Data Structure
2022, Fall

CSE @ PNU

학과 정보컴퓨터공학부 학번: 201924437 이름 : 김윤하

HOMEWORK # 5

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 파일를 지정한 폴더에 제출한다.

[1] Queue의 확장개념으로 double-ended queue 혹은 deque (dequeue와 차별되게 'deck'이라 발음한다)은 queue의 양쪽에서 insert와 delete를 지원하는 자료구조이다. (교재 5.3 참조). 이 자료구조의 ADT의 function들은 5.3.1을 보고 정리하시오. 또한 다음의 연산에 대해 그 출력을 설명하시오. (Text Book Exercise No. R-5.10)

insertFront(3), insertBack(8), insertBack(9), insertFront(5), removeFront(), eraseBack(), first(), insertBack(7), removeFront(), last(), eraseBack()

A. ① deque의 ADT

- insertFront(o) : deque 객체의 처음, 즉 front에 obj를 삽입한다.
- insertBack(o) : deque 객체의 마지막, 즉 끝에 obj를 삽입한다.
- removeFront() : deque 객체의 처음, 즉 front의 obj를 제거한다.
  (if, deque isEmpty()라면, 예외처리 필요하다.)
- eraseBack() : deque 객체의 마지막, 즉 끝의 원소(obj)를 제거한다.
  (if, deque isEmpty()라면, 예외처리 필요하다.)
- first() : deque 객체의 처음 obj를 반환해준다.
  (if, deque isEmpty()라면, 예외처리 필요하다.)
- last() : deque 객체의 마지막 obj를 반환해준다.
  (if, deque isEmpty()라면, 예외처리 필요하다.)

A. ② 연산의 출력

- insertFront(3) ; deque 객체 ⇒ (3)
- insertBack(8) : deque 객체 ⇒ (3, 8)
- insertBack(9) : deque 객체 ⇒ (3,8,9)
- insertFront(5) : deque 객체 ⇒ (5,3,8,9)
- removeFront() : deque 객체 ⇒ (3,8,9)
- eraseBack() : deque 객체 ⇒ (3,8)
- first() : deque 객체 ⇒ (3,8), output ⇒ 3
- insertBack(7) : deque 객체 ⇒ (3,8,7)
- removeFront() : deque 객체 ⇒ (8.7)
- last() : deque 객체 ⇒ (8,7), output ⇒ 7
- eraseBack() : deque 객체 ⇒ (8)

2. Describe how to implement the stack ADT using two queues. What is the running time of the push and pop functions in this case? (Text Book Exercise No. C-5.5)

A. ① 두 Queue 의 ADT.

↳ Queue1 , Queue2 라 했을 때, Queue1은 main elements 를 저장하고, Queue2는 부수적인 elements 를 저장한다. (main Queue, 임시 Queue)

- bool isEmpty() : Queue1 이 비어있는지를 반환
- int size() : Queue1 의 size를 반환
- void Push(obj) : Queue1 의 front 에 원소를 삽입한다.
- Obj Pop() : ① while (! Queue1 . isEmpty()) ⇒ main Queue가 비어있음 ⇒ 빈 stack

```
{ Queue2 . enqueue ( Queue1 . dequeue() ); }
Object obj = Queue2 . dequeue() ;
```

② while (! Queue2 . isEmpty()) ⇒ main Queue 에 하나 남을 때까지 임시큐에 넣음,

```
{ Queue1 . enqueue ( Queue2 . dequeue() ); 임시큐 → 다시 main 큐
return obj ; }
```

마지막 하나 반환.

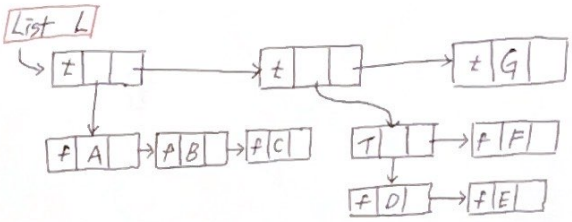
A. ② running time

시간복잡도 위주에서, push()] ⇒ O(1) 의 시간복잡도를 갖는다.
isEmpty()
size()

pop()] ⇒ O(n) 의 시간복잡도를 갖는다.

3. 범용리스트 (generalized list) L 은 $n(n \geq 0)$ 개의 원소를 갖는 유한한 시퀀스이다. 원소 e는 하나의 원자(원자 원소) 아니면 또 다른 범용리스트이다. 원소가 아니라는 것은 L에 sublist 라는 뜻이다. 예를 들어, L이 ((A,B,C), ((D,E),F), G) 라고 하자. L은 3개 원소를 갖는데, 부분리스트 (A,B,C), 부분리스트 ((D,E),F), 원자 G 이다. 부분리스트 ((D,E),F)는 부분리스트 (D,E)와 원자 F, 2개의 원소를 갖는다. 범용리스트 노드를 위한 클래스는 다음과 같이 설계할 수 있다. 이 클래스를 이용하여 임의의 범용리스트 $\$L\$$이 입력으로 주어지면 아래 그림과 같이 범용리스트를 생성하는 방법을 제시하시오.

```
class GeneralizedListNode {
private :
    GeneralizedListNode * next ;
    bool tag ;
    union {
        char data ;
        GeneralizedListNode * down ;
    };
};
```



Lnew = ($, $, ((A,B,C), ((D,E),F), G), $, $) ; put.