

정렬된 nums를 입력받아이진 검색으로 target에 해당하는 인덱스를 찾아라.

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

Output: 4

1.재귀

class Solution:

```
def search(self, nums: List[int], target: int) -> int:
```

```
def binarySearch(left, right):
```

```
    if left > right:
```

```
        return -1
```

```
    result = (left + right) // 2
```

```
    if nums[result] < target:
```

```
        return binarySearch(result + 1, right)
```

```
    elif nums[result] > target:
```

```
        return binarySearch(left, result - 1)
```

```
    else:
```

```
        return result
```

```
    return binarySearch(0, len(nums) - 1)
```

- (left + right) // 2 가 오버플로 될수 있음 --->>> 파이썬은 상관x
- left + ((right - left) // 2)로 변경시 오버플로 없음

2.반복

class Solution:

```
def search(self, nums: List[int], target: int) -> int:
```

```
    left = 0
```

```
    right = len(nums) - 1
```

```
    while left <= right:
```

```
        result = (left + right) // 2
```

```
        if nums[result] < target:
```

```
            left = result + 1
```

```
        elif nums[result] > target:
```

```
            right = result - 1
```

```
        else:
```

```
            return result
```

```
    return -1
```

- (left + right) // 2 가 오버플로 될수 있음 --->>> 파이썬은 상관x
- left + ((right - left) // 2)로 변경시 오버플로 없음

2.파이썬 이진 검색 모듈

import bisect

class Solution:

```
def search(self, nums: List[int], target: int) -> int:
```

```
    index = bisect.bisect_left(nums, target)
```

```
    if index < len(nums) and nums[index] == target:
```

```
        return index
    else:
        return -1
```

3.파이썬 다운

class Solution:

```
    def search(self, nums: List[int], target: int) -> int:
        if target in nums:
            return nums.index(target)
        else:
            return -1
```