Find The Index of Rotation

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

* smallest no. idx in RS array.
* largest " " " " " " "

(5)

ans = 1 (idx).

eg.

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

idx = -1 ~~3~~ 2

ignore sorted part

but first store potential answer m

i j

6 < 2

m → j sorted.

1 < 3

ans = 3

smallest idx

6	7	9	2	3	4	5
0	1	2	3	4	5	6

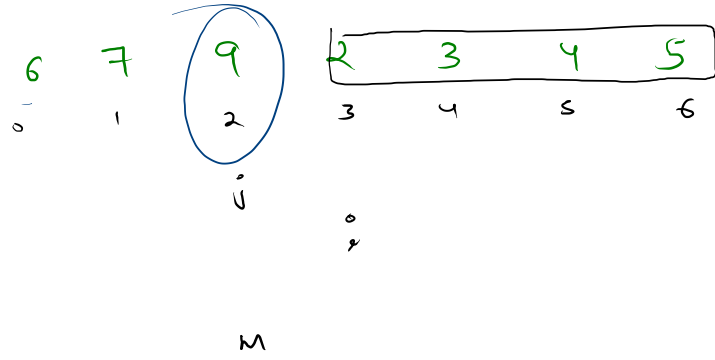
j

i

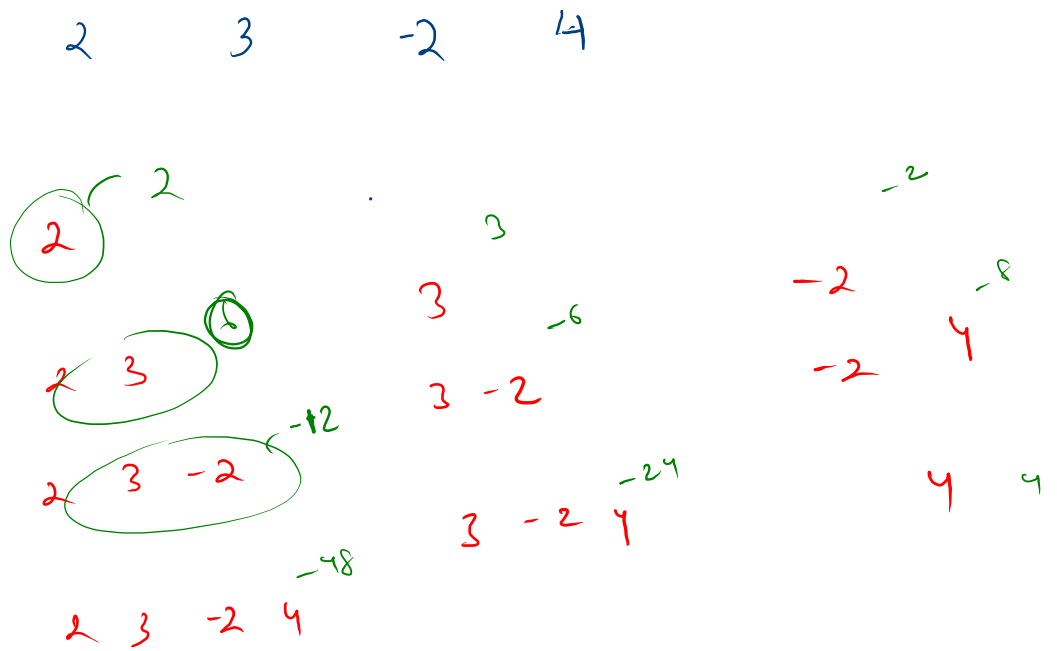
m

9 < 2

```java
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 i = 0;
        int j = n-1;
        int idx = -1;         //ans
        int min = Integer.MAX_VALUE;
        while(i<=j){
            int m = (i+j)/2;
            if(A[i] <= A[m]){        //left sorted
                if(A[i] < min){
                    min = A[i];
                    idx = i;
                }
                i = m + 1;
            }
            else if(A[m] <= A[j]){    //right sorted
                if(A[m] < min){
                    min = A[m];
                    idx = m;
                }
                j = m - 1;
            }
        }
        System.out.println(idx-1);
    }
}
```



Maximum Product Subarray 2



curr max = (A[i], prev min * A[i], prev max * A[i])

2	3	-2	-12	4
0	1	2	3	4

max = ~~2~~ ~~6~~ 144 576

prev max = ~~2~~ ~~6~~ -2 144 576

prev min = ~~2~~ ~~3~~ -12 -12 -48

currmax = 576

currmin = -48

4, -48, 576

```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 max = A[0];
        int prevMax = A[0];
        int prevMin = A[0];

        for(int i = 1; i < n; i++){
            int currMax = Math.max(A[i], Math.max(A[i] * prevMax, A[i] * prevMin));
            int currMin = Math.min(A[i], Math.min(A[i] * prevMax, A[i] * prevMin));
            max = Math.max(currMax, max);
            prevMax = currMax;
            prevMin = currMin;
        }
        System.out.println(max);
    }
}
```