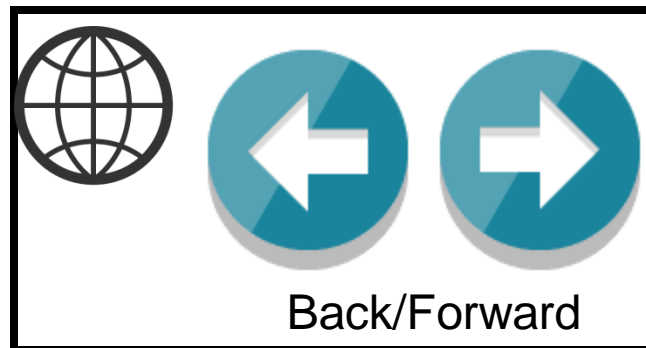
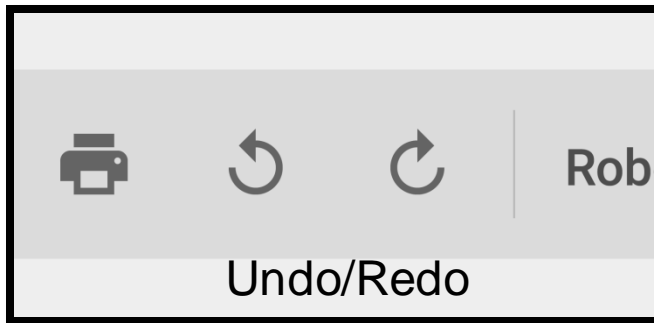


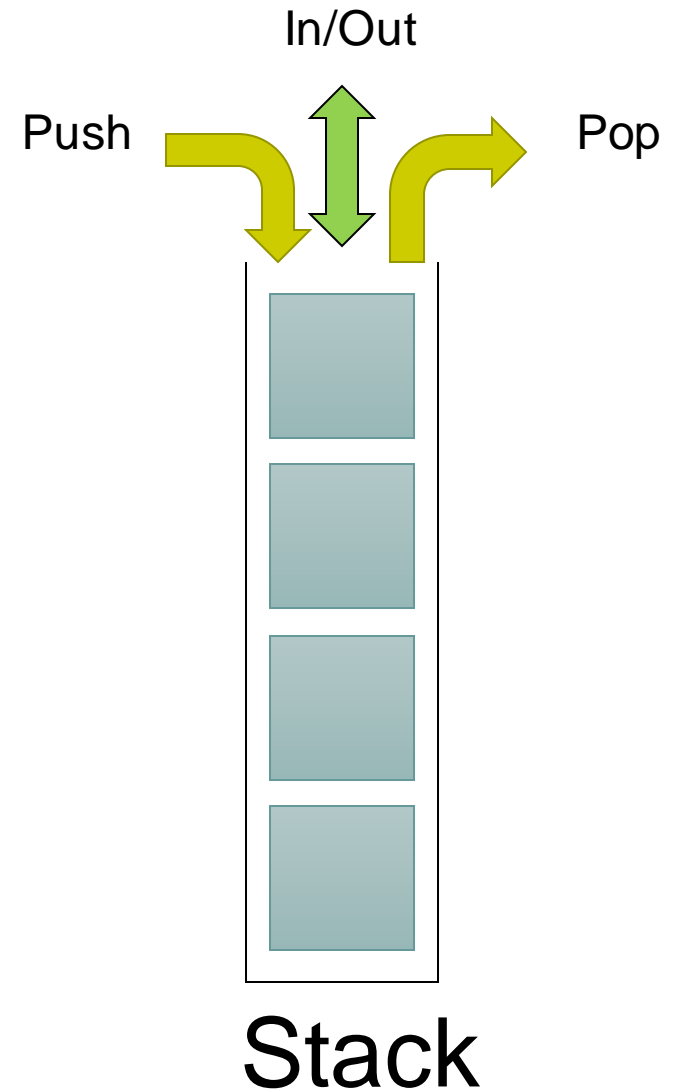
Stacks and Queues

Stack

> LIFO: Last-in First-out



Function call/return

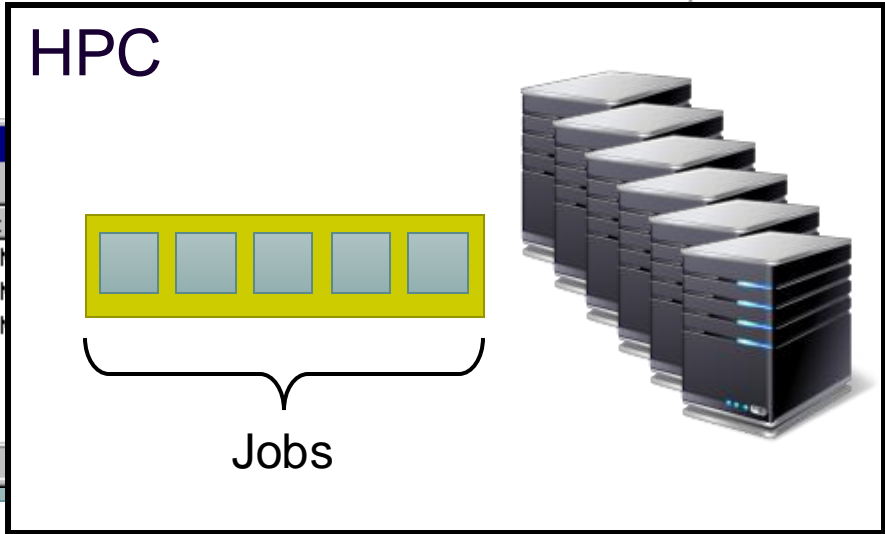


Queue

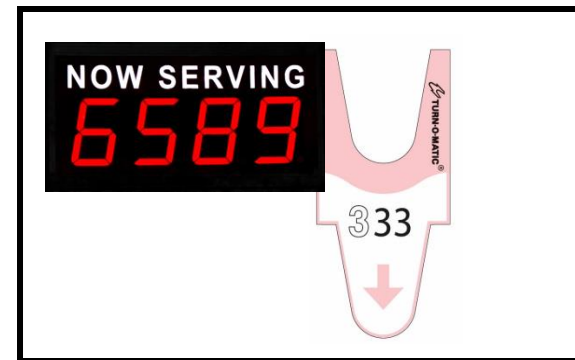
> FIFO: First-in First-out

Document Name	Status	Owner	Progress	Started At
Job 2994	Printed	Sue Smith...	24.0KB	5:01:42 PM
Job 3004	Warning	Bernie Leon...	42.5KB	5:36:57 PM
Job 3005	Printing	Sue Smith...	0 bytes of 2...	5:53:02 PM

