统计一个数字在有序数组中出现的次数。

//找出特殊的，比如空的，比如只有1个元素的。

//注意min <= max，上下限。

**package** facetest;  
  
**public class** BinaryChop {  
 **public static void** main(String[] args) {  
*// int[] arrays = {1, 2, 3, 3, 3, 4, 5, 6, 7};* **int**[] arrays = {};  
 **int** key = 3;  
 **int** c = *BinaryChop*(arrays, key);  
 System.***out***.println(c);  
 }  
  
 **public static int** BinaryChop(**int**[] arrays, **int** key) {  
  
 **int** length = arrays.**length**;  
 **int** min = 0;  
 **int** max = length-1;  
 **if** (length == 0) {  
 **return** 0;  
 }  
 **if** (key > arrays[max] || key < arrays[0]) {  
 **return** 0;  
 }  
 **if** (key == arrays[0] && length == 1) {  
 **return** 1;  
 }  
 **int** first = *GetFirst*(arrays,key,min,max);  
 **int** last = *GetLast*(arrays,key,min,max);  
  
 **if** (first == 0 && last == 0) {  
 **return** 0;  
 }  
 **int** count = last - first + 1;  
 **return** count;  
 }  
  
 **public static int** GetFirst(**int**[] arrays, **int** key, **int** min, **int** max) {  
 **while** (min <= max) {  
 **int** mid = (min + max) / 2;  
 **if** (arrays[mid] == key) {  
 **if** (mid - 1 < 0) {  
 **return** mid;  
 } **else if** (arrays[mid - 1] == key) {  
 max = mid-1;  
 } **else** {  
 **return** mid;  
 }  
 } **else if** (arrays[mid] > key) {  
 max = mid-1;  
 } **else** {  
 min = mid+1;  
 }  
 }  
 **return** 0;  
 }  
  
 **public static int** GetLast(**int**[] arrays, **int** key, **int** min, **int** max) {  
 **while** (min <= max) {  
 **int** mid;  
 **if** (min == max) {  
 mid = max;  
 } **else** {  
 mid = (min + max) / 2 + 1;  
 }  
 **if** (arrays[mid] == key) {  
 **if** (mid + 1 > arrays.**length** - 1) {  
 **return** mid;  
 } **else if** (arrays[mid + 1] == key) {  
 min = mid + 1;  
 } **else** {  
 **return** mid;  
 }  
 } **else if** (arrays[mid] > key) {  
 max = mid-1;  
 } **else** {  
 min = mid+1;  
 }  
 }  
 **return** 0;  
 }  
}