DSA through C++

Stack

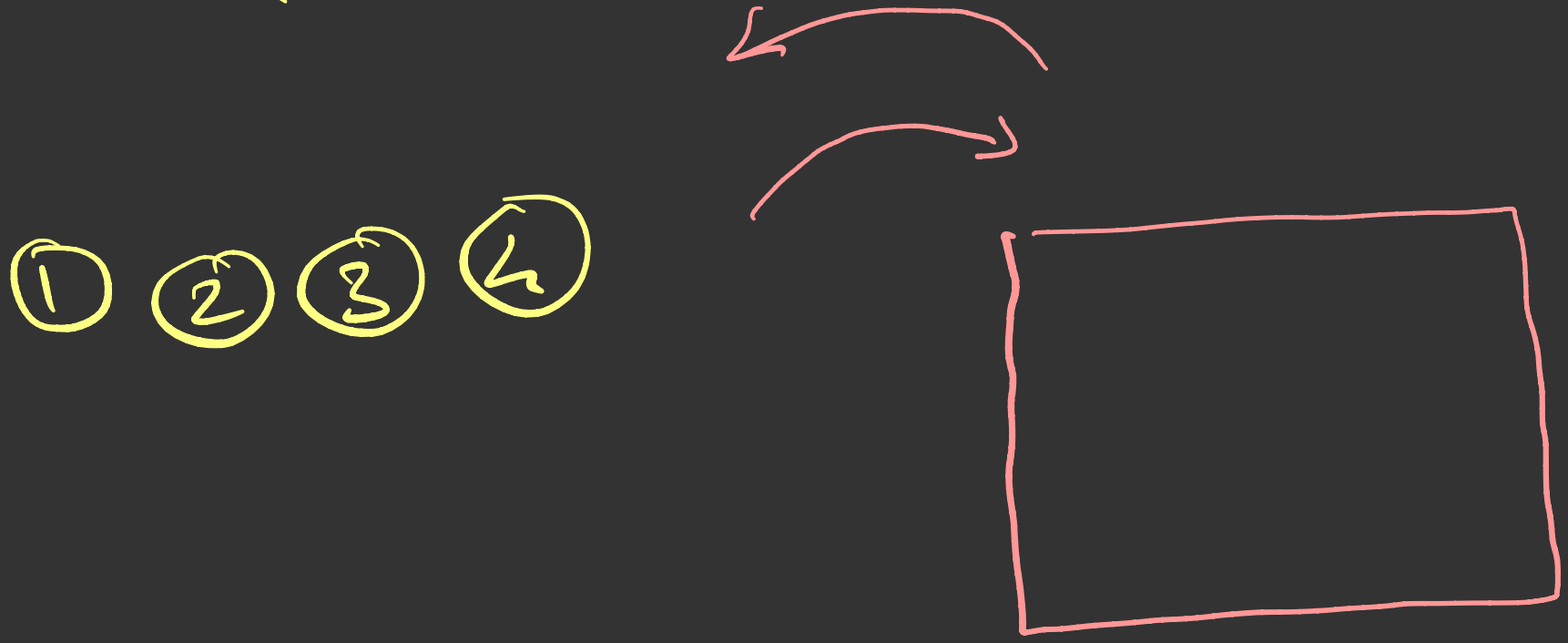
Saurabh Shukla (MySirG)

Agenda

1. What is STACK?
2. Operations on STACK
3. Ways to implement STACK
4. Polish Notation
5. Algorithm to convert infix to postfix
6. Algorithm to evaluate postfix expression

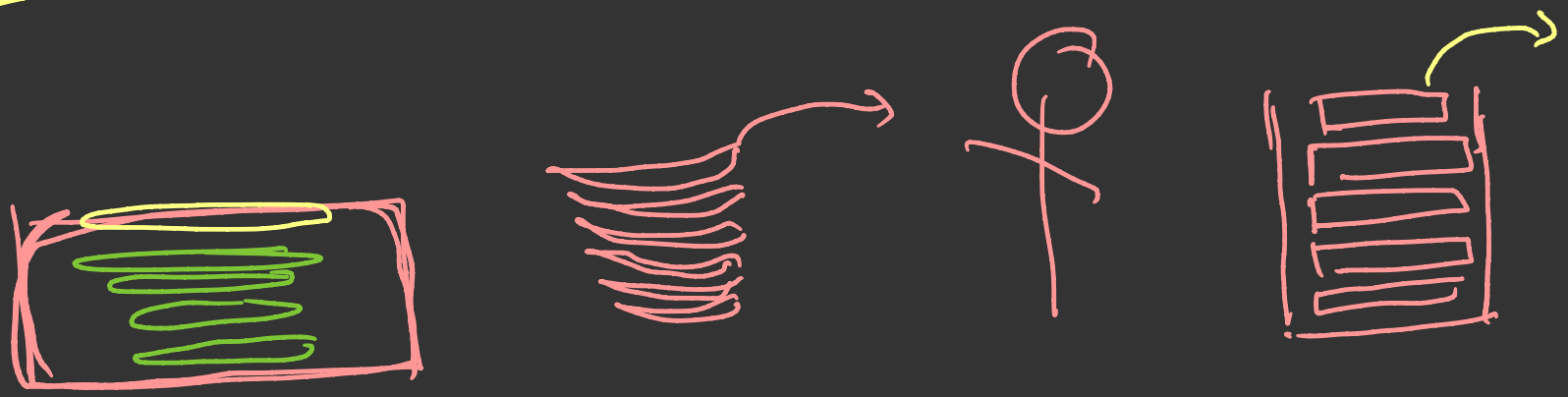
What is STACK?