3 jobs in queue



Queue



Queue and Stack ADT

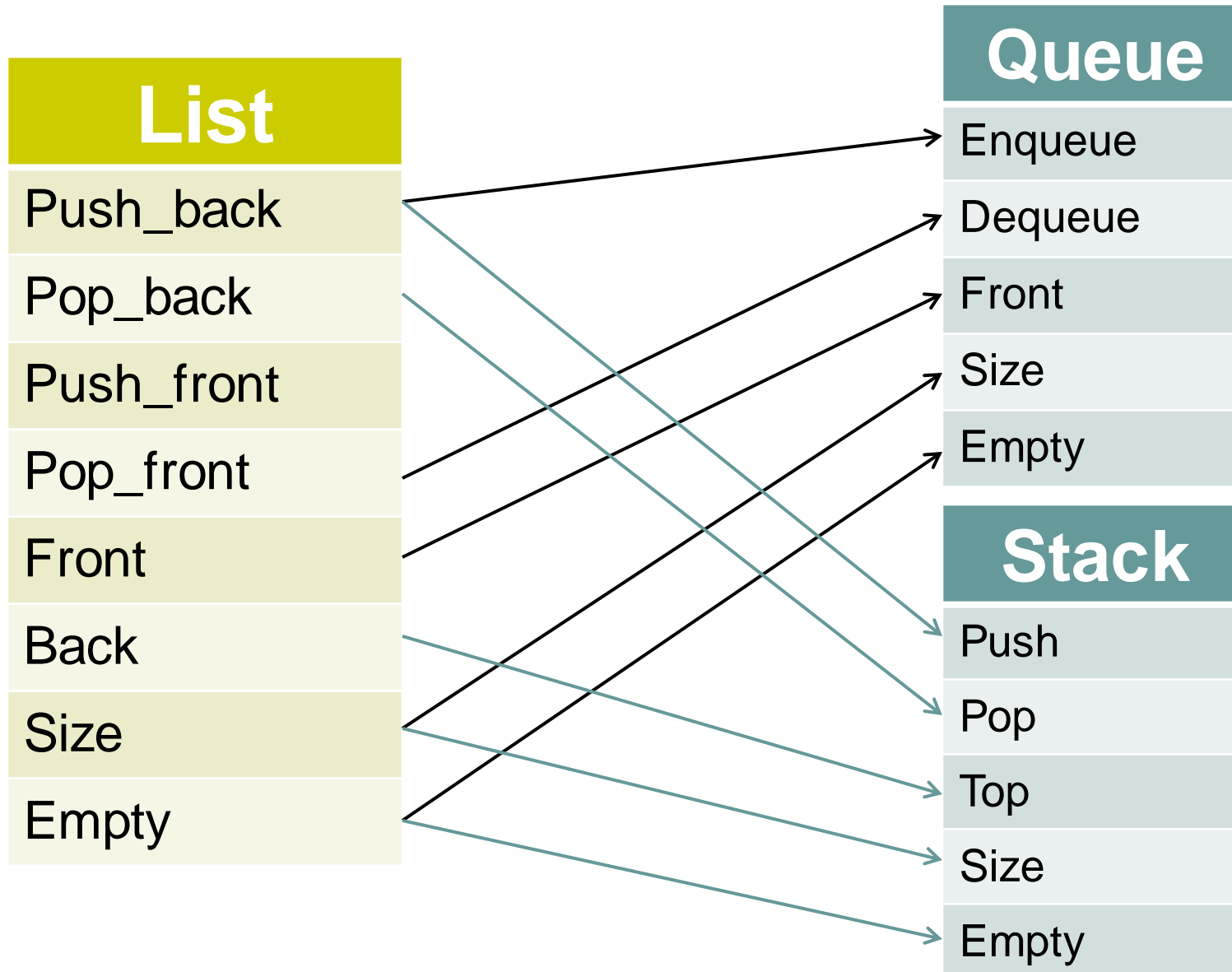
Queue

- › Enqueue
- › Dequeue
- › Front
- › Size
- › Empty?

Stack

- › Push
- › Pop
- › Top
- › Size
- › Empty?

Stack/Queue Implementation

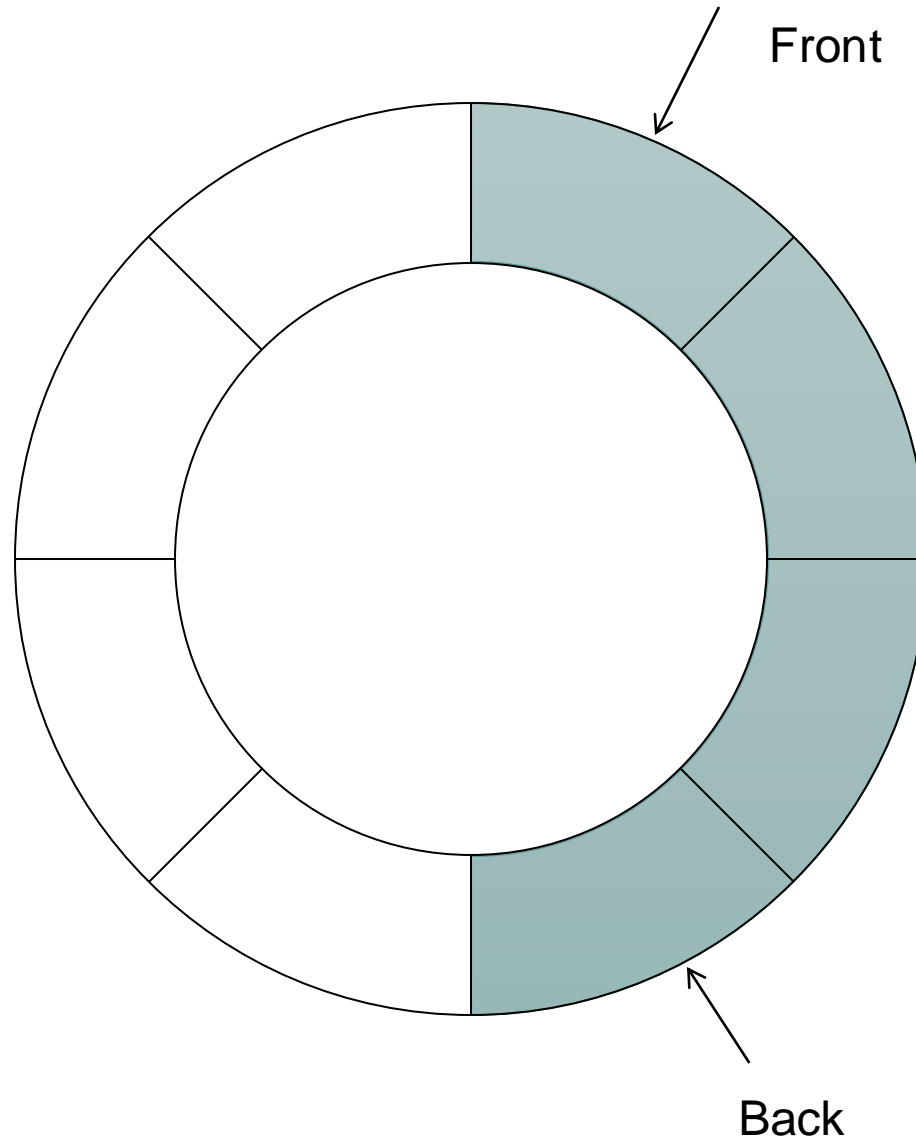


Queue Implementation



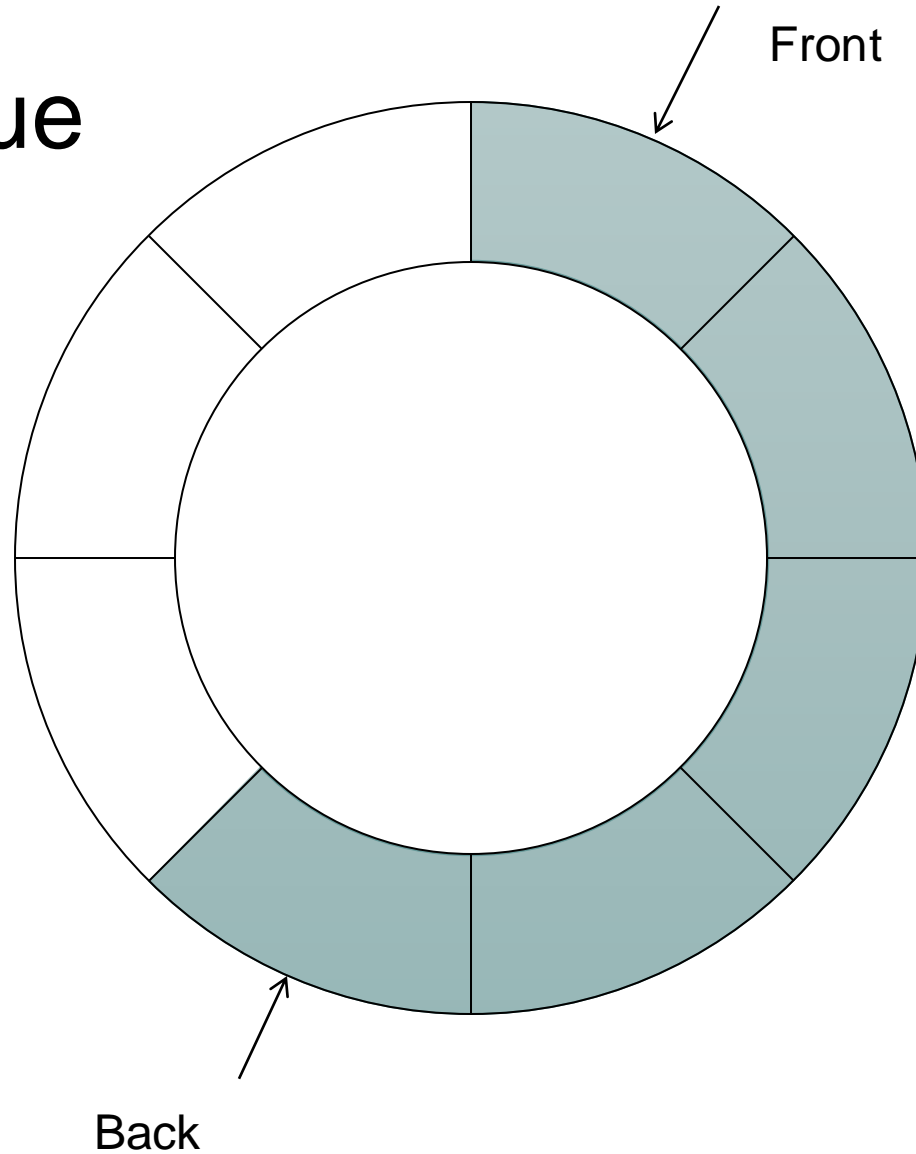
Queue	Array Impl.	Linked List Impl.
Enqueue	$O(1)$	$O(1)$
Dequeue	$O(n)$	$O(1)$
Front	$O(1)$	$O(1)$
Memory overhead	Small	Big
Random access	$O(1)$	$O(n)$

