33题

方法一：

var search=function(nums,target){

if(nums.length<1) return -1;

var array=nums.concat(nums);

var low = 0;

var high = nums.length-1;

for(var i=0;i<array.length-1;i++){

if(array[i+1]<array[i]){

low=i+1;

high=low+nums.length;

break;

}

}

while(low<=high){

var mid=Math.floor((low+high)/2);

if(array[mid]===target){

return mid%(nums.length);

}else if(array[mid]<target){

low=mid+1;

}else{

high=mid-1;

}

}

return -1;

};

方法二：（分情况讨论）

var search = function(nums, target) {

var low=0,

high=nums.length-1,

index=-1;

if(nums.length===0)return -1;

while(low<=high){

var mid=Math.floor((low+high)/2);

if(nums[mid]===target){

index=mid;

break;

}

else if(nums[low]<=nums[mid]){

if(nums[low]<=target&&target<=nums[mid]){

high=mid-1;

}else{

low=mid+1;

}

}else if(nums[low]>=nums[mid]){

if(nums[mid]<=target&&target<=nums[high]){

low=mid+1;

}else{

high=mid-1;

}

}

}

return index;

};

方法三：（offset方法 C++代码）

int l = 0,h = nums.size()-1;  
    while(l<h){  
        int mid = (l+h)/2;  
        if(nums[mid] > nums[h]) l= mid+1;  
        else h =mid;  
    }  
    cout<<l<<endl;  
    int rot = l;  
    l = 0,h=nums.size()-1;  
    while (l<=h){  
        int mid = (l+h)/2;  
        int rm = (mid+rot)%nums.size();  
        cout<<rm<<" "<<mid<<endl;  
        if (nums[rm] == target) return rm;  
        else if(nums[rm]>target) h=mid-1;  
        else l=mid+1;  
    }  
    return -1;

81题：

方法一：

var search=function(nums,target){

if(nums.length<1) return false;

var array=nums.concat(nums);

var low = 0;

var high = nums.length-1;

for(var i=0;i<array.length-1;i++){

if(array[i+1]<array[i]){

low=i+1;

high=low+nums.length;

break;

}

}

while(low<=high){

var mid=Math.floor((low+high)/2);

if(array[mid]===target){

return true;

}else if(array[mid]<target){

low=mid+1;

}else{

high=mid-1;

}

}

return false;

};

方法二:（和上题的方法二作对比）

var search = function(nums, target) {

var low=0,

high=nums.length-1,

mid=0;

while(low<=high){

mid=Math.floor((low+high)/2);

if(nums[mid]===target)return true;

else if(nums[low]<nums[mid]){

if(nums[low]<=target&&target<=nums[mid]){

high=mid-1;

}else{

low=mid+1;

}

}else if(nums[low]>nums[mid]){

if(nums[mid]<=target&&target<=nums[high]){

low=mid+1;

}else{

high=mid-1;

}

}else{

low++;

}

}

return false;

};

方法三：（offset方法）

34题：

方法一：

var searchRange = function(nums, target) {

var array=new Array();

var low=0,

high=nums.length-1;

if(nums.length<1){

array.push(-1);

array.push(-1);

return array;

}else if(nums.length===1){

if(nums[0]===target){

array.push(0);

array.push(0);

return array;

}else{

array.push(-1);

array.push(-1);

return array;

}

}

while(low<=high){

if(nums[low]<target){

++low;

}

if(nums[high]>target){

--high;

}

if(nums[low]===target&&nums[high]===target){

array.push(low);

array.push(high);

return array;

}

}

array.push(-1);

array.push(-1);

return array;

};

方法二：

var binary=function(nums,target){

var low= 0,

high=nums.length- 1;

while(low<=high){

var mid=Math.floor((low+high)/2);

if(nums[mid]>=target){

high=mid-1;

}else{

low=mid+1;

}

}

return low;

}

var searchRange=function(nums,target){

var array1=new Array();

var index1=binary(nums,target);

var index2=binary(nums,target+1)-1;

if(index1<nums.length&&nums[index1]==target){

array1.push(index1);

array1.push(index2);

return array1;

}else{

array1.push(-1);

array1.push(-1);

return array1;

}

};

int l = 0,h = nums.size()-1;  
    while(l<h){  
        int mid = (l+h)/2;  
        if(nums[mid] > nums[h]) l= mid+1;  
        else h =mid;  
    }  
    cout<<l<<endl;  
    int rot = l;  
    l = 0,h=nums.size()-1;  
    while (l<=h){  
        int mid = (l+h)/2;  
        int rm = (mid+rot)%nums.size();  
        cout<<rm<<" "<<mid<<endl;  
        if (nums[rm] == target) return rm;  
        else if(nums[rm]>target) h=mid-1;  
        else l=mid+1;  
    }  
    return -1;