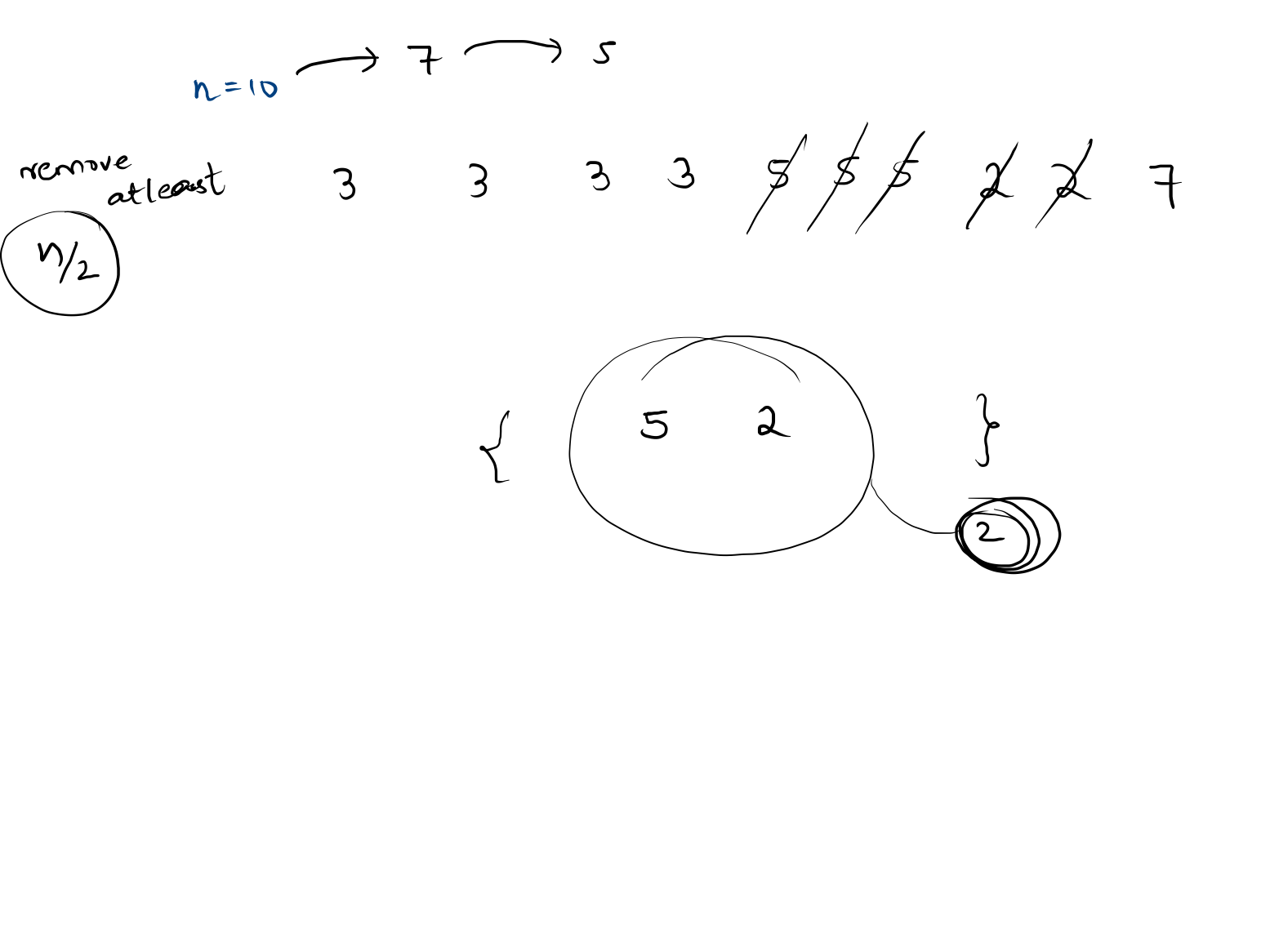

n=10
7
5
remove atleast
3 3 3 3
n/2
{
5 2
}
2

$n = 10$

3 3 3 3 5 5 5 2 2 7

3	4
5	3
2	2
7	1

ans = ~~0~~ ~~1~~ (2)

rem = ~~0~~ ~~4~~ (7)