Circular Array Queue



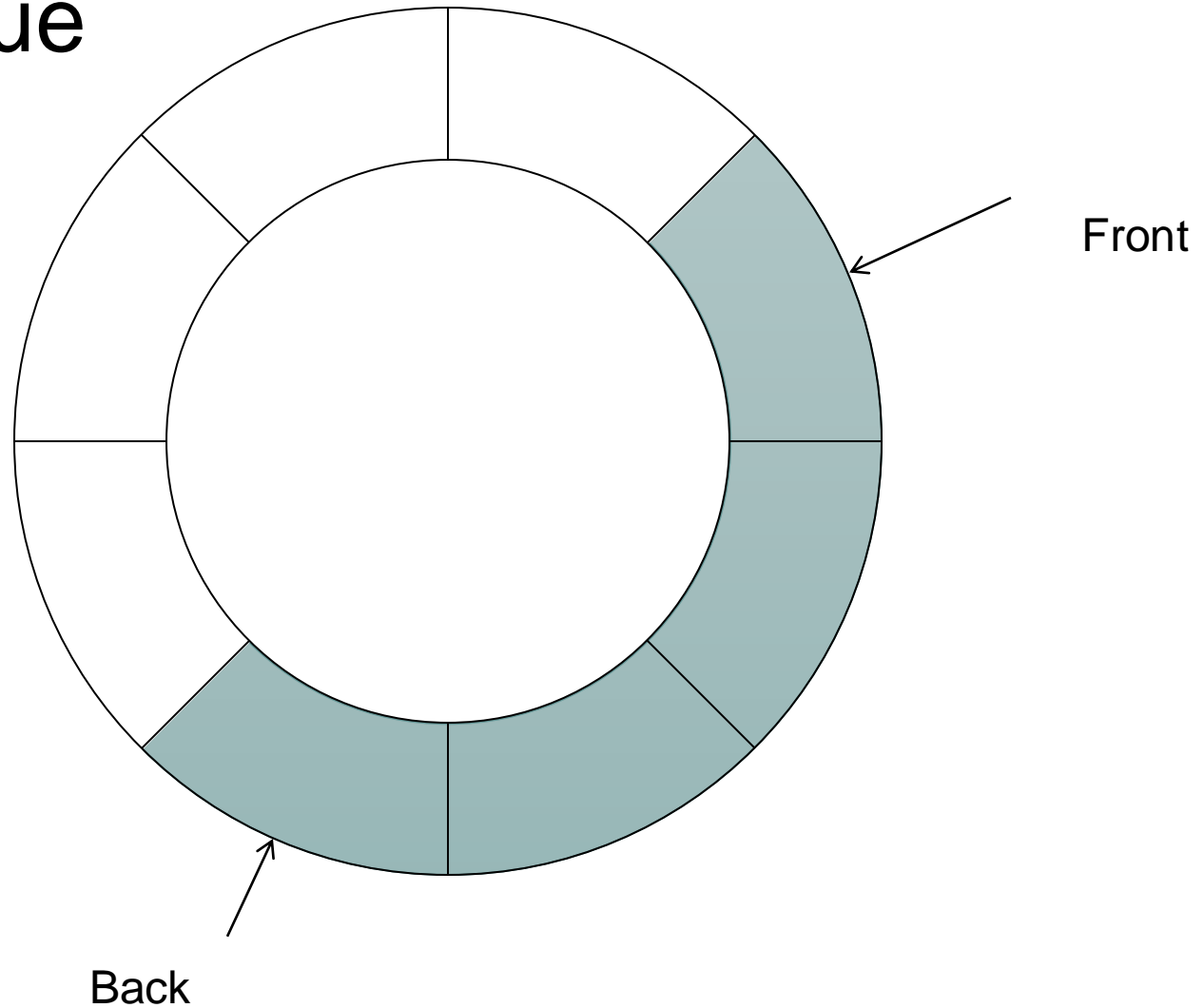
Circular Array Queue

Enqueue

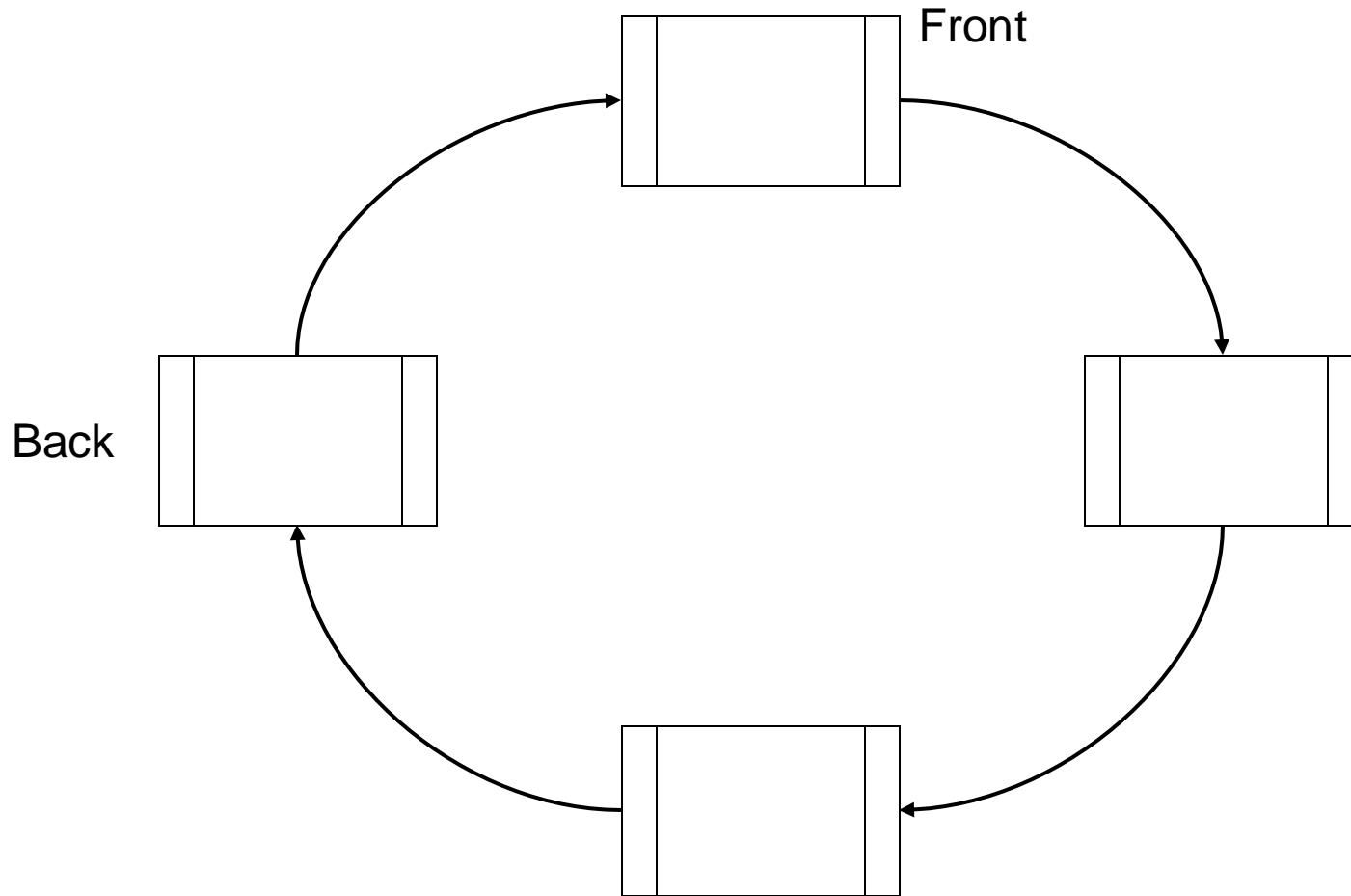


Circular Array Queue

Dequeue



Circular Linked List Queue



Queue Implementation



Queue	Circular Array Impl.	Circular Linked List Impl.
Enqueue	$O(1)$	$O(1)$
Dequeue	$O(1)$	$O(1)$
Front	$O(1)$	$O(1)$
Memory overhead	Small	Big

Standard Template Library (STL)



- Lists, stacks, and queues are all implemented in STL
- In a *real* program, you would better use them; why?
- For the sake of learning, you are not allowed to use STL during this class unless otherwise mentioned

Stack Applications

- Expression evaluation
- Human-friendly infix expressions
 - The operator falls between the two operands

$$3 + 2 \times 5 = 13$$
 - Easier to read and understand
 - Can be easily broken into pieces
- Machine-friendly postfix expressions
 - The operator is placed *after* the two operands

