**数组**

## 一.数组概述

**1.  定义:**数组是一种数据结构，用来存储同一类型值的集合。

**2.  数组是存在下标索引的，**通过下标可以获取指定位置的元素，数组小标是从0开始的也就是说下标0对应的就是数组中第1个元素，可以很方便的对数组中的元素进行存取操作。

**3.  数组存在一维数组和多维数组。**

**4. 数组好处**

可以自动给数组中的元素从0开始编号，方便操作这些元素。存数据和不存数据有什么区别吗？数组的最大好处就是能都给存储进来的元素自动进行编号. 注意编号是从0开始。方便操作这些数据。例如 学生的编号，使用学号就可以找到对应的学生。

# **5 数组和集合的区别**

（1）数组是大小固定的，并且同一个数组只能存放类型一样的数据（基本类型/引用类型）

（2）JAVA集合可以存储和操作数目不固定的一组数据。 (3)若程序时不知道究竟需要多少对象，需要在空间不足时自动扩增容量，则需要使用容器类库，array不适用。

（3）联系：使用相应的toArray()和Arrays.asList()方法可以回想转换。

**6 方法传参数组，本身初始化或者参与计算不会修改值，如果修改属性值会生效**

## 二.一维数组

**1.     声明方式一**

A.     数据类型数组名= new 数据类型[个数]

String s[] = new String[10];  
定义了一个长度为10的String类型的一维数组。

数据类型[]数组名= new 数据类型[个数]  
String[] s = new String[10];  
定义了一个长度为10的String类型的一维数组。

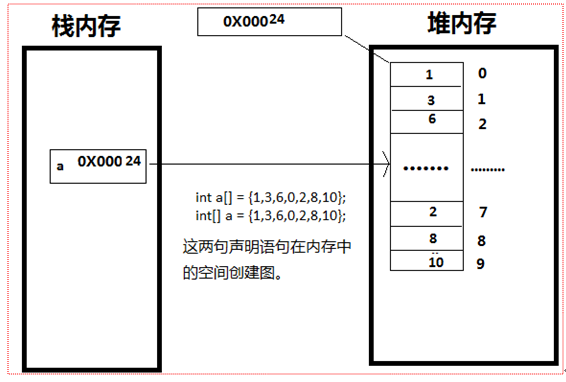
**2.     声明格式二**

A.     数据类型数组名[] ={元素1,元素2,元素3,……数值n};

int a[] = {1,3,6,0,2,8,10};  
声明一个为数组长度为7的int类型数组，并静态的为数组每一个小标对应的元素赋值。

B.     数据类型[]数组名={元素1,元素2,元素3,……数值n};  
int[] a = {1,3,6,0,2,8,10};

声明一个为数组长度为7的int类型数组，并静态的为数组每一个小标对应的元素赋值。

[](http://blog.photo.sina.com.cn/showpic.html#url=http://album.sina.com.cn/pic/a042f0dbgx6CEka8kj3cd)

1. **创建一个匿名数组**  
   new int[]{1,3,6,0,2,8,10};

**4.  数组与数组之间的赋值关系**

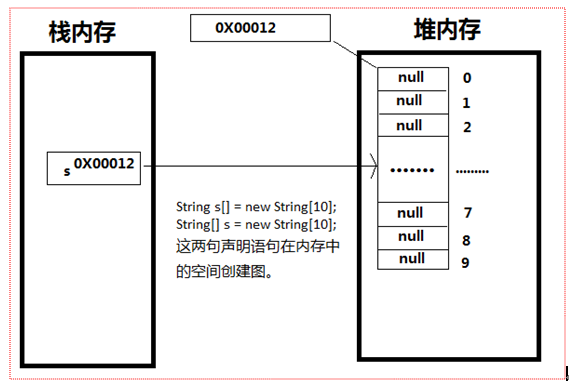
String s[] = new String[10];

String t[] = s;

**5 一维数组长度:数组名.length**

**6.  一维数组的有效最大角标:数组名.length-1**

**7 数组的内存分析**

[](http://blog.photo.sina.com.cn/showpic.html#url=http://album.sina.com.cn/pic/a042f0dbgx6CEk8A7nyb0)

