Infix
prefix
postfix
} calculation.

traversal. { Infix → Human Maths.
prefix
postfix }

Equations. (i) Operand. (ii) operator.

BDMAS

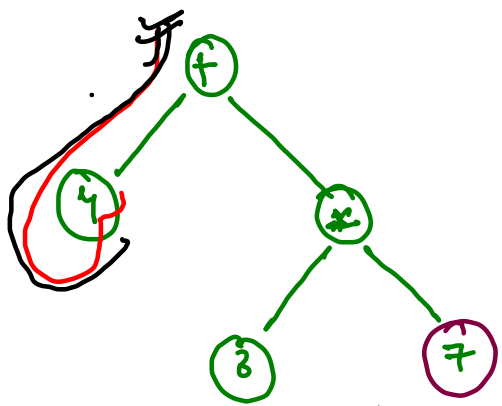
Priority.
1. Bracket
2. Division
3. Multiplication
4. Addition
5. subtraction

Example (1)

N

4+3*7

Euler trace-



4+3*7 = 25

7*7 = 49

InOrder → 4 + 3 * 7 → Infix

PreOrder → + 4 * 3 7 → Prefix

PostOrder → 4 3 7 * + → postfix

Bracket

Division + Multiplication

Addition + Subtraction

1.1 Infix Evaluation
1.2 Infix to prefix
1.3 Infix to postfix

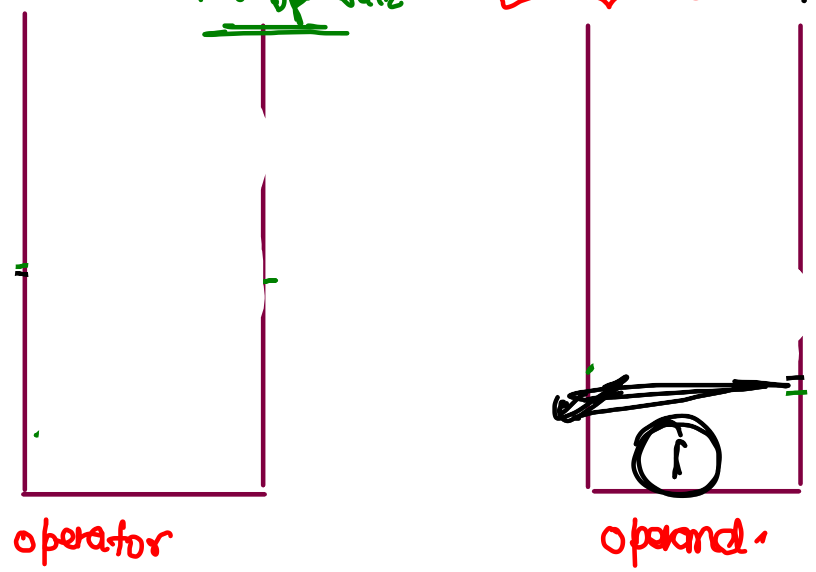
2.1 postfix Evaluation
2.2 postfix to Infix
2.3 postfix to prefix

3.1 prefix Evaluation
3.2 prefix to Infix
3.3 prefix to postfix

Example : → 2 + 3 * 2 − (3 * 4 * 1) + 1

a operator b ⇒ b operator a

val1 operator val2



priority :

( ) → 3

*, / → 2

+, − → 1

Higher priority operator is valid on lower priority operator

operator = *

val2 = 2 .

val1 = 3

val1 operator val2

NOTE: (Algo) Expres: $2 + 3 * 2 - (3 + 4 * 1)$

1. High priority on top of operator stack is valid.
2. Low priority operator can't overtake high priority operator, if it is going to happen then we will resolve stack & remove from stack too.
   - → get val2, then val1 from operand stack
   - → get operator from operator stack.
   - → Solve val1 operator val2.
   - → push result in operand stack

   * How to change the priority.
   * How to solve this
3. If closing bracket is encounter then solve until opening bracket is not hit on operator stack.

Priority } operator

Evaluate } → valid operator valid.

↳ number. = ▷ → operand. } → operator

↳) → resolve } → until opening bracket } →

↳ → opening push.