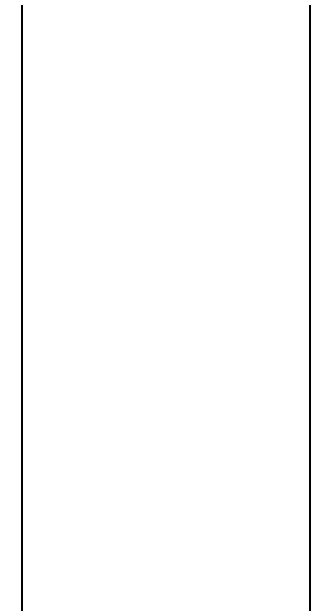
$$325 \times + = 13$$
 - Easier to compute in one pass
 - No need for parentheses

Evaluate postfix expressions



- › Infix: $(3 \times 5 + 4/2) \times 2 = 34$
- › Postfix: $35 \times 42/+2 \times$

3	5	×	4	2	/	+	2	×
---	---	---	---	---	---	---	---	---

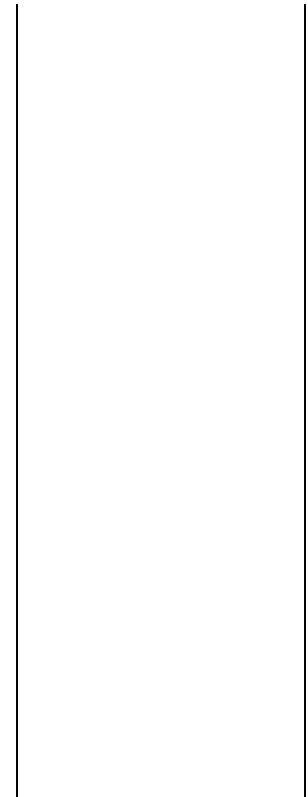


Stack of
operands

Postfix Evaluation Example

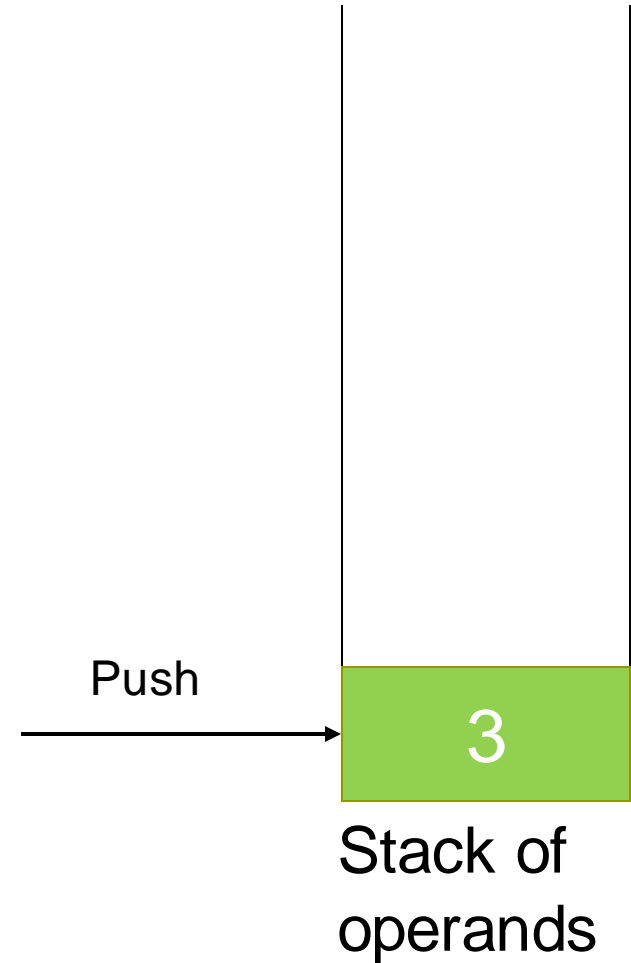
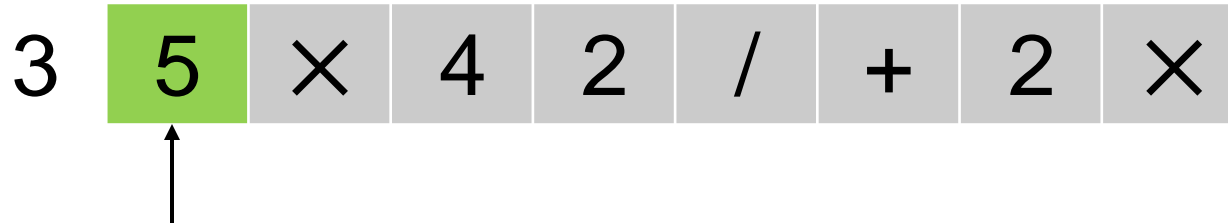


3 5 × 4 2 / + 2 ×

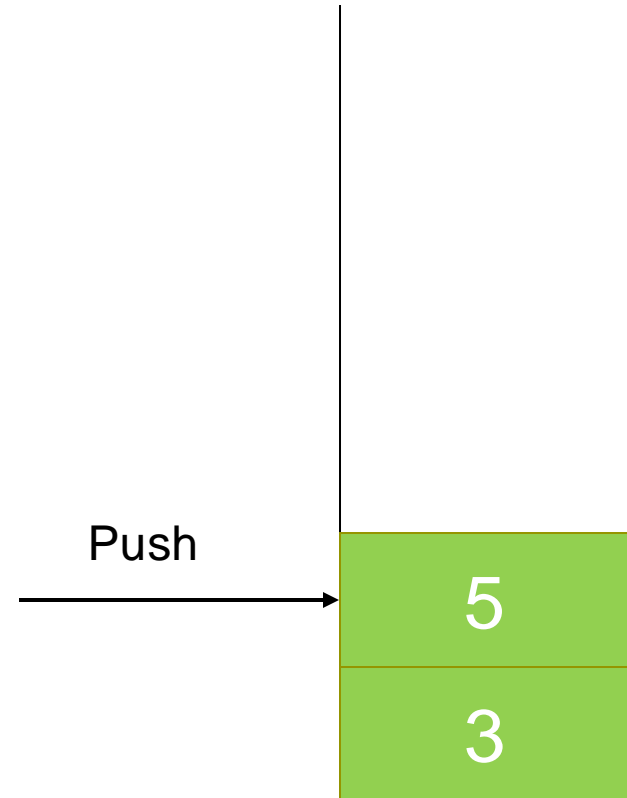
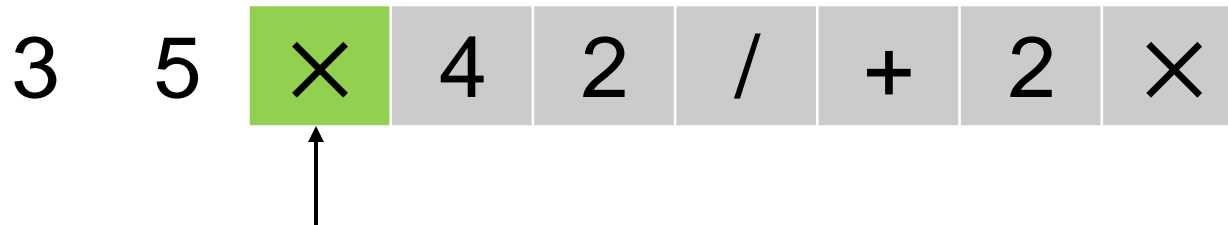


Stack of
operands

Postfix Evaluation Example



Postfix Evaluation Example



Stack of
operands

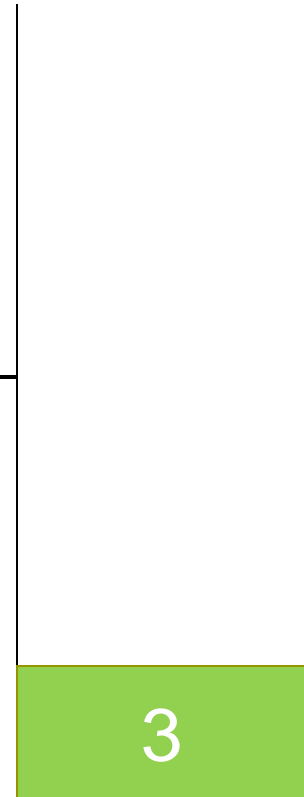
Postfix Evaluation Example

3 5 × 4 2 / + 2 ×



5

Pop



Stack of operands

Postfix Evaluation Example

3 5 **×** 4 2 / + 2 ×



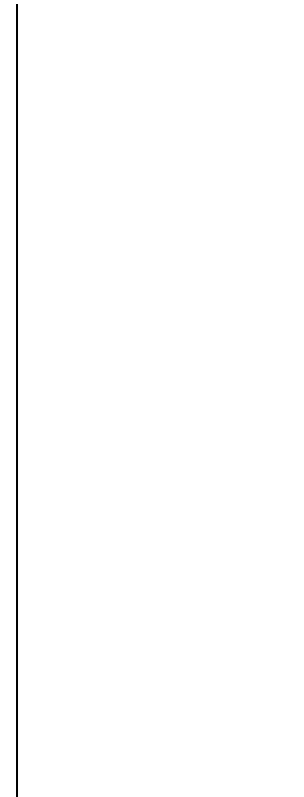
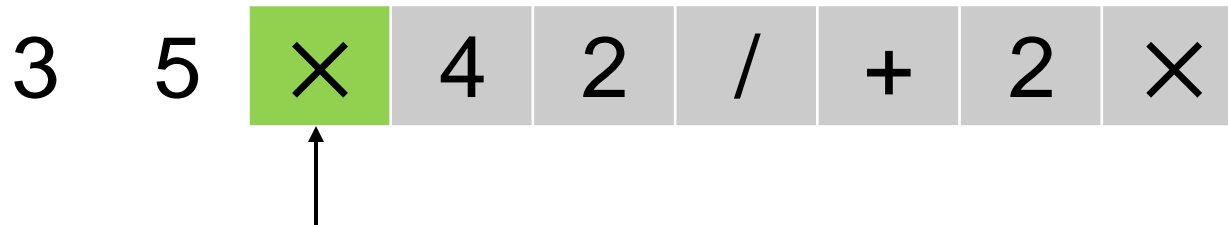
5

