Discussion 3

Biman Tang, Guoteng Rao

UM-SJTU Joint Institute

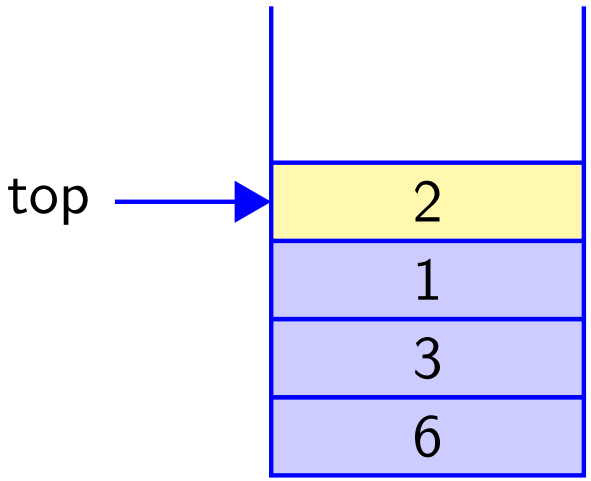
October 16, 2012

CONTENT

1. STACK
   - STACK MODEL
   - POSTFIX EXPRESSIONS
   - INFIX TO POSTFIX CONVERSION
   - POSTFIX EVALUATION
2. QUEUE

Stack Model

- LIFO list – last in, first out.
- Insertions and deletions can be performed in only one position – the end of the list, called the **top**.
- **push**, which is equivalent to an insert.
- **pop**, which deletes the most recently inserted element.



Postfix Expressions

POSTFIX Every operator follows all of its operands.

Example 1

INFIX

$$3 + 4$$

POSTFIX

$$3 \quad 4 \quad +$$

Example 2

INFIX

$$6 \times [5 + (2 + 3) \times 8 + 3]$$

POSTFIX

$$6 \quad 5 \quad 2 \quad 3 \quad + \quad 8 \quad \times \quad + \quad 3 \quad + \quad \times$$

Infix to Postfix Conversion

RULES

We start with an initially empty stack. We concentrate on a small version of the general problem by allowing only the operators $+$, $\times$, (,).

- Operand Immediately place it onto the output.
- Right Parenthesis Pop the stack, writing symbols until we encounter a (corresponding) left parenthesis, which is popped but not output.
- Any other symbol($+$, $\times$, () Pop entries from the stack until we find an entry of lower priority. But we never remove a '(' from the stack except when processing a ')'. When poping is done, we push the operator onto the stack.
- End of input Pop the stack until it is empty.

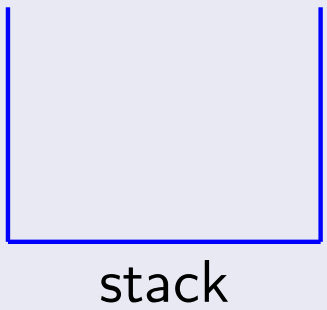
Infix to Postfix Conversion (cont'd)

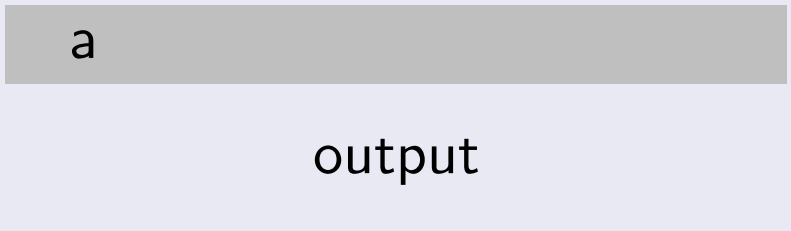
Example 3

$$a + b \times c + (d \times e + f) \times g$$

Solution

PROCESSING CHARACTER a





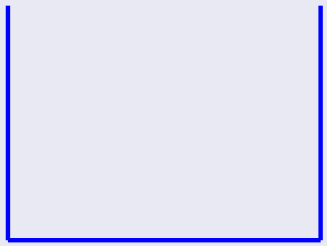
Infix to Postfix Conversion (cont'd)

