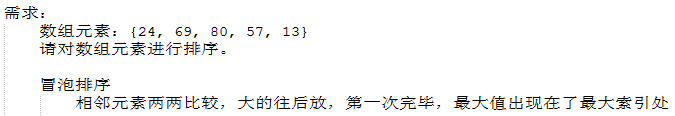
菜鸟教程：http://www.runoob.com/java/java-array.html

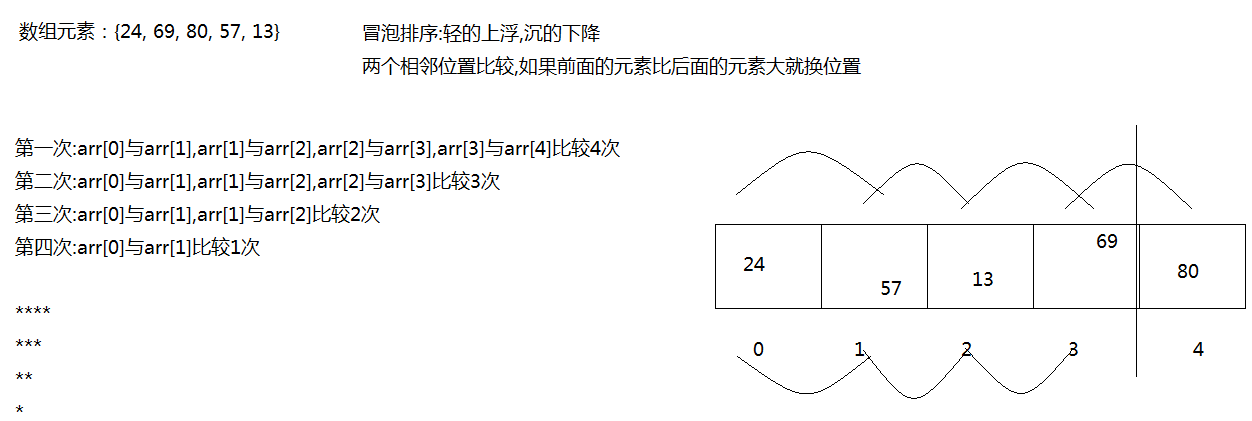
# 排序

## 需求



## 冒泡排序

### 图解



### 代码实现

**public** **class** Demo1\_Array {

/\*\*

\* 数组高级冒泡排序代码

\*/

**public** **static** **void** main(String[] args) {

**int**[] arr = {24, 69, 80, 57, 13};

*bubbleSort*(arr);

*print*(arr);

}

/\*

\* 冒泡排序

\* 1,返回值类型,void

\* 2,参数列表,int[] arr

\* 第一次:arr[0]与arr[1],arr[1]与arr[2],arr[2]与arr[3],arr[3]与arr[4]比较4次

第二次:arr[0]与arr[1],arr[1]与arr[2],arr[2]与arr[3]比较3次

第三次:arr[0]与arr[1],arr[1]与arr[2]比较2次

第四次:arr[0]与arr[1]比较1次

\*/

**public** **static** **void** bubbleSort(**int**[] arr) {

**for** (**int** i = 0; i < arr.length - 1; i++) { //外循环只需要比较arr.length-1次就可以了

**for** (**int** j = 0; j < arr.length - 1 - i; j++) { //-1为了防止索引越界,-i为了提高效率

**if**(arr[j] > arr[j+1]) {

int temp = arr[j];

arr[j] = arr[j + 1];

arr[j+1] = temp;

}

}

}

}

/\*

\* 打印数组

\* 1,返回值类型void

\* 2,参数列表int[]arr

\*/

**public** **static** **void** print(**int**[] arr) {

**for** (**int** i = 0; i < arr.length; i++) {

System.*out*.print(arr[i] + " ");