3

Pop

Stack of
operands

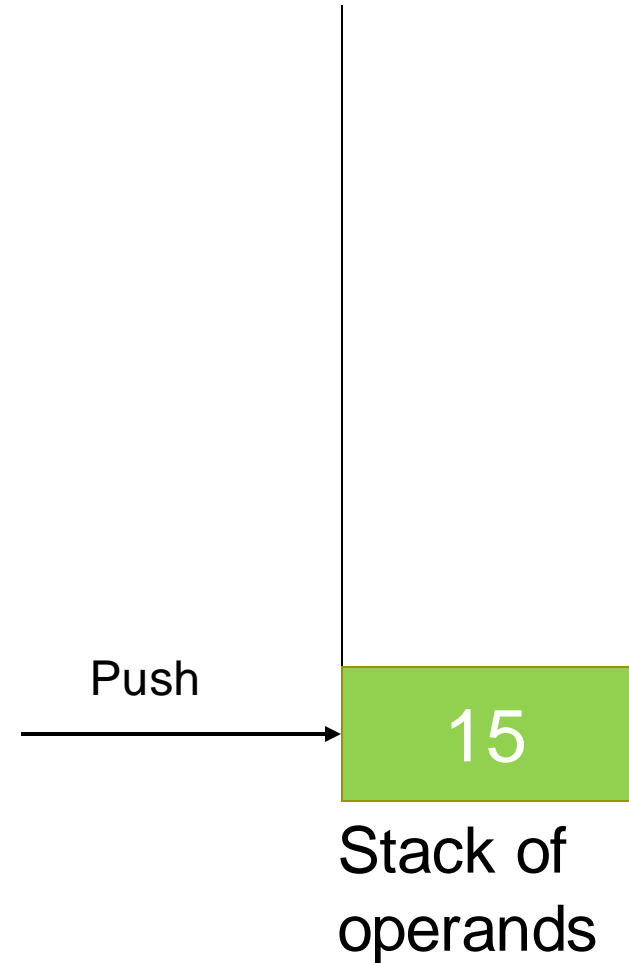
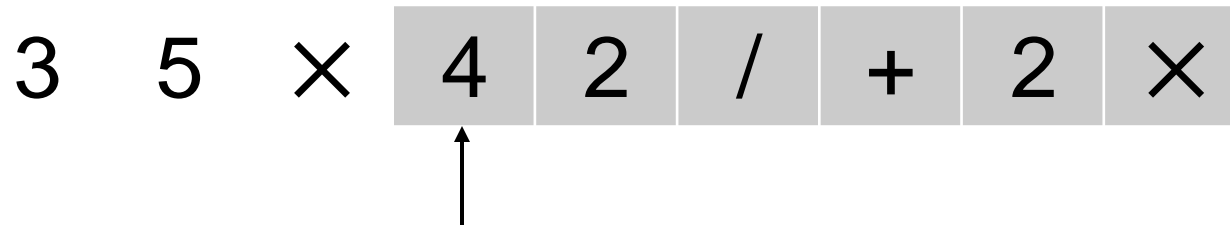
Postfix Evaluation Example



Stack of
operands

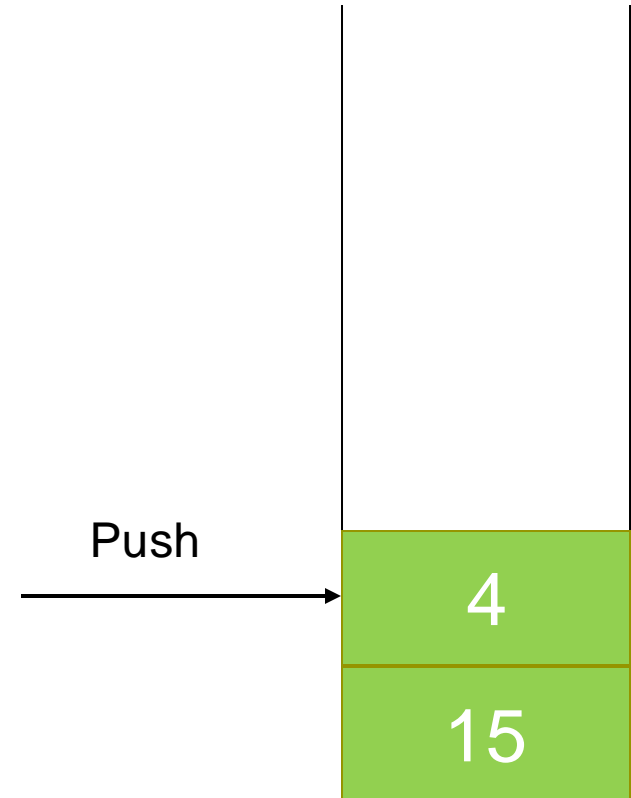
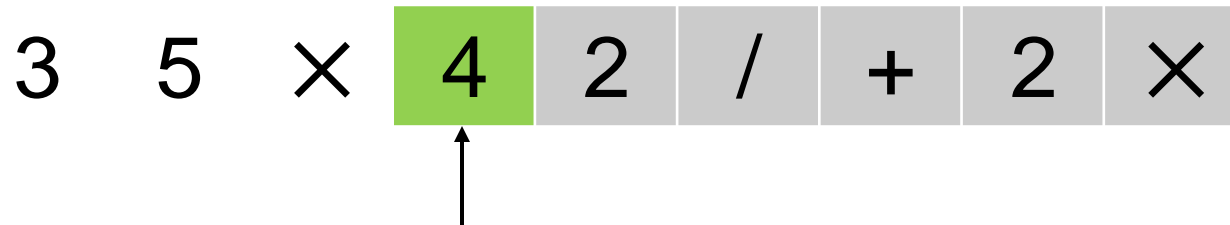
Postfix Evaluation Example

3 5 × 4 2 / + 2 ×



Postfix Evaluation Example

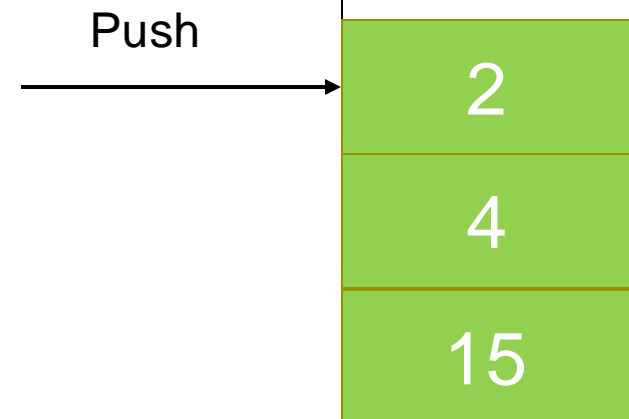
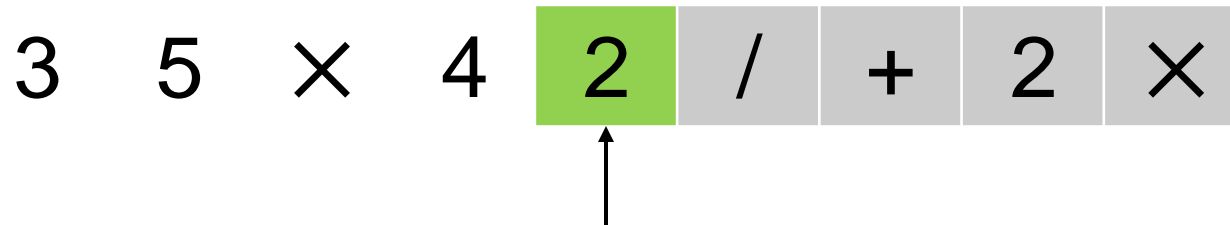
3 5 × 4 2 / + 2 ×



