Reduce Array Size to the half 1

You are given an integer array arr. You can choose a set of integers and remove all the occurrences of these integers in the array.

Return the minimum size of the set so that at least half of the integers of the array are removed.

Sample Input 0

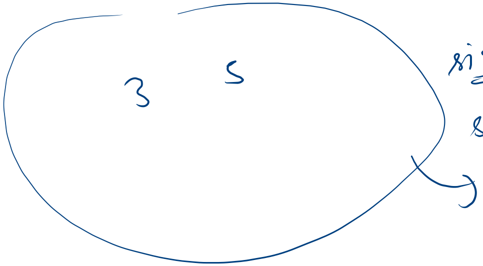
```
10
3
3
3
3
5
5
5
2
2
7
```

~~3~~ ~~3~~ ~~3~~ ~~3~~ ~~5~~ ~~5~~ ~~5~~ 2 2 7

n/2

Sample Output 0

```
2
```



size of set

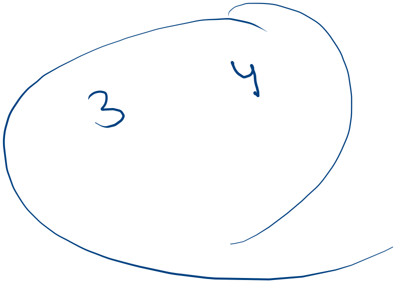
n = 10.

~~3~~ ~~3~~ ~~3~~ 7 5 6 ~~4~~ 1 2 ~~4~~

atleast
'5'

rem = ~~3~~ 5

step = ~~1~~ 2



$n=10$

3 3 3 1 4 5 4 2 6 7

atleast 5

freq.

1 →	1
4 →	2
5 →	1
3 →	3
6 →	1
7 →	1
2 →	1

3 4

freq.

✓ 1 →	1
4 →	2
5 →	1
3 →	3
6 →	1
7 →	1
2 →	1

✓ pq (max)

