04.10.2024

n = total # of symbols / input numbers

$\boxed{n = 11}$

: Calculator :

$8 + 3 * \underbrace{5 \wedge 2} - 9 * 67 = ?$

$+, -, *, \backslash, \wedge$

power.

$\wedge >$ BDMAS.

$= 8 + \boxed{3 * 25} - 9 * 67$

$= 8 + 75 - 9 * 67$

$\Theta(n) \; \Theta(n)$

$= 8 + 75 - 603$

$\Theta(n)$

$= 83 - 603$

$= -520$ ..

Use: 2 stacks:

- numbers
- operators.

+

−

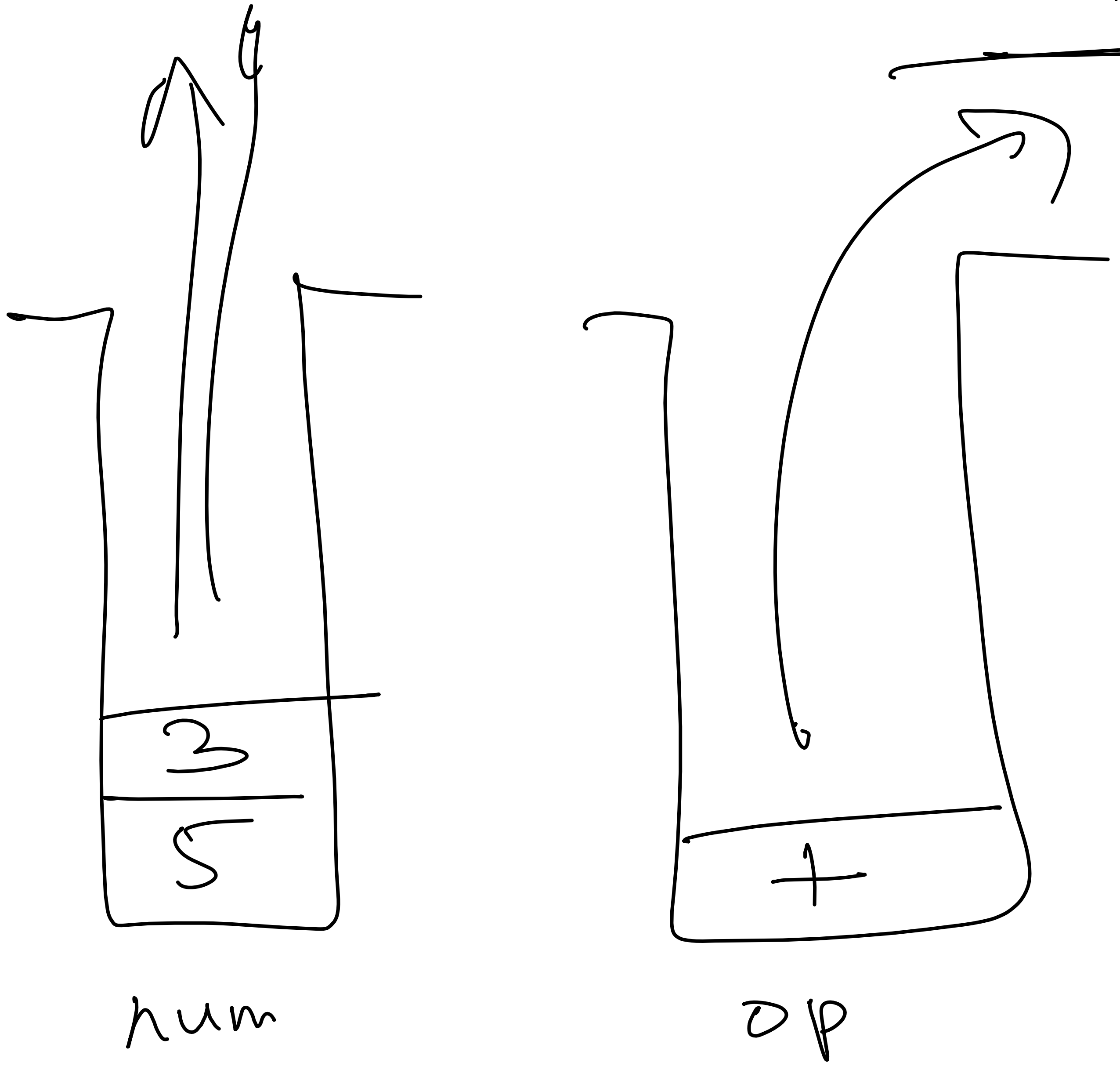
\

*

a \ b

(~~no~~ assumptions: ✓ no ( )

✓ $2 \wedge 3 \wedge 5$)