Stack of
operands

Postfix Evaluation Example

3 5 × 4 2 / + 2 ×

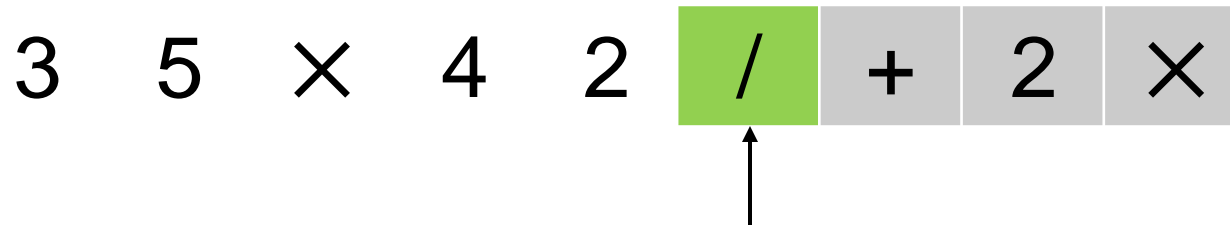
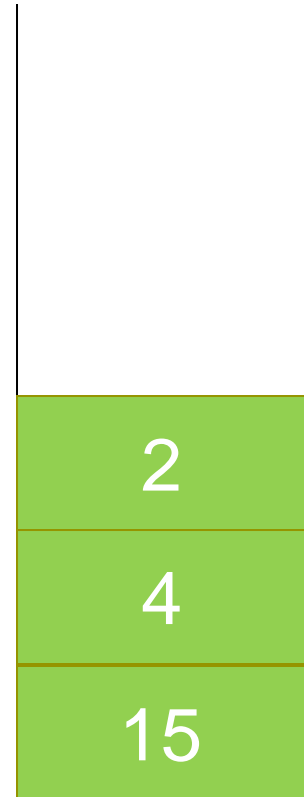


Stack of
operands

Postfix Evaluation Example



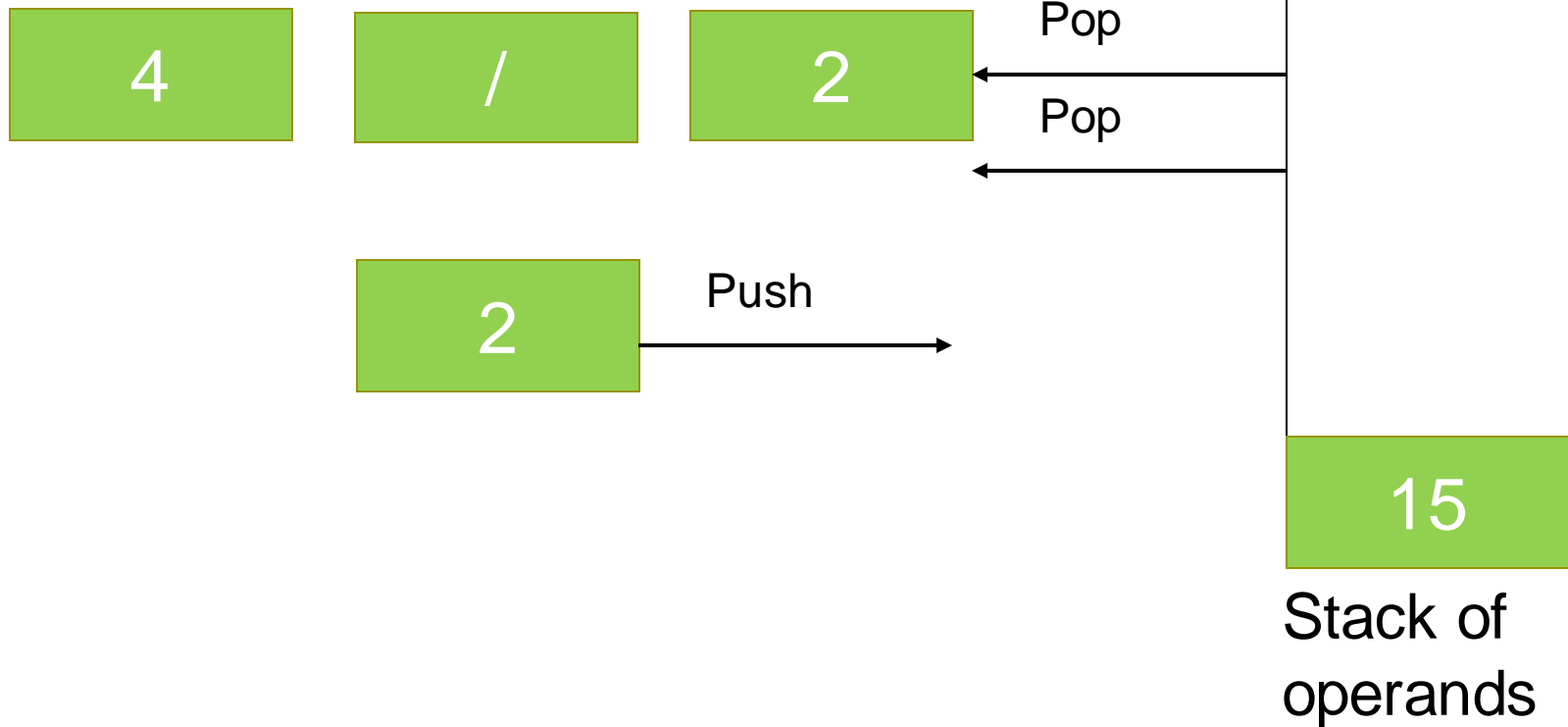
3 5 × 4 2 / + 2 ×

A horizontal sequence of tokens: '3', '5', '×', '4', '2', '/', '+', '2', '×'. The '/' token is highlighted in a light green box, and a black arrow points upwards from below it to the center of the box.

