，不错，但是你就没有想过优化一下嘛？来，让我们想想：每次取完位数，我们可以把它和对应位数互换，一次一次进行循环。

## 数组从新排列

有2个多维数组分别是  $2\ 3\ 4$ ,  $4\ 6\ 8$  和  $1\ 5\ 2\ 8$ ,  $5\ 9\ 10\ -3$ ,  $2\ 7\ -5\ -18$ 。按照如下方式进行运算。生成一个2行4列的数组。此数组的第1行1列是 $21+35+42$ 第1行2列是 $25+39+47$  第2行1列是 $41+65+8*2$  依次类推。

```

public static void main(String[] args) {
int[][] a = { { 2, 3, 4 }, { 4, 6, 8 } };
int[][] b = { { 1, 5, 2, 8 }, { 5, 9, 10, -3 }, { 2, 7, -5, -18 } };

int c[][] = new int[2][4];
int k=0,j=0,add=0;
for (k=0;k<2;k++) {

```

```

        for (j=0;j<4;j++) {
            for (int i = 0; i < 3; i++) {
                add = add + a[k][i] * b[i][j];
            }
            c[k][j] = add;
        }
    }
    //遍历二维数组
    for(int[] e:a){
        for(int n:e){
            System.out.println(n+",");
        }
    }
}

```

### 判定年份是否为闰年

编写程序，判断给定的某个年份是否是闰年。闰年的判断规则如下：（1）若某个年份能被4整除但不能被100整除，则是闰年。（2）若某个年份能被400整除，则也是闰年。

```

public static void main(String[] args) {
    System.out.println("请输入一个整数年份：");
    for(;;){
        Scanner s = new Scanner(System.in);
        int a = s.nextInt();
        if(a%400==0){
            System.out.println("今年是闰年啊！");continue;
        }else if(a%4==0&& a%100!=0){
            System.out.println("今年是闰年啊！");continue;
        }else{
            System.out.println("不是闰年哦！！");continue;
        }
    }
}

```

### 求大于200的最小质数

```

public static void main(String[] args) {
    boolean f = true;
    for(int i=200;;i++) {
        for(int j=2;j<i;j++) {
            if(i%j == 0) {
                //如果找到一个i能被j整除，则不是质数
                f = false;
            }
        }
        if(f) {

```

```

        System.out.println("大于200的最小质数: "+i);
        break;
    }
    f = true;
}
}

```

## 求1-100以内奇数和值

编写程序求截止到某位数之前所有奇数的和值。

```

public static void main(String[] args) {
    System.out.println("请输入一个正整数: ");
    Scanner sr = new Scanner(System.in);
    int n = sr.nextInt();
    int s = 0;
    //此处为for循环的关键点, a+=2: 求奇数
    for(int a = 1;a<=n;a+=2){
        s = s+a;
    }
    System.out.println("截止到"+n+"所有奇数和为: "+s);
}

```

- 思想: 输入一个数, 求这个数范围内奇数和。我们需要去想, 怎末才会得到奇数的问题, 最后再把他们相加起来呢? 奇数 = 1+2、3+2、5+2, 也就是说: 起点为奇数, 每次加2, 结果得到下一位奇数, 依次循环并相加这些奇数。提示: 运用for循环。

## 个人所得税

//计算个人所得税, 3500, 5000, 3500与5000税后金额差, 输入税前工资 显示税后工资 显示交了多少税

```

/*
 * 税前工资 4,500.00
 * 五险一金 300.00
 * 起征点 3,500.00
 *
 * 起征点: 指税法规定对征税对象开始征税的起点数额。征税对象的数额达到起征点的就全部数额征税, 未达到起征点的不征税。
 *
 * 应纳税所得额 = 税前工资 - 五险一金 - 起征点
 * = 4,500.00 - 300.00 - 3,500.00
 * = 700.00
 *
 * 个人所得税是采用速算扣除数法计算超额累进税率的所得税时的计税公式是:
 *
 * 速算扣除数是指用快捷方法计算税款时, 可以扣除的数额。个人所得税的计算, 可以将应纳税所得额分成不同级距和相应的税率来计算。

