10.1-6.

Show how to implement a quque using two stacks. Analyze the running time of the queue operations.

Answer.

Using two stacks S_1 and S_2, we can ENQUEUE an element to the queue Q by PUSHing it into S_1 and DEQUEUE the tail element by POPing it from S_2. If stack S_2 is empty when a DEQUEUE is requested, or S_1 is full when an ENQUEUE is requested, we transmit as many elements in S_1 to S_2 as possible. A queue is said to be empty if and only if both of its stacks S_1 and S_2 are empty. Likewise, the queue becomes full when S_1 and S_2 are both full.

```
QUEUE-EMPTY(Q)
1  if STACK-EMPTY(S_1) and STACK-EMPTY(S_2)
2       return TRUE
3  else return FALSE

QUEUE-FULL(Q)
1  if STACK-FULL(S_1) and STACK-FULL(S_2)
2       return TRUE
3  else return FALSE

ENQUEUE(Q, x)
1  if QUEUE-FULL(Q)
2       error "overflow"
3  else
4       if STACK-FULL(S_1)
5            TRANSMIT(S_1, S_2)
6       PUSH(S_1, x)

DEQUEUE(Q)
1  if QUEUE-EMPTY(Q)
2       error "underflow"
3  else
4       if STACK-EMPTY(S_2)
5            TRANSMIT(S_1, S_2)
6       POP(S_2)

TRANSMIT(S_1, S_2)
1  if STACK-EMPTY(S_2)
2       POUR(S_1, S_2)
3  else
4       T = CREAT-STACK()
5       POUR(S_2, T)
6       POUR(S_1, T)
7       POUR(T, S_2)

POUR(S_1, S_2)
1  while not STACK-FULL(S_2)
2       PUSH(S_2, POP(S_1))
```


Email address: informlarry@gmail.com

In the worst case, an ENQUEUE or DEQUEUE operation is performed when stack S_1 is full while stack S_2 is empty. We must transfer at most n elements in S_1 to S_2 and they both take time $O(n)$. However, we could armotize the cost of the transfering over all the ENQUEUE and DEQUEUE operations. So the average-case running time of ENQUEUE and DEQUEUE takes time $O(1)$.