atleast half

$4 < n/2$

✗ ~~$7 < n/2$~~

pq max

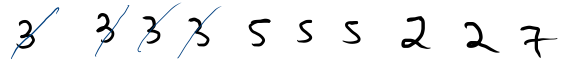
~~4~~

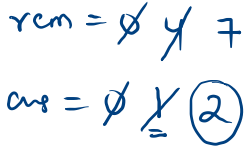
~~3~~

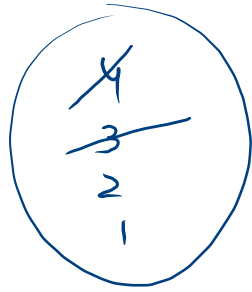
2

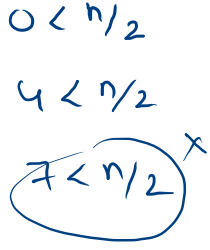
1

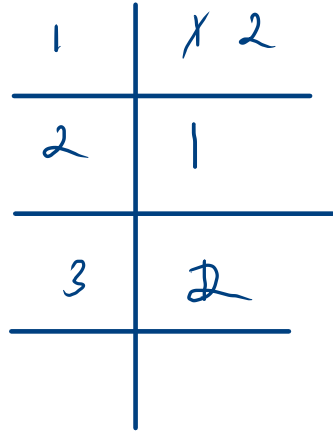
```java
import java.io.*;
import java.util.*;
public class Solution {
    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int []  A = new int[n];
        for(int i = 0; i < n; i++){
            A[i] = scn.nextInt();
        }
        HashMap<Integer, Integer> hm = new HashMap<>();
        for(int i = 0; i < n; i++){
            hm.put(A[i], hm.getOrDefault(A[i], 0)+1);
        }
        PriorityQueue<Integer> pq = new PriorityQueue<>(Collections.reverseOrder());
        for(int k : hm.keySet()){
            pq.add(hm.get(k));
        }
        int rem = 0;
        int ans = 0;
        while(rem < n/2){
            rem += pq.remove();
            ans++;
        }
        System.out.println(ans);
    }
}
```












```
for(int i = 0; i < n; i++){
    hm.put(A[i], hm.getOrDefault(A[i], 0)+1);
}
```

Maximum Product Subarray 2

Problem | Submissions | Leaderboard | Discussions

Sofia is a student who is struggling with her grades in math class. Her teacher has given her a list of **N** integers, and she needs to find the **contiguous subarray** with the largest **product**. Can you help Sofia find the solution to this problem?

Note: According to testcases answer will not overflow long

n = 6

2 3 -1 4 2 3

24

Case 1 all +ve.

1 2 3 4 5 prod.

Case 2 all -ve

-1 -2 -3

n → odd

→ even

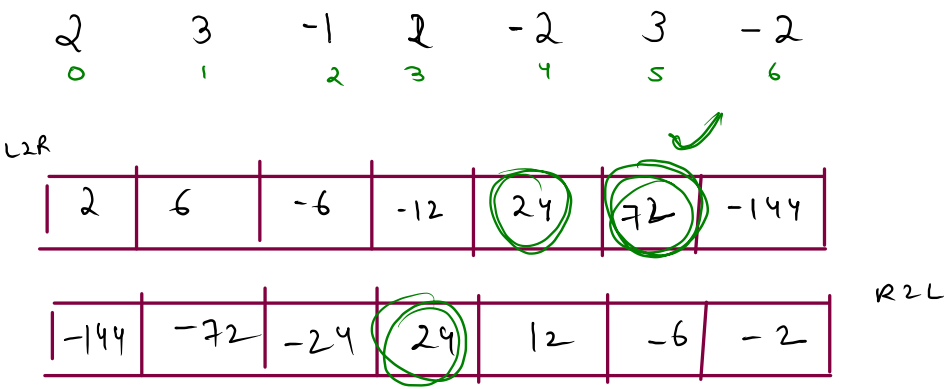
-1 -2 -3 -4 ... prod

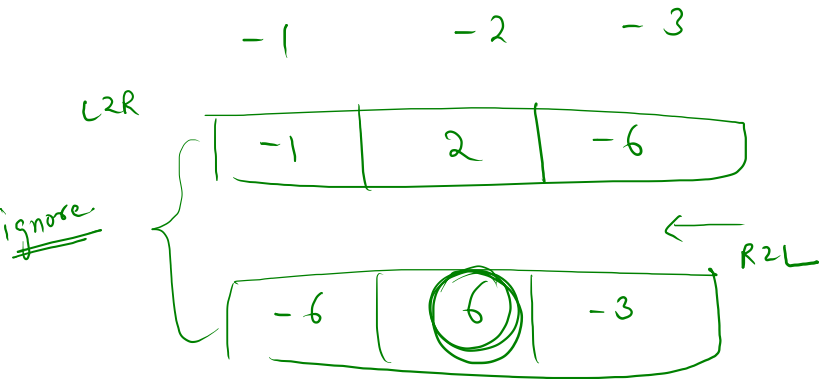
mix.

