```cpp
#include <iostream>
#include<vector>
#include<algorithm>
using namespace std;
//find target in rotated array
int findTarget(int arr[], int target)
{
    int start = 0, end = 8;
    while (end >= start)
    {
        int mid = (start + end) / 2;
        if (arr[mid] == target)
        {
            return mid;
            break;
        }
        // Check if the left part is sorted
        if (arr[mid] >= arr[start])
        {
            if (arr[start] <= target && target < arr[mid])
            {
                end = mid - 1; // Target is in the left part
            }
            else
            {
                start = mid + 1; // Discard the left part
            }
        }
        // Otherwise, the right part must be sorted
        else
        {
            if (arr[mid] < target && target <= arr[end])
            {
                start = mid + 1; // Target is in the right part
            }
            else
            {
                end = mid - 1; // Discard the right part
            }
        }
    }
    return -1;
}
int main()
{
    int arr[] = {7, 8, 9, 1, 2, 3, 4, 5, 6};
    int target = 5;
    int ans = findTarget(arr, target);
    cout << ans;
    return 0;
}
```

```cpp
//Book Allocation
int main(){
    int arr[] = {10,20,30,40};
    // int arr[] = {12,34,67,90};
    int student = 2;
    int size = sizeof(arr)/sizeof(arr[0]);
    int start = 0,end = 0, ans = -1;
    for(int i=0; i<size; i++){
        start = max(start,arr[i]);
        end += arr[i];
    }
    while(start<=end){
        int mid = (start+end)/2;
        int pages = 0, count=1;
        for(int i=0; i<size; i++){
            pages += arr[i];
            //if pages sixe exceed the max no. of pages that start assigning to next
student
            if(pages>mid){
                count++;
                pages = arr[i];
            }
        }
        //if we can assign mid no. of pages to the students , and we want to minimize
the max no. of pages assigned to a student
        //then we can try for a better solution in the left half of the array
        if(count<=student){
            ans = mid;
            end = mid-1;
        }
        //if we cannot assign mid no. of pages to the students, then we have to try for
a better solution in the right half of the array
        //eg if we have to assign max 20 pages to a student and require more than 2
students, then we have to increase the max no. of pages assigned to a student because if
we cannot able to assign max 20 pages than obviously we cannot assign less than 20 pages
also to a student
        else{
            start = mid+1;
        }
    }
    cout<<"ANS: "<<ans;
    return 0;
}



//aggresive cows
bool isPossible(vector<int> &arr, int mid, int cows){
    int count = 1;
    int lastPos = arr[0];
    for(int i=1; i<arr.size(); i++){
        if(arr[i]-lastPos>=mid){
            count++;
```

```
            lastPos = arr[i];
        }
        if(count >= cows){
            return true;
        }
    }
    return false;
}

int main(){
    vector<int> arr = {4,2,1,3,6};
    int cows = 2;
    int size = arr.size();
    int start = 0, end = 0, ans = -1;
    for(int i=0; i<size; i++){
        end = max(end,arr[i]);
    }
    sort(arr.begin(),arr.end());
    while(start<=end){
        int mid = (start+end)/2;
        if(isPossible(arr,mid,cows)){
            ans = mid;
            start = mid+1;
        }
        else{
            end = mid-1;
        }
    }
    cout<<"ANS: "<<ans;
    return 0;
}



//search element in row and column wise sorted matrix
//T.C = O(n+m) and S.C = O(1)
int main(){
    int arr[4][4] = {{10,20,30,40},
                     {15,25,35,45},
                     {27,29,37,48},
                     {32,33,39,50}};
    int target = 29;
    int row = 4;
    int col = 4;
    int i=0,j=col-1;
    while(i<row && j>=0){
        if(arr[i][j] == target){
            cout<<"Element found at: "<<i<<","<<j;
            return 0;
        }
        else if(arr[i][j]>target){
            j--;
        }
        else{
```

```
                i++;
            }
        }
        cout<<"Element not found";
        return 0;
    }
```