Example 3

$$a + b \times c + (d \times e + f) \times g$$

Solution

Processing Character b



stack

a b

output

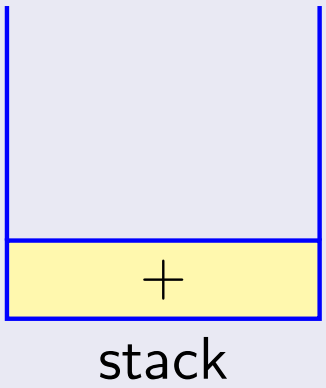
Infix to Postfix Conversion (cont'd)

Example 3

$$a + b \times c + (d \times e + f) \times g$$

Solution

Processing Character +



a b

output

Infix to Postfix Conversion (cont'd)

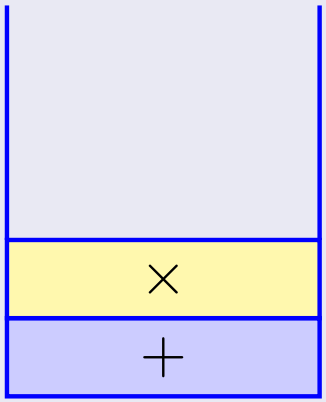
Example 3

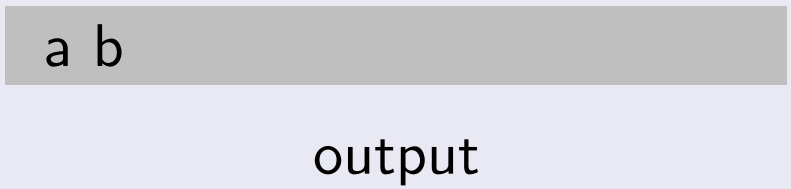
$$a + b \times c + (d \times e + f) \times g$$

Solution

PROCESSING CHARACTER $\times$

The top entry on the stack $+$ has lower precedence than $\times$, so noting is output and $\times$ is put on the stack.





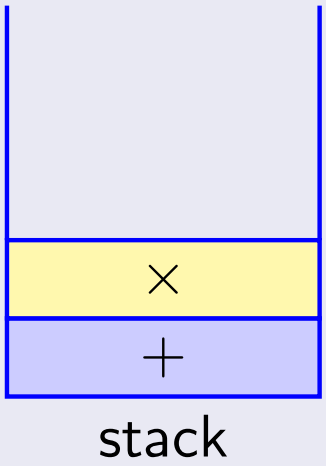
Infix to Postfix Conversion (cont'd)

Example 3

$$a + b \times c + (d \times e + f) \times g$$

Solution

PROCESSING CHARACTER c



a b c

output

Infix to Postfix Conversion (cont'd)

Example 3

$$a + b \times c + (d \times e + f) \times g$$

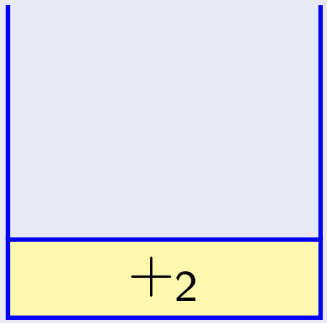
Solution

PROCESSING CHARACTER $+$

Top $\times$ is of higher priority, pop it and place it on the output.

New top $+$ is not of lower but equal priority, pop and output it.

Push current $+$;



a b c $\times$ $+_1$

output

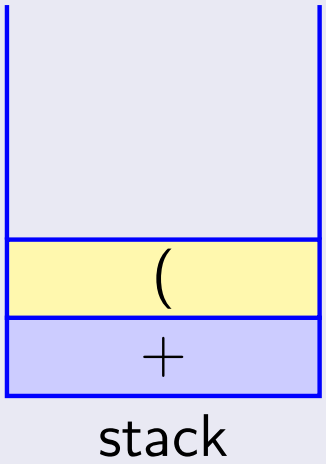
Infix to Postfix Conversion (cont'd)

Example 3

$$a + b \times c + (d \times e + f) \times g$$

Solution

Processing Character (, being of highest precedence, so push it.



