题目一 数字在排序数组中出现的次数

面试题 53：在排序数组中查找数字

题目一：数字在排序数组中出现的次数。

统计一个数字在排序数组中出现的次数。例如，输入排序数组{1, 2, 3, 3, 3, 3, 4, 5}和数字 3，由于 3 在这个数组中出现了 4 次，因此输出 4。

解法一 O(n)

二分查找，先找到这个数字，然后从这个数字开始往前找到第一个，往后找到最后一个

```
int GetNumberOfK(vector<int> data ,int k) {
    if(data.size()==0)
        return 0;
    int left=0,right=data.size()-1;
    while(left<=right)
    {
        int mid=(left+right)>>1;
        if(k<data[mid])
            right=mid-1;
        if(k>data[mid])
            left=mid+1;
        if(k==data[mid])
        {
            return numofk(data,mid,k);
        }
    }
    return 0;
}
int numofk(vector<int> tmp,int index,int number)
{
    int count=1;
    int i=index,j=index;
    while(tmp[++i]==number)
        count++;
    while(tmp[--j]==number)
        count++;

    return count;
}
```

解法二 直接用二分查找法找第一个数字和最后一个数字 O(logn)

```
class Solution {
public:
    int GetNumberOfK(vector<int> data ,int k) {
        if(data.empty())
            return 0;
        int n=data.size();
        int i=getFirstk(data,k,0,n-1);
```

```
        int j=getLastk(data,k,0,n-1);
        if(i==-1 && j==-1)
            return 0;
        return j-i+1;
    }
    int getFirstk(vector<int> data,int k,int start,int end)
    {
        if(start>end)
            return -1;
        int mid=(start+end)>>1;
        if(data[mid]==k)
        {
            if((mid>0 && data[mid-1]!=k) || mid==0)
                return mid;
            else
                end=mid-1;
        }
        else if(data[mid]>k)
        {
            end=mid-1;
        }
        else
            start=mid+1;
        return getFirstk(data,k,start,end);
    }
    int getLastk(vector<int> data,int k,int start,int end)
    {
        if(start>end)
            return -1;
        int n=data.size();
        int mid=(start+end)>>1;
        if(data[mid]==k)
        {
            if((mid<n-1 && data[mid+1]!=k) || mid==n-1)
                return mid;
            else
                start=mid+1;
        }
        else if(data[mid]>k)
            end=mid-1;
        else
            start=mid+1;

        return getLastk(data,k,start,end);
    }
};
```

```
int GetNumberOfK(int* data, int length, int k)
{
    int number = 0;

    if(data != nullptr && length > 0)
    {
        int first = GetFirstK(data, length, k, 0, length - 1);
        int last = GetLastK(data, length, k, 0, length - 1);

        if(first > -1 && last > -1)
            number = last - first + 1;
    }

    return number;
}
```

题目二 0~n-1中缺失的数字

题目二：0～n–1 中缺失的数字。

一个长度为 n–1 的递增排序数组中的所有数字都是唯一的，并且每个数字都在范围 0～n–1 之内。在范围 0～n–1 内的 n 个数字中有且只有一个数字不在该数组中，请找出这个数字。

解

递增排序数组，用二分查找，缺失的数字是第一个 值和下标不相等时 的下标

```
int firstLost(vector<int> nums)
{
   if(nums.empty())
      return -1;
   int n=nums.size();
   int left=0,right=n-1;
   while(left<=right)
   {
      int mid=(left+right)>>1;
      if(nums[mid]!=mid)
      {
         if(mid==0 || nums[mid-1]==mid-1)
            return mid;
         else
            right=mid-1;
      }
      else
         left=mid+1;
   }
   if(left==n)
      return n;
```

```
    return -1;
}
```

题目三

题目三：数组中数值和下标相等的元素。

假设一个单调递增的数组里的每个元素都是整数并且是唯一的。请编程实现一个函数，找出数组中任意一个数值等于其下标的元素。例如，在数组{-3, -1, 1, 3, 5}中，数字 3 和它的下标相等。

解

单调递增，二分查找

因为如果 m[i]>i，那么它右边的数也一定大于其下标；同理如果m[i]<i，它左边的数也一定小于其下标

```
int getNumber(vector<int> nums)
{
    int n=nums.size();
    if(n==0)
        rturn -1;
    int left=0,right=n-1;
    while(left<=right)
    {
        int mid=(left+right)>>1;
        if(nums[mid]==mid)
        {
            return mid;
        }
        else if(num[mid]>mid)
        {
            right=mid-1;
        }
        else
            left=mid+1;
    }
    return-1;
}
```