



In the nearly sorted array, the element present at the ith index in the sorted array can present in 3 places i.e (i-1) the index or ith index or (i+1) the index.

i=0 in sorted away can be present at -1 or 0 or 1st index in nearly sorted away & is found at 1st index.

i=1 in Sorted away can be present at 0,1 or 2nd index in the nearly sorted away. I hence is found at 0th index.

Similarly we can verify for all the elements as the condition will always be true.

Approach can be that we can apply linear search however it has time complexity = 0(n) but can we solve in the logn approach / time complexity.

Other approach can be like we can sort

the array & then apply binary search but the complexity of this solution will be $O(n \log n)$.

Algorithm
Sorted
Find mid = S+e &

compare arr [mid] and target.

Find mid & compared target with value at mid, mid-lor mid+l index and return index in each case.

	_ CIAS
(194)	Date
	Page
1	

	target > arr (m)
	target > aru (mid), target > aru (mid) search in right side side i.e s=mid+1 i.e s=mid+1 he added 2
	seauch in right state side i.e s= mid
	search in right side side i.e s=mid+1 i.e s=mid+1 We added 2 as
	we have already
	checked mid+1
	index.
	Jan Barrier Ba
	tanget < any [mid], Here we will
3)_	cong a root crimos,
	Search in left side search in left i.e e= side i.e e=
Variable of the	
	mid-2 as we
MAGO	have already -
	checked mid-1
	index·
١. ت	a playman sur digit ad on during t
((')	Code man money at the second of the
	in can we said of mire an and a
	int binary Search (vector <int> aver , int target)</int>
	the second secon
4	Int S = O just place to the second
	int e = over size()-1;
	int mid = S+ (e-S)/2;
	while (S < = e) {
=	if (our [mid] = = target) {
	XP+117n no.: 1
thousand	to Start Valid Inday of
3 315 1	if (mid-1 >= 0 { & ovr (mid-1) = = target)}
1 1	return mid-li
ومع المسالح	2011 12 12 101
	if (mid+1 < arr. Size () lfarr[mid+1] = = target
	return mid +1;
	3

Scarnieu with Cam

	if (target > and [mid]) {
	if (target > aur [mid]) { S=mid+2;
	3
	else {
	e=mid-2;
	3
	mid = s + (e-s)/2;
	3
	retwin - 13 dal marine
	3
Vote	We can optimize the above code by
- 110	modifying the condition of valid index
	check to other condition.
	control of the state of the sta
	$mid-1>=0 \rightarrow mid-1>=5$
1	mid+1 < over size() -> mid+1 <= e
	Time complexity: O(logn) same as that
	of binary search.
	0
Q2	Divide two numbers using binary search divisor polividend
	$i/b \rightarrow dividend = 10$ $\Rightarrow 2 \int 10 (5)$
	divisor = 2 10 quotient.
	quotient = ?
	remainder
	0/b -> 5
	Now the question is how can we use binary.
Maria Company	search algorithm to find the quotient. The
温祥	similar kind of approach of finding the square noot of a number using binary search.
	square noot of a number using billowy search.
	Scarnieu with ca

Scarnieu with CaM

	Class	
196	Date Page	MAte

	Here we will take the search space from
	0 to dividend.
	Dividend = Quotient * Divisor + Remainder Also we can modify the above formulae to
	quotient * remainder <= dividend
	Algorithm for 10/2
	Start = 0
	end = 10
36	mid = 0 + 10 = 5
	2
	mid * divisor = 5 * 2 = = 10 return mids
	111/10 (CEWIII 10/10)
	Algorithm for 2217
Jack	Start = O (min)
9	end = 22
	mid = 0 + 22 = 11
dia	2 UPIVIZ
100010	mid * divisor = 11 * 7 = 77 > 22
	move to the left side by e=mid-1.
Journal of	10 75.20 Age siac by e=mid-1.
૨)	Start = 0
	end = 10
	mid = 0+10 =5
121	2
, b	mid * divisor = 5 * 7 = 35 > 22
	Again Search in the left side by e=mid-
diffe	U By e=mid=
	D NOW THE STATE OF

Scarineu with Cam

3) start = 0

end = 4

mid = 0 + 4 = 2

mid * divisor = 2 * 7 = 14 < = 22

(i) Store ans as ans = mid as it

might be the answer.