-ve → even

+ve → odd.

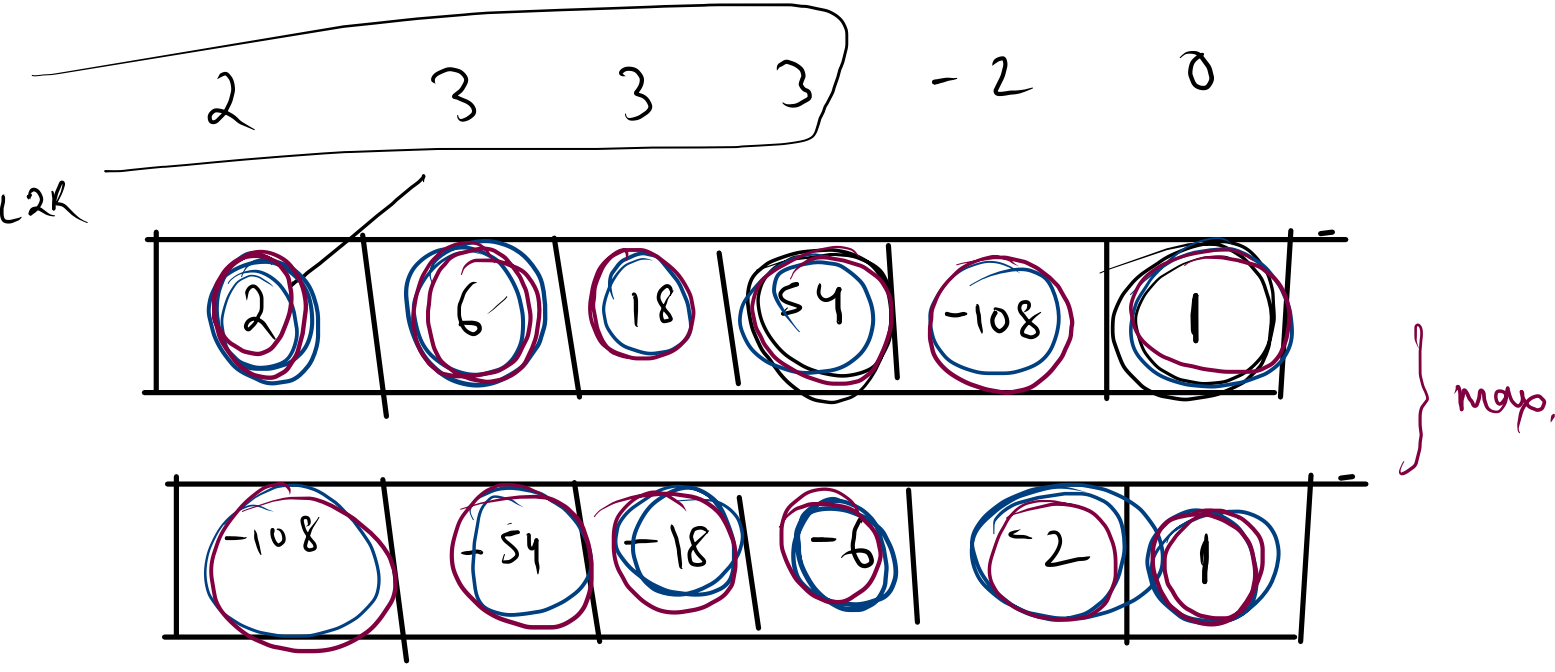
1 2 -3 -4 5 6

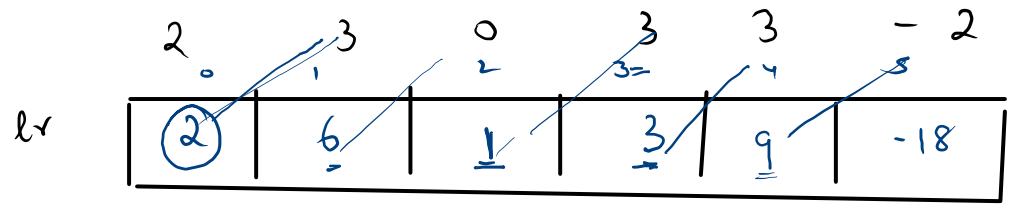

2 3 -1 2 -2 3 -2
0 1 2 3 4 5 6
L2R
2 6 -6 -12 24 72 -144
R2L
-144 -72 -24 24 12 -6 -2


-1 -2 -3
L2R
-1 2 -6
ignore
R2L
-6 6 -3

ma

2	3	0	3	3	-2
2	6	1	3	9	-18
6	3	1	-18	-6	-2

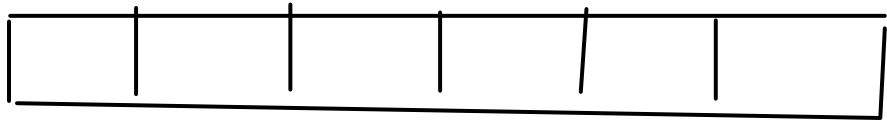

2 3 3 3 -2 0
L2R
2 6 18 54 -108 1
-108 -54 -18 -6 -2 1
map.




```
16
17      int ans = Integer.MIN_VALUE;
18
19      for(int i = 0; i < n; i++){
20          if(i == 0){
21              lr[i] = A[i];
22          }
23          else{
24              lr[i] = lr[i-1] * A[i];
25          }
26          if(lr[i] == 0){
27              lr[i] = 1;
28          }
29
30          ans = Math.max(ans, lr[i]);
31      }
```

i = 0 1 2 3 4 5

rl

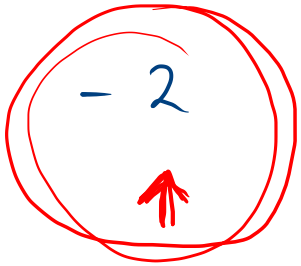


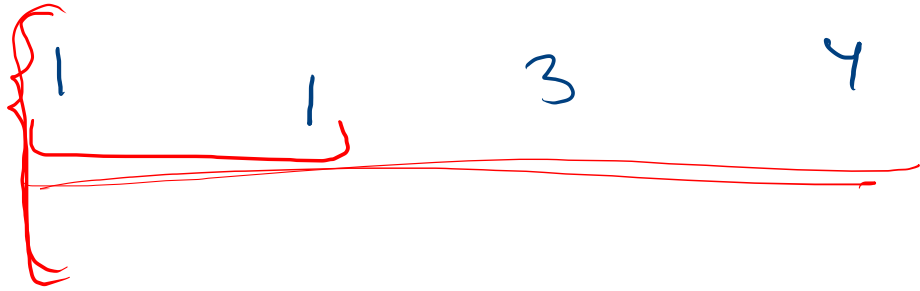
```java
import java.io.*;
import java.util.*;
public class Solution {
    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int [] A = new int[n];
        for(int i = 0; i < n; i++){
            A[i] = scn.nextInt();
        }
        int [] lr = new int[n];
        int [] rl = new int[n];
        int ans = Integer.MIN_VALUE;
        for(int i = 0; i < n; i++){
            if(i == 0){
