Postfix expression calculation

Theory

- Infix expression :- ((4+5)*(7-3)) // a+b
- Prefix expression :- *+45-73 // +ab
- Postfix expression :- 45+73-* // ab+

⇒ ((4+5)*(7-3))

⇒ ((45+)*(7-3))

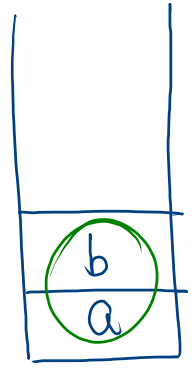
⇒ ((45+)*(73-))

⇒ 45+73-*

idea

$\Rightarrow (a+b) =$

$a\ b\ +$



why stack

Note:- top element of stack will always represent the previous potential element to make pair.

Imp

dry run

str = "4 5 + 7 3 − *" ;

0 1 2 3 4 5 6

i

curr = '+', '−', '*'

num1 = st.pop() = ~~3~~ 4

num2 = st.pop() = ~~7~~ 9

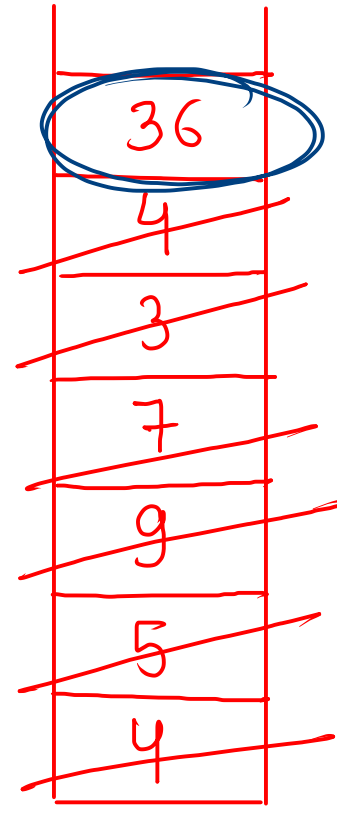
ans = num2 + num1

ans = num2 − num1

ans = num2 * num1

push ans back in stack

gmp



stack

(character)

(integer)

psudo code)

1) declare stack

2) traverse in string

2.1) check char if digit then blindly push

2.2) else if char is operator

ch == + , - , * , /

num1 = st.pop()

num2 = st.pop()

ans = num2 ch num1

imp

push ans back in stack

code

```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    String str = scn.nextLine();
    int ans = evaludatePostfixExp(str);
    System.out.println(ans);
}
public static int evaludatePostfixExp(String str) {
    Stack<Integer> st = new Stack<>();
    for (int i = 0; i < str.length(); i++) {
        char curr = str.charAt(i);
        if ( curr >= '0' && curr <= '9' ) { // Character.isDigit(curr)
            st.push( curr - '0' );
        } else {
            int num1 = st.pop();
            int num2 = st.pop();
            int ans = 0;
            if ( curr == '+' ) {
                ans = num2 + num1;
            } else if ( curr == '-' ) {
                ans = num2 - num1;
            } else if ( curr == '*' ) {
                ans = num2 * num1;
            } else if ( curr == '/' ) {
                ans = num2 / num1;
            }
            st.push(ans);
        }
    }
    return st.peek();
}
```

T.C = O(n)

n = str.len

Asteroid Collision (Most Imp)

arr =

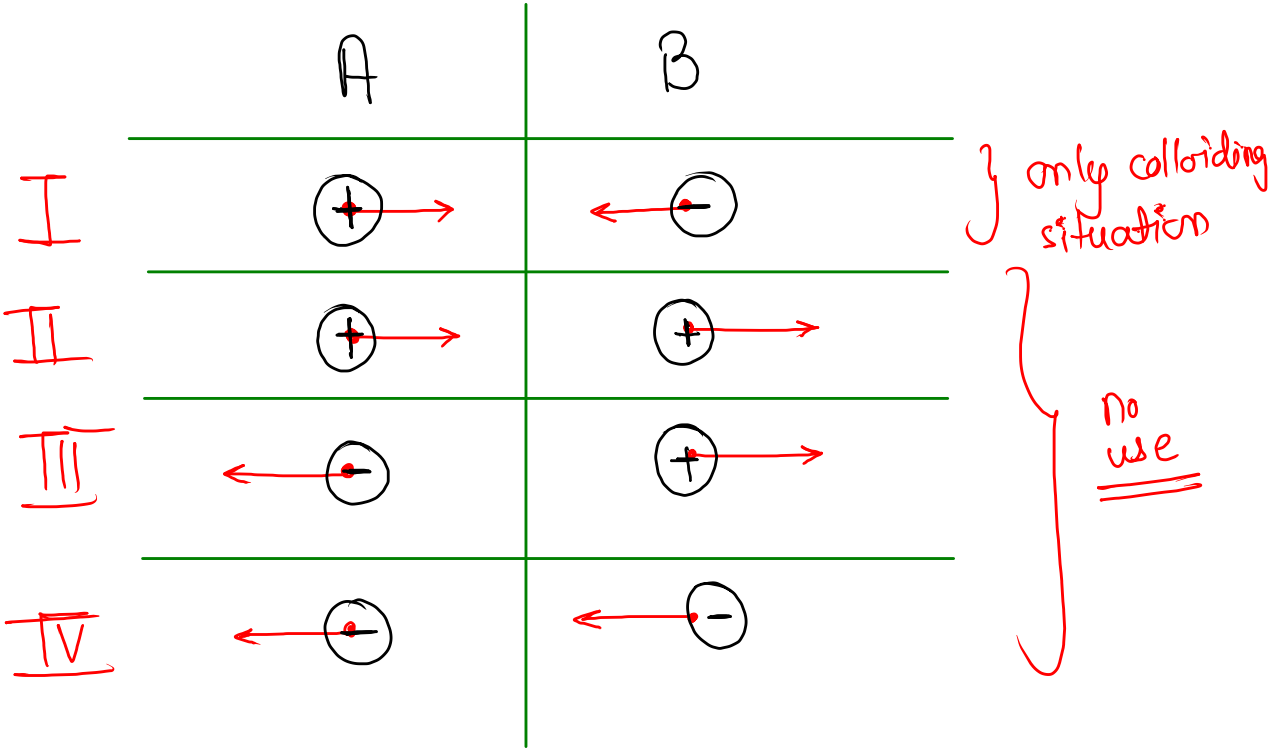
0	1	2	3	4	5
-5	10	-15	7	-2	-8

ans = [-5, -15, -8]