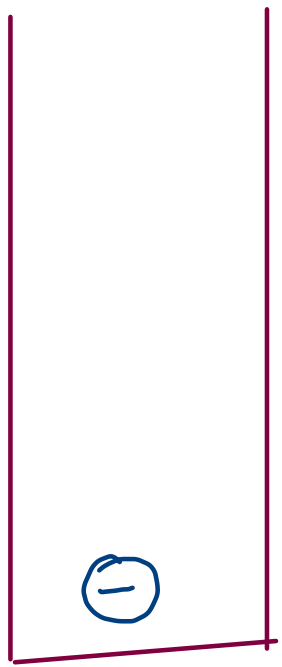
(operator / opening bracket

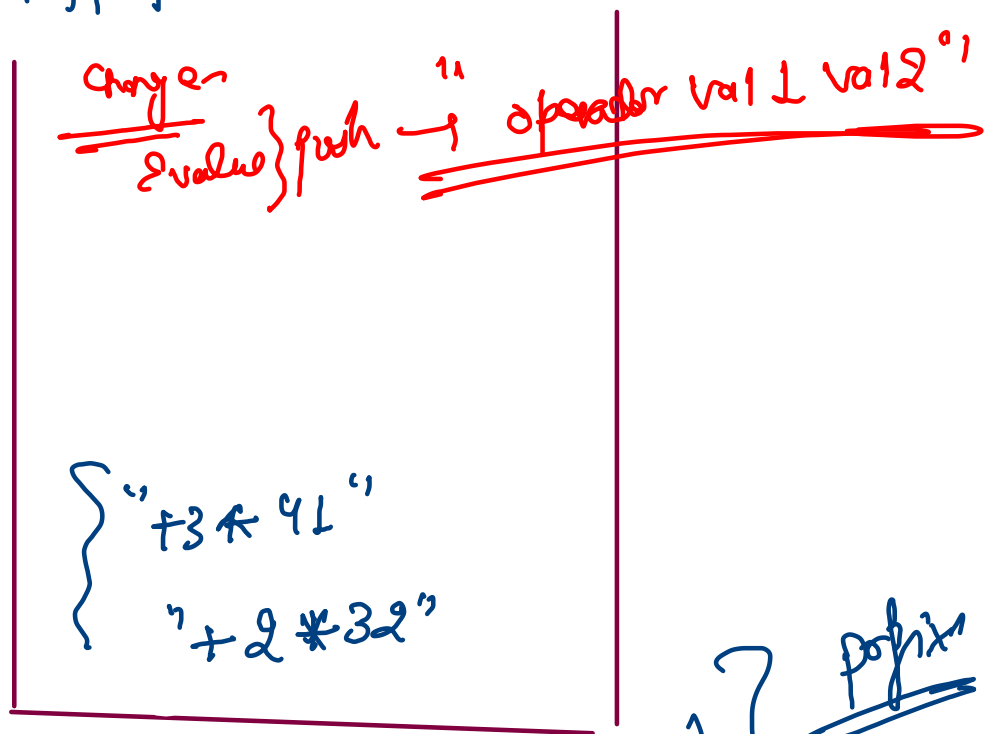
→ operator

Example :

Convert prefix

Expression :→ 2 + 3 * 2 − (3 + 4 * 1)