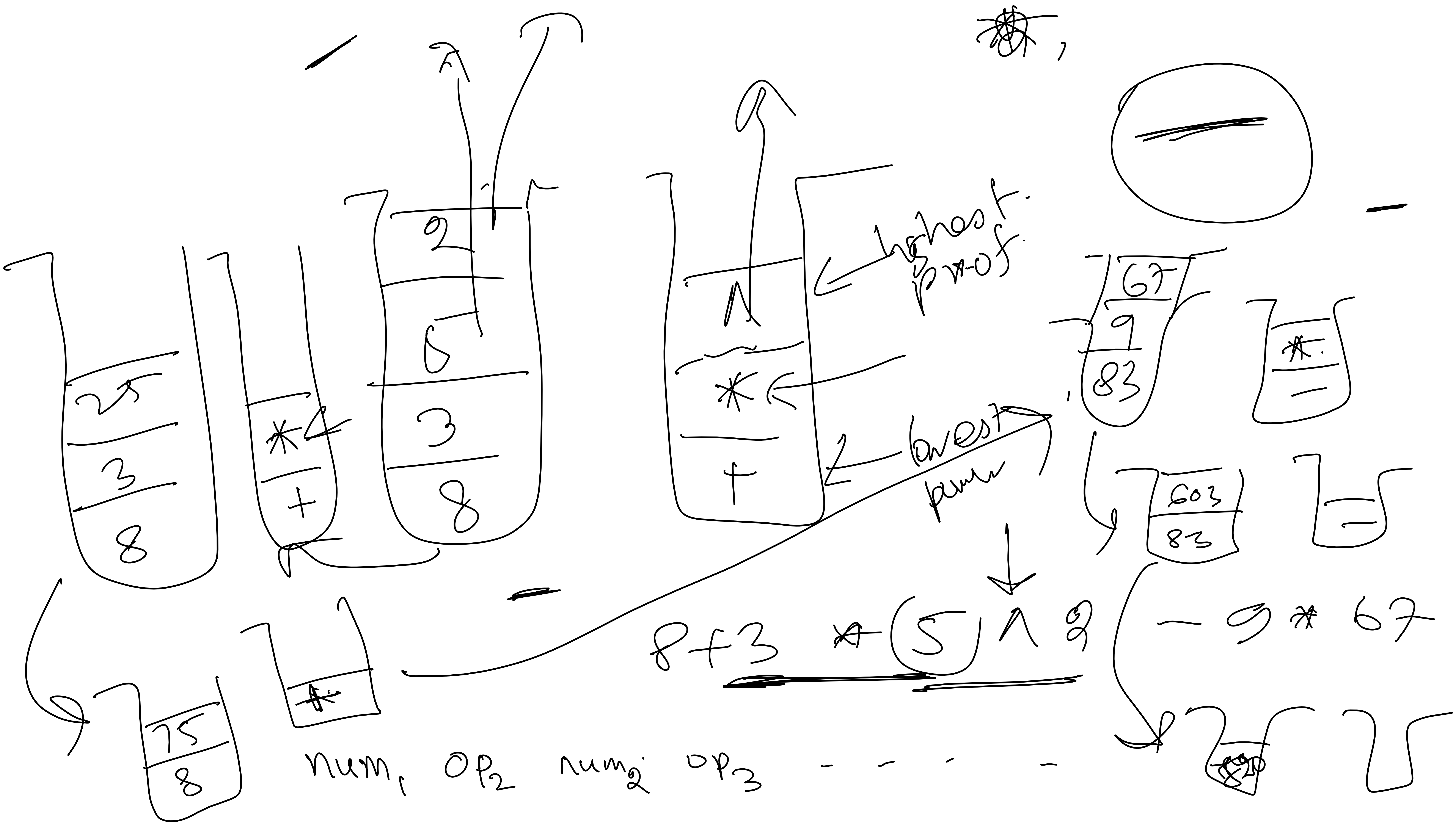


~~(5 + 3)~~

5 $\underline{+}$ 3

```
Procedure     PopN Operate
        // numStack, opStack .
        operator ← op. Stack. pop ( )
        number1 ← numStack. pop ( )
        number 2 ← numStack. pop ( )

        Operate ( number1 , number 2, operator ) .
```



25
3
8
*
+
2
5
3
8
^
*
+
highest prec.
lowest prec.
67
9
83
*
-
603
83
-
75
8
*
8+3 * (5 ^ 2) - 9 * 67
num1 op2 num2 op3 - - - -
-520

lesson: popNoperate whenever next symbol is less than the opStack.top symbol.

<

Priority:

+

−

\

*

(^)

!

```
Procedure    Calculator. (Input X)
    // X. next() gives me. ~~next~~ op/num.