1 ~~2~~

1 ~~3~~ 1 1

1

$\boxed{\text{Step} = \cancel{0}\ \cancel{1}\ 2}$

removed = ~~0~~ ~~3~~ 5

while removed < n/2

↳ remove

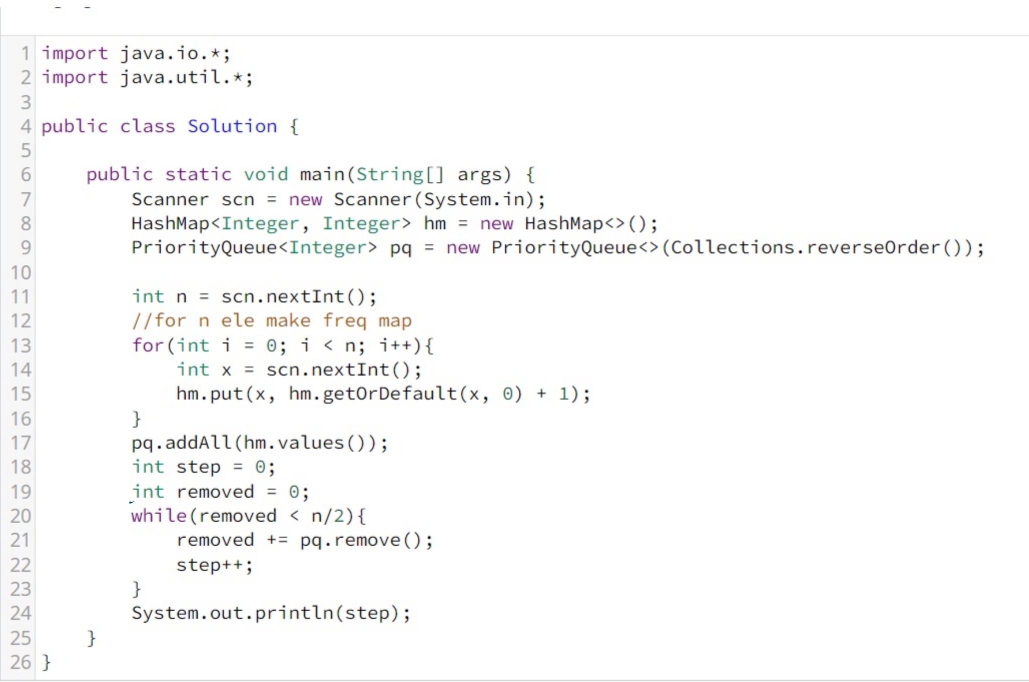
3

2

5 < 10/2

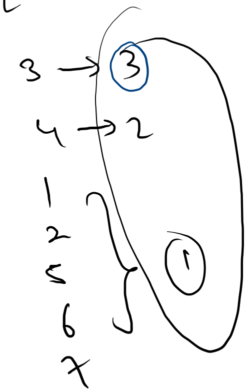
5 < 5 ✗

3 3 3 4 4 1 5 7 6 2



```java
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        HashMap<Integer, Integer> hm = new HashMap<>();
        PriorityQueue<Integer> pq = new PriorityQueue<>(Collections.reverseOrder());

        int n = scn.nextInt();
        //for n ele make freq map
        for(int i = 0; i < n; i++){
            int x = scn.nextInt();
            hm.put(x, hm.getOrDefault(x, 0) + 1);
        }
        pq.addAll(hm.values());
        int step = 0;
        int removed = 0;
        while(removed < n/2){
            removed += pq.remove();
            step++;
        }
        System.out.println(step);
    }
}
```

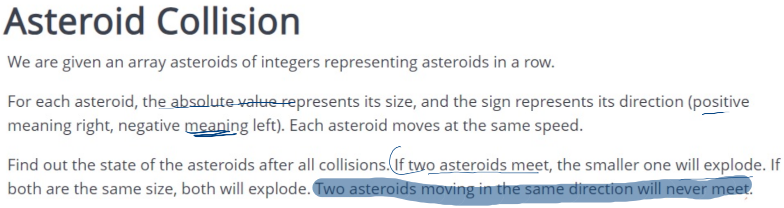
freq. map



pq

removed = ~~0~~ ~~3~~ 5 0 < 5 5 < 5

step = ~~0~~ ~~1~~ 2 3 < 5



Asteroid Collision

We are given an array asteroids of integers representing asteroids in a row.

For each asteroid, the absolute value represents its size, and the sign represents its direction (positive meaning right, negative meaning left). Each asteroid moves at the same speed.

Find out the state of the asteroids after all collisions. If two asteroids meet, the smaller one will explode. If both are the same size, both will explode. Two asteroids moving in the same direction will never meet.