数组属于引用数据类型，它指向的是内存中的一个对象的地址。只要使用new对象产生一个对象或数组都会在堆内存中创建一块内存空间，然后将这块空间的内存地址赋给引用变量，以上只是声明创建一个大小为10的数组，但是没有给每一个元素赋值，所以体现出来的都是这个数据类型的默认值，即String类型的默认值为null。

**8 Arrays类的使用**

java.util.Arrays 类能方便地操作数组，它提供的所有方法都是静态的。

具有以下功能：

给数组赋值：通过 fill 方法。

对数组排序：通过 sort 方法,按升序。

比较数组：通过 equals 方法比较数组中元素值是否相等。

查找数组元素：通过 binarySearch 方法能对排序好的数组进行二分查找法操作。

…….

## 三.二维数组

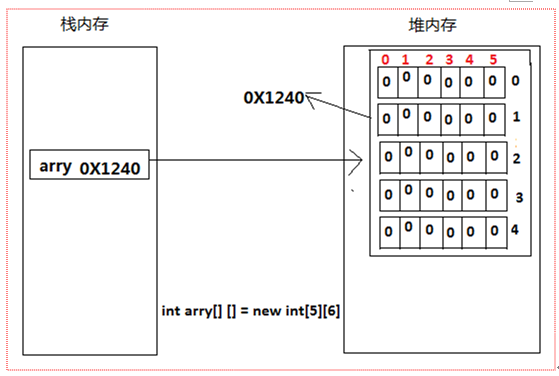
**1.     定义:**二维数组实际上是指数组元素为一维数组的一维数组，简单说也就是数组中的数组。

**2.     声明格式一**  
数据类型数组名[][] = new数据类型[个数][个数];

数据类型[][]数组名= new数据类型[个数][个数];

-->int arry[] [] = new int[5][6];

上述语句声明了一个二维数组，其中[5]表示该数组有(0~4)5行，每行有(0~5)6个元素，因此该数组有30个元素，其中[5]也可说是有5个一维数组，[6]可以说是每一个一维数组中有6个int型号的元素。

[](http://blog.photo.sina.com.cn/showpic.html#url=http://album.sina.com.cn/pic/a042f0dbgx6CEkh4sxcea)

数组new出来后，就在堆内存中开辟的内存空间，并将所在内存空间地址返回给arry数组变量，因为以上语句只是声明了一个数组并未没一个元素赋值，所以都是类型默认值，为0。

-->arry[0][0]=3;即给第一个一维数组中的第一个元素赋值。依此方式还可以和其他元素赋值。

**3.     声明格式二**

int[][] arr = new int[3][];

此二维数组中含有3个一维数组，每个一维数组的默认初始化值都为null，每个一维数组中的元素值都为0；可对每个一维数组赋值，如：  
 arr[0] = new int[3];

  arr[1] = new int[2];

  arr[2] = new int[1];

对每个一维数组中某个元素赋值：如:

arr[0][2]=90;

arr[1][1]=89;

arr[2][2]=78;

**4.     声明格式三**

显式初始化二维数组：int[][] arr = {{1,5,7},{4,8},{3,9,20,12}}

A.     以上数组定义了一个长度为3的数组，arr.length=3

B.     数组中第一维数组的长度:arr[0].length=3

四.  数组的应用代码  
1. 获取一个数组中的最大值和最小值

public class ArrayMaxMinValue

{

public static void main(String[] args)

{

int[] array = {7,9,6,3,9,8,0,10,2,1};

System.out.println("最大值:　"+getMaxValue(array));

System.out.println("最小值:　"+getMinValue(array));

}

public static int getMaxValue(int a[])

{

int max = a[0];//一开始假设最大值为a[0]

for(int i = 1 ; i < a.length ;i++)

{

if(a[i] > max)

{

max = a[i];//每一次比较将大值赋值给max变量存储

}

}

return max;

}

public static int getMinValue(int a[])

{

int min = 0;//一开始假设最小值为角标为0的元素

for(int i = 1 ; i < a.length ;i++)

{

if(a[i] < a[min])

{

min = i;//每一次比较将最小值的角标赋值给min变量存储

}

}

return a[min];