    opstack ← empty stack, numStack ← empty stack.
    while ( X. next () ≠ NULL).
        if ( x. next() is an operator)
            if (outpriority ( X.next()) < inpriority (opStack. peek()) ) ).
                popNOperate. ( ).
            else
                ~~pu~~ OpStack. push ( X. next()).
        else if (x. next() is a number)
```

```
            numStack. push (~~s.~~ next ( )).
        else.
    end while.
    while ( opStack is  not  empty )
            popNoperate ( ).

    return   Numstack. top ( ).
```

Parenthesis ( )

Idea:

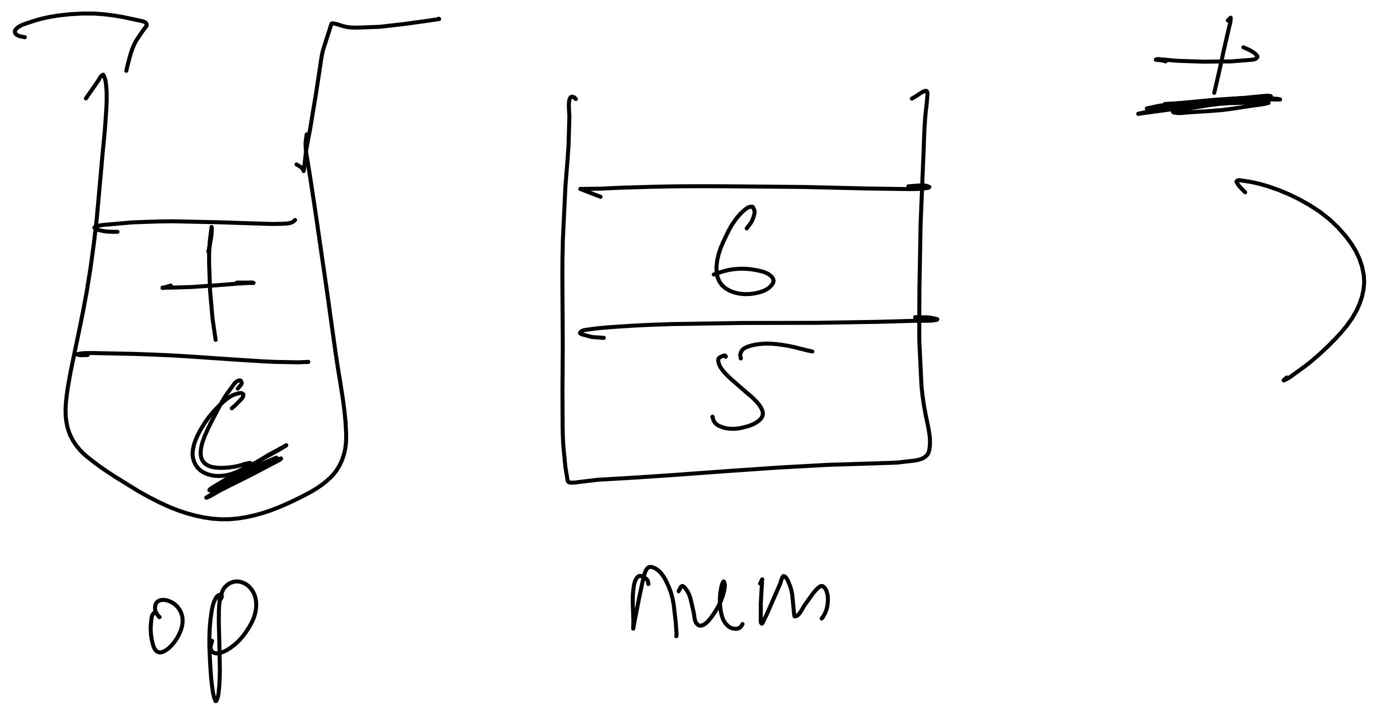
(≡ Serves as a new bottom of stack.

