Stacks -4

Lecture-48

Raghav Garg

Today's checklist

Infix, Prefix, Postfix

Infix Expressions

'human readable & solve'

'BODMAS'

s/str = "2 + 6 * 4 / 8 - 3" i

D, M > A, S

2 1

we will need 2 stacks

v1 op v2

Rules :

1) If s[i] is a digit, then push into val

2) If s[i] is +, -, *, /

a) if op stack is empty then push

b) if pr(st top) < pr(s[i]) then push

c) w pr(st.top) >= pr(s[i]) → Kaam Solve

char ch = -

int val2 = 3

int val1 = 5

int ans = 5 - 3 = 2

Infix Evaluation → 'VIMP'

'BODMAS'

D, M > A, S
1 1
2 1

s/str = "2 + 6 * 4 / 8 - 3"

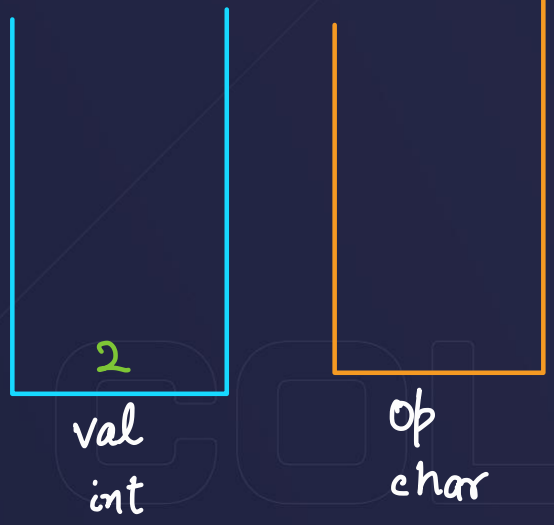
we will need 2 stacks

v1 op v2

Rules :

1) If s[i] is a digit, then push into val

2) If s[i] is +, -, *, /
   a) if op stack is empty then push
   b) if pr(st top) < pr(s[i]) then push
   c) w pr(st.top) >= pr(s[i]) → Kaam Solve



char ch = -

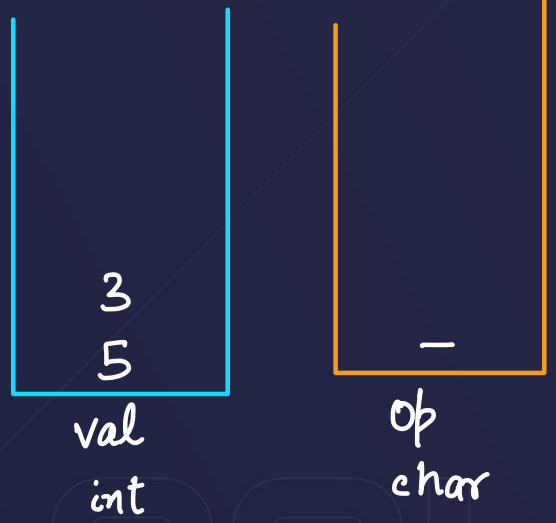
int val2 = 3

int val1 = 5

int ans = 5 - 3 = 2

Infix Evaluation (with brackets)

s/str = " 1 + (2 + 6) * 4 / 8 - 3 "



1) If s[i] is a digit, then push into val

2) If s[i] is +, -, *, /, (,)

a) if op stack is empty then push

e) if pr(st top) < pr(s[i]) then push

b) if (s[i] == '(') push

c) if (op.top() == '(') push

f) w pr(St.top) >= pr(s[i]) → Kaam Solve

d) If (s[i] == ')')

Kaam, jabtak '(' pop the C

char ch = +

int val2 = 4

int val1 = 1

int ans = 1 + 4 = 5

Prefix Expressions

In "2 + 6"

Pre "+ 2 6"