+ −

Sample Input 0

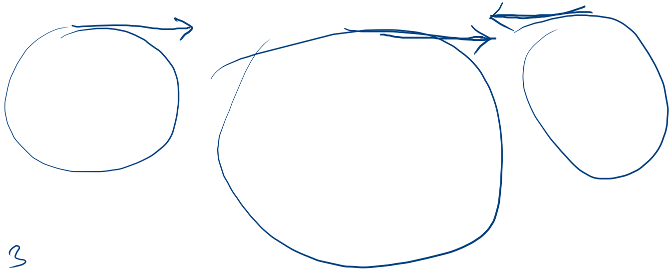
```
3
5
10
-5
```

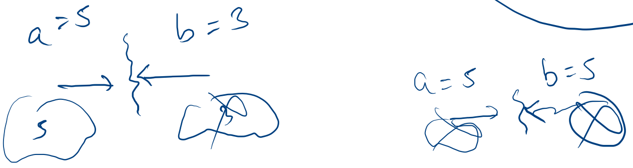
Sample Output 0

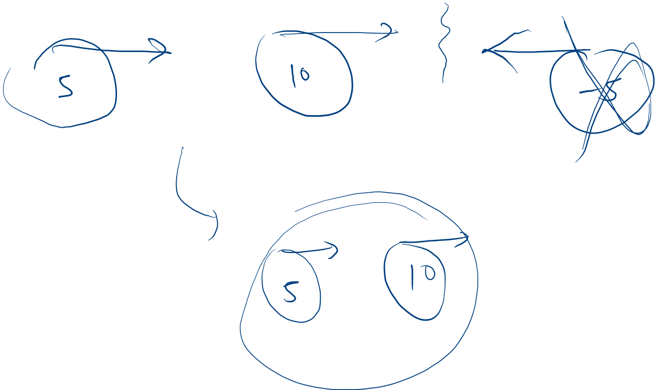
```
5 10
```

n=3

A → 5 10 −5



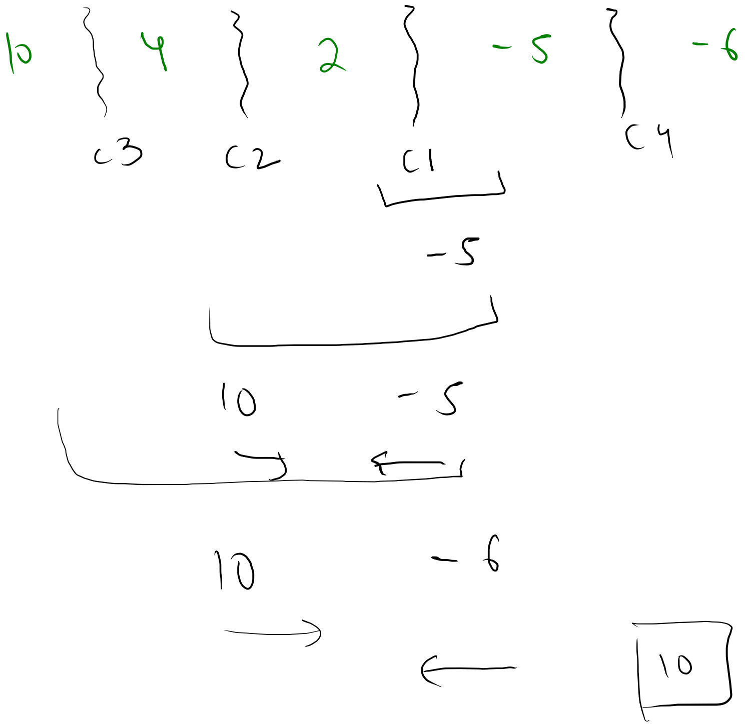




eg.1	2 →	2 →	⇒	2 2		
eg2.	10 →	← -5	⇒	10		
eg3.	2 →	← 2	⇒	[ ]		
* eg4.	2 →	1 →	=	2 1	+ +	No collision
eg 5.	-2	-2	⇒	-2 -2	- -	
eg 6.	-2	2	⇒	-2 2	- +	
eg4.	3	-2	⇒	3	+ -	collision.

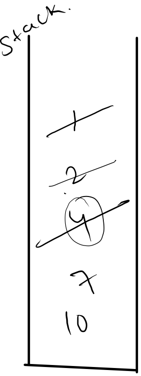
eg: n=5

+ − → collision



10	7	4	2	1	-2	-5	-6
0	1	2	3	4	5	6	7

i



curr = A[i]

