#LeetCode33

https://leetcode.com/problems/search-in-rotated-sorted-array/description/

Yifeng Zeng

#题目描述

Search in Rotated Sorted Array

#思路报告

The primitive idea is to traverse the array to see if target is in it which is O(n) time. If we want to do better we may want to consider throw away half of the impossible part of the array, which would be a binary search. Since the array is rotated, we can split it into two halves by where the retation starts. So it can be split by the last number of the array. One half is larger than the last number, the other half is smaller than the last number. If mid lays on the upper half, then if target lays between start and mid, throw the second half, end = mid, otherwise throw the first half, start = mid. If mid lays on the lower half, then if target lays between mid and end, throw the first half, start = mid, otherwise throw the second half, end = mid.

代码如下：

```
public int search(int[] nums, int target) {
    if (nums == null || nums.length == 0) {
        return -1;
    }

    int start = 0;
    int end = nums.length - 1;
    int last = nums[end];

    while (start + 1 < end) {
        int mid = (end - start) / 2 + start;
        if (nums[mid] > last) {
            if (nums[start] <= target && target <= nums[mid]) {
                end = mid;
            } else {
                start = mid;
            }
```

```
            } else {
                if (nums[mid] <= target && target <= nums[end]) {
                    start = mid;
                } else {
                    end = mid;
                }
            }
        }

        if (nums[start] == target) {
            return start;
        }
        if (nums[end] == target) {
            return end;
        }
        return -1;
    }
```

#套路总结

- 把mid的位置分成多个section分类讨论