observations

- magnitude will represent size
- +ve means right side & -ve means left

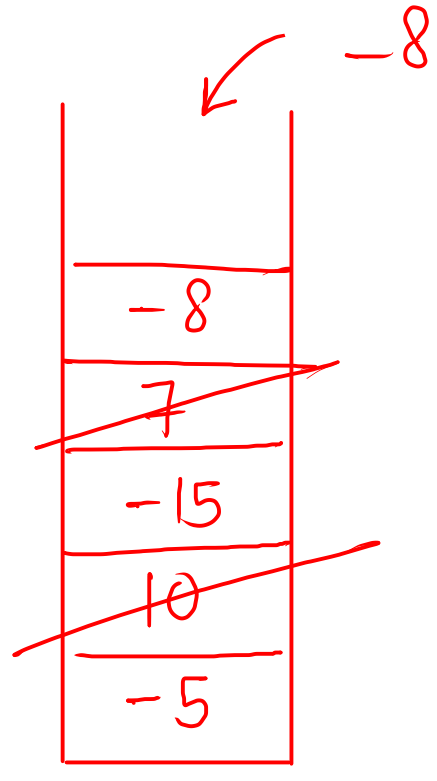
observations



Ex 0

arr =

0	1	2	3	4	5
-5	10	-15	7	-2	-8



arr = [-5 -15 -8]

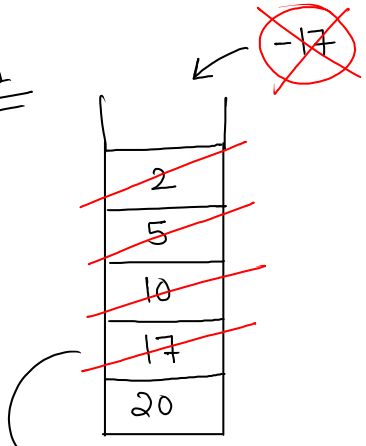
psudo code

1) declare stack
2) traverse in array
   2.1) if curr is +ve
        then push
   2.2) else (evaluate)
        while(curr < 0 && peek > 0 && |peek| < |curr|)
        pop

Ex 1

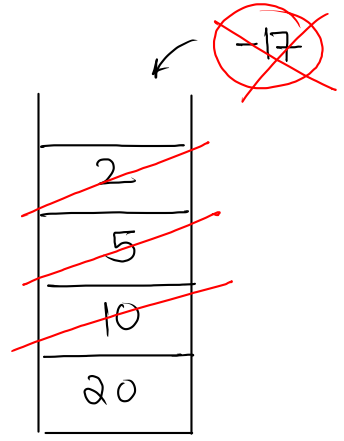
$$arr = [\overset{0}{5}, \overset{1}{\cancel{3}}, \overset{2}{\cancel{0}}, \overset{3}{\cancel{1}}, \overset{4}{-4}]$$

Ex1



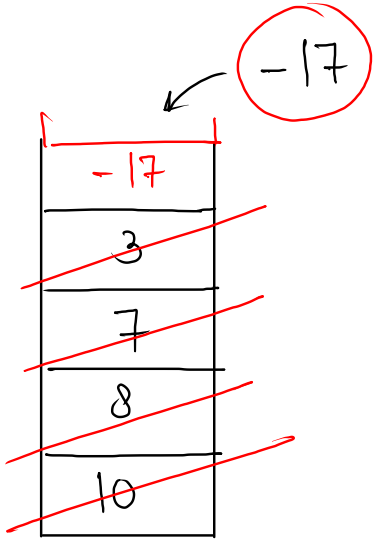
pop 17 and do not push -17

Ex2



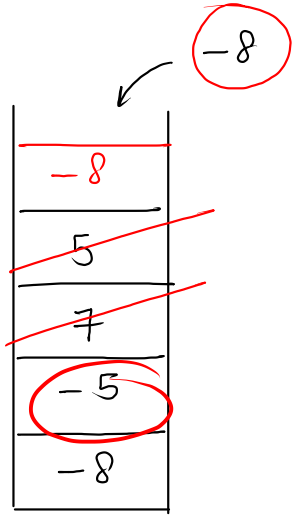
if curr small then just do not push it in stack

Ex3



if stack became empty then push curr ele in stack

Ex 4



if top element is -ve then push curr ele in stack

Code

```java
public static ArrayList<Integer> 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 {
            // important
            // we are coming here for current element -ve
            while ( st.size() > 0 && st.peek() > 0 && st.peek() < -1 * arr[i] ) {
                st.pop();
            }
            if ( st.size() > 0 && st.peek() == -1 * arr[i] ) {
                st.pop();
            } else if ( st.size() == 0 || st.peek() < 0 ) {
                st.push( arr[i] );
            }
        }
    }

    ArrayList<Integer> ans = new ArrayList<>();
    while (st.size() > 0) {
        int val = st.pop();
        ans.add(0, val);
    }
    return ans;
}
```