curr = ~~-5~~ ~~6~~ 0

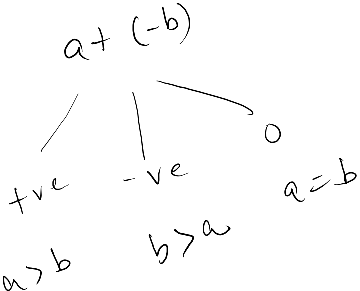
if curr<0
peek>0

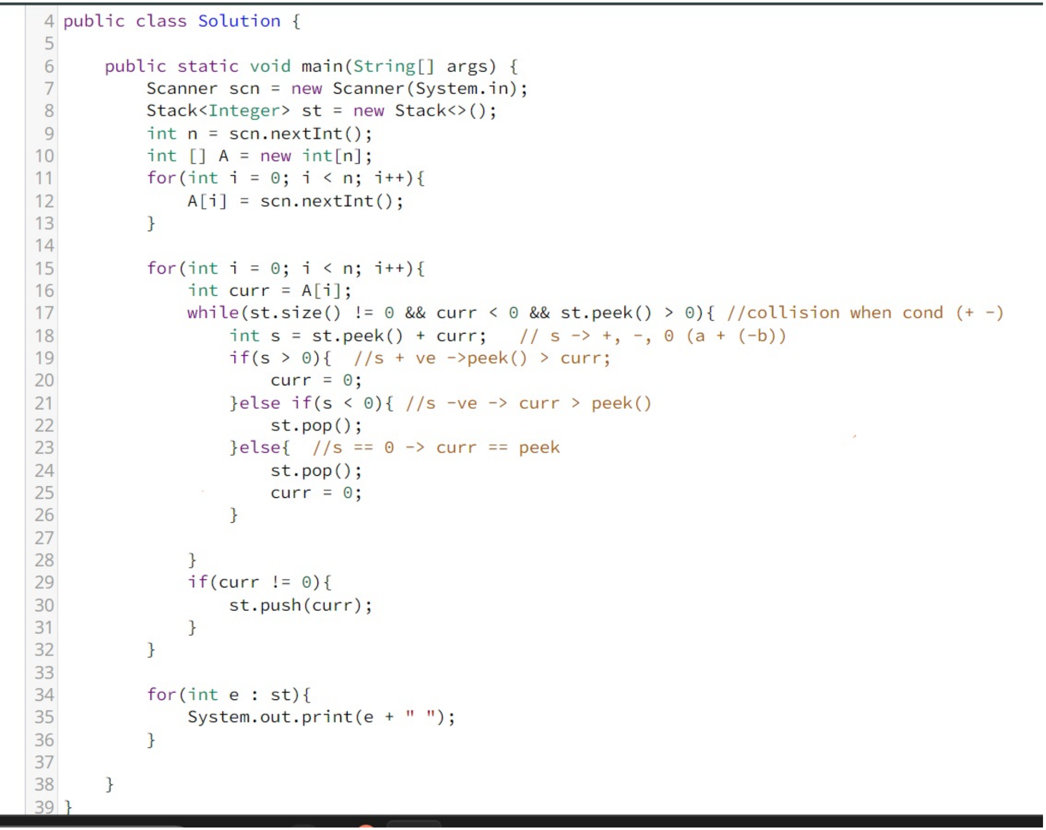
1. s = 1 - 2 = -ve → curr > peek
↳ st.pop()

2. s = 2 - 2 = 0 → curr == peek
↳ st.pop && curr=0

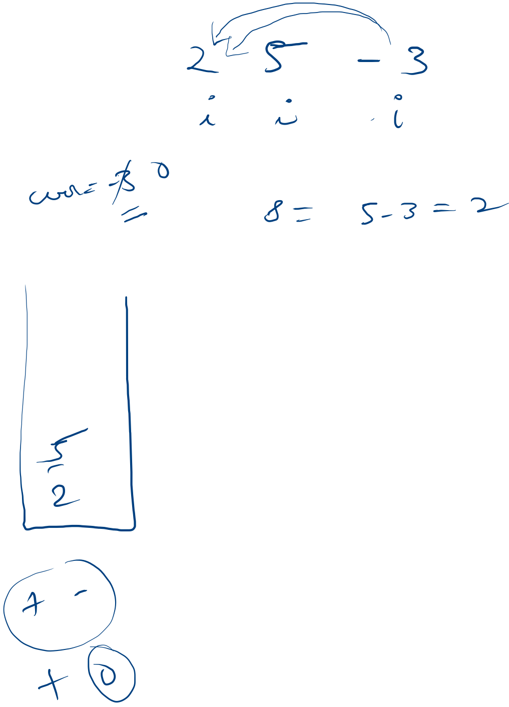
3. s = 7 - 5 = +ve
↳ * curr < peek.