In → v1 op v2

Pre → op v1 v2

Post → v1 v2 op

In 2 + 6 * 4 / 8 - 3

→ 2 + * 6 4 / 8 - 3

→ 2 + / * 6 4 8 - 3

→ + 2 / * 6 4 8 - 3

Pre - + 2 / * 6 4 8 3

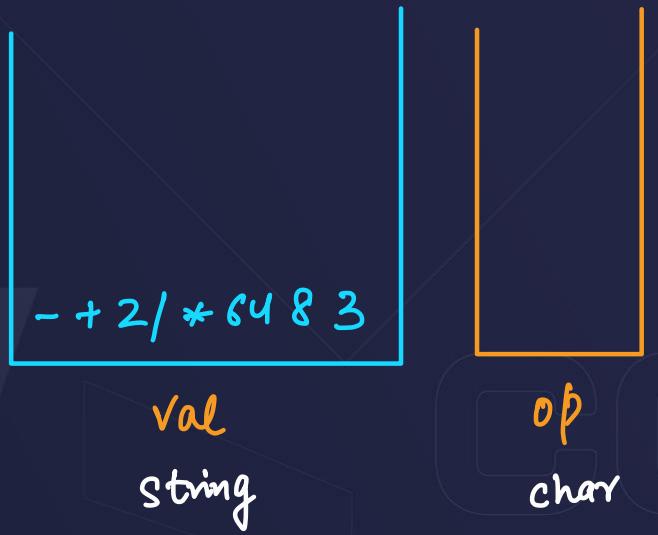
Infix to Prefix Conversion

prefix

op V1 V2

Eg 2 + 6 * 4 / 8 - 3

i



Char ch = -

string val2 = 3

string val1 = + 2 / * 6 4 8

string ans = - + 2 / * 6 4 8 3

Infix to Prefix Conversion

str = "(7 + 9) * 4 / 8 - 3"

+ 7 9 * 4 / 8 - 3

* + 7 9 4 / 8 - 3

/ * + 7 9 4 8 - 3

- / * + 7 9 4 8 3 prefix

-/*+79483

Postfix Expressions

In "2 + 6"

Post "2 6 +"

In → v1 op v2

Pre → op v1 v2

Post → v1 v2 op

In "(7 + 9) * 4 / 8 - 3"

→ 7 9 + * 4 / 8 - 3

→ 7 9 + 4 * / 8 - 3

→ 7 9 + 4 * 8 / - 3

Post → 7 9 + 4 * 8 / 3 -

79+4*8/3-

Infix to Postfix Conversion

↓

ditto same as

infix to prefix

↳ solve → v1 v2 op

Postfix Evaluation (Simplest)

S = 7 9 + 4 * 8 / 3 -

i

int val2 = 3

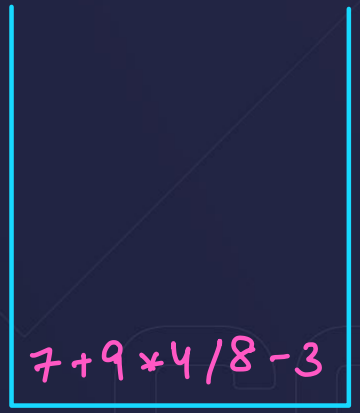
int val1 = 8

int ans = 8 - 3 = 5

Postfix to Infix Conversion

Infix → v1 op v2

S = 7 9 + 4 * 8 / 3 −
i



string val2 = 3

string val1 = 7+9*4/8

string ans = 7+9*4/8-3

Postfix to Prefix Conversion

↓

Copy paste of Post to In code

↓

only diff is → solve

↓

In: v1 op v2

Pre op v1 v2

Prefix Evaluation

→ we traverse in reverse order
→ val2, val1

`- / * + 7 9 4 8 3` prefix

i

int val1 = 8

int val2 = 3

int ans = 8 - 3 = 5

Prefix to Infix Conversion

"Homework"

↓

Solved in Class

Prefix to Postfix Conversion

"Homework"

↓

Solved in Class

Summary:

Similar
- Infix Eval
- Infix to Pre
- In to Post

]] same

Similar
- Post Eval
- Post to In
- Post to Pre

]] same

Similar
- Pre Eval
- Pre to In
- Pre to Post

]] same

reverse order
val1, val2

THANK YOU!