(ii) Search in right part S = mid + 1;

4) start = 3

end = 4

mid = 3 + 4 = 3

mid * divisor = 3 * 7 = 21

(i) store ans as ans = mid

ans = 3

(ii) Search in right part

s = mid + 1

5) Start = 4

end = 4

mid = 4 + 4 = 4

mid * divisor = 4 * 7 = 28 > 22

e = mid-1; 3 Search in the left part

Now start > end, exit the loop.

Code

int solve (int dividend, int divisor) {

Scarnieu with cam

3.	CIASSMAS
	198) Date Page
	int o - o i
	int s = 0; int e = abs(dividend);
	int mid = S + (e-s)/2;
	int ans = 1;
	int ans -1)
	while (S < = e) {
	While (3 < - c) c
	if (abs (mid * divisor) = = abs (dividend)){
	ans = mid : // got the answer
	break;
	3
	if (abs (mid * divisor) > abs (dividend)){-
	e=mid-1; // Search in left
	3
	else {
	ans=midi //Store answer
	S=mid+el; // Search in right
	3
	mid = S + (e-s)/2j
	3
	1/ To handle the -ve case
	if (divisor < 0 && dividend < 0) (divisor >0 44
	dividend >0){
	retwin ans j
	3 3 3 4 4 4 4 4
t	else {
	return -ansi
	3
	5
Wote >	$abs(-3)j \rightarrow 3$ $abs(3)j \rightarrow 3$
11	Scarnieu With C

Scarnieu with Cam



	abs () always return the +ve value.
23	Find the odd occurring element in the away.
	i/b→1 1 2 2 3 3 4 4 3 600 600 4 4
	In this question all the elements occur even
	number of times except one. Also all the
	repeating occurrence of element appear in the
	pairs and the pairs are not adjacent · Also
	note that there can not be more than 2
	consecutive occurrence of any element. We have
	to tind the element that appears odd number
	of times.
	•
	Algorithm - 1
	Brute force approach can be doing XOR of
	all the elements of the away. Time complexity of this approach is O(n) but can we
	of this approach is O(n) but can we
	<u> </u>
	Algorithm-2
	1 1 2 2 3 3 4 45 31 600 600 44
_	ans
	deft of ans -> pair
	First value Second value
	on the even on the odd
	index inden
	Right of ans -> pair
	First value Last/second value
	on odd index on even index.
	Scarnieu With C

	_ 0	554
(200)	Date	- SWAF
	Page	1
W.		

	Also from observation, and always lies on the even index.
	Also from observations
	on the even index.
	aidened as even
Wote	- 0th index will be considered as even
	in this question.
10000	
	S = 0
	e = n - 1
= 1	e = n - 1 $mid = s + e$? Change in code to $s + (e-s)$;
	while (s<=e) {
	if (S = = e) 2 Only one element
-	return si J remaining case
Case-	if (mid 1/. 2 = =0){→ Even index
	Teft side of answer
	if (our [mid] = = our [mid+1]) {
\	// Search in the right part
	s = mid + 2
	3 - As midtl abready
	checked.
1	else à
	e=mid j // if we do e=mid-1, then
	3 we might loose the answer_
	3 as mid may be an answer.
	ren kritari zate de de de de
Case-	else { -) odd index
*********	Pleft part as mid 15 ca
	if (aur [mid-1] = aur (mid]) {
	// Search in right part
	5 = mid + (1)
J. 4	4 Here 1 4 not 2 because
· /	3 midtlis not explored

Scarned with Cam



	else {
	e = mid-1; // Here mid can not be
	unswer as the mid index
	is odd but answer is at
	even index.
	<u>Code mon vous, add unidous marals (1)</u>
	int odd Occurence (vector <int>v) {</int>
	int s = oi
	inte = V·size() - 1;
	int mid = S + (e-s)/2;
	while (S < = e) {
	if (S = = e)
	OCCUMIT S 2
	if (mid 1/0 2 = = 0) {
	if (V[mid] = = V[mid+1]){
	S=mid+2;
	else }
	e=mid;
	2,
	else {
	if (v(mid) = = v(mid-1)) {
	5=mid+1)
	3
	else {
	e = mid-1;
*	3
The state of the s	

Scanned with CaM

	CLASSMAS
(202)	Date
(Page
-	

	A suge
	mid = s + (e-s)/2j
	3
	retwin - 1
	3
	Types of questions in binary search
	1 7 PCS 0 6 9/0000
1)	Classic Questions like lower bound,
	Upper bound etc.
2)	Search space predicate function question
	Search space predicate function question such as aggressive cows, bookallocation
~	etc.
3)	Observing index value like the question of
	Observing index value like the question of finding odd occurring element.
	U U

Scarrieu with Cam