Data Structure
2022, Fall

CSE @ PNU

학과 정보컴퓨터공학부 학번: 201924437 이름 : 김윤하

HOMEWORK #4

제출방법 : 프로그램 코드를 요구하는 프로그래밍을 직접해야 하는 경우를 제외하고 모든 숙제(과제)는 **손으로 답안을 작성(워드프로세서, 편집기 사용하지 않음)**해야 한다. 제출방법은 제출장소에 마감시간 이전에 직접 제출하거나, 온라인에 제출해야 하는 경우는 손으로 작성한 리포트를 스캔한 파일을 지정한 폴더에 제출한다.

1 Suppose an initially empty stack S has performed a total of 25 push operations, 12 top operations, and 10 pop operations, 3 of which generated a StackEmpty exception that was caught and ignored. What is the current size of S? (Text Book Exercise No: R-5.3)

A. 25 개의 push(o), 10 개의 pop(). 이중 3개는 시행이 안 됐으므로 7개의 pop()만 시행됐다. 따라서 현재 S의 사이즈는 25개의 object − 7 개의 pop 작업(object 없어짐) = 18개 이다.

답: 18개

2. Alice has three array-based stacks, A, B, and C, such that A has capacity 100, B has capacity 5, and C has capacity 3. Initially, A is full, and B and C are empty. Unfortunately, the person who programmed the class for these stacks made the push and pop functions private. The only function Alice can use is a static function, transfer(S,T), which transfers (by iteratively applying the private pop and push functions) elements from stack S to stack T until either S becomes empty or T becomes full. So, for example, starting from our initial configuration and performing transfer(A,C) results in A now holding 97 elements and C holding 3. Describe a sequence of transfer operation that starts from the initial configuration and results in B holding 4 elements at the end.

(Text Book Exercise No: C-5.10)

A.

A	B	C	
100	0	0	→ 현재
100	5	3	→ 용량
?	4	?	→ 설명해야 하는 것.

A에서 B를 모두 채우는 연산, 즉 transfer(A, B) 시행 시,

A	B	C
95	5	0

이 된다, 여기서 C로 transfer(B, C)를 하면, B에 2개만 남는다. 즉,

A	B	C
95	2	3

이 된다. 여기서 transfer(C, A) 시행 시,

A	B	C
98	2	0

가 되는데, B를 C로 옮기면 (transfer(B, C))

A	B	C
98	0	2

가 된다. 이 때, 다시 transfer(A.B) 시행 시

A	B	C
93	5	2

가 되고, 한 번만 transfer(B, C) 시행 시

A	B	C
93	4	3

을 얻을 수 있다!!

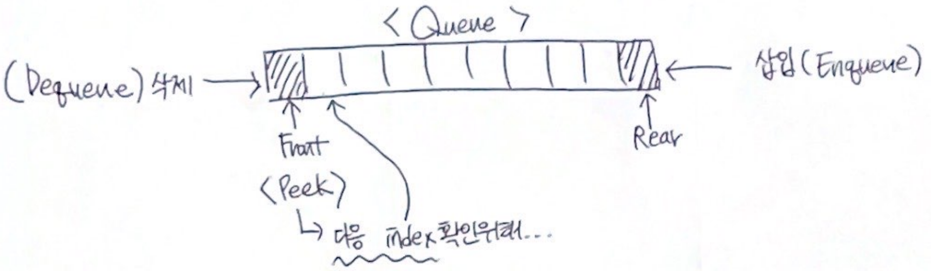
$\boxed{3}$ 수식 $\boxed{((5+2)*(8-3))/4}$ 를 postfix notation (후위표기식)으로 바꾸는 과정을 스택을 사용하여 설명하고 그 결과인 후위표기식 5 2 + 8 3 - * 4 / 을 스택을 이용하여 계산하는 과정을 설명하시오. (Text Book Exercise No: C— 5.8)

A.

다음토큰	스택	출력
없음	공백	없음.
(	(	없음.
(	((	없음.
5	((	5
+	((+	5
2	$(\boxed{(+}$	5 2
$\boxed{)}$	(	5 2 +
*.	(*	5 2 +
(	(*(	5 2 +
8	(*(	5 2 + 8
—	(*(-	5 2 + 8
3	$(*\boxed{(-}$	5 2 + 8 3
$\boxed{)}$	$\boxed{(*}$	5 2 + 8 3 -
$\boxed{)}$	공백	5 2 + 8 3 - *
/	/	5 2 + 8 3 -*.
4	/	5 2 + 8 3 -* 4
완료	공백	5 2 + 8 3 -* 4 /

$\boxed{4.}$ Queue의 자료구조를 설명하고 이를 위한 ADT를 설계하시오.

A. Queue는 Stack의 '후입선출 (Last in First Out)' 과는 다르게, '선입선출 (First in First Out)' 의 구조를 갖고있다. 따라서, 처음 위치와 마지막 위치는 기억해야 한다.



따라서 ADT는

① Enqueue (삽입) : 큐 맨 마지막 자리에 element 추가.
② Dequeue (삭제) : 큐 맨 앞쪽의 element 삭제.
③ Peek : 큐 맨 앞의 위치를 확인함. 다음 index 알아냄.
④ front : 큐의 맨 앞 위치 반환
⑤ rear : 큐의 맨 뒤 위치 반환.

$\boxed{5}$. Describe the output for the following sequence of queue operations : ①enqueue(5), ②enqueue(3), ③dequeue(), ④enqueue(2), ⑤enqueue(8), ⑥dequeue(), ⑦dequeue(), ⑧enqueue(9), ⑨enqueue(1), ⑩dequeue(), ⑪enqueue(7), ⑫enqueue(6), ⑬dequeue(), ⑭dequeue(), ⑮enqueue(4), ⑯dequeue(), ⑰dequeue()

(Text Book Exercise No : R—5.9)

$\boxed{\text{front = rear = -1}}$; 초기화.

```
① enqueue(5) ;
      rear+=1
      queue[rear] = 5        [5]

② enqueue(3) ;
      rear+=1
      queue[rear] = 3        [5|3]

③ dequeue() ;
      front+=1
      return queue[front]    [3]

④ enquequeue(2) ;
      rear+=1
      queue[rear] = 2        [3|2]

⑤ enqueue(8) ;
      rear+=1
      queue[rear] = 8        [3|2|8]

⑥ dequeue();
      front+=1
      return queue[front]    [2|8]

⑦ dequeue();
      front+=1
      return queue[front]    [8]

⑧ enqueue(9)
      rear+=1
      queue[rear] = 9        [8|9]

⑨ enqueue(1)
      rear+=1
      queue[rear] = 1        [8|9|1]

⑩ dequeue();
      front+=1, return queue[front]        [9|1]

⑪ enqueue(7);
      rear+=1, queue[rear] = 7             [9|1|7]

⑫ enqueue(6);
      rear+=1, queue[rear] = 6             [9|1|7|6]

⑬ dequeue();
      front+=1, return queue[front]        [1|7|6]

⑭ dequeue();
      front+=1, return queue[front]        [7|6]

⑮ enqueue(4);
      rear+=1, queue[rear]=4               [7|6|4]

⑯ dequeue();
      front+=1, return queue[front]        [6|4]

⑰ dequeue();
      front+=1, return queue[front]        [4]
```