Search Insert Position

Given a sorted array of distinct integers and a tar
-get value return the index if target is found. If not
return the index where it would be if it were in
-serted in order
You must write an algorithm with O(log n) runtime comple
-xity

Ex:
Input: nums = [1,3,5,6], target = 5
output: 2

constraints:
- 1 <= nums.length <= 10^4
- -10^4 <= nums[i] <= 10^4
- nums contains distinct values sorted in ascending order
- -10^4 <= target <= 10^4

Igorithm:
- Intialize left=0, right = nums.length-1, start & end of the array.
Input array 'nums' and target value 'target'
- Itelate while left <= right.
calculate middle index: mid = left + (right-left)/2
- we use this instead of $\frac{left+right}{2}$ to avoid integer overflow
- Compare nums[mid] with target:
  - If equal, return mid (position found)

- If nums[mid] < target, move left=mid+1

Because the target is larger than mid, so it must lie in right half

- If nums[mid] ≥ target, move right=mid-1

Because the target is greater than mid, it must lie in left half

→ When the loop ends, return left, that's where the target should be inserted to keep array sorted

code:

```
int left=0;      // start of array
int right=nums.length-1; // end of array
while(left<=right){
        int mid=left+(right-left)/2
        // prevents overflow
    if(nums[mid]==target){
      return mid;  // match found
    }
else if(nums[mid]<target){  // target bigger than mid
    left=mid+1    // so move pointer to right half
  }
else{
    right=m-1;   // target less than mid, so move
                 right pointer to left half
  }
}
return left;  // if target not found, left pointer is correct index to insert at
```