Stack of
operands

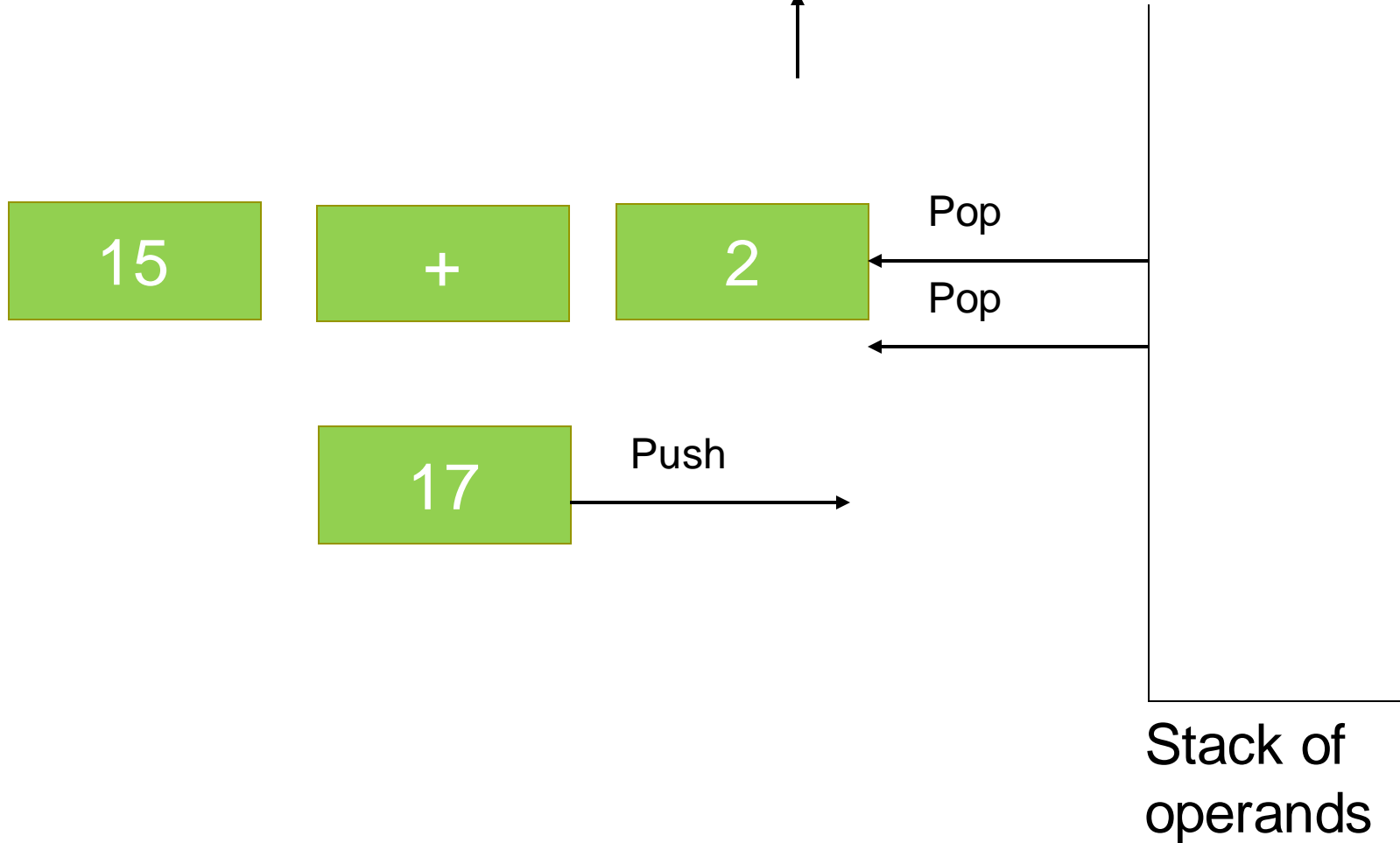
Postfix Evaluation Example

3 5 × 4 2 / + 2 ×



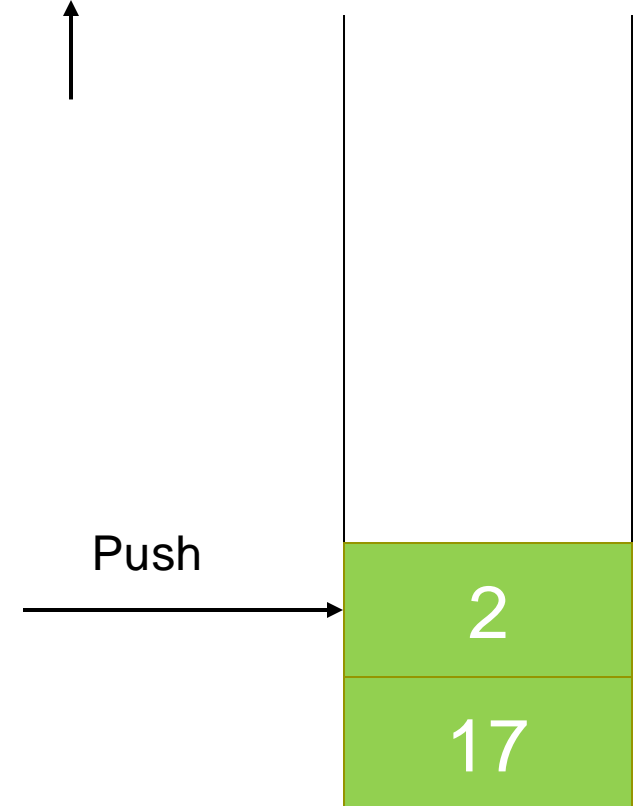
Postfix Evaluation Example

3 5 × 4 2 / + 2 ×



Postfix Evaluation Example

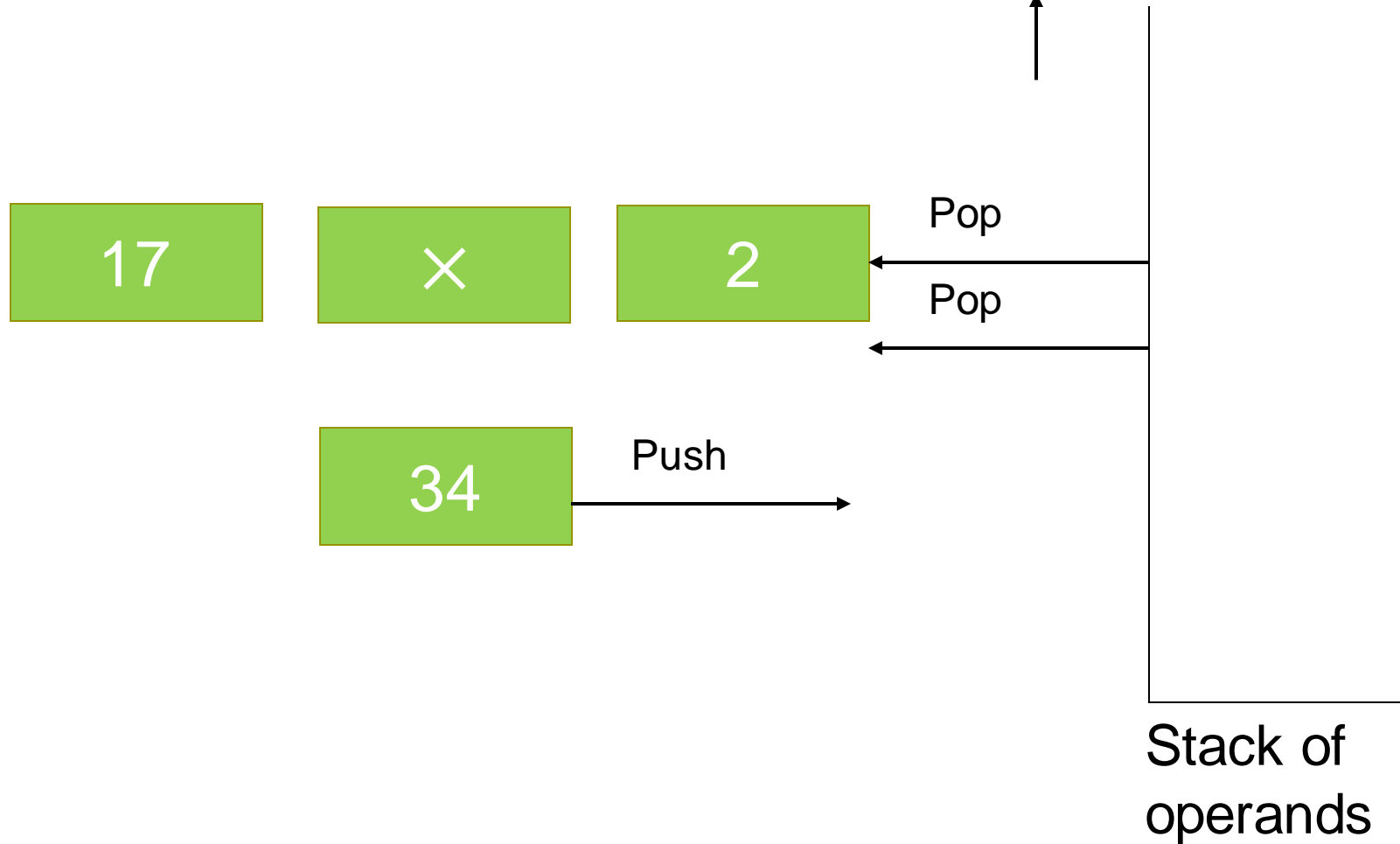
3 5 × 4 2 / + 2 ×



Stack of
operands

Postfix Evaluation Example

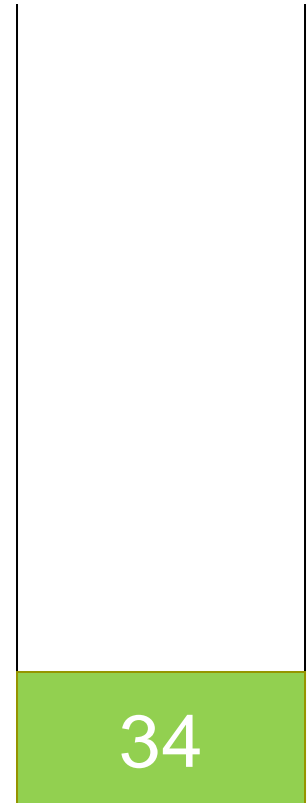
3 5 × 4 2 / + 2 ×



Postfix Evaluation Example



3 5 × 4 2 / + 2 ×



Stack of
operands

Infix to Postfix Conversion

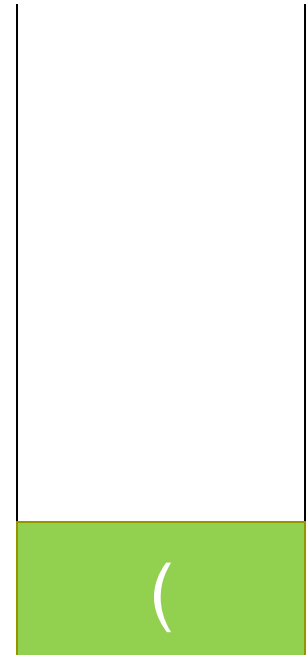
- Convert the input into a sequence of operators and operands
- Account for operator precedence
- Account for parentheses
- Example
 - Infix (input): $(3 \times 5 + 4/2) \times 2$
 - Postfix (desired output): $35 \times 42/+2 \times$