- STACK is a linear Data Structures
- STACK's working principle is Last In First Out (LIFO)



Travell

Real world examples of Stack

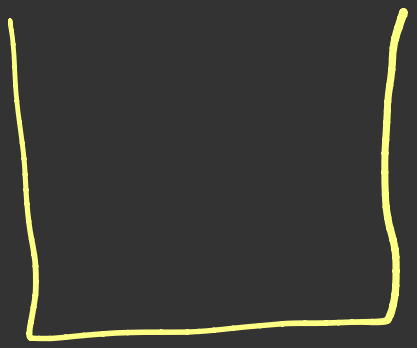


Programming world examples of stack

Recursion | function call

f1() → f2() → f3() → f4()

f1()
f2()
f3()



Operations on STACK

Insert	PUSH( )
Delete	POP( )
View top element	PEEK( )

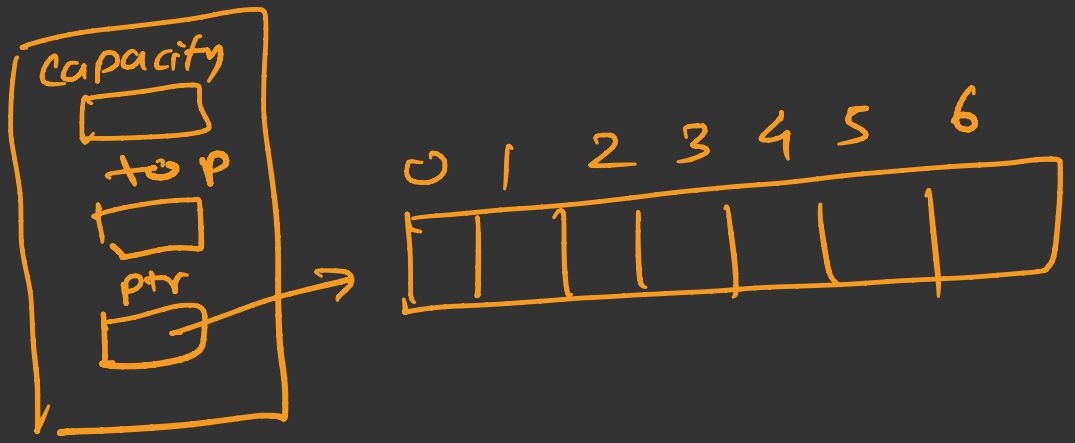
top

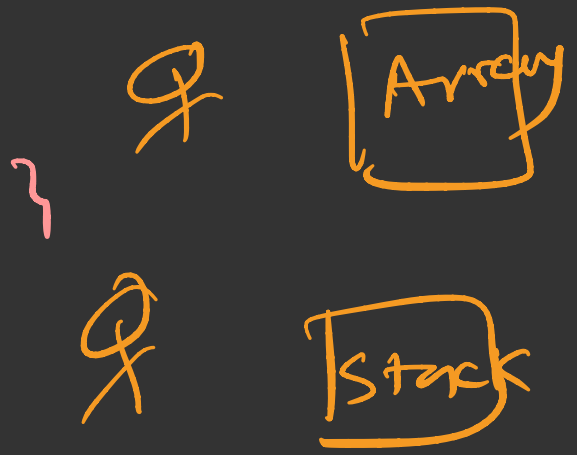
Ways to implement STACK

1. Using Arrays
2. using Dyn Arrays
3. using Linked List

Inheritance

```
class STACK
{
}
```





Polish Notation

The method of writing operators of an expression either before their operands or after them is called the Polish Notation

- Infix Notation — A + B
- Prefix Notation — + A B
- Postfix Notation — A B +

Practice

Infix : A+B*C

Prefix : +A*BC

Postfix : ABC*+

Infix : A*B-C/(D+E)

Prefix : -*AB/C+DE

Postfix : AB*CDE+/-

Infix : A+B*(C-D)+E

Prefix : ++A*B-CDE

Postfix : ABCD-*+E+

A+B*CD-+E

A+BCD-*+E

ABCD-*++E

ABCD-*+E+

A*B-C/+DE

*AB-C/+DE

*AB- /C+DE

-*AB/C+DE

A*B-C/(D+E)

A*B-C/DE+

AB*-C/DE+

AB*- CDE+/

AB*CDE+/-

A+B*-CD+E

A+*B-CD+E

+A*B-CD+E

++A*B-CDE

A + B * C → A B C * +

(A + B) * C → A B + C *

Infix to Postfix

Suppose Q is an arithmetic expression written in infix notation. This algorithm finds the equivalent posttix expression P.

1. PUSH '(' onto the STACK and add ')' to the end of Q.
2. Scan Q from left to right and repeat steps 3 to 6 for each element of Q until the STACK is empty.
3. If an operand is encountered add it to P
4. If a left parenthesis is encountered, PUSH it onto the STACK.

5. If an operator (say #) is encountered, then:
    a. Repeatedly pop from STACK and add to P each operator which has the same precedence than #
    b. Add # to STACK
6. If a right parenthesis is encountered, then:
    a. Repeatedly pop from the STACK and add to P each operator until a left parenthesis is encountered
    b. Remove the left parenthesis
7. Exit.

Evaluate Postfix

Let P be an arithmetic expression written in postfix notation. We uses a STACK to hold operands.

This algorithm finds the VALUE of an arithmetic expression P written in Postfix notation.

1. Add a right parenthesis ')' at the end of P

2. Scan P from left to right and repeat step 3 and 4 for each element of P until the sentinel `)' is encountered.
3. If an operand is encountered, put it on the STACK
4. If an operator # is encountered, then
   a. Remove the two top elements of STACK, where X is the to element and Y is the next to top element
   b. Evaluate Y # X
   c. Place the result of (b) back on STACK
5. set VALUE equal to the top element on STACK
6. Exit