```



\* 如扣除3500元后的应纳税所得额在1500元以内的，按3%税率计算，1500-4500元之间的部分（即3000元），按10%的税率计算。

\*

\* 应纳税额 = 应纳税所得额 × 税率 - 速算扣除数

\* = 700.00 × 3% - 0.00

\* = 21.00

\* 税后工资 = 税前工资 - 五险一金 - 应纳个税

\* = 4,500.00 - 300.00 - 21.00

\* = 4,179.00

\*/

```
public static void main(String[] args) {
    sum();
}
```

```
}
```

```
//求差
```

```
private static void sum(){
```

```
    double s1 = 0;
```

```
    s1 = query1(s1);
```

```
    double s2 = 0;
```

```
    s2 = query2(s2);
```

```
    double c=0;
```

```
    if(s1==0){
```

```
        s1 = query1(s1);
```

```
    }
```

```
    else{
```

```
        if(s1>s2){
```

```
            c = s1-s2;
```

```
            System.out.println("税后工资"+s1+"与"+"税后工资"+s2+"两者之间的
```

```
差为: "+c);
```

```
        }else{
```

```
            c = s2-s1;
```

```
            System.out.println("税后工资"+s1+"与"+"税后工资"+s2+"两者之间的
```

```
差为: "+c);
```

```
        }
```

```
    }
```

```
    if(s2==0){
```

```
        s2 = query1(s2);
```

```
    }
```

```
    else{
```

```
        if(s1>s2){
```

```
            c = s1-s2;
```

```
            System.out.println("税后工资"+s1+"与"+"税后工资"+s2+"两者之间的
```

```
差为: "+c);
```

```
        }else{
```

```
            c = s2-s1;
```

```
            System.out.println("税后工资"+s1+"与"+"税后工资"+s2+"两者之间的
```

```
差为: "+c);
```

```
        }
```

```
    }
    System.out.println();

}
//3500元
protected static double query1(double s1){
    Scanner sc = new Scanner(System.in);

    System.out.println("请输入税前金额(此时起征点为3500):");
    double money = sc.nextDouble();
    if(money>=3500){
        System.out.println("请输入五险一金金额:");
        double w = sc.nextDouble();
        // System.out.println("请输入起征点金额:");
        // double q = sc.nextDouble();
        // 应纳税所得额 = 税前工资 - 五险一金 - 起征点
        double y = money-w-3500;
        System.out.println("应纳税所得金额: "+ y);

        // 速算扣除数 : 0
        // 应纳税额 = 应纳税所得额 × 税率 - 速算扣除数
        double ys = 0;//应纳税金额
        if(y>0&&y<=1500){

            ys = y*0.03-0.00;
            System.out.println("应纳税金额:"+ ys);
        }
        else if(y>1500&&y<=4500){

            ys = y*0.1-105.0;
            System.out.println("应纳税金额:"+ ys);
        }
        else if(y>4500&&y<=9000){

            ys = y*0.20-555.0;
            System.out.println("应纳税金额:"+ ys);
        }
        else if(y>9000&&y<=35000){

            ys = y*0.25-1005.0;
            System.out.println("应纳税金额:"+ ys);
        }
        else if(y>35000&&y<=55000){

            ys = y*0.30-2755.0;
            System.out.println("应纳税金额:"+ ys);
        }
        else if(y>55000&&y<=8000000){

            ys = y*0.35-5505.0;
            System.out.println("应纳税金额:"+ ys);
        }
    }
}
```

```
        else if(y>8000000){
            ys = y*0.45-13505.0;
            System.out.println("应纳税金额："+ ys);
        }
//    税后工资 = 税前工资 - 五险一金 - 应纳个税
    s1 = money-w-ys;
    System.out.println("税后工资"+s1);
    System.out.println();
}else{
    System.out.println("家里有矿啊!这点钱还交税????");
}
    return s1;
}

//5000
protected static double query2(double s2){
    Scanner sc = new Scanner(System.in);

    System.out.println("请输入税前金额(此时起征点为5000):");
    double money = sc.nextDouble();
    if(money>=5000){
        System.out.println("请输入五险一金金额:");
        double w = sc.nextDouble();
//    System.out.println("请输入起征点金额:");
//    double q = sc.nextDouble();
//    应纳税所得额 = 税前工资 - 五险一金 - 起征点
        double y = money-w-5000;
        System.out.println("应纳税所得金额:    "+ y);

//    速算扣除数 : 0
//    应纳税额 = 应纳税所得额 × 税率 - 速算扣除数
        double ys = 0;//应纳税金额
        if(y>0&&y<=1500){

            ys = y*0.03-0.00;
            System.out.println("应纳税金额："+ ys);
        }
        else if(y>1500&&y<=4500){

            ys = y*0.1-105.0;
            System.out.println("应纳税金额："+ ys);
        }
        else if(y>4500&&y<=9000){

            ys = y*0.20-555.0;
            System.out.println("应纳税金额："+ ys);
        }
        else if(y>9000&&y<=35000){

            ys = y*0.25-1005.0;
            System.out.println("应纳税金额："+ ys);
        }
        else if(y>35000&&y<=55000){
```

```
        ys = y*0.30-2755.0;
        System.out.println("应纳税金额:"+ ys);
    }
    else if(y>55000&&y<=8000000){

        ys = y*0.35-5505.0;
        System.out.println("应纳税金额:"+ ys);
    }
    else if(y>8000000){
        ys = y*0.45-13505.0;
        System.out.println("应纳税金额:"+ ys);
    }
    // 税后工资 = 税前工资 - 五险一金 - 应纳个税
    s2 = money-w-ys;
    System.out.println("税后工资"+s2);
}else{
    System.out.println("家里有矿啊!这点钱还交税????");
}
System.out.println();
System.out.println();
return s2;

}
```