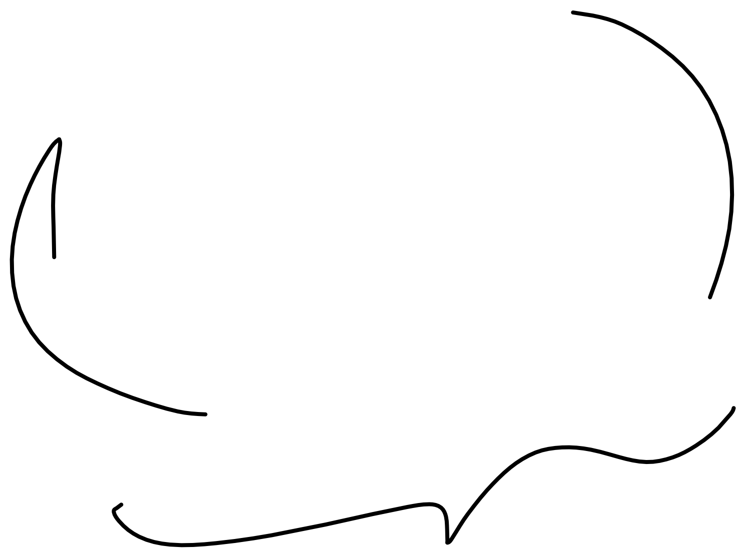
) ≡ Serves as a new end of input.

'(' must get the lowest priority.

