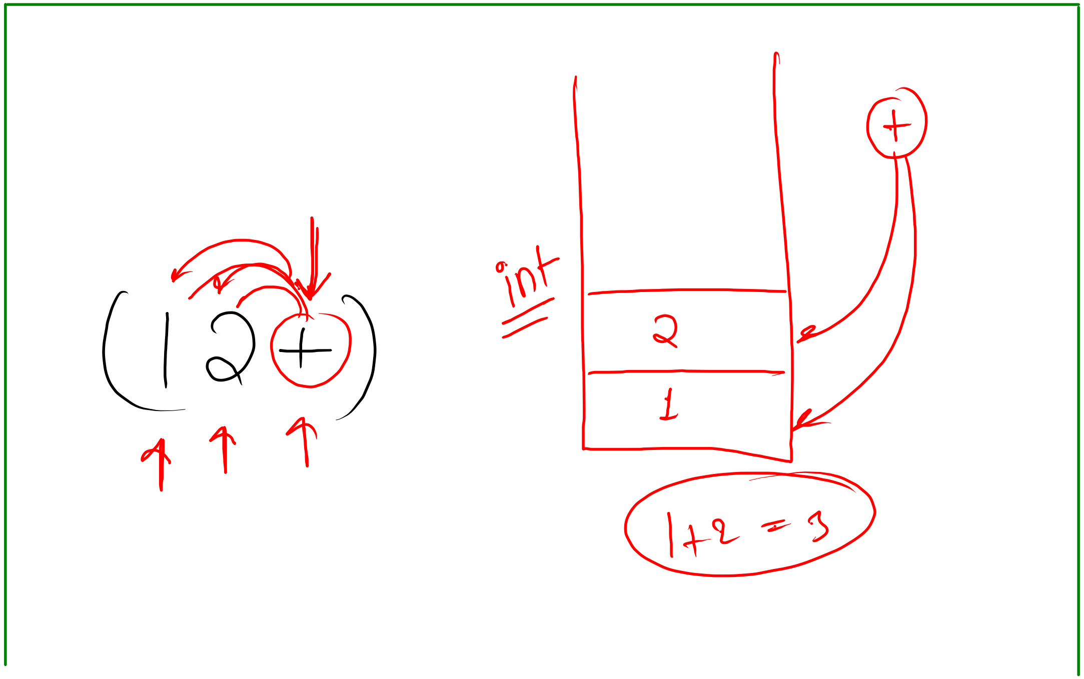
Postfix expression calculation

Infix exp:– $((4+5)*(7-6))$, $((2/3)*((7+4)-(3-2)))$

prefix exp:– $*+45\ -76$, $*/23\ -+74\ -32$

postfix exp:– $45+76-*$, $23/\ 74+32--*$



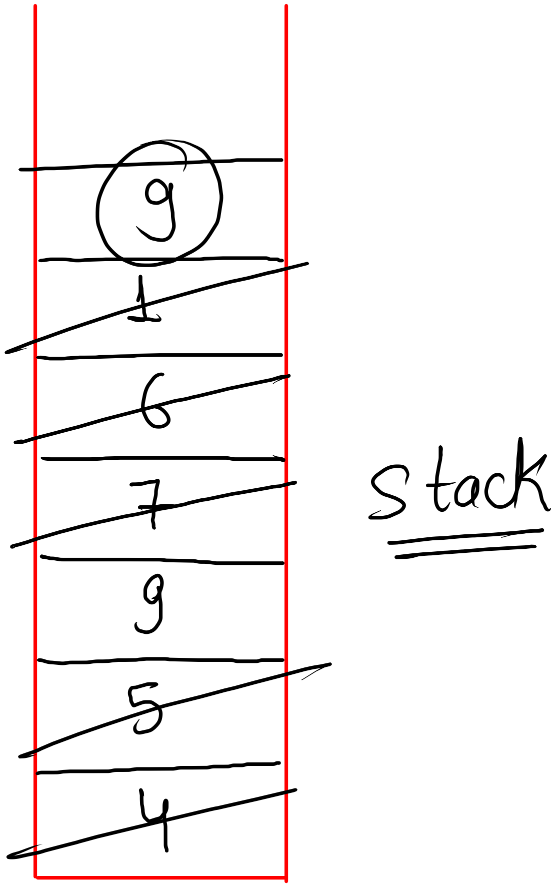
ex1

$$\underbrace{\overset{\downarrow}{4}\;\overset{\downarrow}{5}\;\overset{\downarrow}{+}}\;\underbrace{\overset{\downarrow}{7}\;\overset{\downarrow}{6}\;\overset{\downarrow}{-}}\;\overset{\downarrow}{*}$$

```
if( number)
        push
else
    top1 = ~~5~~ ~~6~~  1
    top2 = ~~4~~ ~~7~~  9
```