## 打印99表

利用for while dowhile循环打印 9\*9 表?

```
public static void main(String[] args) {
    //for 循环行列进行打印
    /*    for(int i=0;i<9;i++){

        for(int j=1;j<9;j++){
            System.out.print(" * ");
        }
        System.out.println(" * ");
    }
    System.out.println();
    */

    //while循环

    /*    int i=0;
    while(i<9){
        i++;
        for(int j=1;j<9;j++){
            System.out.print(" * ");
        }
        System.out.println(" * ");
    }
    */
}
```



```
public static void main(String[] args) {
//    正三角形
    int l=5;
    for(int i=0;i<l;i++){

        for(int j=0;j<l-1-i;j++){
            System.out.print(" ");
        }

        for( int j=0;j<2*i+1;j++){
            System.out.print("*");
        }

        System.out.println();
    }
}
```

```
-----
public static void main(String[] args) {
    //倒三角形
    int line=6;//定义行数
    for(int i=0;i<line;i++){
        for(int j=0;j<i;j++){
            System.out.print("^");
        }
        //打印@的个数
        for( int j=0;j<2*(line-i)-1;j++){
            System.out.print("@");
        }
        //打印完记得换行
        System.out.println();
    }
    /*

    //打印一行
    //打印上半部分
    for(int i=0;i<line;i++){
        //打印i个空格
        for(int j=0;j<line-1-i;j++){
            System.out.print(" ");
        }
        //打印*的个数
        for( int j=0;j<2*i+1;j++){
            System.out.print("@");
        }
        //打印完* 记得换行
        System.out.println();
    }
    */

    //控制台输入
    //    System.out.print("打印一个倒立的正三角形，底边长为（请输入大于或等于三的奇数）：");
}
```

```
//      Scanner in=new Scanner(System.in);
//      int m=in.nextInt();
//      int n=(m-1)/2;
//      for(int i=0;i<=n;i++){
//          for(int j=0;j<=i;j++){
//              System.out.print(" ");
//          }
//          for(int j=0;j<=2*n-2*i;j++){
//              System.out.print("@");
//          }
//          System.out.println();
//      }
}
```

### 数组排序找出最大值及其下标

在一个有8个整数（18, 25, 7, 36, 13, 2, 89, 63）的数组中找出其中最大的数及其下标

```
public static void main(String[] args) {
    int[] a = {18,25,7,36,13,2,89,63};
    //      Arrays类中的sort()使用的是经过调优的快速排序法;
    Arrays.sort(a);
    int i=0;
    for(i=0;i<a.length;i++){
        if(i==(a.length-1)){
            System.out.println("数组下标为: "+i);
        }
    }
    System.out.println(a[a.length-1]);
}
}
```