a b c × +

output

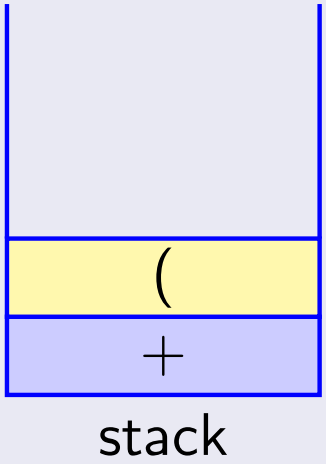
Infix to Postfix Conversion (cont'd)

Example 3

$$a + b \times c + (d \times e + f) \times g$$

Solution

Processing Character d



a b c × + d

output

Infix to Postfix Conversion (cont'd)

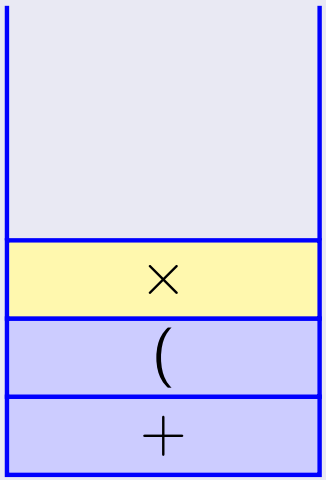
Example 3

$$a + b \times c + (d \times e + f) \times g$$

Solution

PROCESSING CHARACTER $\times$

Since open parentheses do not get removed except when a closed parenthesis is being precessed, there is no output.



stack

a b c × + d

output

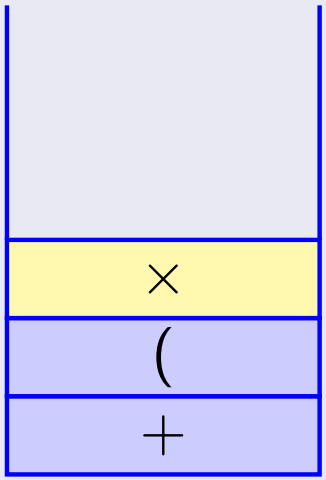
Infix to Postfix Conversion (cont'd)

Example 3

$$a + b \times c + (d \times e + f) \times g$$

Solution

PROCESSING CHARACTER e



a b c × + d e

output

Infix to Postfix Conversion (cont'd)

Example 3

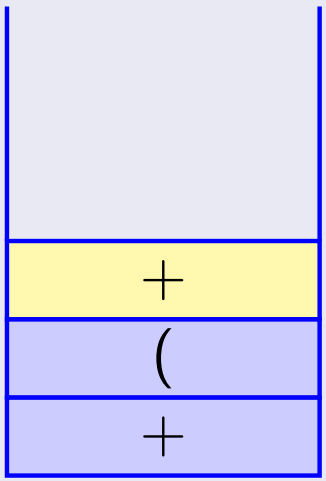
$$a + b \times c + (d \times e + f) \times g$$

Solution

Processing Character $+$, following is f.

Pop and output $\times$ and then push the $+$.

Output f.



a b c × + d e × f

output

Infix to Postfix Conversion (cont'd)

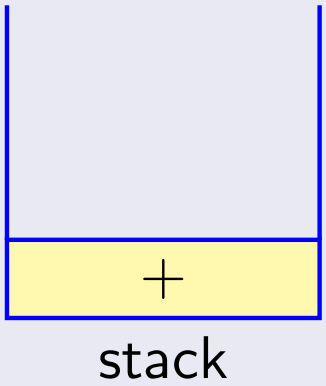
Example 3

$$a + b \times c + (d \times e + f) \times g$$

Solution

PROCESSING CHARACTER)

The stack is emptied back to the (. So output a +.



a b c × + d e × f +

output

Infix to Postfix Conversion (cont'd)

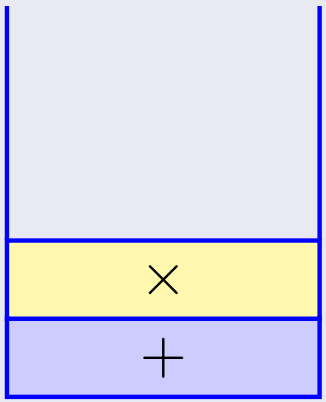
Example 3

$$a + b \times c + (d \times e + f) \times g$$

Solution

Processing Character $\times$, following is g.