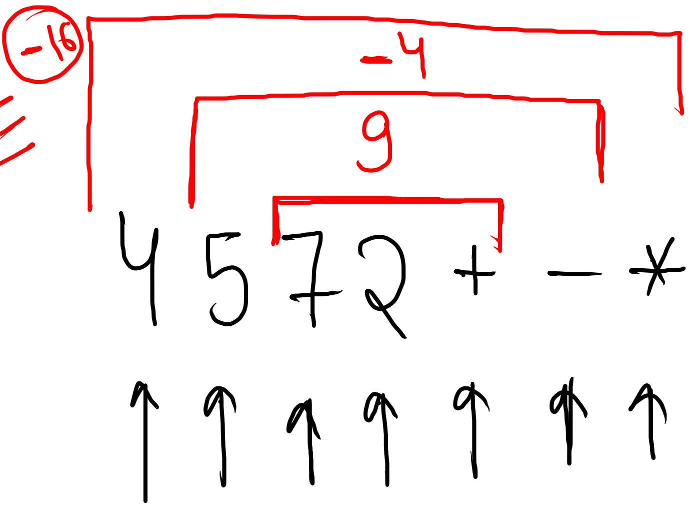
$\underline{\underline{top2 - top1}}$

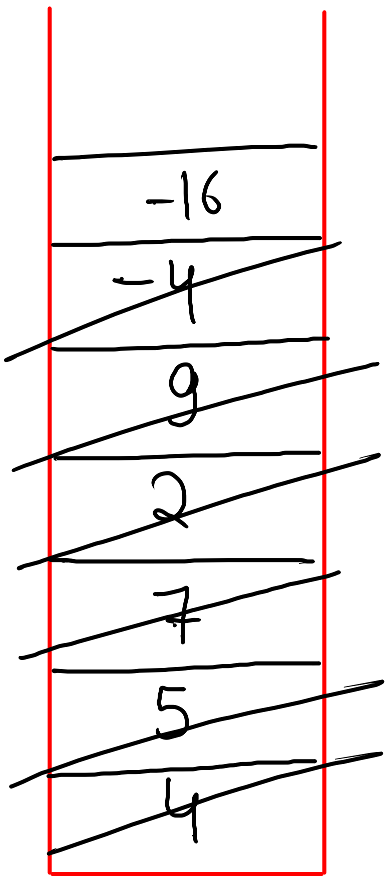
$top2 * top1$



ans = 9 ←
ans = 1 ←
ans = 9

Ex2





top1 = ~~2~~ ~~9~~ -4

top2 = ~~7~~ ~~5~~ 4

ans = 7+2 = 9

ans = 5-9 = -4

ans = 4*(-4) = -16

psudo code

```
1) declare stack
2) traverse in string
    2.1) if number
            push
    2.2) else
            top1 , top2
            calculate
            push back ans
3) return top element
```

```
+   top2 + top1
-   top2 - top1
*   top2 * top1
/   top2 / top1
```