### 清除数组中重复元素

按照如下方式进行运算。生成一个2行4列的数组。此数组的第1行1列是 $2*1+3*5+4*2$ 第1行2列是 $2*5+3*9+4*7$ 第2行1列是 $4*1+6*5+8*2$  依次类推。（知识点：多维数组定义和创建、数组遍历、数组元素访问）

```
public static void main(String[] args) throws ParseException {
    /* 有2个多维数组分别是 2 3 4   和   1 5 2 8
    4 6 8       5 9 10 -3
                2 7 -5 -18
    */
    //      将一个数组中的重复元素保留一个其他的清零
    String[] a = {"1","2","2","3","3","3","6"};
    TreeSet ts = new TreeSet();
    for(String s:a) {
```

```

        ts.add(s);
    }
    String[] array = (String[])ts.toArray(new String[]{});
    System.out.println(Arrays.toString(array));
}

```

## 数组拷贝

```

public static void main(String[] args) {
//将一个字符数组的值 (neusofteducation) 拷贝到另一个字符数组中
    String[] a = {"neusofteducation"};

    String[] b = new String[1];

    System.arraycopy(a, 0, b, 0, a.length);

    for(String n:b){
        System.out.print(n);
    }

}

```

## 数组查重拷贝

```

public static void main(String[] args) {
    // TODO Auto-generated method stub
    //1.冒泡排序
    //2.输入不超过5位数 实现逆序 输出
    //3. 1/2+2/3+3/4+.....99/100=?求结果

    //4. int[] a={12,34,54,33,31,67,33,98,31,87};
    // 定义数组b 将去重后的a数组 拷贝到b数组中
    //a数组是正整数集合
    int[] a={12,34,54,33,31,67,33,98,31,87};
    int[] c=new int[a.length]; //{0,0,0.....}
    // {12,34,54,33,31,67,0}
    //i=0 12
    //i=1 12,34
    //i=2 12,34,54
    //i=3 12,34,54,33
    //....
    //i=5 12,34,54,33,31,67
    //i=6 12,34,54,33,31,67,33==>0
    //i=6 12 ~33 34 ~33 54~33 [33~33]不用比较 break;

    for(int i=0;i<a.length;i++){
        boolean flag=false; //如果出现重复 true , 不重复false
    }
}

```



```
        for(int j=0;j<i;j++){ //控制比较次数
            if(a[i]==a[j]){ //出现了重复
                flag=true;
                break;
            }else{
                flag=false;
            }
        }

        if(flag==true){
            c[i]=0;
        }else{
            c[i]=a[i]; //赋值
        }
    }

    for(int n:c){
        System.out.print(n+",");
    }

    //b数组有多长
    int blen=0;
    for(int i=0;i<c.length;i++){
        if(c[i]!=0){
            blen++;
        }
    }
    System.out.println();
    System.out.println(blen);

    //定义b数组
    int[] b=new int[blen];
    int j=0;
    //将c数组 拷贝到b数组当中 不包括0;
    for(int i=0;i<c.length;i++){

        if(c[i]==0){

        }else{
            b[j]=c[i];
            j++;
        }
    }
    // 遍历b数组
    for(int n:b){
        System.out.print(n+",");
    }

}
```

## 计算器

```

public static void main(String[] args) {

    double a=0,b=0,c=0;
    double r=1;
    Scanner sa = new Scanner(System.in);

    System.out.println("请输入一个数: ");
    a=sa.nextDouble();

    System.out.println("请输入一个运算符: ");
    String s = sa.next();

    System.out.println("请输入一个数: ");
    b=sa.nextDouble();

    for(int i=0;i<=s.length()-1;i++) {
        if(s.equals("+")) {
            System.out.println("运算+得: "+(c=a+b));
        }else if(s.equals("-")){
            System.out.println("运算-得: "+(c=a-b));
        }else if(s.equals("*")){
            System.out.println("运算*得: "+(c=a*b));
        }else if(s.equals("/")){
            if(b==0){
                System.out.println("除数不能为0! ");
            }else{
                System.out.println("运算/得: "+(c=a/b));
            }
        }
        else if(s.equals("^")){
            for(int j=0;j<b;j++) {
                r = r *a;//当j等于1 进行r= r*a 由于j<b 所以再一次进行运算
                r*a(r为第一次运算的)
            }
            System.out.println("运算得: "+r);
        }
    }
}

