<https://leetcode.com/problems/binary-search/>

**Binary Search**

**Given an array of integers nums which is sorted in ascending order, and an integer target, write a function to search target in nums. If target exists, then return its index. Otherwise, return -1.**

**You must write an algorithm with O(log n) runtime complexity.**

Example 1:

Input: nums = [-1,0,3,5,9,12], target = 9

Output: 4

Explanation: 9 exists in nums and its index is 4

Example 2:

Input: nums = [-1,0,3,5,9,12], target = 2

Output: -1

Explanation: 2 does not exist in nums so return -1

Constraints:

1 <= nums.length <= 104

-104 < nums[i], target < 104

All the integers in nums are unique.

nums is sorted in ascending order.

**Method 1: (recursion)**

Time Complexity: O(logn) *[]*

Space Complexity: O(1) *[]*

int binSearch(vector<int>& nums, int target, int l, int r){ //using recursion

        if(l>r)

            return -1;

        int mid = l + (r-l)/2;

        if(nums[mid]==target)

            return mid;

        else if(target<nums[mid])

            return binSearch(nums, target, l, mid-1);

        else

            return binSearch(nums, target, mid+1, r);

    }

    int search(vector<int>& nums, int target) {

        return binSearch(nums, target, 0, nums.size()-1);

    }

**Method 2: (loop)**

int search(vector<int>& nums, int target) {

        int l = 0, r = nums.size()-1, mid;

        while(l<=r){

            mid = l + (r-l)/2;

            if(target==nums[mid])

                return mid;

            else if(target<nums[mid])

                r = mid - 1;

            else

                l = mid + 1;

        }

        return -1;

    }