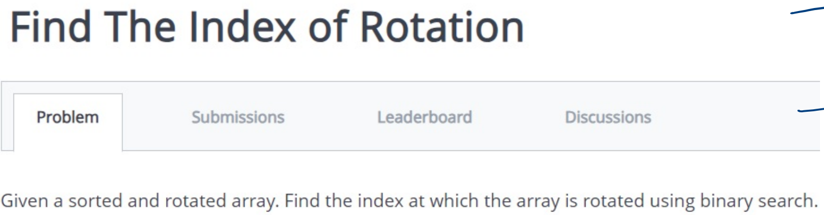
| 10 | 7 |





```java
public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        Stack<Integer> st = new Stack<>();
        int n = scn.nextInt();
        int [] A = new int[n];
        for(int i = 0; i < n; i++){
            A[i] = scn.nextInt();
        }

        for(int i = 0; i < n; i++){
            int curr = A[i];
            while(st.size() != 0 && curr < 0 && st.peek() > 0){ //collision when cond (+ -)
                int s = st.peek() + curr;   // s -> +, -, 0 (a + (-b))
                if(s > 0){  //s + ve ->peek() > curr;
                    curr = 0;
                }else if(s < 0){ //s -ve -> curr > peek()
                    st.pop();
                }else{  //s == 0 -> curr == peek
                    st.pop();
                    curr = 0;
                }

            }
            if(curr != 0){
                st.push(curr);
            }
        }

        for(int e : st){
            System.out.print(e + " ");
        }

    }
}
```





Find The Index of Rotation

Given a sorted and rotated array. Find the index at which the array is rotated using binary search.

→ ** min in rotated sorted array.

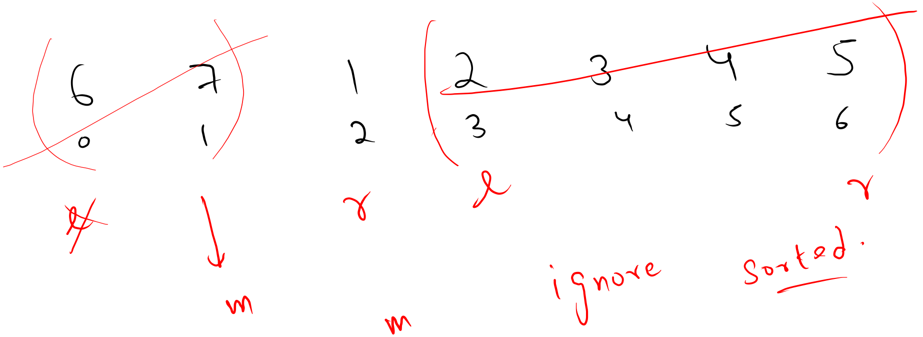
→ max in " " " "