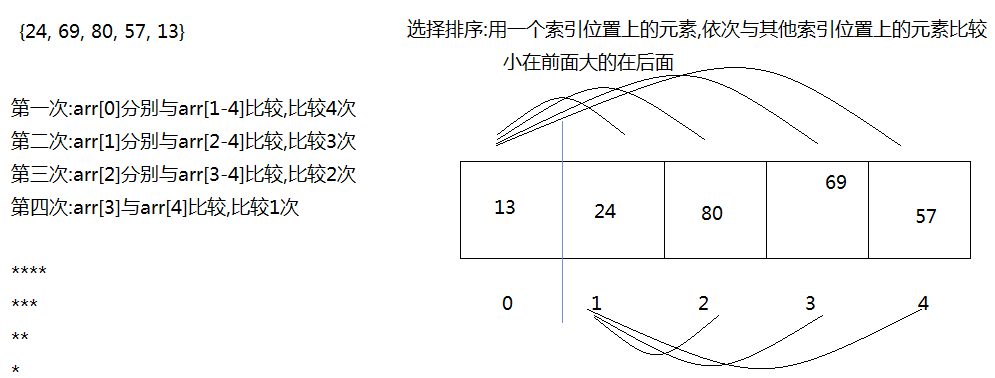
}

}

}

## 选择排序

### 图解



### 代码实现

**public** **class** Demo1\_Array {

**public** **static** **void** main(String[] args) {

**int**[] arr = {24, 69, 80, 57, 13};

*selectSort*(arr);

*print*(arr);

}

/\*

\* 打印数组

\* 1,返回值类型void

\* 2,参数列表int[]arr

\*/

**public** **static** **void** print(**int**[] arr) {

**for** (**int** i = 0; i < arr.length; i++) {

System.*out*.print(arr[i] + " ");

}

}

/\*

\* 选择排序

\* 1,返回值类型void

\* 2,参数列表int[] arr

\*

\* 第一次:arr[0]分别与arr[1-4]比较,比较4次

第二次:arr[1]分别与arr[2-4]比较,比较3次

第三次:arr[2]分别与arr[3-4]比较,比较2次

第四次:arr[3]与arr[4]比较,比较1次

\*/

**public** **static** **void** selectSort(**int**[] arr) {

**for** (**int** i = 0; i < arr.length - 1; i++) { //只需要比较arr.length-1次

**for** (**int** j = i + 1; j < arr.length; j++) {

**if**(arr[i] > arr[j]) {

int temp = arr[i];

arr[i] = arr[j];

arr[j] = temp;

}

}

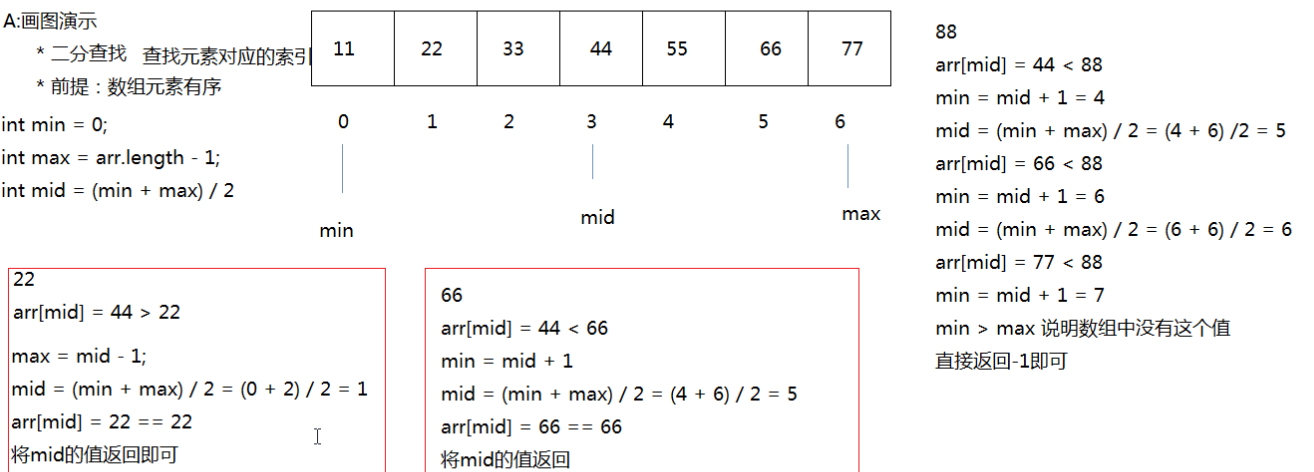
}

}

# 二分查找



## 图解



## 代码实现

**public** **class** Demo2\_Array {

/\*\*

\* \* A:案例演示

\* 数组高级二分查找代码

\* B:注意事项

\* 如果数组无序，就不能使用二分查找。

\* 因为如果你排序了，但是你排序的时候已经改变了我最原始的元素索引。

\*/

**public** **static** **void** main(String[] args) {

**int**[] arr = {11,22,33,44,55,66,77};

//返回索引

System.*out*.println(*getIndex*(arr, 22));