Push $\times$ and output g.



a b c × + d e × f + g

output

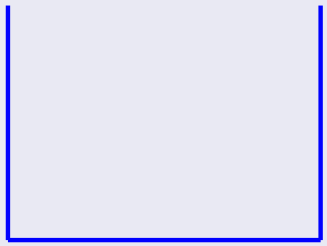
Infix to Postfix Conversion (cont'd)

Example 3

$$a + b \times c + (d \times e + f) \times g$$

Solution

The input is now empty, so we pop and output symbols from the stack until it is empty.



stack

a b c × + d e × f + g × +

output

Postfix Evaluation

RULES

We start with an initially empty stack.

- NUMBER Immediately push it onto the stack.
- OPERATOR It is applied to the 2 numbers (symbols) that are popped from the stack, and the result is pushed onto the stack .

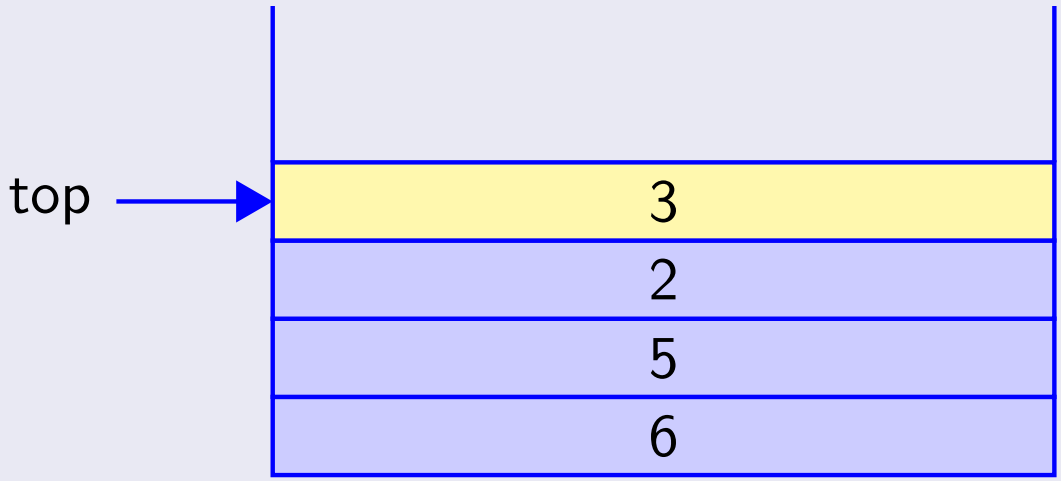
Postfix Evaluation (cont'd)

Example 4

$$6 \quad 5 \quad 2 \quad 3 \quad + \quad 8 \quad \times \quad + \quad 3 \quad + \quad \times$$

Solution

The first 4 numbers are placed on the stack.



Postfix Evaluation (cont'd)

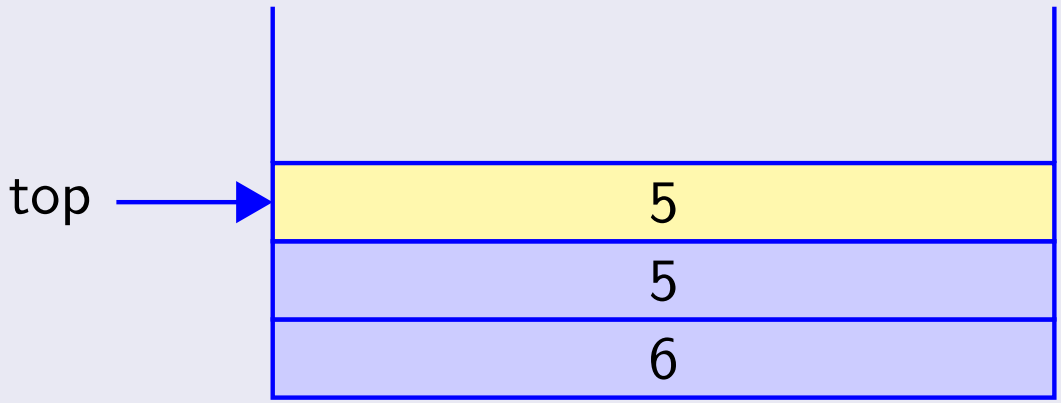
Example 4

$$6 \quad 5 \quad 2 \quad 3 \quad + \quad 8 \quad \times \quad + \quad 3 \quad + \quad \times$$

Solution

PROCESSING CHARACTER $+$

So 3 and 2 are popped and their sum 5 is pushed.



