Strivers-A2Z-DSA-Sheet-main\02.Binary Search\1D Arrays\04.Search_insert_position.cpp

```cpp
/*
QUESTION:
Given a sorted array of distinct integers and a target value, return the index if the target is found. If not, return the index where it would be if it were inserted in order.

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

Example:

Input: nums = [1,3,5,6], target = 5
Output: 2
Example 2:

Input: nums = [1,3,5,6], target = 2
Output: 1
*/

/*
APPROACH:
We can use the lower_bound function from the C++ standard library to find the index where the target should be inserted. The lower_bound function returns an iterator pointing to the first element that is not less than the target. By subtracting the beginning iterator from the lower_bound iterator, we get the index where the target should be inserted.

CODE:
*/

int searchInsert(vector<int>& nums, int target) {
    auto ans = lower_bound(nums.begin(), nums.end(), target) - nums.begin();
    return ans;
}

// TIME COMPLEXITY: O(log n) due to the use of lower_bound function
// SPACE COMPLEXITY: O(1)

```