K’th Smallest/Largest Element in Unsorted Array | Set 2 (Expected Linear Time)

We recommend to read following post as a prerequisite of this post.

K’th Smallest/Largest Element in Unsorted Array | Set 1

Given an array and a number k where k is smaller than size of array, we need to find the k’th smallest element in the given array. It is given that ll array elements are distinct.

Examples:

```
Input: arr[] = {7, 10, 4, 3, 20, 15}
       k = 3
```

```
Output: 7

Input: arr[] = {7, 10, 4, 3, 20, 15}
       k = 4
Output: 10
```

We have discussed three different solutions here.

In this post method 4 is discussed which is mainly an extension of method 3 (QuickSelect) discussed in the previous post. The idea is to randomly pick a pivot element. To implement randomized partition, we use a random function, rand() to generate index between l and r, swap the element at randomly generated index with the last element, and finally call the standard partition process which uses last element as pivot.

Following is C++ implementation of above Randomized QuickSelect.

```
// C++ implementation of randomized quickSelect
#include<iostream>
#include<climits>
#include<cstdlib>
using namespace std;

int randomPartition(int arr[], int l, int r);

// This function returns k'th smallest element in arr[l..r] using
// QuickSort based method.  ASSUMPTION: ALL ELEMENTS IN ARR[] ARE DISTINCT
int kthSmallest(int arr[], int l, int r, int k)
{
    // If k is smaller than number of elements in array
    if (k > 0 && k <= r - l + 1)
    {
        // Partition the array around a random element and
        // get position of pivot element in sorted array
        int pos = randomPartition(arr, l, r);

        // If position is same as k
        if (pos-l == k-1)
            return arr[pos];
        if (pos-l > k-1)  // If position is more, recur for left subarray
            return kthSmallest(arr, l, pos-1, k);

        // Else recur for right subarray
        return kthSmallest(arr, pos+1, r, k-pos+l-1);
    }

    // If k is more than number of elements in array
    return INT_MAX;
}

void swap(int *a, int *b)
{
    int temp = *a;
    *a = *b;
    *b = temp;
}

// Standard partition process of QuickSort().  It considers the last
// element as pivot and moves all smaller element to left of it and
// greater elements to right. This function is used by randomPartition()
int partition(int arr[], int l, int r)
{
    int x = arr[r], i = l;
    for (int j = l; j <= r - 1; j++)
    {
        if (arr[j] <= x)
        {
            swap(&arr[i], &arr[j]);
            i++;
        }
    }
    swap(&arr[i], &arr[r]);
    return i;
}

// Picks a random pivot element between l and r and partitions
// arr[l..r] arount the randomly picked element using partition()
int randomPartition(int arr[], int l, int r)
{
    int n = r-l+1;
    int pivot = rand() % n;
    swap(&arr[l + pivot], &arr[r]);
    return partition(arr, l, r);
}

// Driver program to test above methods
int main()
{
    int arr[] = {12, 3, 5, 7, 4, 19, 26};
    int n = sizeof(arr)/sizeof(arr[0]), k = 3;
    cout << "K'th smallest element is " << kthSmallest(arr, 0, n-1, k);
    return 0;
}
```

Output:

```
K'th smallest element is 5
```

Time Complexity:

The worst case time complexity of the above solution is still O(n^2). In worst case, the randomized function may always pick a corner element. The expected time complexity of above randomized QuickSelect is Θ(n), see CLRS book or MIT video lecture for proof. The assumption in the analysis is, random number generator is equally likely to generate any number in the input range.

Sources:

MIT Video Lecture on Order Statistics, Median
Introduction to Algorithms by Clifford Stein, Thomas H. Cormen, Charles E. Leiserson, Ronald L.