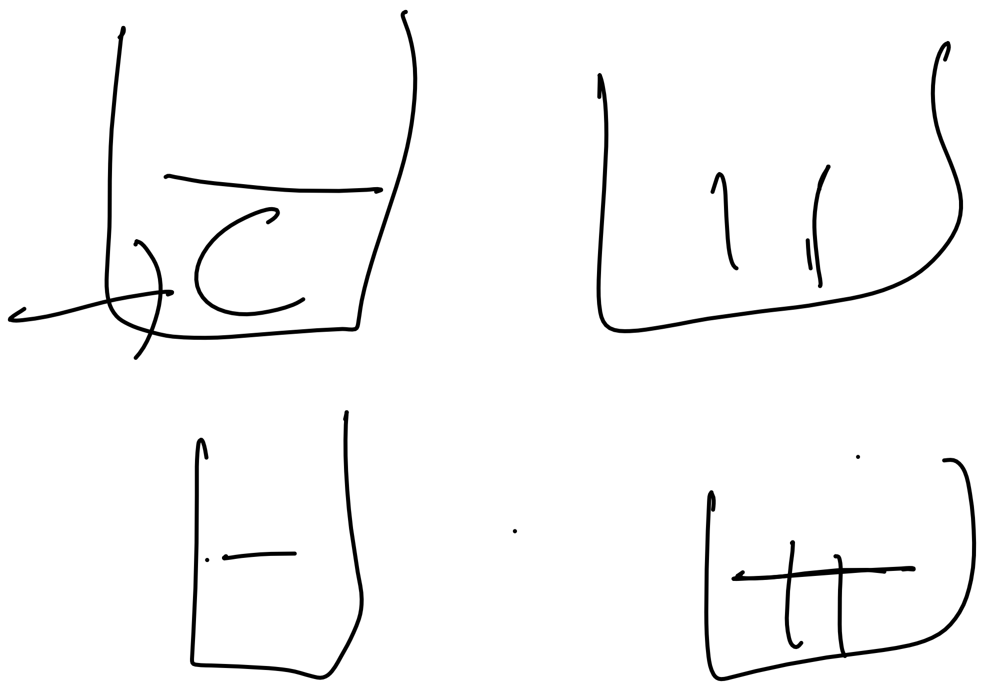
```
else if ( S.next is '(' ).
    while ( S. next() is not ')' ).
        while loop from
        previous pseudo code
        will be repeated
        here.
    if (S. next ')' )
```

(5 + 6)





new environment for calculator.

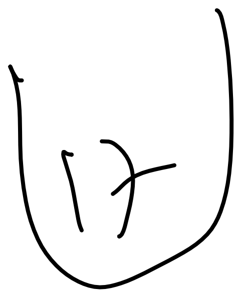


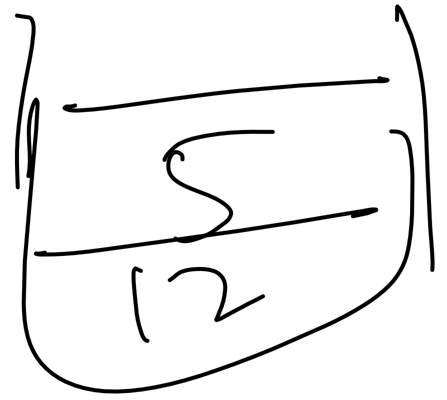
opstack. pop()