```

## 对象与方法练习题

### 使用封装方法对三角形进行判断

```

//对三角形类进行封装
public class SJX {
    private int a;

```

```
private int b;
private int c;
public int getA() {
    return a;
}
public void setA(int a) {
    this.a = a;
}

public int getB() {
    return b;
}
public void setB(int b) {
    this.b = b;
}

public int getC() {
    return c;
}
public void setC(int c) {
    this.c = c;
}
}
//调用封装方法 进行计算、比较
public class SJXTest {
    public static void main(String[] args) {

        SJX s = new SJX();
        Scanner sc = new Scanner(System.in);
        System.out.println("请输入三角形的A边长度为: ");
        s.setA(sc.nextInt());

        if(s.getA()<=0){
            System.out.println("输入错误! 请重新输入: ");
            s.setA(sc.nextInt());
        }else{

        }

        System.out.println("请输入三角形的B边长度为: ");
        s.setB(sc.nextInt());

        if(s.getB()<=0){
            System.out.println("输入错误! 请重新输入: ");
            s.setB(sc.nextInt());
        }else{

        }

        System.out.println("请输入三角形的C边长度为: ");
        s.setC(sc.nextInt());

        if(s.getC()<=0){
            System.out.println("输入错误! 请重新输入: ");
```

```

        s.setC(sc.nextInt());
    }else{

    }

    if(((s.getA() + s.getB()) > s.getC()) && ((s.getA() + s.getC()) >
s.getB())&&((s.getB() + s.getC()) > s.getA()))
    {
        if((s.getA()==s.getB()) || (s.getB()==s.getC()) ||
(s.getA()==s.getC())){
            if((s.getA()==s.getB()) && (s.getB()==s.getC()) &&
(s.getA()==s.getC())){
                System.out.println("这是一个等边三角形");
                System.out.println("此三角形的A边长度为: "+s.getA());
                System.out.println("此三角形的A边长度为: "+s.getB());
                System.out.println("此三角形的A边长度为: "+s.getC());
            }else{
                System.out.println("这是一个等腰三角形");
                System.out.println("此三角形的A边长度为: "+s.getA());
                System.out.println("此三角形的A边长度为: "+s.getB());
                System.out.println("此三角形的A边长度为: "+s.getC());
            }
        }
    }else{
        System.out.println("这是一个不等边三角形");
        System.out.println("此三角形的A边长度为: "+s.getA());
        System.out.println("此三角形的A边长度为: "+s.getB());
        System.out.println("此三角形的A边长度为: "+s.getC());
    }
}
else{
    System.out.println("还输入三角形边长。家里有旷啊! ");
}
}
}

```

## 文件管理类的使用方法

### 构造函数

```

public class FileDemo {
    public static void main(String[] args){
        //构造函数File(String pathname)
        File f1 =new File("c:\\abc\\1.txt");
        //File(String parent,String child)
        File f2 =new File("c:\\abc","2.txt");
        //File(File parent,String child)
        File f3 =new File("c:"+File.separator+"abc");//separator 跨平台分隔
符
        File f4 =new File(f3,"3.txt");
        System.out.println(f1);//c:\abc\1.txt
    }
}

```

```
    }  
}
```

## 创建方法

- 1.boolean createNewFile() 不存在返回true 存在返回false
- 2.boolean mkdir() 创建目录
- 3.boolean mkdirs() 创建多级目录

## 删除方法

- 1.boolean delete()
- 2.boolean deleteOnExit() 文件使用完成后删除

```
import java.io.File;  
import java.io.IOException;  
  
public class FileDemo2 {  
    public static void main(String[] args){  
        File f =new File("d:\\1.txt");  
        try {  
            System.out.println(f.createNewFile());//当文件存在时返回false  
            System.out.println(f.delete());//当文件不存在时返回false  
        } catch (IOException e) {  
            // TODO Auto-generated catch block  
            e.printStackTrace();  
        }  
    }  
}
```

## 判断方法

- 1.boolean canExecute()判断文件是否可执行
- 2.boolean canRead()判断文件是否可读
- 3.boolean canWrite() 判断文件是否可写
- 4.boolean exists() 判断文件是否存在
- 5.boolean isDirectory()
- 6.boolean isFile()
- 7.boolean isHidden()
- 8.boolean isAbsolute()判断是否是绝对路径 文件不存在也能判断

## 获取方法

- 1.String getName()
- 2.String getPath()

- 3.String getAbsolutePath()
- 4.String getParent()//如果没有父目录返回null
- 5.long lastModified()//获取最后一次修改的时间
- 6.long length()
- 7.boolean renameTo(File f)
- 8.File[] listRoots()//获取机器盘符
- 9.String[] list()
- 10.String[] list(FilenameFilter filter)

