Array Assignment: 03

16. 3Sum Closest

Given an integer array nums of length n and an integer target, find three integers in nums such that the sum is closest to target.

Return the sum of the three integers.

You may assume that each input would have exactly one solution.

```
//Time Complexity: O(n^2)

//Space Complexity: O(1)

class Solution {

public:

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

        sort(nums.begin(),nums.end());

        int n=nums.size();

       int ClosestSum=nums[0]+nums[1]+nums[n-1];



       for(int i=0;i<n-2;i++)

       {

          int start=i+1;

          int end=n-1;

          while(start<end)

          {

              int CurrSum=nums[i]+nums[start]+nums[end];

              if(CurrSum>target)
```

```
                {
                    end--;
                }
                else
                {
                    start++;
                }
                if(abs(CurrSum-target)<abs(ClosestSum-target))
                {
                    ClosestSum=CurrSum;
                }
            }

        }
        return ClosestSum;
    }
};
```

17. 35. Search Insert Position

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.

```
//Time Complexity: O(log n)

//Space Complexity:O(1)

class Solution {

public:

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

    int low=0, high=nums.size()-1, mid;

    if(target<nums[low])

    {

        return 0;

        //cout<<nums[left];

    }

     if(target>nums[high])

    {

        return nums.size();

        //cout<<nums[right];

    }

    while(low<=high)

    {

        mid=low+(high-low)/2;
```

```
        if(target>nums[mid])
        {
            low=mid+1;
            cout<<low;
        }
        else if(target<nums[mid])
        {
            high=mid-1;
            cout<<high;
        }
        else
        {
            return mid;
            cout<<mid;
        }
    }
    return low;
    cout<<low;
    }
};
```