Code

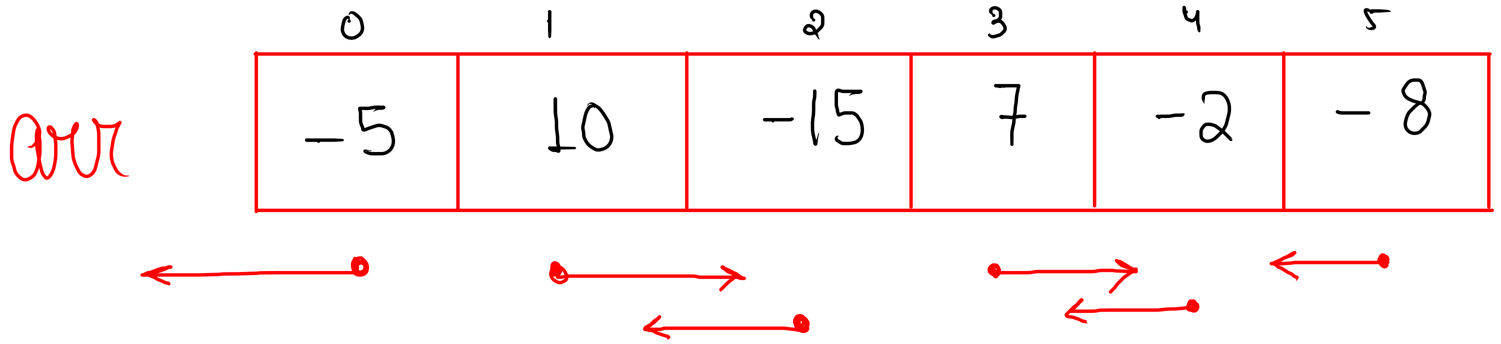
```java
public static int postfixExp(String str) {
    Stack<Integer> st = new Stack<>();
    for (int i = 0; i < str.length(); i++) {
        char ch = str.charAt(i);
        if ( Character.isDigit(ch) ) {
            st.push( (ch - '0') );
        } else {
            int top1 = st.pop();
            int top2 = st.pop();
            int ans = 0;
            if ( ch == '+' ) {
                ans = top2 + top1;
            } else if ( ch == '-' ) {
                ans = top2 - top1;
            } else if ( ch == '*' ) {
                ans = top2 * top1;
            } else {
                ans = top2 / top1;
            }
            st.push( ans );
        }
    }
    return st.peek();
}
```