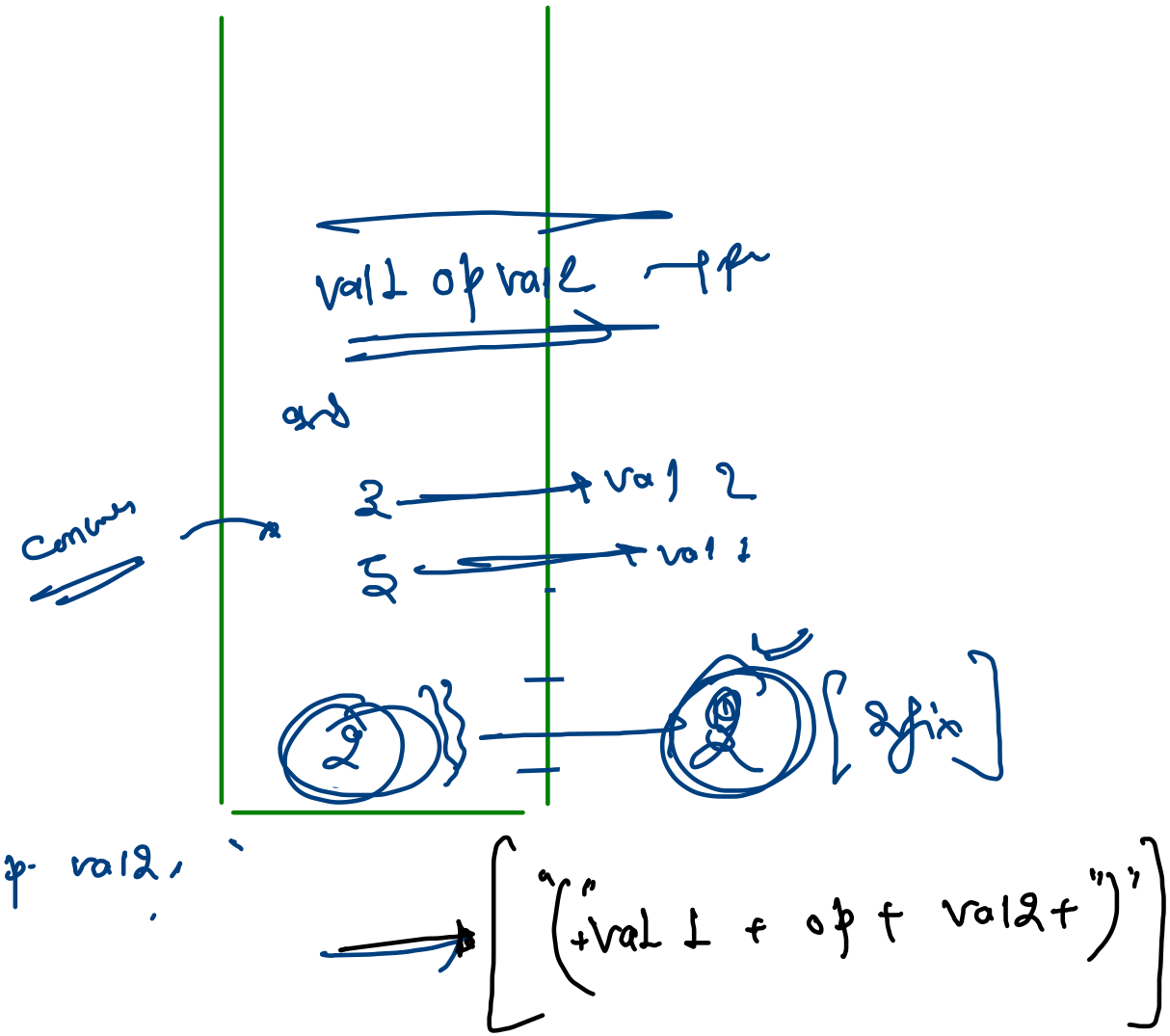


(− +2*32 +3*41)

postfix

2 6 4 * 8 / + 3 -

val1 val2 operator



*

val2 = st.pop();

val1 = st.pop();

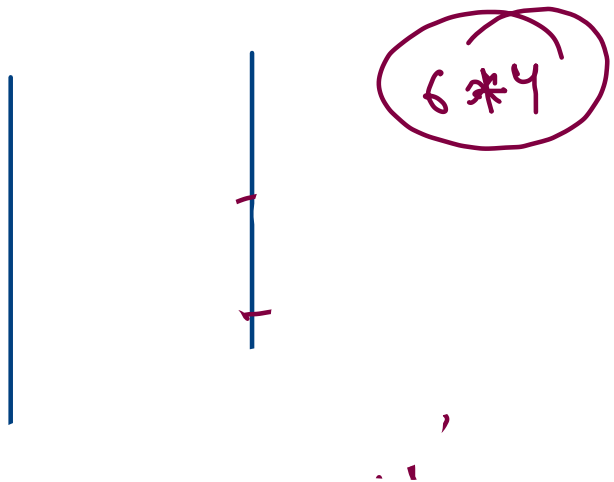
int res = val1 op val2;