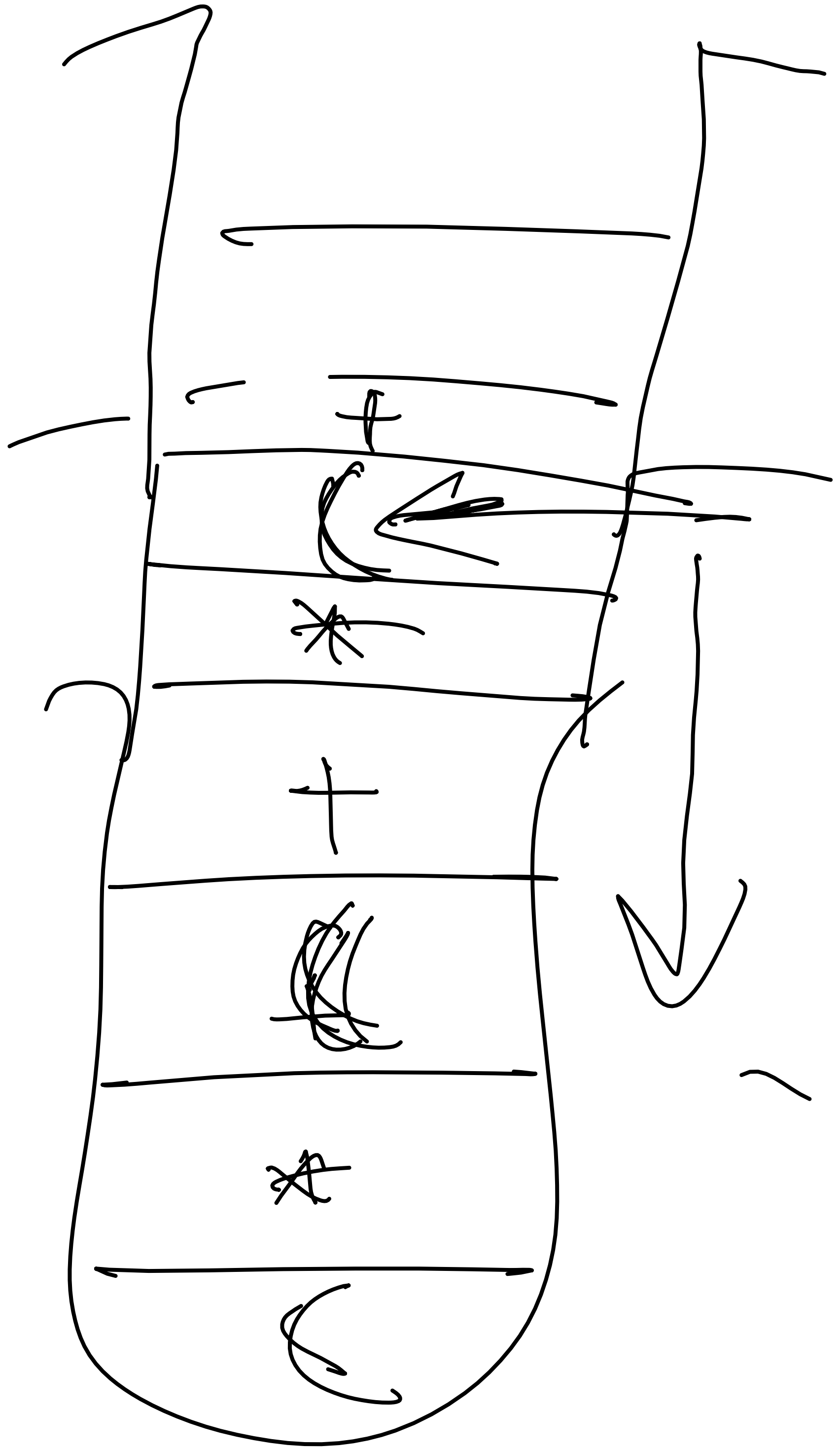
Associativity

if you have

$$(5 * (3 + 10\cancel{)} * \underbrace{(\overset{\uparrow}{12} + \overset{\uparrow}{5})}_{17})))$$







)

Associativity

$2 \wedge 3 \wedge 4$ → right assoc. → $\wedge$

$(2 \wedge 3) \wedge 4$

$\boxed{2 \wedge (3 \wedge 4)}$

$2 \setminus 3 \setminus 4$ → left assoc.