Note

Imp

always use top2 first and then top1

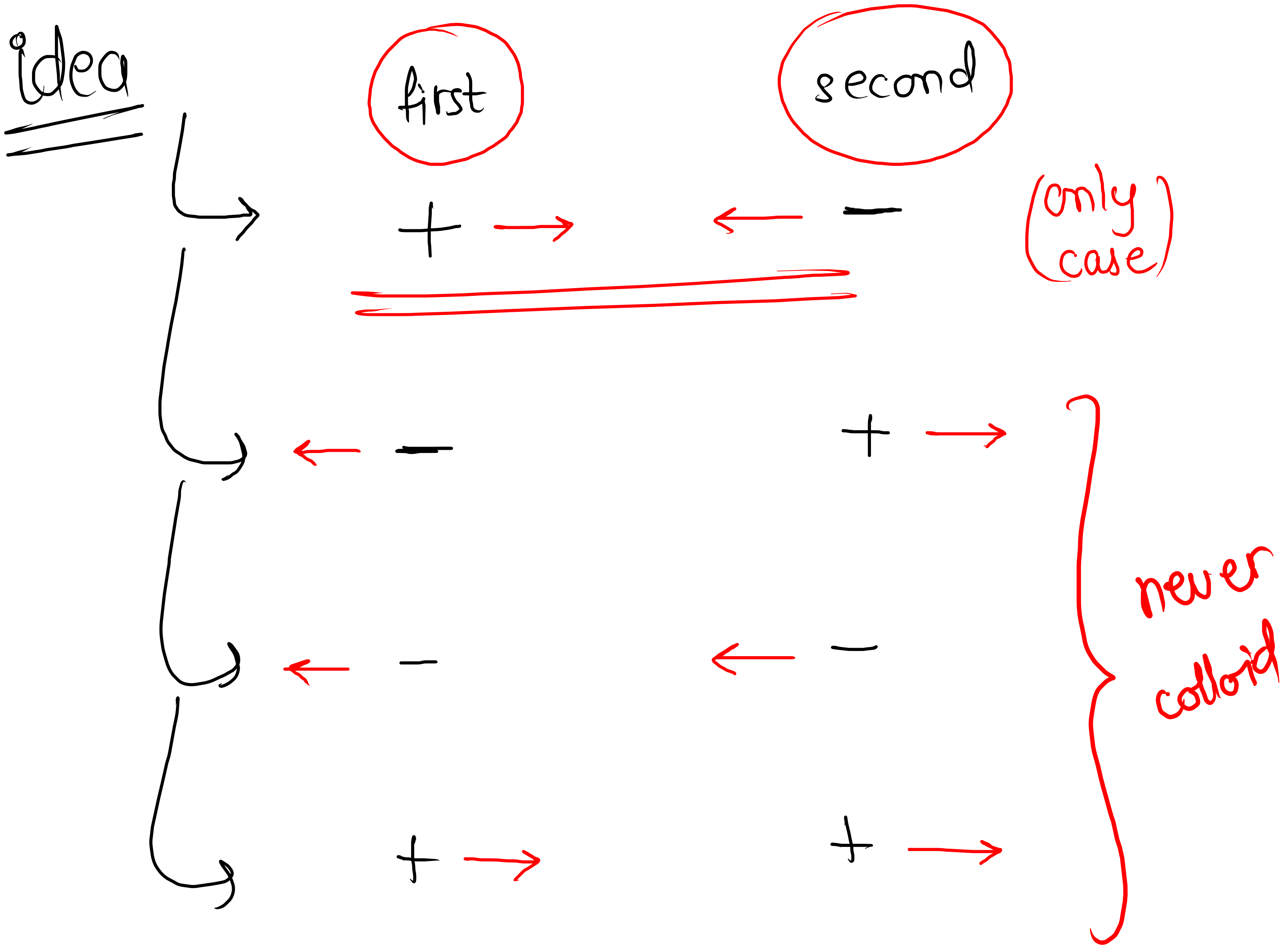
Asteroid Collision



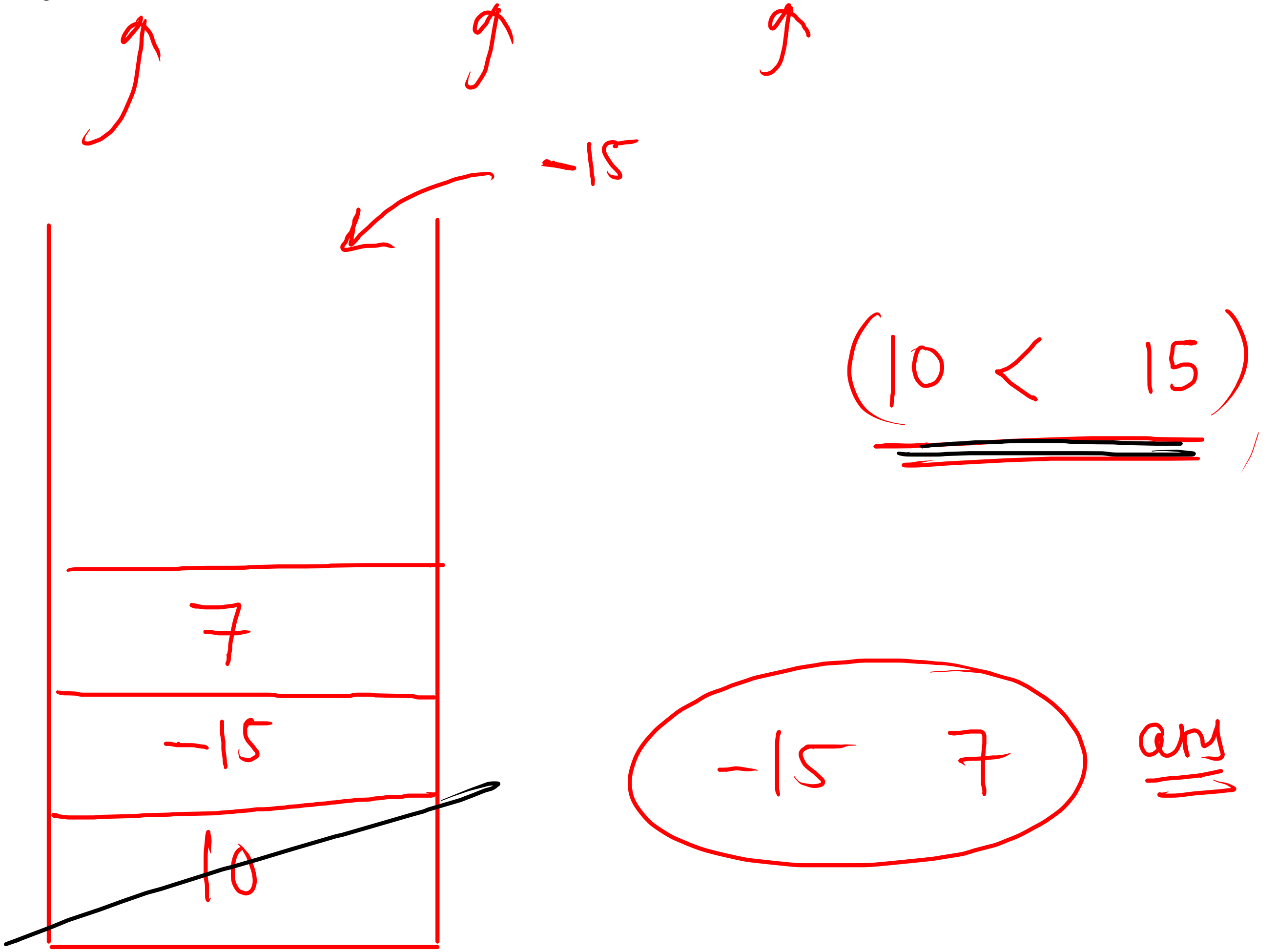
Note:- ↳ absolute value represents size

↳ + direction means going to right side

↳ − direction means going to left - side idea



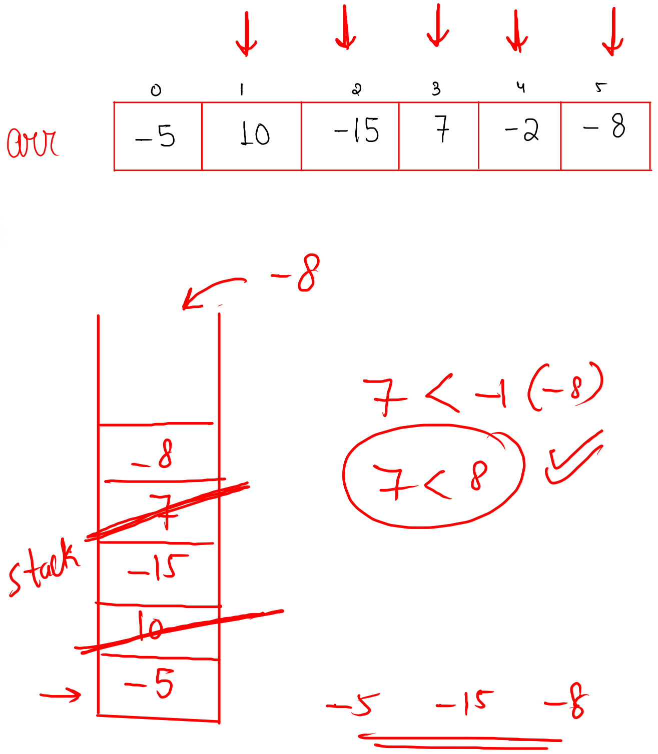
arr [10 -15 7]



T.C = O(N)   S.C = O(N)

```java
public static void astroidCollision(int[] arr, int n) {
    Stack<Integer> st = new Stack<>();
    for (int i = 0; i < n; i++) {
        if ( arr[i] > 0 ) {
            st.push( arr[i] );
        } else {
            while ( !st.isEmpty() && st.peek() > 0 && st.peek() < -1 * arr[i] ) {
                st.pop();
            }
            if ( !st.isEmpty() && st.peek() == -1 * arr[i] ) {