### 列出磁盘下的文件和文件夹

```
public class FileDemo3 {
    public static void main(String[] args){
        File[] files =File.listRoots();
        for(File file:files){
            System.out.println(file);
            if(file.length()>0){
                String[] filenames =file.list();
                for(String filename:filenames){
                    System.out.println(filename);
                }
            }
        }
    }
}
```

### 文件过滤

```
import java.io.File;
import java.io.FilenameFilter;
public class FileDemo4 {
    public static void main(String[] args){
        File[] files =File.listRoots();
        for(File file:files){
            System.out.println(file);
            if(file.length()>0){
                String[] filenames =file.list(new FilenameFilter(){
                    //file 过滤目录 name 文件名
                    public boolean accept(File file,String filename){
                        return filename.endsWith(".mp3");
                    }
                });
                for(String filename:filenames){
                    System.out.println(filename);
                }
            }
        }
    }
}
```

```
    }  
  
    }  
  
    File[] listFiles()  
    File[] listFiles(FilenameFilter filter)
```

## 利用递归列出全部文件

代码如下:

```
public class FileDemo5 {  
    public static void main(String[] args){  
        File f =new File("e:\\音樂");  
        showDir(f);  
    }  
    public static void showDir(File dir){  
        System.out.println(dir);  
        File[] files =dir.listFiles();  
        for(File file:files){  
            if(file.isDirectory())  
                showDir(file);  
            else  
                System.out.println(file);  
        }  
    }  
}
```

## 移动文件

找出d盘下所有的 .java 文件, 拷贝至 c:\jad 目录下, 并将所有文件的类型由.java 修改为.jad 。

```
public class Test5 {  
    public static void main(String[] args){  
        File f1 = new File("d:\\");  
        moveFile(f1);  
    }public static void moveFile(File dir){  
        File[] files=dir.listFiles();  
        for(File file:files){  
            if(file.isDirectory())  
                moveFile(file);  
            else{  
                if(file.getName().endsWith(".java"))  
                    file.renameTo(new File("c:\\jad\\"+  
  
file.getName().substring(0,file.getName().lastIndexOf('.')+".jad"));  
            }  
        }  
    }  
}
```

```

    }
}

```

## 查找文件夹及文件名

在本机的磁盘系统中，找一个文件夹，利用 File类的提供方法，列出该文件夹中的所有文件的 文件名和文件的路径，执行效果如下：[必做题] • 路径是xxx的文件夹内的文件有： 文件名：

```

abc.txt  路径名: c:\temp\abc.txt  -----
---  文件名: def.txt  路径名: c:\temp\def.txt

```

\* 第一个版本

```

public class getFile {
    public static void main(String[] args) {
        getFileListame("C:\\Users\\shuai\\Desktop\\作业");
    }
    public static void getFileListame(String strPath) {
        File dir = new File(strPath);
        File[] files = dir.listFiles(); // 该文件目录下文件全部放入数组
        if (files != null) {
            for (int i = 0; i < files.length; i++) {
                System.out.println(files[i].getName());
                if (files[i].isDirectory()) { // 判断是文件还是文件夹
                    getFileListame(files[i].getAbsolutePath()); // 获取文件
                    System.out.println(files[i].getAbsolutePath() +
files[i].getName());
                }
            }
        }
    }
}

```

\* 第二个 完善版

```

public class Test22 {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("请输入要打印出的文件夹：(windows下分隔符请使用“\\”)");
        String ML = sc.next();
        File file = new File(ML);
        getFile(file);
    }
    public static void getFile(File file) {
        //listFiles() 返回一个抽象路径名数组这些路径名表示此抽象路径名表示的目录中的
        // 该文件目录文件放入数组
        File[] files = file.listFiles();
        if(file.exists()){
            if (files != null) {
                for (int i = 0; i < files.length; i++) {

```



```
        System.out.println("文件名= " + files[i].getName());
        //获取文件名
        /*File tempFile =new File(
files[i].getAbsolutePath().trim());
        String fileName = tempFile.getName();
        System.out.println("文件名= " + fileName);*/

        // 获取文件绝对路径
        System.out.println("文件的绝对路径= " +
files[i].getAbsolutePath());
        System.out.println("-----");
    }
}
}else{
    System.out.println("输入有误! ");
}
}
}
```