







package yangs.demo;  
public class BinarySearchDemo {  
 public static void main(String[] args) {  
 int[] arr = {1, 2, 3, 5, 7, 11, 13};  
 *//创建对象* BinarySearchDemo bsd = new BinarySearchDemo();  
 *//执行方法* bsd.BinarySearch(arr, 8);  
 bsd.BinarySearch(arr, 6);  
 bsd.BinarySearch(arr, 0);  
 bsd.BinarySearch(arr, 1);  
  
 }  
  
 void BinarySearch(int[] arr, int target) {  
 int left = 0;  
 int right = arr.length - 1;  
 int middle = 0;  
 while (left <= right) {  
 middle = (left + right) / 2;  
 if (target == arr[middle]) {  
*// return middle;//如果找到target，则返回target位置* System.*out*.println("小于"+ target +"的最大元素位置 i 是 " + middle);  
 System.*out*.println("大于"+ target +"的最小元素位置 j 是 " + middle);  
 System.*out*.println();  
 }  
 if (target > arr[middle]) {  
 left = middle + 1;*//当 target > arr[middle] ，在右半段查找* } else {  
 right = middle - 1;*//当 target < arr[middle] ，在左半段查找* }  
 }  
 if (left == 0) {*//当target小于数组中全部元素是，小于target的元素不存在* System.*out*.println("小于"+ target +"的最大元素位置 i 不存在");  
 System.*out*.println("大于"+ target +"的最小元素位置 j 是 " + left);  
 System.*out*.println();  
 }  
 if (right == arr.length) {*//当target大于数组中的全部元素时，大于target的元素不存在* System.*out*.println("小于"+ target +"的最大元素位置 i 是 " + right);  
 System.*out*.println("大于"+ target +"的最小元素位置 j 是 不存在");  
 System.*out*.println();  
 } else {*//当target的值大于数组的最大元素且小于数组的最小元素时，返回 i j* System.*out*.println("小于"+ target +"的最大元素位置 i 是 " + (left-1));  
 System.*out*.println("大于"+ target +"的最小元素位置 j 是 " + left);  
 System.*out*.println();  
 }  
  
 }  
}