}

/\*

\* 二分查找

\* 1,返回值类型,int

\* 2,参数列表int[] arr,int value

\*/

**public** **static** **int** getIndex(**int**[] arr, **int** value) {

**int** min = 0;

**int** max = arr.length - 1;

**int** mid = (min + max) / 2;

**while**(arr[mid] != value) { //当中间值不等于要找的值,就开始循环查找

**if**(arr[mid] < value) { //当中间值小于了要找的值

min = mid + 1; //最小的索引改变

}**else** **if** (arr[mid] > value){ //当中间值大于了要找的值

max = mid - 1; //最大的索引改变

}

mid = (min + max) / 2; //无论最大还是最小改变,中间索引都会随之改变

**if**(min > max) { //如果最小索引大于了最大索引,就没有查找的可能性了

**return** -1; //返回-1

}

}

**return** mid;

}

}

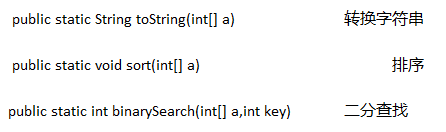
# Arrays类

## Arrays类概述

针对数组进行操作的工具类。

提供了排序，查找等功能。

## 成员方法





Arrays.*sort*(arr);

System.*out*.println(Arrays.*toString*(arr));

## 代码实现

**public** **class** Demo3\_Arrays {

/\*\*

\* public static String toString(int[] a)

\* public static void sort(int[] a)

\* public static int binarySearch(int[] a,int key)

\*

\* public static String toString(int[] a) {

if (a == null) //如果传入的数组是null

return "null"; //返回null

int iMax = a.length - 1; //iMax最大索引

if (iMax == -1) //如果数组中没有元素

return "[]"; //返回[]

StringBuilder b = new StringBuilder(); //线程不安全,效率高

b.append('['); //将[添加到字符串缓冲区中

for (int i = 0; ; i++) { //遍历数组,判断语句没有写默认是true

b.append(a[i]); //把第一个元素添加进字符串缓冲区

if (i == iMax) //如果索引等于了最大索引值

return b.append(']').toString(); //将]添加到字符串缓冲区,在转换成字符串并返回

b.append(", "); //如果不等于最大索引就将, 添加到缓冲区

}

}

private static int binarySearch0(int[] a, int fromIndex, int toIndex,

int key) {

int low = fromIndex; //最小索引0

int high = toIndex - 1; //最大索引数组长度-1

while (low <= high) { //最小索引小于等于最大索引可以循环判断

int mid = (low + high) >>> 1; //求出中间索引值,(最小+最大)/2

int midVal = a[mid]; //通过中间索引获取中间值

if (midVal < key) //中间索引对应的值小于查找的值

low = mid + 1; //最小索引变化

else if (midVal > key) //中间索引对应的值大于查找的值

high = mid - 1; //最大索引变化

else

return mid; // key found //找到了

}

return -(low + 1); // key not found.//-插入点 - 1

}

\*/

**public** **static** **void** main(String[] args) {

**int**[] arr = {33,22,11,44,66,55};

System.*out*.println(Arrays.*toString*(arr)); //数组转字符串

Arrays.*sort*(arr); //排序

System.*out*.println(Arrays.*toString*(arr));

**int**[] arr2 = {11,22,33,44,55,66};

System.*out*.println(Arrays.*binarySearch*(arr2, 22)); //-插入点-1

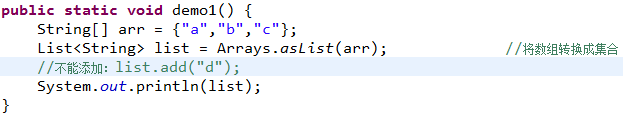
}

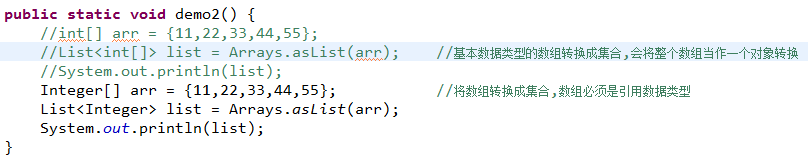
}

## 常用方法

### 数组转集合—aslist()

数组转换成集合虽然不能增加或减少元素,但是可以用集合的思想操作数组,也就是说可以使用其他集合中的方法





### 集合转数组—toArray()