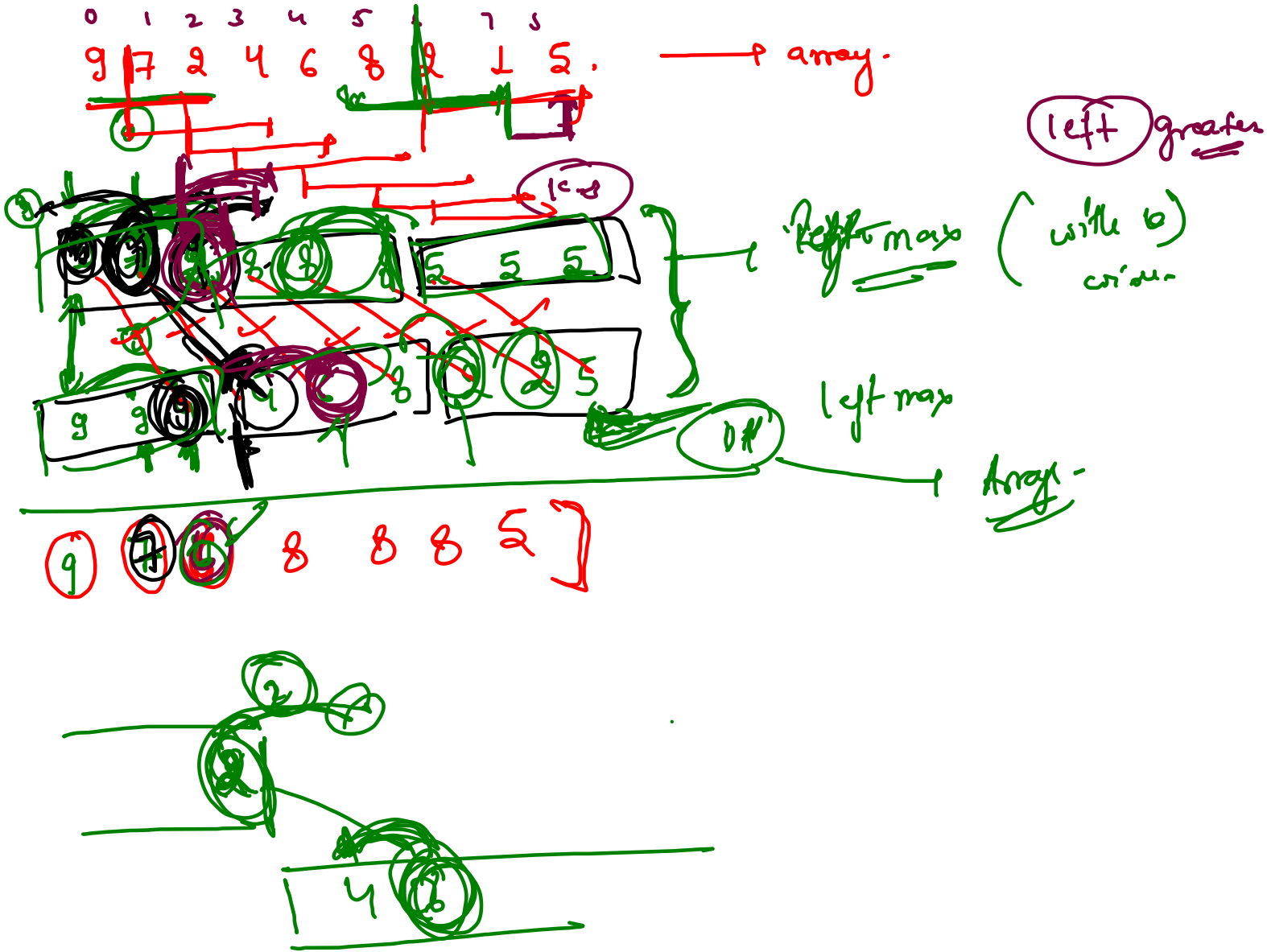
st.push(res);

→ ["(" + val1 + op + val2 + ")"]

prefix.

operator val1 val2,




0 1 2 3 4 5 6 7 8
9 7 2 4 6 8 2 1 5.
array.
left greater
left max (with b)
left max
array.
9 7 6 8 8 8 5
4 6