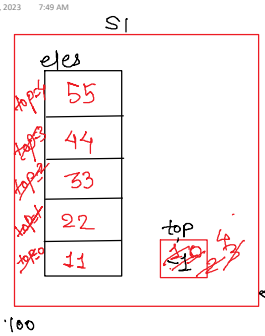
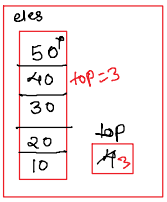


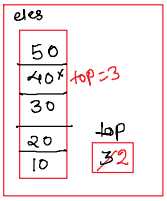

```
BOOL is_full(STACK *p)
{

    return p->top == SIZE-1 ? TRUE : FALSE;
}

void push_element(STACK *p,const ELEMENT *e)
{
    p->top++;
    p->eles[p->top] = *e;
}//add element on stack
```

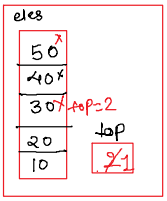
p
100

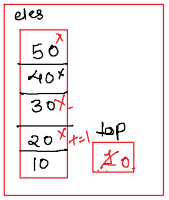
p->eles[0] = 11

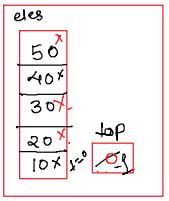
p->top











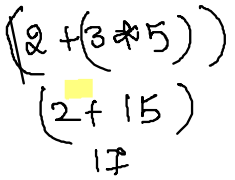

```
BOOL is_empty(STACK *p)
{
    return p->top == -1 ? TRUE : FALSE ;
}

ELEMENT peek_element(STACK *p)
{
    return p->eles[p->top];
}
void pop_element(STACK *p)
{
      p->top--;
}
```

50 40 30 20 10

Tuesday, June 20, 2023 9:27 AM

1. + -
2. * / %
3. ^
4. $

Infix solve manual method



Infix to postfix

(2 + (3 * 5))
(2 + 35*)
235*+

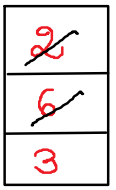
Infix to postfix
((3 + ((6 * 2) / 4)) - (5 ^ (10 $ 4)))
((3 + ((6 * 2) / 4)) - (5 ^ 104$))
((3 + ((6 * 2) / 4)) - 5104$^)
((3 + (62* / 4)) - 5104$^)
((3 + 62*4/) - 5104$^)
(362*4/+ - 5104$^)
3 6 2 * 4 / + 5 10 4 $ ^ -

Infix to prefix
((3 + ((6 * 2) / 4)) - (5 ^ (10 $ 4)))
((3 + ((6 * 2) / 4)) - (5 ^ $104))
((3 + ((6 * 2) / 4)) - ^5$104)
((3 + (*62 / 4)) - ^5$104)
((3 + /*624) - ^5$104)
(+3/*624 - ^5$104)
-+3/*624^5$104

Tuesday, June 20, 2023 10:20 AM

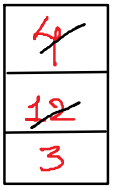
3 6 2 * 4 / + 5 10 4 $ ^ -

3 is read is operand push on stack
6 is read is operand push on stack
2 is read is operand push on stack

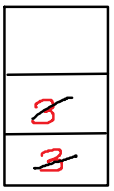


*is operator is read pop() twice
1st pop() === 2 == right operand
2nd pop() === 6 == left operand
After evaluation 6*2=12 as operand hence push

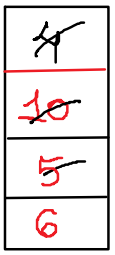
4 is read is operand push on stack
/ is read operator pop() twice
1st pop() === 4 ==== right operand
2nd pop() === 12 ==== left operand
After evaluation 12/4=3 as operand hence push



+ is read is operator == pop() twice
1st pop() === 3 === right operand
2nd pop() === 3 ==== left operand
After evaluation 3+3 = 6 as operand hence push



5 is read is operand push on stack
10 is read is operand push on stack
4 is read is operand push on stack

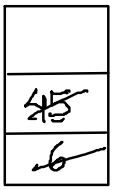


$ is read is operator pop() twice
1st pop() === 4 === right operand
2nd pop() === 10 === left operand
After evaluation 10 $ 4 = 40 as operand hence push

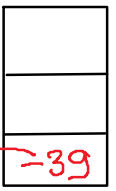
00000101
00101000

101101

2nd pop() == 5 === left operand
After evaluation 5 ^ 40 ==== 45 as operand push



- Is read is operator === pop() twice
1st pop() === 45 === right operand
2nd pop() === 6 ===left operand
6 - 45 = -39 as operand hence push

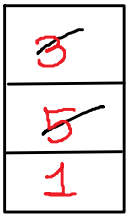


pop() de
which is left on stack
it is a result of
postfix eval'n

Tuesday, June 20, 2023 10:55 AM

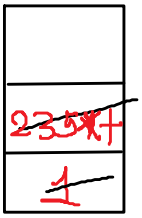
2 + 3 * 5 - 1
-+2*351
153*2+-

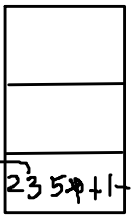
1 is read is operand push on stack
5 is read is operand push on stack
3 is read is operand push on stack
*Is read is operator pop() twice
1st pop() === 3 ==left operand
2nd pop() == 5 ==right operand
After conversion 35* is operand hence push



2 is read is operand push on stack
+ is read is operator pop() twice
1st pop() === 2 == left operand
2nd pop() == 35* === right operand
After conversion 235*+ as operand hence push

- is read is operator pop() twice
1st pop() ==== 235*+ === left operand
2nd pop() === 1 ==== right operand
After conversion 235*+1- as operand hence push





pop the ele which is left in stack is postfix for given prefix