This article is contributed by **Shivam**. Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above

Related Topics:

- Find Union and Intersection of two unsorted arrays
- Pythagorean Triplet in an array
- Maximum profit by buying and selling a share at most twice
- Design a data structure that supports insert, delete, search and getRandom in constant time
- Print missing elements that lie in range 0 – 99
- Iterative Merge Sort
- Group multiple occurrence of array elements ordered by first occurrence
- Given a sorted and rotated array, find if there is a pair with a given sum

Tweet 3 g+1 2

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

12 Comments GeeksforGeeks 1 Login

Recommend Share Sort by Newest

Join the discussion…

Goku • 2 months ago

Cleaner Solution using the same method.
http://ideone.com/VXOiWH

Language : C++14

• Reply • Share ›

Guest • 4 months ago

can some one explain kthSmallest(arr, pos+1, r, k-pos+l-1) ..
why (k-pos+l-1)??

• Reply • Share ›

cfh → Guest • 2 months ago

this is the case when (pos-l<k-1), this="" means="" that="" kth="" smallest="" element="" will="" not="" be="" in="" left="" part="" arr[l...pos]="" when="" sorted,="" thus="" will="" be="" in="" right="" part="" a[pos+1...r].="" thus="" in="" right="" part="" we="" look="" for="" the="" (k-(pos-l+1))th="" element="" i.e.="" (k-pos+l-1)th="" element="" :)="">

• Reply • Share ›

Bhagwat Singh • 5 months ago

Bhagwat Singh • 5 months ago

I think in place of "%" there should be * . int pivot = rand() % n;
It should be rand()*n;

• Reply • Share ›

GeeksforGeeks Mod Bhagwat Singh • 5 months ago

The above implementation looks correct. Please refer http://www.cplusplus.com/refer... for rand()

• Reply • Share ›

Bhagwat Singh GeeksforGeeks • 5 months ago

WHAT IF THE NUMBER IS GREATER THAN 32767 BECAUSE THE STANDARD MAX IS THIS ONLY.

• Reply • Share ›

Raj • 5 months ago

```
#include<iostream>
#include <queue> // std::priority_queue
#include <functional> // std::greater
using namespace std;

class mycomparison;

typedef std::priority_queue<int,std::vector<int>,mycomparison> mypq_type;

//By default priority queue is max heap means max value will be in front
//if you want to max it min heap just implement below mycomparison class
int swapValue =0;
class mycomparison
{
bool reverse;
public:
mycomparison(const bool& revparam=false)//True for ascending
{reverse=revparam;}
bool operator() (const int& lhs, const int&rhs) const
```

see more

• Reply • Share ›

kaushik • 5 months ago

we can use min heap or maxheap and easily we can get kth min or max in linear time.

• Reply • Share ›

K. kaushik • 5 months ago

Shouldn't that be O(k*lgn) ?

• Reply • Share ›

Deepak Mishra K. • 5 months ago

Shouldn't we evaluate first (k * log n) is greater or (n) before applying the algorithm

• Reply • Share ›

Deepak Mishra Deepak Mishra • 5 months ago

Sorry my mistake heap sort is n * log k

• Reply • Share ›

kaushik K. • 5 months ago

yes I got

• Reply • Share ›

Subscribe Add Disqus to your site Privacy

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

- Follow @GeeksforGeeks

Recent Comments

- lt_k

  i need help for coding this function in java...

  Java Programming Language · 1 hour ago

- Piyush

  What is the purpose of else if (recStack[*i])...

  Detect Cycle in a Directed Graph · 1 hour ago

- Andy Toh

  My compile-time solution, which agrees with the...

  Dynamic Programming | Set 16 (Floyd Warshall Algorithm) · 1 hour ago

- lucy

  because we first fill zero in first col and...

  Dynamic Programming | Set 29 (Longest Common Substring) · 2 hours ago

- lucy

  @GeeksforGeeks i don't n know what is this long...

  Dynamic Programming | Set 28 (Minimum insertions to form a palindrome) · 2 hours ago

- manish

  Because TAN is not a subsequence of RANT. ANT...

  Given two strings, find if first string is a subsequence of second · 2 hours ago