GeeksforGeeks
A computer science portal for geeks

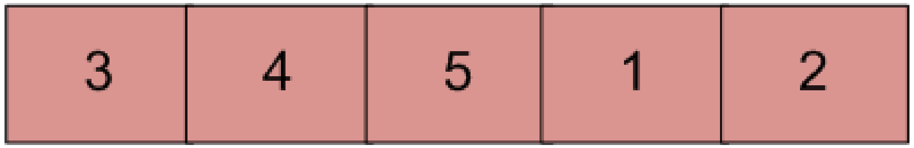
GeeksQuiz

Login

Home Algorithms DS GATE Interview Corner Q&A C C++ Java Books Contribute Ask a Q About
Array Bit Magic C/C++ Articles GFacts Linked List MCQ Misc Output String Tree Graph

Search an element in a sorted and pivoted array

Question:
An element in a sorted array can be found in O(log n) time via binary search. But suppose I rotate the sorted array at some pivot unknown to you beforehand. So for instance, 1 2 3 4 5 might become 3 4 5 1 2. Devise a way to find an element in the rotated array in O(log n) time.



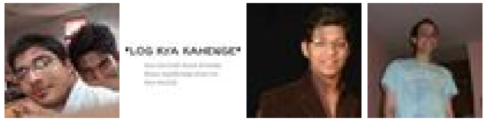
3	4	5	1	2

Solution:
Thanks to Ajay Mishra for initial solution.

Algorithm:
Find the pivot point, divide the array in two sub-arrays and call binary search.
The main idea for finding pivot is – for a sorted (in increasing order) and pivoted array, pivot element is the only only element for which next element to it is smaller than it.
Using above criteria and binary search methodology we can get pivot element in O(logn) time

```
Input arr[] = {3, 4, 5, 1, 2}
Element to Search = 1
  1) Find out pivot point and divide the array in two
      sub-arrays. (pivot = 2) /*Index of 5*/
```

Google™ Custom Search



Interview Experiences
Advanced Data Structures
Dynamic Programming
Greedy Algorithms
Backtracking
Pattern Searching
Divide & Conquer
Mathematical Algorithms
Recursion
Geometric Algorithms

```
  2) Now call binary search for one of the two sub-arrays.
      (a) If element is greater than 0th element then
             search in left array
      (b) Else Search in right array
          (1 will go in else as 1 < 0th element(3))
  3) If element is found in selected sub-array then return index
     Else return -1.
```

Implementation:

```
/* Program to search an element in a sorted and pivoted array*/
#include <stdio.h>

int findPivot(int[], int, int);
int binarySearch(int[], int, int, int);

/* Searches an element no in a pivoted sorted array arrp[]
   of size arr_size */
int pivotedBinarySearch(int arr[], int arr_size, int no)
{
   int pivot = findPivot(arr, 0, arr_size-1);

   // If we didn't find a pivot, then array is not rotated at all
   if (pivot == -1)
     return binarySearch(arr, 0, arr_size-1, no);

   // If we found a pivot, then first compare with pivot and then
   // search in two subarrays around pivot
   if (arr[pivot] == no)
     return pivot;
   if (arr[0] <= no)
     return binarySearch(arr, 0, pivot-1, no);
   else
     return binarySearch(arr, pivot+1, arr_size-1, no);
}

/* Function to get pivot. For array 3, 4, 5, 6, 1, 2 it will
   return 3. If array is not rotated at all, then it returns -1 */
int findPivot(int arr[], int low, int high)
{
   // base cases
   if (high < low)  return -1;
   if (high == low) return low;

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

Geometric Algorithms

Popular Posts

All permutations of a given string

Memory Layout of C Programs

Understanding “extern” keyword in C

Median of two sorted arrays

Tree traversal without recursion and without stack!

Structure Member Alignment, Padding and Data Packing

Intersection point of two Linked Lists

Lowest Common Ancestor in a BST.

Check if a binary tree is BST or not

Sorted Linked List to Balanced BST

```
   int mid = (low + high)/2;   /*low + (high - low)/2;*/
   if (mid < high && arr[mid] > arr[mid + 1])
     return mid;
   if (mid > low && arr[mid] < arr[mid - 1])
     return (mid-1);
   if (arr[low] >= arr[mid])
     return findPivot(arr, low, mid-1);
   else
     return findPivot(arr, mid + 1, high);
}

/* Standard Binary Search function*/
int binarySearch(int arr[], int low, int high, int no)
{
   if (high < low)
       return -1;
   int mid = (low + high)/2;  /*low + (high - low)/2;*/
   if (no == arr[mid])
     return mid;
   if (no > arr[mid])
     return binarySearch(arr, (mid + 1), high, no);
   else
     return binarySearch(arr, low, (mid -1), no);
}


/* Driver program to check above functions */
int main()
{
   // Let us search 3 in below array
   int arr1[] = {5, 6, 7, 8, 9, 10, 1, 2, 3};
   int arr_size = sizeof(arr1)/sizeof(arr1[0]);
   int no = 3;
   printf("Index of the element is %d\n",  pivotedBinarySearch(arr1, a

