Infix to Postfix

CSC-114 Data Structure and Algorithms

Slides credit: Ms. Saba Anwar

Infix to Postfix Conversion

- **Infix Expression:** a + b * c
  - Precedence of * is higher than +, So convert the multiplication

    a + (b c *)
  - Convert the addition

    a (b c *) +
  - Remove parenthesis
- **Postfix Expression:** a b c * +

- **Infix Expression: (**a +b) * c
  - Convert addition

    (a b +)* c
  - Convert the multiplication

    (a b + c *)
  - Remove parenthesis
- **Posfix Expression:** a b + c*

Keep in Mind:

1. Relative order of variables is not changed
2. No parenthesis in postfix/prefix
3. Operators are arranged according to precedence
4. If same precedence operators, then evaluate left to right

More Examples

Infix	**Postfix**
(a + b) * (c – d)	a b + c d - *
a – b / (c + d * e)	a b c d e * + / -
((a + b) * c – (d – e))/(f + g)	a b + c * d e - - f g + /
(300+23)*(43-21)/(84+7)	300 23 + 43 21 -* 84 7 + /

- Can we use **stack** to convert an infix expression into post fix expression

Infix to Postfix: Algorithm

- Input: a infix expression string
- Output: a postfix expression string of given input
- Steps:

1. Let P is string, and S is character stack
2. While(input expression has more characters)
3. Read the next character
4. If character is an operand
5. append it to P
6. Else If character is an operator
7. pop S, until top of the S has an element of lower precedence
8. append popped character to P
9. Then push the character to S
10. Else If character is "("
11. Push it to S
12. Else If character is ')'
13. pop S until we find the matching '('
14. append popped character to P
15. pop "(" // '**(**' has the lowest precedence when in the stack but has the highest precedence when in the input
16. End If
17. End While
18. Pop until the stack is empty and append popped character to P

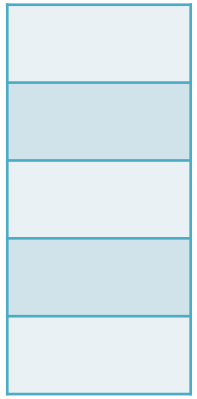
Keep in Mind:

1. Relative order of variables is not changed
2. No parenthesis in postfix
3. Operators are arranged according to precedence
4. If same precedence operators, then evaluate left to right

Infix to Postfix: Example-1

3+4*5/6

- Stack:

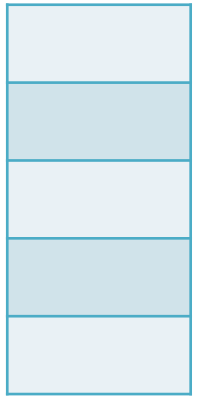


- Output:

Infix to Postfix: Example-1

3+4*5/6

- Stack:

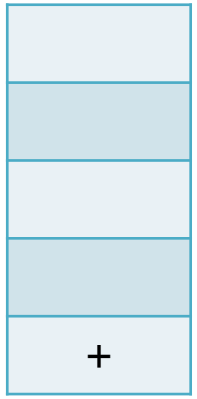


- Output: 3

Infix to Postfix: Example-1

3+4*5/6

- Stack:

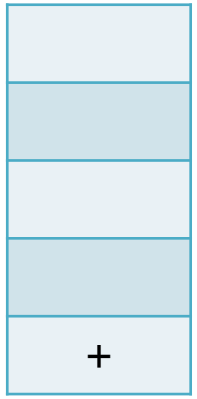


- Output: 3

Infix to Postfix: Example-1

3+4*5/6

- Stack:

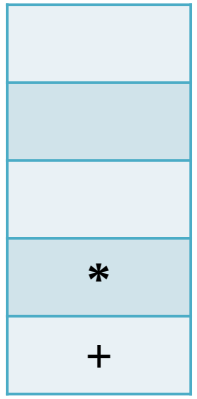


- Output: 3 4

Infix to Postfix: Example-1

3+4*5/6

- Stack:

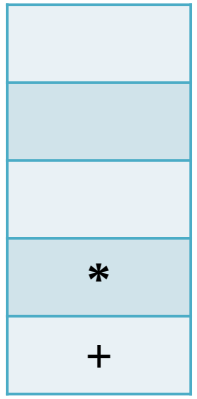


- Output: 3 4

Infix to Postfix: Example-1

3+4*5/6

- Stack:

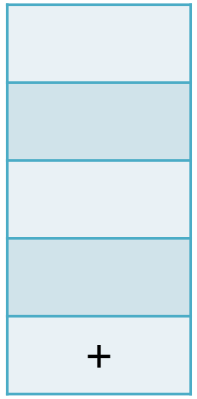


- Output: 3 4 5

Infix to Postfix: Example-1

3+4*5/6

- Stack:

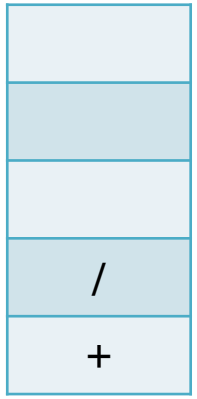


- Output: 3 4 5 *

Infix to Postfix: Example-1

3+4*5/6

- Stack:

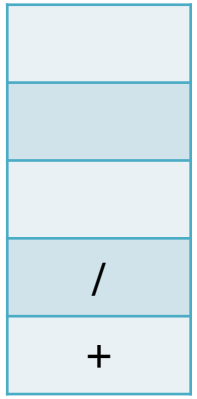


- Output: 3 4 5 *

Infix to Postfix: Example-1

3+4*5/6

- Stack:

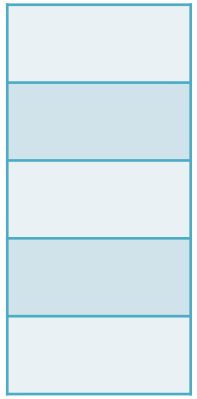


- Output: 3 4 5 * 6

Infix to Postfix: Example-1

3+4*5/6

- Stack:

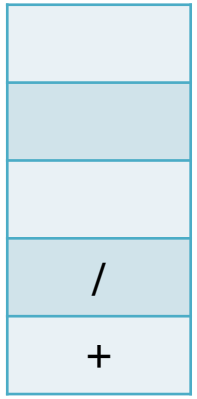


- Output: **3 4 5 * 6 / +**

Infix to Postfix: Example-2

(((1 + 2) * 3) + 6) / (2 + 3)

- Stack:

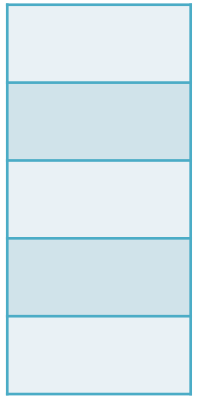


- Output:

Infix to Postfix: Example-2

(((1 + 2) * 3) + 6) / (2 + 3)

- Stack:

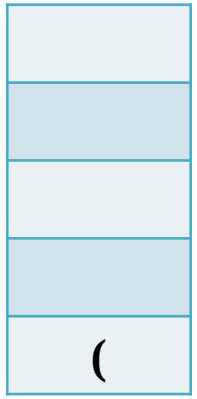


- Output:

Infix to Postfix: Example-2

(((1 + 2) * 3) + 6) / (2 + 3)

- Stack:

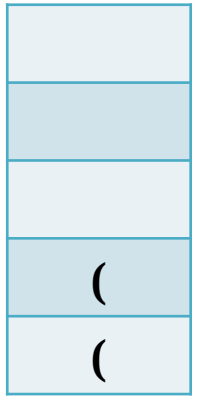


- Output:

Infix to Postfix: Example-2

(((1 + 2) * 3) + 6) / (2 + 3)

- Stack:

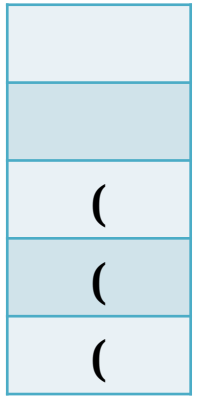


- Output:

Infix to Postfix: Example-2

(((1 + 2) * 3) + 6) / (2 + 3)

- Stack:

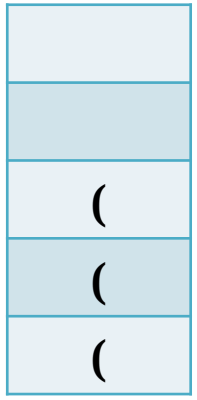


- Output:

Infix to Postfix: Example-2

(((1 + 2) * 3) + 6) / (2 + 3)

- Stack:

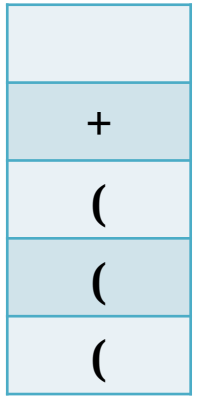


- Output: 1

Infix to Postfix: Example-2

(((1 + 2) * 3) + 6) / (2 + 3)

- Stack:

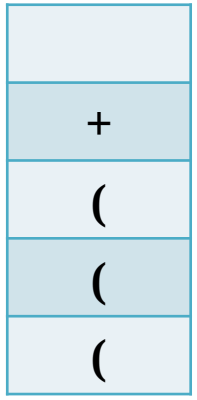


- Output: 1

Infix to Postfix: Example-2

(((1 + 2) * 3) + 6) / (2 + 3)

- Stack:

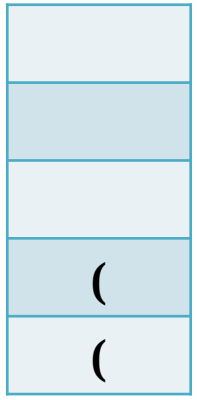


- Output: 1 2

Infix to Postfix: Example-2

(((1 + 2) * 3) + 6) / (2 + 3)

- Stack:

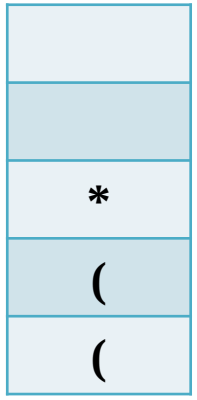


- Output: 1 2 +

Infix to Postfix: Example-2

(((1 + 2) * 3) + 6) / (2 + 3)

- Stack:

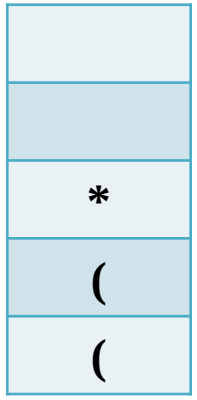


- Output: 1 2 +

Infix to Postfix: Example-2

(((1 + 2) * 3) + 6) / (2 + 3)

- Stack:

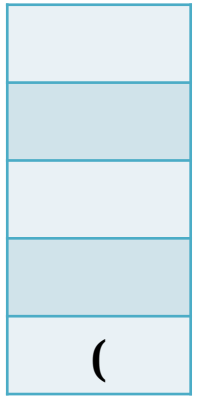


- Output: 1 2 + 3

Infix to Postfix: Example-2

(((1 + 2) * 3) + 6) / (2 + 3)

- Stack:

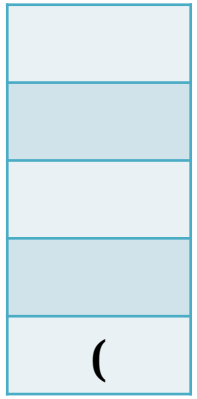


- Output: 1 2 + 3 *

Infix to Postfix: Example-2

(((1 + 2) * 3) + 6) / (2 + 3)

- Stack:

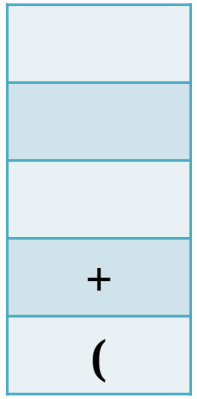


- Output: 1 2 + 3 *

Infix to Postfix Conversion

(((1 + 2) * 3) + 6) / (2 + 3)

- Stack:

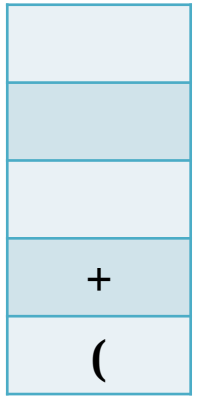


- Output: 1 2 + 3 *

Infix to Postfix: Example-2

(((1 + 2) * 3) + 6) / (2 + 3)

- Stack:

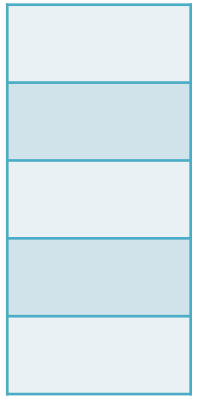


- Output: 1 2 + 3 *

Infix to Postfix: Example-2

(((1 + 2) * 3) + 6) / (2 + 3)

- Stack:

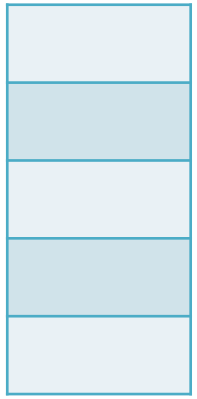


- Output: 1 2 + 3 * +

Infix to Postfix: Example-2

(((1 + 2) * 3) + 6) / (2 + 3)

- Stack:

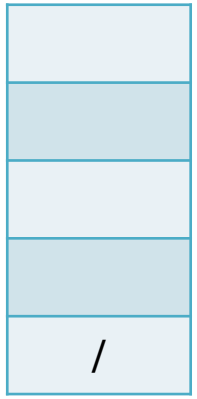


- Output: 1 2 + 3 * +

Infix to Postfix: Example-2

(((1 + 2) * 3) + 6) / (2 + 3)

- Stack:

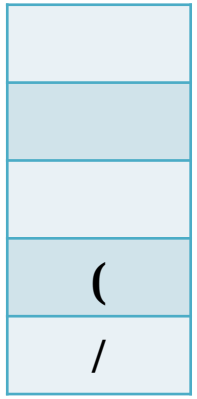


- Output: 1 2 + 3 * +

Infix to Postfix: Example-2

(((1 + 2) * 3) + 6) / (2 + 3)

- Stack:

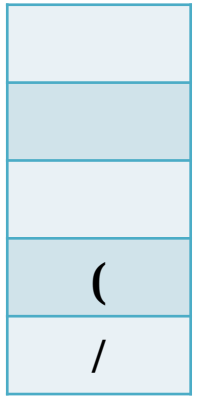


- Output: 1 2 + 3 * +

Infix to Postfix: Example-2

(((1 + 2) * 3) + 6) / (2 ± 3)

- Stack:

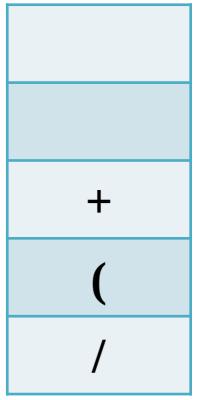


- Output: 1 2 + 3 * + 2

Infix to Postfix: Example-2

(((1 + 2) * 3) + 6) / (2 + 3)

- Stack:

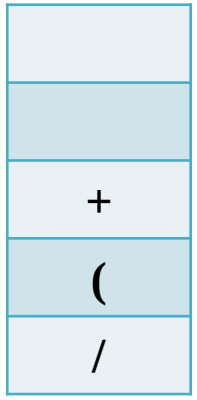


- Output: 1 2 + 3 * + 2

Infix to Postfix: Example-2

(((1 + 2) * 3) + 6) / (2 + 3)

- Stack:

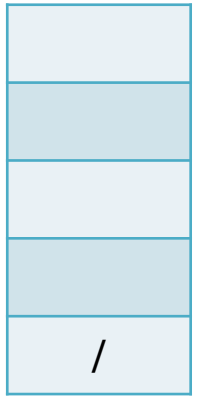


- Output: 1 2 + 3 * + 2 3

Infix to Postfix: Example-2

(((1 + 2) * 3) + 6) / (2 + 3)

- Stack:

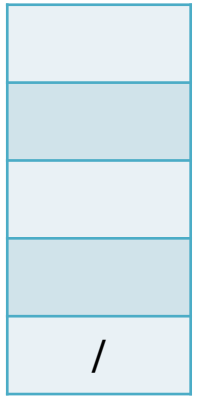


- Output: 1 2 + 3 * + 2 3 +

Infix to Postfix: Example-2

(((1 + 2) * 3) + 6) / (2 + 3)

- Stack:

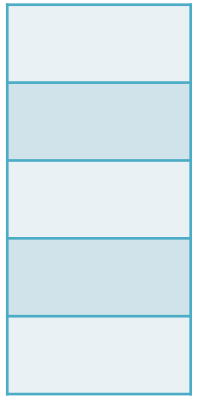


- Output: 1 2 + 3 * + 2 3 +

Infix to Postfix: Example-2

(((1 + 2) * 3) + 6) / (2 + 3)

- Stack:

