Rotated Array

Suppose a sorted array A is rotated at some pivot unknown to you beforehand.

Find the minimum element.

Given that the array will not contain duplicates.

An integer n followed by an array A of size n.

Output should contain the minimum element in that array A.

5

3 4 5 1 2

1

1 <= n <= 10<sup>6</sup>

<br>1 <= arr[i] <= 10<sup>9</sup>

<br>where arr[i] is any element in the array.

Here the MInimum element is 1.

#include<bits/stdc++.h>

#define ll long long int

using namespace std;

ll rotateFindMin(ll\* arr, ll n) {

//int n = arr.size();

int high = n - 1;

int low = 0;

int ans = 0, mid;

while(low <= high) {

if(n==1) return arr[ans];

mid = low + ((high - low) / 2);

if( (arr[mid] < arr[(mid+1)%n]) && (arr[mid] < arr[(mid-1+n)%n]) ) {

return arr[mid];

}

else if(arr[mid] < arr[high]) {

high = mid;

}

else if(arr[mid] > arr[high]) {

low = mid + 1;

}

}

return arr[ans];

}

int main() {

ll n;

cin >> n;

ll\*arr = new ll[n+1];

for(ll i = 0; i < n; i++) {

cin >> arr[i];

}

cout << rotateFindMin(arr, n);

}

#include<bits/stdc++.h>

#define ll long long int

using namespace std;

ll rotateFindMin(ll\* arr, ll n) {

}

int main() {

ll n;

cin >> n;

ll\*arr = new ll[n+1];

for(ll i = 0; i < n; i++) {

cin >> arr[i];

}

cout << rotateFindMin(arr, n);

}

Here,in this problem we have to find the minimum element in the array,

- Pivot is the minimum element

- We need to find out the pivot element from the array.

- In sorted part we can apply binary search, we took initially high = n-1, low = 0.

- if element to the right and element to the left of the mid element is greater - the element is the pivot element.

- else :<br>

if :(element at mid is greater than the last) :<br>

a) Pivot is present in the right part

<br>else:

<br>b) Pivot is present in the left part.

<br><br> Here is the code :

<pre><code class="lang-cpp">

#include&lt;bits/stdc++.h&gt;

#define ll long long int

using namespace std;

ll rotateFindMin(ll\* arr, ll n) {

int high = n - 1;

int low = 0;

int ans = 0, mid;

while(low <= high) {

if(n==1) return arr[ans];

mid = low + ((high - low) / 2);

if( (arr[mid] < arr[(mid+1)%n]) && (arr[mid] < arr[(mid-1+n)%n]) ) {

return arr[mid];

}

else if(arr[mid] < arr[high]) {

high = mid;

}

else if(arr[mid] > arr[high]) {

low = mid + 1;

}

}

return arr[ans];

}

int main() {

ll n;

cin >> n;

ll\*arr = new ll[n+1];

for(ll i = 0; i < n; i++) {

cin >> arr[i];

}

cout << rotateFindMin(arr, n);

}