Example

Input



Output



Stack of operators

Example

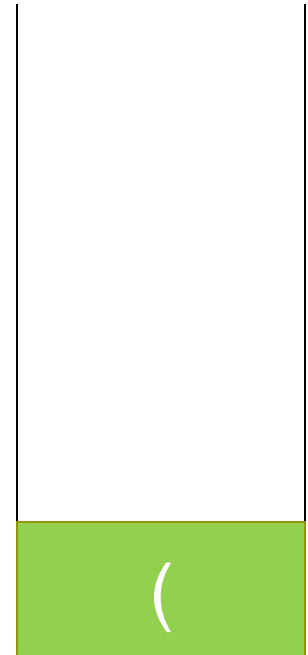
Input

(3 × 5 + 4 / 2) × 2



Output

3



Stack of operators

Example

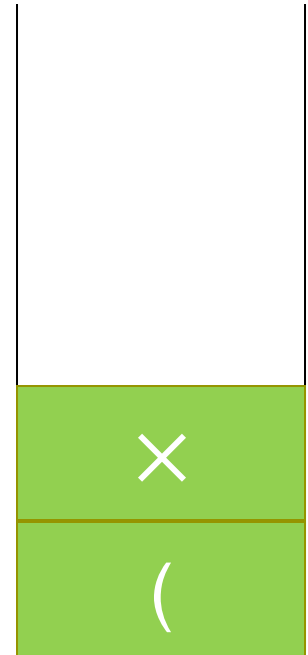
Input

(3 × 5 + 4 / 2) × 2



Output

3



Stack of operators

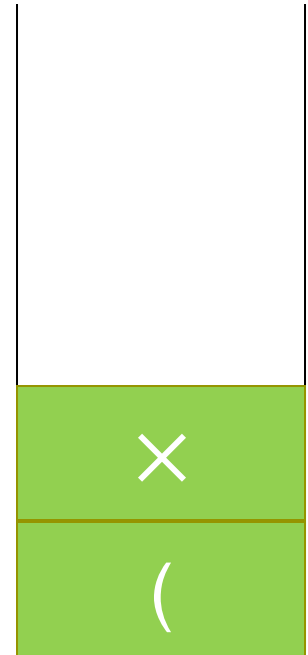
Example

Input

(3 × 5 + 4 / 2) × 2

Output

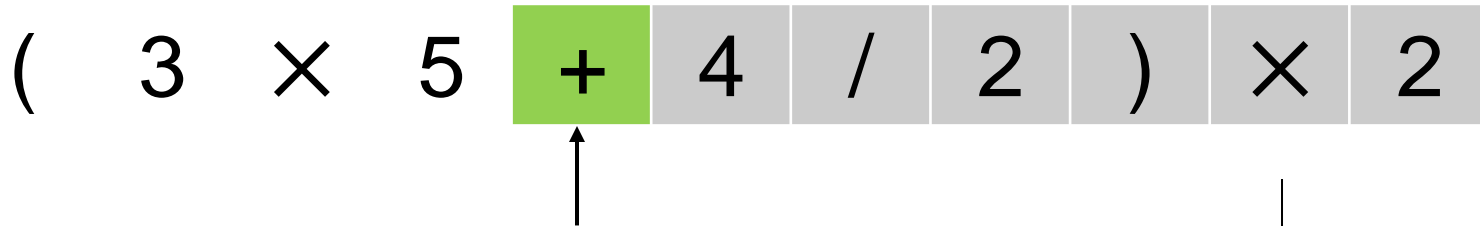
35



Stack of operators

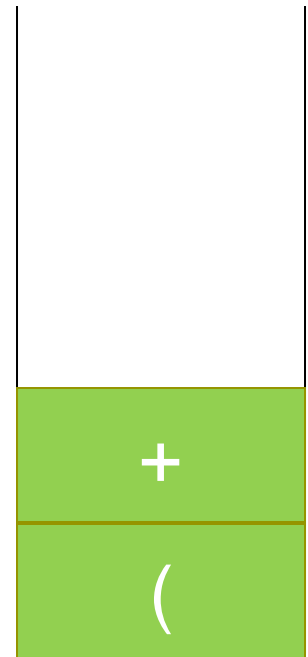
Example

Input



Output

35×



Stack of operators

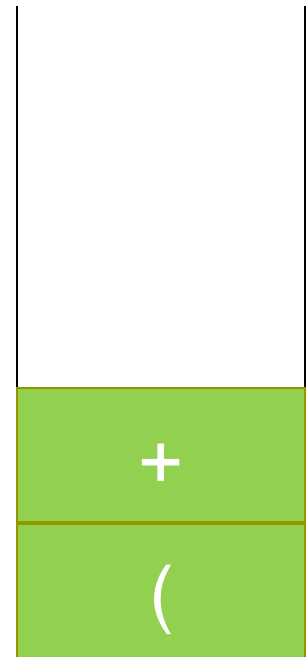
Example

Input

(3 × 5 + 4 / 2) × 2

Output

35 × 4



Stack of operators

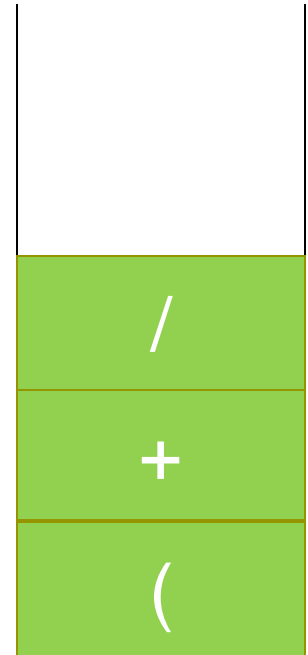
Example

Input

(3 × 5 + 4 / 2) × 2

Output

35 × 4



Stack of operators

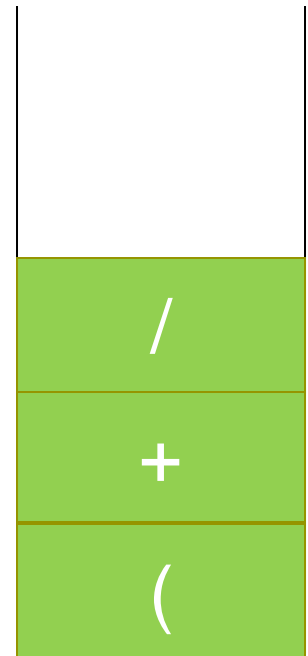
Example

Input

(3 × 5 + 4 / 2) × 2

Output

35 × 42



Stack of operators

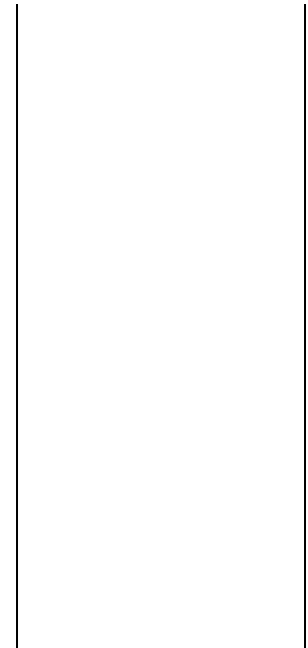
Example

Input

(3 × 5 + 4 / 2) × 2

Output

35×42/+



Stack of operators

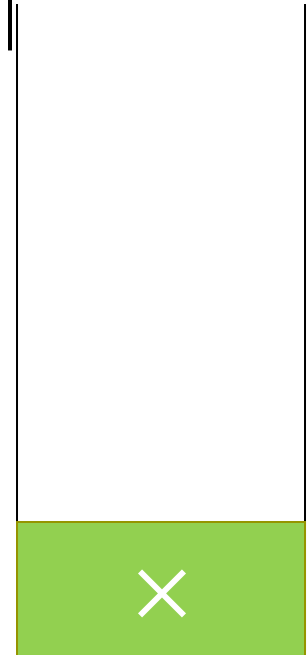
Example

Input

(3 × 5 + 4 / 2) × 2

Output

35×42/+



Stack of operators

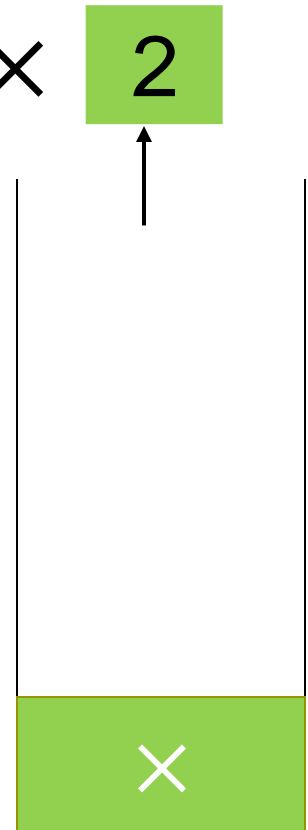
Example

Input

(3 × 5 + 4 / 2) × 2

Output

35×42/+2



Stack of operators

Example

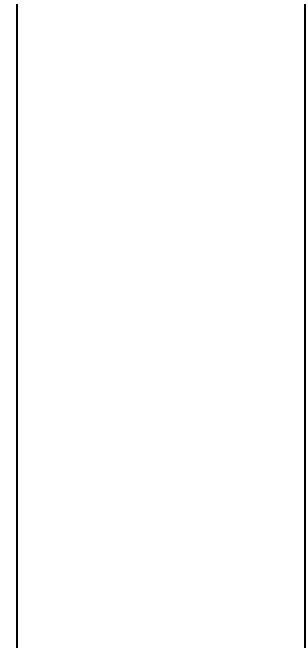


Input

(3 × 5 + 4 / 2) × 2

Output

35×42/+2×



Stack of operators