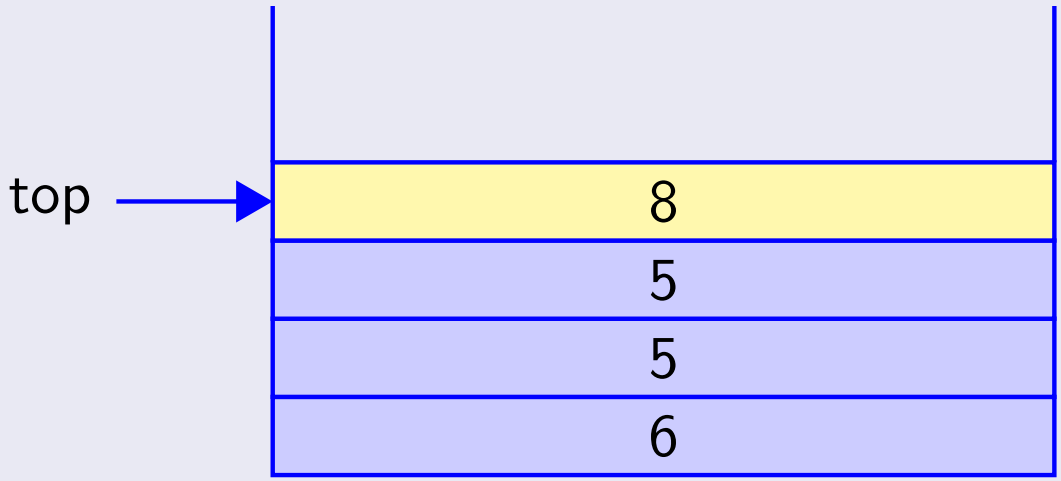
Postfix Evaluation (cont'd)

Example 4

$$6 \quad 5 \quad 2 \quad 3 \quad + \quad 8 \quad \times \quad + \quad 3 \quad + \quad \times$$

Solution

8 is pushed.



Postfix Evaluation (cont'd)

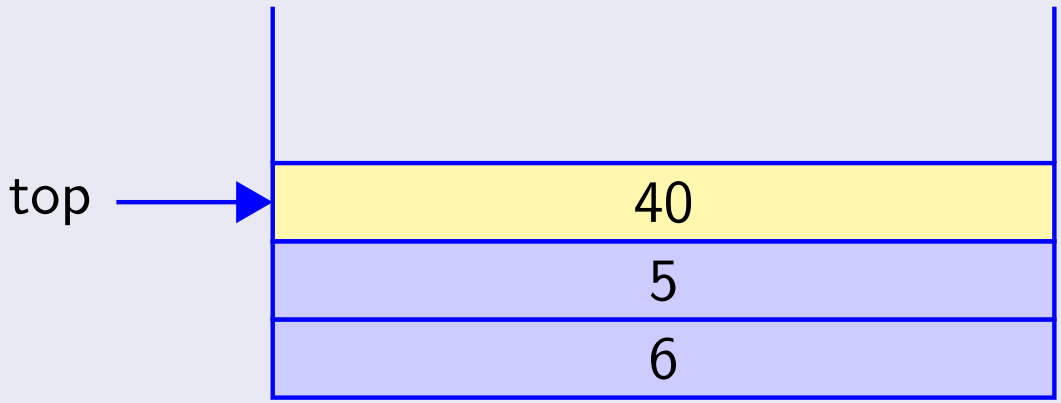
Example 4

$$6 \quad 5 \quad 2 \quad 3 \quad + \quad 8 \quad \times \quad + \quad 3 \quad + \quad \times$$

Solution

PROCESSING CHARACTER $\times$

So 8 and 5 are popped and $5 \times 8 = 40$ is pushed.



