class Solution {

public:

int calcLeftBound(const vector<int>& data,int target)

{

int left=0,right=data.size()-1;

while(left<=right)

{

int mid=left+(right-left)/2;

if(data[mid]>target) right=mid-1;

else if(data[mid]<target) left=mid+1;

else if(data[mid]==target) right=mid-1;//收缩右侧边界

}

//检查是否越界

if(left>=data.size()||data[left]!=target) return -1;

return left;

}

int calcRightBound(const vector<int>& data,int target)

{

int left=0,right=data.size()-1;

while(left<=right)

{

int mid=left+(right-left)/2;

if(data[mid]>target) right=mid-1;

else if(data[mid]<target) left=mid+1;

else if(data[mid]==target) left=mid+1;

}

if(right<0||data[right]!=target) return -1;

return right;

}

int GetNumberOfK(vector<int> data ,int k) {

//二分法查询数字的左右边界

//左边界查询

int leftBound=calcLeftBound(data, k);

int rightBound=calcRightBound(data, k);

return (leftBound==-1)?0:(rightBound-leftBound+1);

}

};