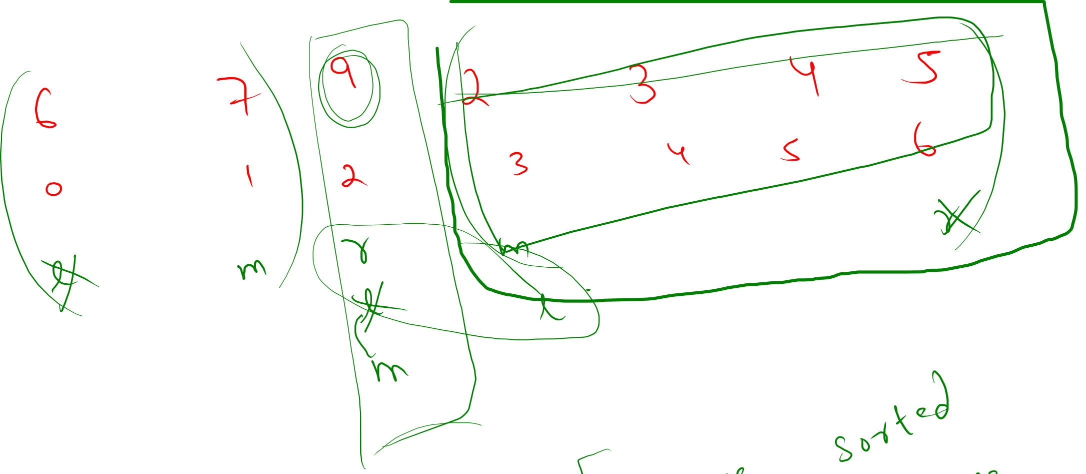
min = ~~∞~~ ~~7~~ 1

min Idx = ~~-1~~ ~~3~~ 2

idx-1

ans=1

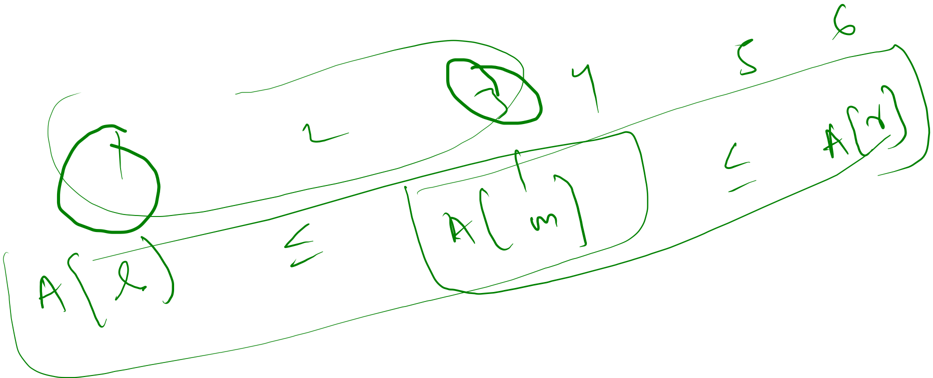


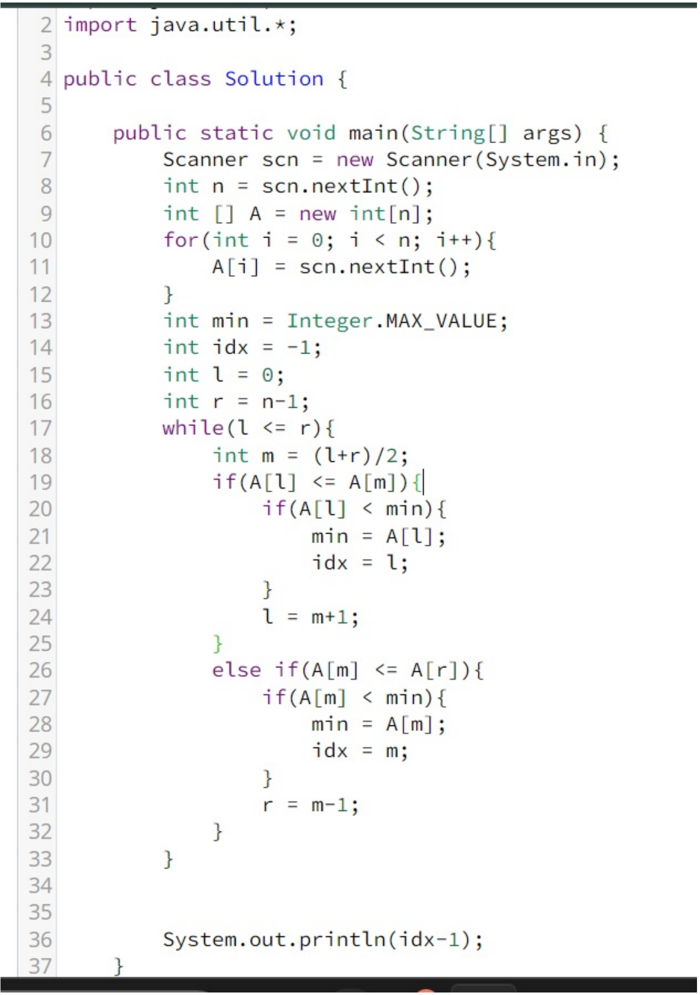


min = ~~∞~~ 2
↳ idx = ~~-1~~ 3

$A[m] \le A[r]$, ignore sorted before ignoring update min

ans = 2





```
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int [] A = new int[n];
        for(int i = 0; i < n; i++){
            A[i] = scn.nextInt();
        }
        int min = Integer.MAX_VALUE;
        int idx = -1;
        int l = 0;
        int r = n-1;
        while(l <= r){
            int m = (l+r)/2;
            if(A[l] <= A[m]){
                if(A[l] < min){
                    min = A[l];
                    idx = l;
                }
                l = m+1;
            }
            else if(A[m] <= A[r]){
                if(A[m] < min){
                    min = A[m];
                    idx = m;
                }
                r = m-1;
            }
        }


        System.out.println(idx-1);
    }
```