Postfix Evaluation (cont'd)

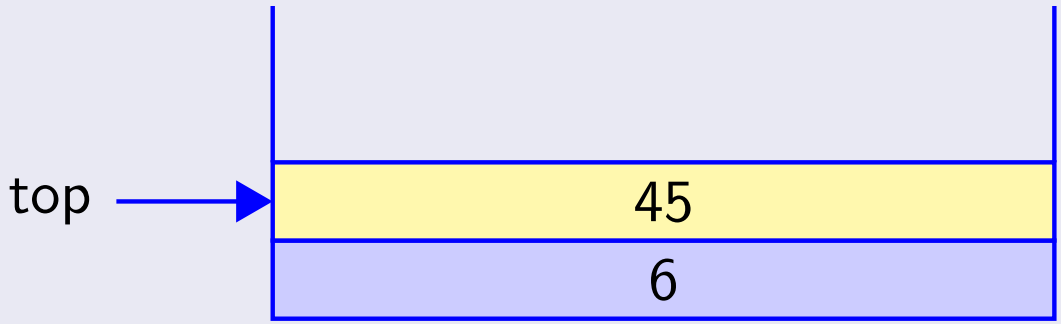
Example 4

$$6 \quad 5 \quad 2 \quad 3 \quad + \quad 8 \quad \times \quad + \quad 3 \quad + \quad \times$$

Solution

PROCESSING CHARACTER $+$

So 40 and 5 are popped and $5 + 40 = 45$ is pushed.



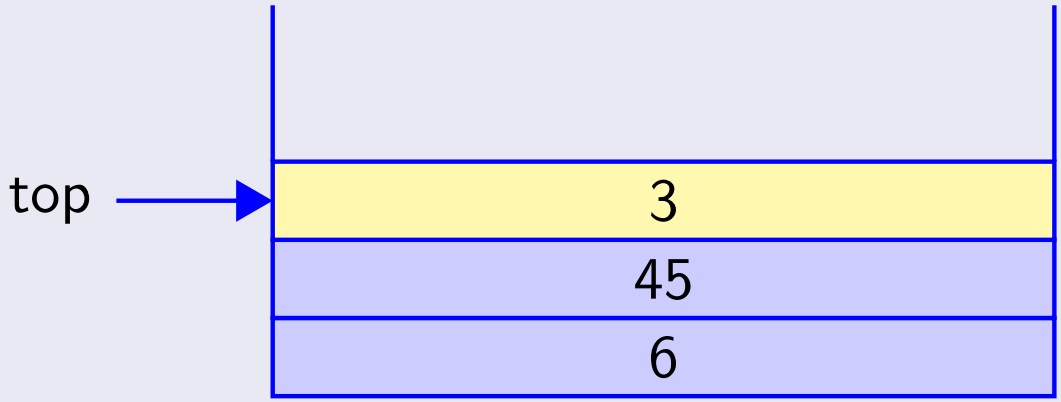
Postfix Evaluation (cont'd)

Example 4

$$6 \quad 5 \quad 2 \quad 3 \quad + \quad 8 \quad \times \quad + \quad 3 \quad + \quad \times$$

Solution

3 is pushed.



Postfix Evaluation (cont'd)

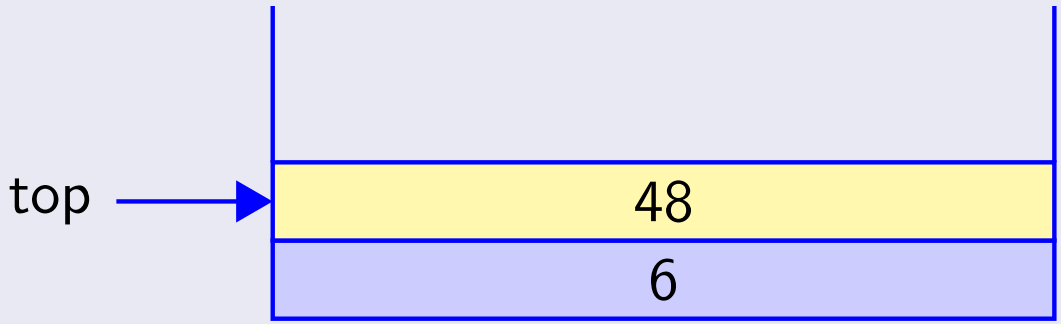
Example 4

$$6 \quad 5 \quad 2 \quad 3 \quad + \quad 8 \quad \times \quad + \quad 3 \quad + \quad \times$$

Solution

Processing Character $+$

$+$ pops 3 and 45 and pushes $45 + 3 = 48$.



