Binary Search.

FIND HOW MANY TIMES HAS AN ARRAY HAS BEEN ROTATED?

arr = [3 4 5 1 2] ans = 3.

it will have unique numbers.

ideally a Sorted array is arr = [1 2 3 4 5]

But the array given is arr = [3 4 5 1 2], so it is seen that array is rotated.

∴ Lastly we have solved, the minimum in Rotated Sorted Array.

Index → 0 1 2 3 4

arr[] = [3 4 5 1 2]

The index at which, the minimum element is present is the answer. If we track the no. of index of the minimum element in array that is the number of times array is rotated.

C++ Code.

```
# include
int findnooftimesrotated (vector<int> &arr)
 int low=0; high = arr.size()-1;
 int ans = INT_MAX;
 int index = -1;
while (low<=high)
int mid= (low+high)/2;
  // search space is already sorted
  // they always arr[low] is will be smaller
  // in that search space.

if (arr[low]<=arr[high]) {
      if(arr[low] < ans) {
          index= low;
          ans= arr[low];
       }
     break;
  }
   if(arr[low] <= arr[mid]) {
       if (arr[low] <ans){
           index = low;
           ans = arr[low];
         }
         low=mid+1;
      }
     else {
        high = mid-1;
         if(arr[mid]<ans) {
             index = mid;
             ans= arr[mid];
             }

       }
    }
     return index;
}
```