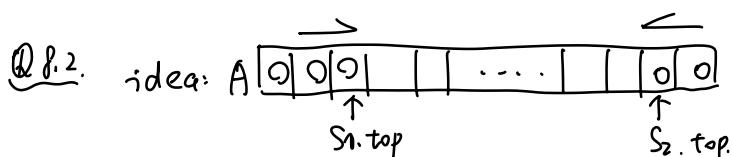


Q. 8.1. 1. S: empty $\Rightarrow \emptyset$  $\Rightarrow \emptyset 4$  $\Rightarrow \emptyset 4 5$  $\Rightarrow \emptyset 4$  $\Rightarrow \emptyset 4 8$  $\Rightarrow \emptyset 4$  $\Rightarrow \emptyset$ 

2. Q: empty

 $\Rightarrow \emptyset$  $\Rightarrow \emptyset 4$  $\Rightarrow \emptyset 4 5$  $\Rightarrow \emptyset 4 5$  $\Rightarrow \emptyset 4 5 8$  $\Rightarrow \emptyset 5 8$  $\Rightarrow \emptyset 8$ 

3. L: empty

 $\Rightarrow \emptyset$  $\Rightarrow \emptyset - \emptyset$  $\Rightarrow \emptyset - \emptyset - \emptyset$  $\Rightarrow \emptyset - \emptyset$  $\Rightarrow \emptyset - \emptyset - \emptyset$  $\Rightarrow \emptyset - \emptyset$  $\Rightarrow \emptyset$ 

Push  $S_1(A, x)$

1. if  $S_1.\text{top} == S_2.\text{top} - 1$
2. error "overflow".
3. else
4.  $S_1.\text{top} = S_1.\text{top} + 1$
5.  $A[S_1.\text{top}] = x.$

Push  $S_2(A, x)$

1. if  $S_2.\text{top} == S_1.\text{top} + 1$
2. error "overflow".
3. else
4.  $S_2.\text{top} = S_2.\text{top} - 1$
5.  $A[S_2.\text{top}] = x.$

Pop  $S_1(A)$

1. if  $S_1.\text{top} == 0$
2. error "underflow".
3. else
4.  $S_1.\text{top} = S_1.\text{top} - 1$
5. return  $A[S_1.\text{top} + 1].$

Pop  $S_2(A)$

1. if  $S_2.\text{top} == A.\text{size} + 1$
2. error "underflow".
3. else
4.  $S_2.\text{top} = S_2.\text{top} + 1$
5. return  $A[S_2.\text{top} - 1].$

Q8.3 Enqueue ( $Q, x$ ).

1. if  $Q.\text{tail} + 1 == Q.\text{head}$
2. error "overflow".
3.  $Q[Q.\text{tail}] = x.$
4. if  $Q.\text{tail} == Q.\text{size}$
5.  $Q.\text{tail} = 1.$
6. else
7.  $Q.\text{tail} = Q.\text{tail} + 1.$

Dequeue ( $Q$ ).

1. if  $Q.\text{tail} == Q.\text{head}$
2. error "underflow".
3. if  $Q.\text{head} == Q.\text{size}$
4.  $Q.\text{head} = 1$
5. else
6.  $Q.\text{head} = Q.\text{head} + 1.$

Q8.V. idea: Construct a object "queue"  $Q$  which uses  $S_1$  to push and  $S_2$  to pop.

Enqueue ( $Q, x$ ).

1. Push ( $Q, S_1, x$ )

$\Rightarrow$  runtime: same as Push

$\Rightarrow O(1)$ .

Dequeue ( $Q$ ).

1. if IsEmpty ( $Q, S_2$ )

2. while not IsEmpty ( $Q, S_1$ )

3. Push ( $Q, S_2, \text{Pop}(Q, S_1)$ ).

4. return Pop ( $Q, S_2$ )

$\Rightarrow$  runtime: worst: need to take the whole  $S_1$  to  $S_2$ .

$\Rightarrow O(n)$ .

amortized: cost per element: mainly

Pop ( $Q, S_1$ ), Push ( $Q, S_2$ ), Pop ( $Q, S_2$ )

$\Rightarrow$  all  $O(1) \Rightarrow O(1)$ .