Postfix Evaluation (cont'd)

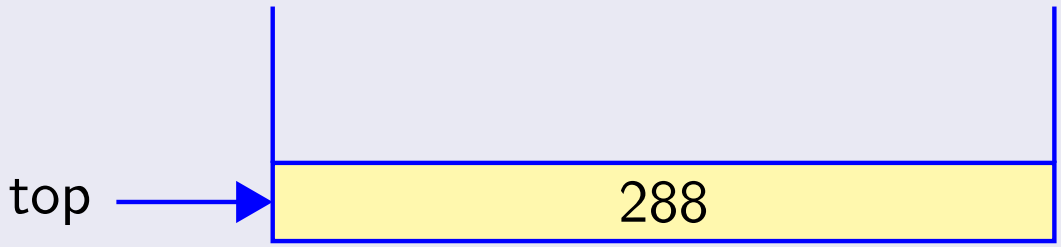
Example 4

$$6 \quad 5 \quad 2 \quad 3 \quad + \quad 8 \quad \times \quad + \quad 3 \quad + \quad \times$$

Solution

PROCESSING CHARACTER $+$

Finally, $\times$ pops 48 and 6 and the result $48 + 6 = 288$ is pushed.

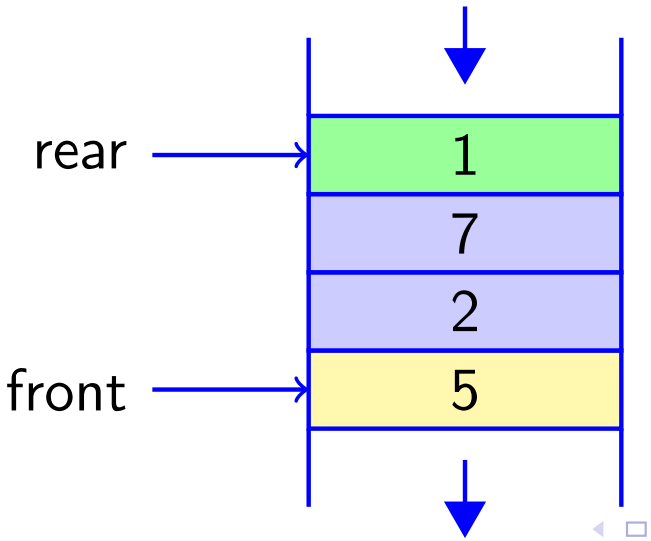


1 STACK

2 QUEUE
- QUEUE MODEL
- MOTHER'S MILK

Queue Model

- FIFO list – first in, first out.
- Insertions is done at one end, whereas deletion is performed at the other end.
- **enqueue**, which inserts an element at the end of the list (called the rear).
- **dequeue**, which deletes (and returns) the element at the start of the list (konwn as the front).



Mother's Milk

Description

Farmer John has three milking buckets of capacity A, B, and C liters.
Initially, buckets A and B are empty while bucket C is full of milk.
Sometimes, FJ pours milk from one bucket to another until the second bucket is filled or the first bucket is empty.
Once begun, a pour must be completed, of course.
Being thrifty, no milk may be tossed out.
Write a program to help FJ determine what amounts of milk he can leave in bucket C when he begins with three buckets as above, pours milk among the buckets for a while, and then notes that bucket A is empty.

Sample 1

$A = 8$, $B = 9$, $C = 10$.
1 2 8 9 10

Mother's Milk

Description

Farmer John has three milking buckets of capacity A, B, and C liters.
Initially, buckets A and B are empty while bucket C is full of milk.
Sometimes, FJ pours milk from one bucket to another until the second bucket is filled or the first bucket is empty.
Once begun, a pour must be completed, of course.
Being thrifty, no milk may be tossed out.
Write a program to help FJ determine what amounts of milk he can leave in bucket C when he begins with three buckets as above, pours milk among the buckets for a while, and then notes that bucket A is empty.

Sample 2

A = 2, B = 5, C = 10.
5 6 7 8 9 10