$\boxed{(2 \setminus 3) \setminus 4}$

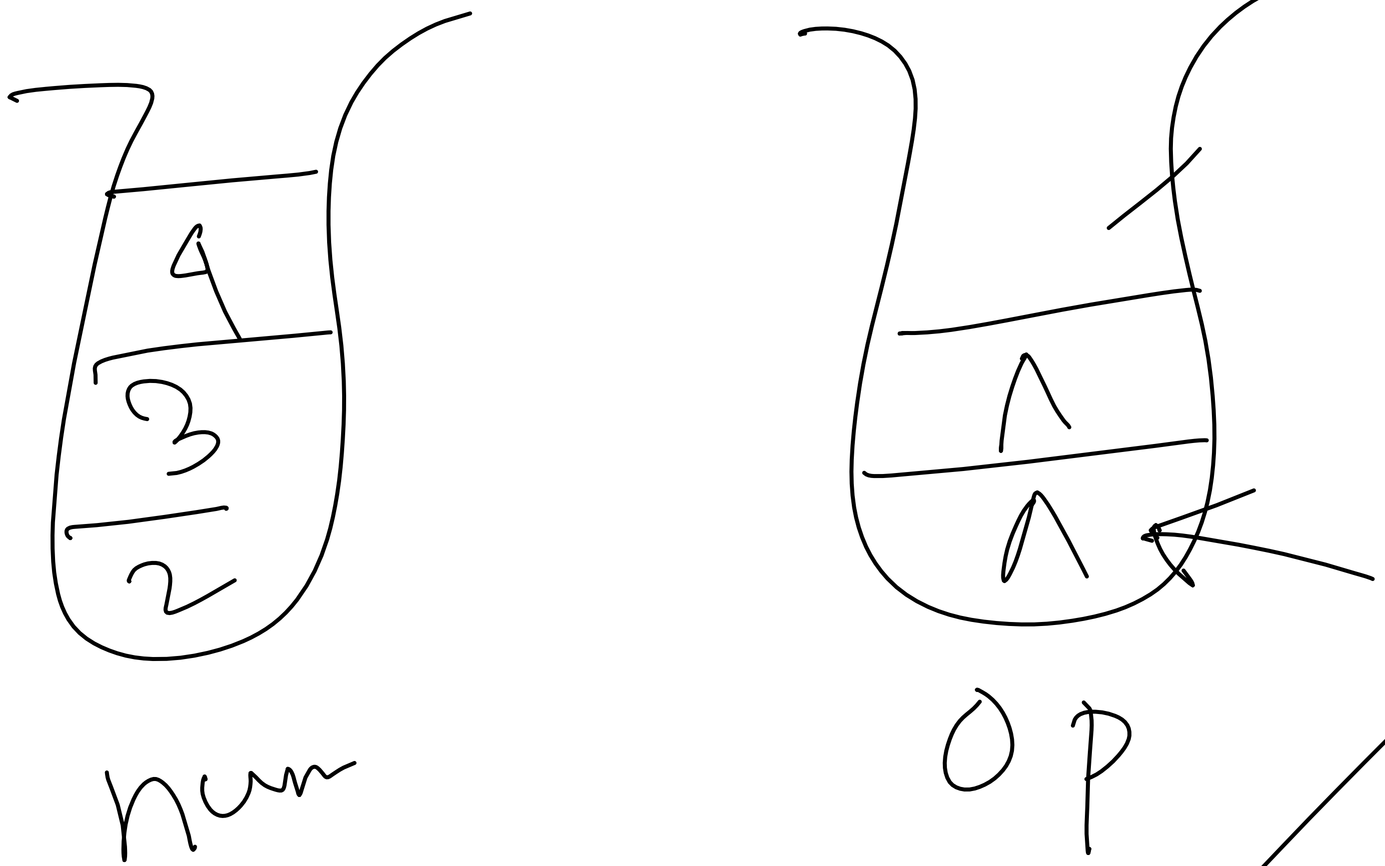
$2 \setminus (3 \setminus 4)$

$+, *, -$

$(5+7)+8$ $5+(7+8)$

$2 \wedge (3 \wedge 4) \rightarrow 2 \wedge 3 \wedge 4$

$(2 \wedge 3) \wedge 4$



$\underline{\wedge}$

<u>Outside</u> > <u>inside</u>

for $\wedge$

otherway

$\underline{2 \wedge (3 \wedge 4)}$

$(2 \wedge 3) \wedge 4$

^ : Out priority > In priority.

/ : outpriority < in priority.

	In P.	Out P.
(	0	–
+, −	1	1
*	2	2
/	4	3
^	5	6

$-5 = (5 - 7) - 3$

$1 = 5 - (7 - 3)$

Correctness: Imitates human behavior.

Timecomplexity: $O(n)$, $n = \#$ input.

Advanced Algorithm

- Randomized algorithms..
- ML: back propagation
- Cryptographic: