تمرین سری دوم

① یک پشته را از [0] و دیگری را از [n-1] در نظر می‌گیریم./

int top1 = 0 , top2 = n-1

```
push1 (x)
1. if (top1 < top2-1)
2.     top1++
3.     arr[top1] = x
4. else Error ("overflow!")
```

```
Push2 (x)
1. if (top1 < top2-1)
2.     top2--
3.     arr[top2] = x
4. else Error ("overflow!")
```

```
Pop1()
1. if (top1 >= 0)
2.     int x = arr[top1]
3.     top1--
4.     return x
5. else Error ("Stack is empty!")
```

```
Pop2()
1. if (top2 < arr.size)
2.     int x = arr[top2]
3.     top2++
4.     return x
5. else Error ("stack is empty!")
```

---

② Stack S1, S2 داشته باشیم. ساخت FIFO

```
enQueve (x) ---> O(n)
1. while (!S1.empty()) // همه‌ی المان‌ها را از S1 به S2 منتقل کنیم
2. |   S2.push(S1.top());
3. |   S1.pop
4. S1.push(x) // ورودی را به S2 اضافه کنیم
5. while (!S2.empty()) // همه المان‌ها را به S1 برگردانیم
6. |   S1.push(S2.top())
7. |   S2.pop()
```

قدیمی ترین ورودی را بالای Stack قرار می‌دهیم تا

* فرض می‌کنیم S1 و S2 overflow پیش نمی‌دهد.

```
O(1) ---- deQueve()
1. if (S1.empty)
2.     Error ("Queve is empty")
3. else
4.     int x = S2.top()
5.     S2.pop();
6.     return x
```

---

③ Queve q_1, q_2 / stack_size = 0 داشته باشیم LIFO ساخت، Queve1 قرار می‌دهیم تا قدیمی ترین ورودی را بالای

```
Push(x) ---> O(n)
1. stack_size ++
2. q2.enQueve(x)
3. while (!q1.empty())
4. |   q2.enQueve(q1.front());
5. |   q1.deQueve()
6. swap(q1, q2)
```

```
pop()
1. if (q1.empty())
2.     return -1
3. else return q1.front()
```

```
pop()
1. if (q1.empty())
2.     Error ("stack is empty!")
3. else
4. |   q1.deQueve()
5. |   stack_size --
```

Push (a)

1. Struct Node * tmp = malloc (sizeof (Node))
2. tmp→data = x
3. tmp→next = L.top
4. L.top = tmp

Pop ()

(4)

1. if L.top == Null
2. Error ("stack is empty!")
3. else
4. temp = top
5. top = top → next
6. temp→ next = Null
7. free (temp)

---

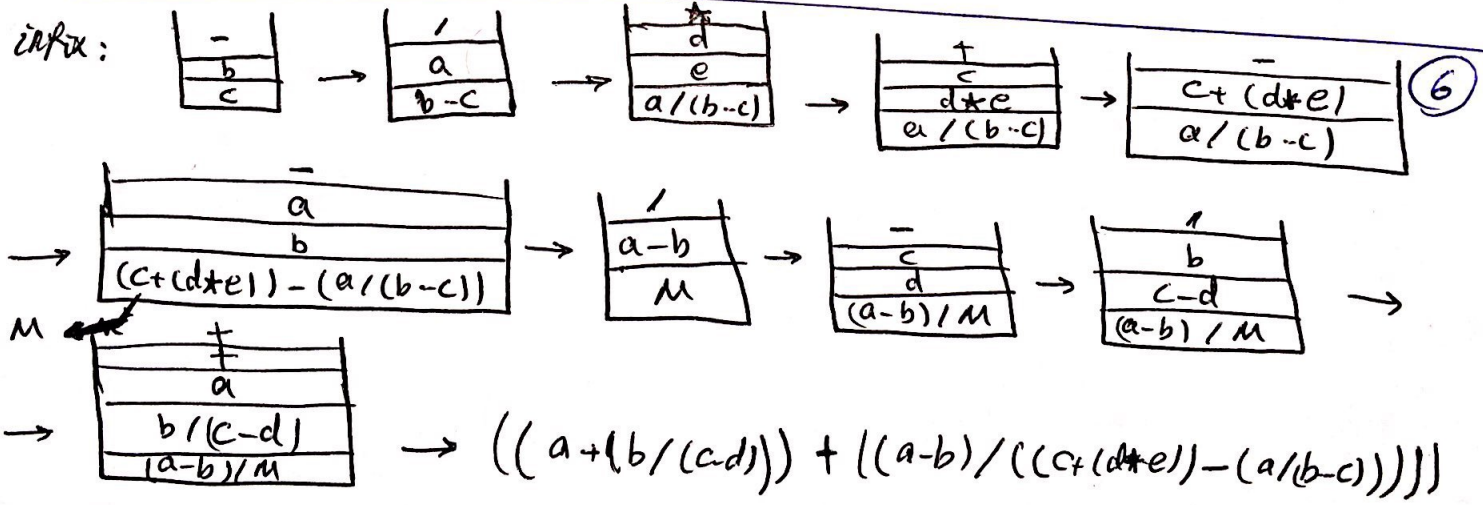
reverse()

1. Node * current = head
2. Node * prev = Null
3. Node * next = Null
4. while (current != Null)
5. next = current → next
6. current → next = prev
7. prev = current
8. current = next
9. head = prev

(5)

---

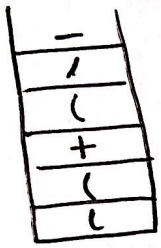
infix:



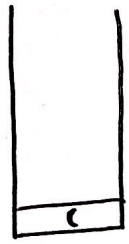
(6)

→ ((a+(b/(c-d)))+((a-b)/((c+(d*e))-(a/(b-c)))))

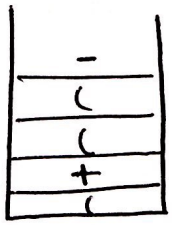
Postfix



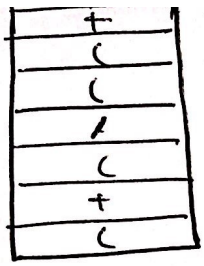
abcd



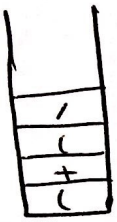
abcd-/+



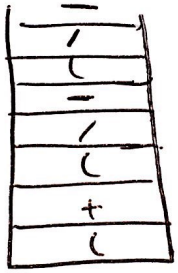
abcd-/+ab



abcd-/+ab-cde



abcd-/+ab-cde*+



abcd-/+ab-cde*+abc

⟵ جواب نهایی : abcd-/+ab-cde*+abc-/-/+