}

}

2.对数组进行排序

public class ArraySortTool

{

public static void main(String[] args)

{

int[] array = {7,9,6,3,9,8,0,10,2,1};

System.out.println("排序前:");

printArray(array);

bubblSort(array);

System.out.println("排序后:");

printArray(array);

}

//选择排序

public static void SelectSort(int[] a)

{

//外层循环写成i < a.length-1，是因为选择排序，

//当到最后一个元素的时候，就不需要进行选择排序的，因为数序已经确定了

for(int i = 0 ; i < a.length-1 ; i++)

{

//内层循环写成i < a.length，是因为每一个选择用来比较的元素都要比较到最后//一个元素，

//int j = i+1是因为没一个元素都不需要和自身比较

for(int j = i+1 ; j < a.length ; j++)

{

if(a[i] > a[j])

{

//交换两个值

int temp = a[j];

a[j] = a[i];

a[i] = temp;

}

}

}

}

//冒泡排序

public static void bubblSort(int a[])

{

//a.length-1冒泡排序是每一次相邻两个元素进行比较，总共要比较a.length-1次

//int i=0 是因为比较是从角标0开始的

for(int i = 0 ; i < a.length-1; i++)

{

//a.length-1-i，第一轮全部计较，每轮比较都产生一个最值，

//下一轮这个最值就不用参与比较了

//int j = i+1因为是相邻比较，所以要改成int j = i+1

for(int j = 0 ; j < a.length-1-i ; j++)

{

//因为是两两比较所以要写成a[j] > a[j+1]

if(a[j] > a[j+1])

{

int temp = a[j+1];

a[j+1] = a[j];

a[j] = temp;

}

}

}

}

//用来打印数组

public static void printArray(int a[] )

{

System.out.print("[");

for(int i = 0 ; i < a.length ; i++)

{

if(i == a.length-1)

{

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

}

else

{

System.out.print(a[i]+",");

}

}

}

}

3.查找数组中指定值的索引位置

public class ArraySearchTool

{

public static void main(String[] args)

{

int[] array = {1,5,9,11,20,21,25,30};

System.out.println(halfSearch2(array , 25));

// System.out.println(oneByoneSearch(array , 10));

}

//遍历判断查找

public static int oneByoneSearch(int[] a , int value)

{

for(int i = 0 ; i < a.length ; i++)

{

if(a[i] == value)

{

return i;

}

}

return -1;

}

//二分查找1

public static int halfSearch(int[] a , int value)

{

int left = 0;

int right = a.length-1;

int mid = (left+right) >> 1;

while( a[mid] != value)

{

if(a[mid] > value)

{

right = mid-1;

}

else

{

left = mid+1;

}

if(left > right)

{

return -1;

}

mid = (left+right)>>1;

}

return mid;

}

//二分查找2

public static int halfSearch2(int[] a, int value)

{

int left = 0;

int right = a.length-1;

int mid;

while( left <= right)

{

mid = (left+right) >> 1;

if(a[mid] == value)

{

return mid;

}

else if(a[mid] > value)

{

right = mid -1;

}

else

{

left = mid+1;

}

}

return -1;

}

}

5.     编写一个类，类中提供方法可以将任何int类型的数值做八进制、二进制、十六进制转换

public class toBintoHextoOctalDemo

{

public static void main(String[] args)

{

System.out.println(toBintoHextoOcta( 500 , 15 , 4 ));

}

public static String toBintoHextoOcta( int n , int a , int m )

{

char[] c = { '0','1','2','3','4','5','6','7','8','9','A','B','C','D','E','F' };

char[] ch = new char[32];

StringBuilder sb = new StringBuilder();

int pos = ch.length;

while( n != 0)

{

ch[--pos] = c[ n & a ];

n = n >>> m ;

}

for(int i = pos ; i < ch.length ; i++)

{

sb.append(ch[i]);

}

return sb.toString();

}

}

**6.     扩展应用**

int[] x ,y[];  àx一维数组，y二维数组。

拆解步骤:

int[] x;

int[] y[];

A.     x[0] = y;错误

B.     y[0] = x;正确

C.     y[0][0]=X;错误

D.    x[0][0] =y;错误

E.     y[0][0] =x[0]正确