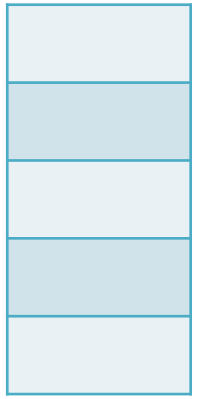


- Output: **1 2 + 3 * + 2 3 + /**

Infix to Postfix: Example-3

3+4-1*5/6

- Stack:

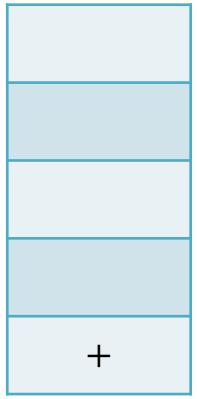


- Output:3

Infix to Postfix: Example-3

3+4-1*5/6

- Stack:

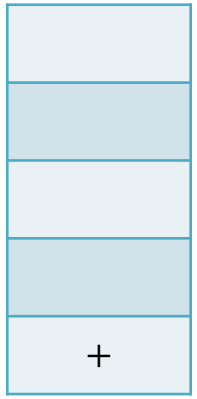


- Output:3

Infix to Postfix: Example-3

3+4-1*5/6

- Stack:

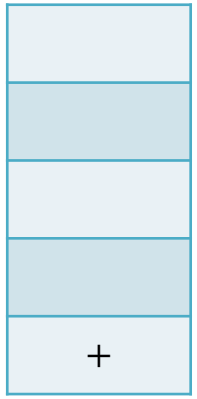


- Output:3

Infix to Postfix: Example-3

3+4-1*5/6

- Stack:

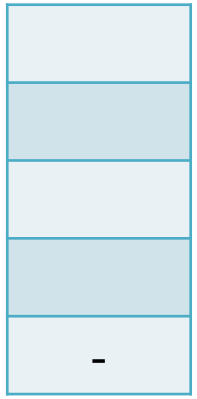


- Output:3 4

Infix to Postfix: Example-3

3+4-1*5/6

- Stack:

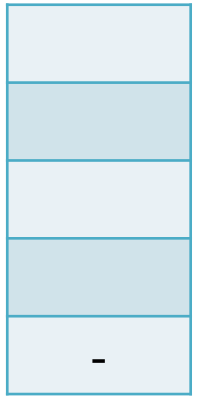


- Output:3 4 +

Infix to Postfix: Example-3

3+4-1*5/6

- Stack:

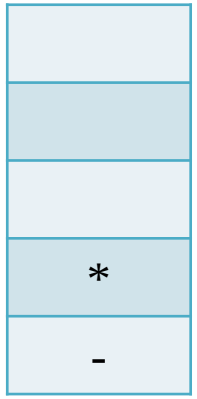


- Output:3 4 + 1

Infix to Postfix: Example-3

3+4-1*5/6

- Stack:

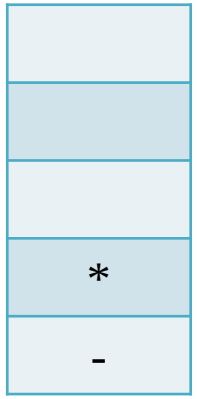


- Output:3 4 + 1

Infix to Postfix: Example-3

3+4-1*5/6

- Stack:

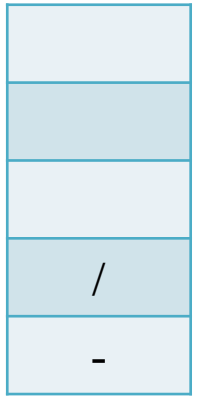


- Output:3 4 + 1 5

Infix to Postfix: Example-3

3+4-1*5/6

- Stack:

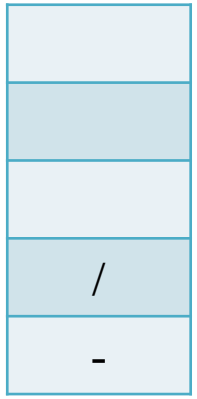


- Output:3 4 + 1 5 *

Infix to Postfix: Example-3

3+4-1*5/6

- Stack:

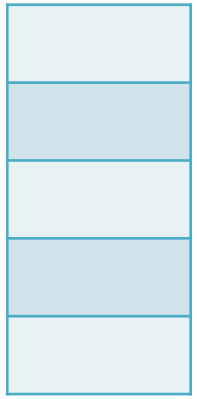


- Output:3 4 + 1 5 * 6

Infix to Postfix: Example-3

3+4-1*5/6

- Stack:



- Output:3 4 + 1 5 * 6 / -