    // Let us search 3 in below array
   int arr2[] = {3, 4, 5, 1, 2};
   arr_size = sizeof(arr2)/sizeof(arr2[0]);
   printf("Index of the element is %d\n",  pivotedBinarySearch(arr2, a

   // Let us search for 4 in above array
   no = 4;
   printf("Index of the element is %d\n",  pivotedBinarySearch(arr2, a

   // Let us search 0 in below array
   int arr3[] = {1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1};
   no = 0;
   arr_size = sizeof(arr3)/sizeof(arr3[0]);
```

```
    printf("Index of the element is %d\n",  pivotedBinarySearch(arr3, a

    // Let us search 3 in below array
    int arr4[] = {2, 3, 0, 2, 2, 2, 2, 2, 2, 2};
    no = 3;
    arr_size = sizeof(arr4)/sizeof(arr4[0]);
    printf("Index of the element is %d\n",  pivotedBinarySearch(arr4, a

    // Let us search 2 in above array
    no = 2;
    printf("Index of the element is %d\n",  pivotedBinarySearch(arr4, a

    // Let us search 3 in below array
    int arr5[] = {1, 2, 3, 4};
    no = 3;
    arr_size = sizeof(arr5)/sizeof(arr5[0]);
    printf("Index of the element is %d\n",  pivotedBinarySearch(arr5, a

    return 0;
}
```

Output:

```
Index of the element is 8
Index of the element is 0
Index of the element is 1
Index of the element is 3
Index of the element is 1
Index of the element is 0
Index of the element is 2
```

Please note that the solution may not work for cases where the input array has duplicates.

Time Complexity O(logn)

Please write comments if you find any bug in above codes/algorithms, or find other ways to solve the same problem.

705 Subscribe

Recent Comments

Aman Hi, Why arent we checking for conditions...

Write a C program to Delete a Tree. · 22 minutes ago

kzs please provide solution for the problem...

Backtracking | Set 2 (Rat in a Maze) · 25 minutes ago

Sanjay Agarwal bool tree::Root_to_leaf_path_given_sum(tree...

Root to leaf path sum equal to a given number · 50 minutes ago

GOPI GOPINATH @admin Highlight this sentence "We can easily...

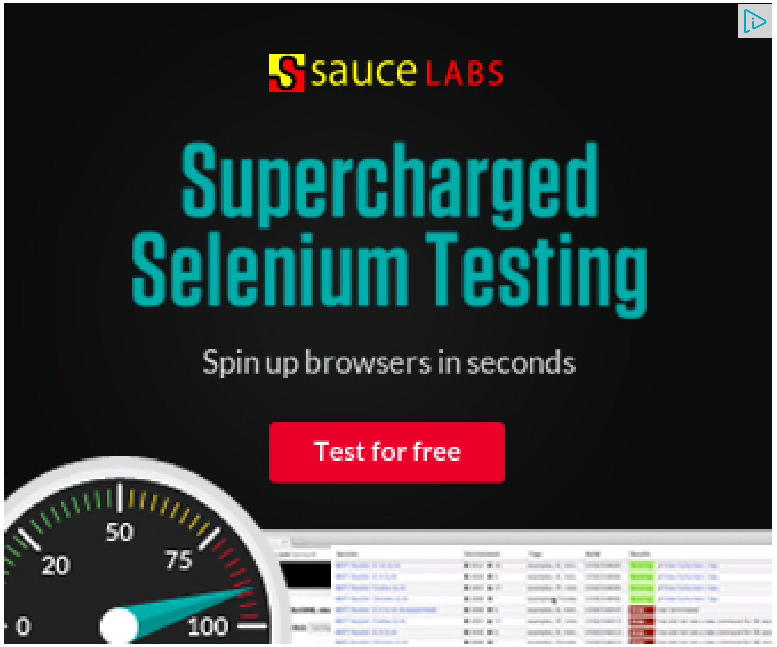
Count trailing zeroes in factorial of a number · 52 minutes ago

newCoder3006 If the array contains negative numbers also. We...

Find subarray with given sum · 1 hour ago

newCoder3006 Code without using while loop. We can do it...

Find subarray with given sum · 1 hour ago



Related Tpoics:

- Remove minimum elements from either side such that 2*min becomes more than max
- Divide and Conquer | Set 6 (Search in a Row-wise and Column-wise Sorted 2D Array)
- Bucket Sort
- Kth smallest element in a row-wise and column-wise sorted 2D array | Set 1
- Find the number of zeroes
- Find if there is a subarray with 0 sum
- Divide and Conquer | Set 5 (Strassen's Matrix Multiplication)
- Count all possible groups of size 2 or 3 that have sum as multiple of 3

7 Tweet 0 0

Writing code in comment? Please use **ideone.com** and share the link here.