07 Asteroid Collision

21 February 2025 16:25

735. Asteroid Collision

Solved

Medium | Topics | Companies | Hint

We are given an array `asteroids` of integers representing asteroids in a row. The indices of the asteriod in the array represent their relative position in space.

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.

Example 1:

```
Input: asteroids = [5,10,-5]
Output: [5,10]
Explanation: The 10 and -5 collide resulting in 10. The 5 and 10 never collide.
```

Example 2:

```
Input: asteroids = [8,-8]
Output: []
Explanation: The 8 and -8 collide exploding each other.
```

Example 3:

```
Input: asteroids = [10,2,-5]
Output: [10]
Explanation: The 2 and -5 collide resulting in -5. The 10 and -5 collide resulting in 10.
```

this example help to understand the example.

Example - {10, 2, -5}

10 → 2 → , -5 ←

10 → ← -5

10 → Final

{-2, -1, 1, 2}

move left side | move right side.

Wrong Answer 58 / 275 testcases passed

submitted at Feb 21, 2025 14:21

Input

asteroids =

[-2,-1,1,2]

Output

[]

Expected

[-2,-1,1,2]

How to know to you the concept of (Stack) because of LIFO { 2 thing Matter

- order of Asteroide
- Direction of Ast.

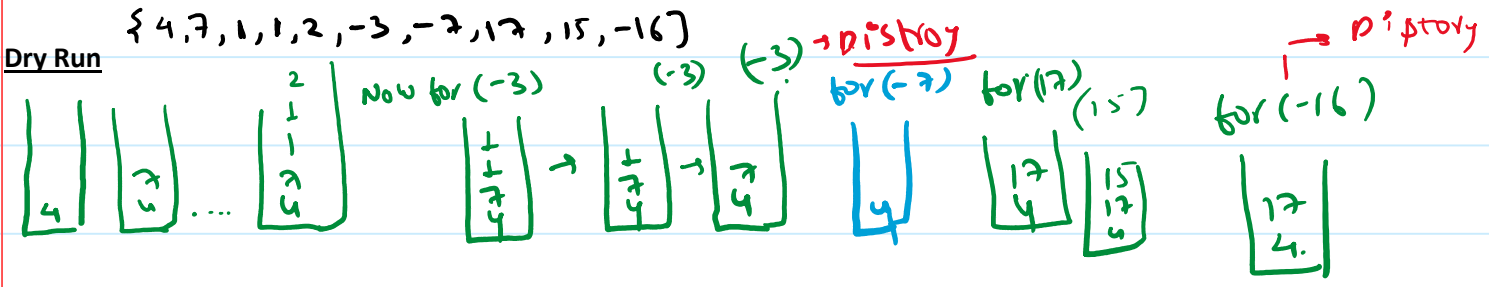
```
public int[] asteroidCollision(int[] asteroids) {
        Stack<Integer> stack = new Stack<>();

        for (int a : asteroids) {
            if (a > 0) {
                stack.push(a);
            } else {
                while (!stack.isEmpty() && stack.peek() > 0 && stack.peek() < -a) {
                    stack.pop();
                }
                if (!stack.isEmpty() && stack.peek() == -a) {
                    stack.pop();
                }
                 else if (stack.isEmpty() || stack.peek() < 0)  {
                    stack.push(a);
                }
            }
        }
        int[] res = new int[stack.size()];
        int i = stack.size() - 1;
        while(!stack.isEmpty()) {
            res[i--] = stack.pop();
        }
        return res;
    }
```

Tc-O(n) …

iterate over the loop … + O(N) … For inserting into array … form stack

SC -- O(2N) (stack & array)

Dry Run

{4, 7, 1, 1, 2, -3, -7, 17, 15, -16}