                lr[i] = A[i];
            }
            else{
                lr[i] = lr[i-1] * A[i];
            }
            if(lr[i] == 0){
                lr[i] = 1;
            }
            ans = Math.max(ans, lr[i]);
        }
        for(int i = n-1; i>=0; i--){
            if(i == n-1){
                rl[i] = A[i];
            }
        }
        for(int i = n-1; i>=0; i--){
            if(i == n-1){
                rl[i] = A[i];
            }
            else{
                rl[i] = rl[i+1] * A[i];
            }
            if(rl[i] == 0){
                rl[i] = 1;
            }
            ans = Math.max(ans, rl[i]);
        }
        System.out.println(ans);
    }
}
```

```java
import java.io.*;
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 ans = Integer.MIN_VALUE;
        int l = 1;
        int r = 1;
        for(int i = 0; i < n; i++){
            if(l==0)
                l = 1;
            if(r==0)
                r = 1;
            l *= A[i];
            r *= A[n-i-1];
            ans = Math.max(ans, Math.max(l,r));
        }
        System.out.println(ans);
    }
}
```

-2 3 1 1 4.

sort





tar = 0

0 - (-2)

= 2

tar = 2

```
6
-2 0 2 4 -2 -8
```

Sample Output 0

```
-2 -2 4
-2 0 2
```