**Question 9:** Find the median of two sorted arrays

**Algorithm:**

Start

Algorithm for method main():

Start

Step 1: Accept a number and store it in an integer variable n.

Step 2: Declare an integer array a of size n and input integers in it.

Step 3: Accept another integer and store it in n.

Step 4: Declare an integer array b of size n and input integers in it.

Step 5: Call subroutine find() with a, length(a), b, length(b) as parameters.

Step 6: Print the returned result and if no result is returned then print an error message.

Stop

Algorithm for subroutine find():

Start

Step 1: Accept two integer arrays a and b and two integers n and m as parameters.

Step 2: Initialise integer variables min\_index with 0, max\_index with n and declare integer variables i and j.

Step 3:

1. i <-- (min\_index + max\_index) / 2
2. j <-- (n + m + 1) / 2 – i
3. if
   1. min\_index <-- i + 1
4. else if
   1. max\_index <-- i – 1
5. else
   1. if i = 0
      1. return b[j-1]
   2. else if j = 0
      1. return a[i-1]
   3. else
      1. return max(a[i-1], b[j-1])

Stop

Stop

**Code:**

#include <iostream>

#include <algorithm>

#include <vector>

#include <optional>

#include <tuple>

std::optional<int> find(const std::vector<int> &a, int n, const std::vector<int> &b, int m)

{

int min\_index = 0, max\_index = n, i, j;

while (min\_index <= max\_index)

{

i = (min\_index + max\_index) / 2;

j = ((n + m + 1) / 2) - i;

if (i < n && j > 0 && b[j - 1] > a[i])

min\_index = i + 1;

else if (i > 0 && j < m && b[j] < a[i - 1])

max\_index = i - 1;

else

{

if (i == 0)

return std::optional<int>{b[j - 1]};

if (j == 0)

return std::optional<int>{a[i - 1]};

else

return std::max(a[i - 1], b[j - 1]);

}

}

return {};

}

int main()

{

int n; std::cin>>n;

std::vector<int> a(n);

for(auto& i : a)

std::cin>>i;

std::cin>>n;

std::vector<int> b(n);

for(auto& i : b)

std::cin>>i;

std::optional<int> ans = find(a, a.size(), b, b.size());

if(ans.has\_value())

std::cout<<ans.value()<<'\n';

else

std::cout<<"Error!!!\n";

}

**Input:**

3

1 4 6

2

3 7

**Output:**

4

**Question 10:** Find the minimum element of a sorted array rotated clockwise arbitrarily.

**Algorithm:**

Start

Algorithm for method main():

Start

Step 1: Accept an integer and store it in an integer variable n.

Step 2: Declare an integer array arr of size n and accept elements in it

Step 3: Call subroutine findMin() with arr, 0 and n-1 as parameters and display the returned result.

Stop

Algorithm for subroutine findMin():

Start

Step 1: Accept an integer array arr and two variables low and high as parameters.

Step 2: if high < low

return arr[0]

Step 3: if high = low

return arr[low]

Step 4: Initialise an integer variable mid with ( low + high ) / 2

Step 5: if mid < high AND arr[mid+1] < arr[mid]

return arr[mid + 1]

Step 6: if mid > low AND arr[mid] < arr[mid – 1]

return arr[mid]

Step 7: if arr[high] > arr[mid]

return findMin(arr, low, mid – 1)

Step 8: return findMin(arr, mid + 1, high)

Stop

Stop

**Code:**

#include <iostream>

#include <vector>

int findMin(const std::vector<int> &arr, int low, int high)

{

if (high < low)

return arr[0];

if (high == low)

return arr[low];

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

if (mid < high && arr[mid+1] < arr[mid])

return arr[mid+1];

if (mid > low && arr[mid] < arr[mid - 1])

return arr[mid];

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

return findMin(arr, low, mid-1);

return findMin(arr, mid+1, high);

}

int main()

{

int n; std::cin>>n;

std::vector<int> arr(n);

for(auto& i : arr)

std::cin>>i;

std::cout<<findMin(arr, 0, n-1)<<'\n';

}

**Input:**

6

5 6 1 2 3 4

**Output:**

1