                st.pop();
            } else if ( st.isEmpty() || st.peek() < 0 ) {
                st.push( arr[i] );
            }
        }
    }
    ArrayList<Integer> ans = new ArrayList<>();
    while ( st.size() > 0 ) {
        int ele = st.pop();
        ans.add( 0, ele );
    }

    for (int i : ans) {
        System.out.print(i + " ");
    }
}
```



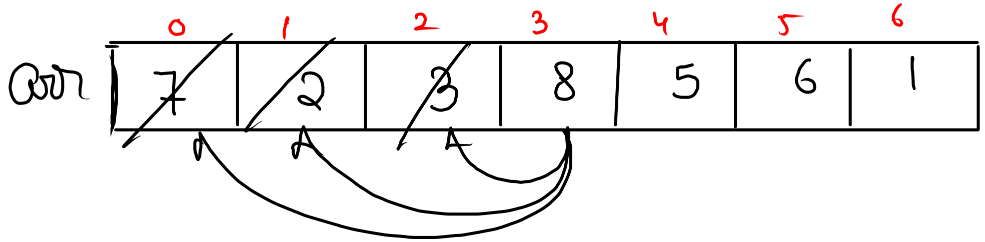
ans = -5, -15, -8

⇒ find next greater element on left

	0	1	2	3	4	5	6
arr	7	2	3	8	5	6	1
ans	-1	7	7	-1	8	8	6

brute force

$O(N^2)$



psudo code

```
1) declare stack
2) traverse in array
    2.1) while ( top <= curr )
                pop()

    2.2) now top element is my answer
            ans[i] = st.top()
            ans[i] = -1 ;
```