9주차 과제 – 스택&큐

※ 입출력에 대한 안내

- 특별한 언급이 없으면 문제의 조건에 맞지 않는 입력은 입력되지 않는다고 가정하라.
- 입출력 예시에서 ↦ 이 후는 각 입력과 출력에 대한 설명이다.

[문제 1-스택] 다음의 스택 ADT를 파이썬 리스트(배열)로 구현하고 테스트하는 프로그램을 작성하시오

- 데이터: 영문자
- 다음의 연산을 지원해야 함
  - push('c') : stack의 top에 데이터를 추가한다.
  - pop ( ) : stack의 top에 있는 데이터를 반환하고 stack에서 제거한다. stack이 비어 있으면 "Stack Empty"를 출력한다.
  - peek( ): stack의 top에 있는 데이터를 화면에 출력한다. stack은 변화하지 않는다. stack이 비어 있으면 "Stack Empty"를 출력한다.
  - duplicate( ): stack의 top에 있는 데이터를 pop해서 두 번 push 한다.
  - upRotate(n): stack의 맨 위 n 개의 데이터를 회전시킨다. 예를 들면 n이 3이고 stack의 top에서부터 elem1, elem2, elem3, ... 이 저장되어 있으면 데이터를 하나씩 위쪽으로 이동시킨다. 맨 위쪽 (top)의 elem1은 n-1번 아래쪽으로 이동해서 스택의 결과는 elem2, elem3, elem1, ...이된다.
    단, n이 데이터의 개수보다 큰 경우에는 아무 작업을 하지 않는다.
  - downRotate(n): stack의 맨 위 n 개의 데이터를 회전시킨다. 예를 들면 n이 3이고 stack의 top에서부터 elem1, elem2, elem3, ... 이 저장되어 있으면 데이터를 하나씩 아래쪽으로 이동시킨다. top에서부터 n번째의 데이터는 top으로 이동해서, 스택의 결과는 elem3, elem1, elem2, ...이된다.
    단, n이 데이터의 개수보다 큰 경우에는 아무 작업을 하지 않는다.
  - print( ) : stack의 모든 데이터를 top에서부터 순서대로 공백 없이 출력한다.

- 입력에 대한 설명 (아래 입출력 예시 참조)
  - 각 연산의 내용이 한 줄에 하나씩 입력되고, 하나의 줄에는 연산의 종류와 그에 필요한 데이터가 입력된다.
  - 연산의 종류: 각 연산 이름은 POP, PUSH , PEEK, DUP, UpR, DownR, PRINT로 주어진다.
  - 프로그램 종료 조건 : 숫자 "-1"이 입력되면 프로그램 종료.

입력 예시 1

```
POP          ↦ pop()
PUSH s       ↦ push('s')
PUSH t       ↦ push('t')
PUSH a       ↦ push('a')
PUSH r       ↦ push('r')
PRINT        ↦ print()
UpR 3        ↦ upRotate(3)
PRINT        ↦ print()
PEEK         ↦ peek()
-1           ↦ 프로그램 종료
```

출력 예시 1

```
Stack Empty  ↦ 1번 POP 연산의 결과
rats         ↦ 6번 PRINT 연산의 결과
atrs         ↦ 8번 PRINT 연산의 결과
a            ↦ 9번 PEEK 연산의 결과
```

입력 예시 2

```
PUSH s       ↦ push('s')
PUSH r       ↦ push('r')
PUSH a       ↦ push('a')
PUSH t       ↦ push('t')
PUSH s       ↦ push('s')
PRINT        ↦ print()
DownR 4      ↦ downRotate(4)
PRINT        ↦ print()
POP          ↦ pop()
POP          ↦ pop()
PRINT        ↦ print()
-1           ↦ 프로그램 종료
```

출력 예시 2

```
stars        ↦ 6번 PRINT 연산의 결과
rstas        ↦ 8번 PRINT 연산의 결과
tas          ↦ 11번 PRINT 연산의 결과
```

입력 예시 3

```
PUSH d       ↦ push('d')
DUP          ↦ duplicate()
PUSH a       ↦ push('a')
PRINT        ↦ print()
-1           ↦ 프로그램 종료
```

출력 예시 3

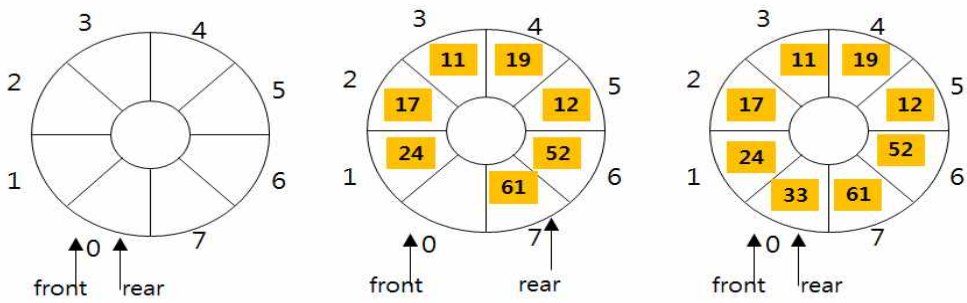
```
add          ↦ 4번 PRINT 연산의 결과
```

[문제 2-큐] 파이썬 리스트(배열)로 구성된 **원형 큐**를 위한 삽입, 삭제 프로그램을 작성하시오.

- **주요 전략 :** 본 문제의 원형 큐에서는 **포화 상태와 공백 상태를 구분**하기 위해 한 자리를 비워둠.
  - front, rear, 배열의 초기 값은 **0**
  - 삽입 시 rear 값을 하나 증가시킨 후 데이터를 큐에 삽입 (출력 예시 1 참고)
  - 삭제 시 front 값을 하나 증가시킨 후 front가 가리키는 데이터를 삭제

- front = rear면 공백 상태로 정의하고, front가 rear보다 하나 앞에 있으면 포화 상태로 정의함

※ **주의 :** front가 맨 앞 원소 위치보다 한 셀 앞 위치를 가리키는 방식으로 정의되었다. **초기 상태에서 맨 처음 삽입되는 위치는 0번이 아니고, 1번이 되어야 함** (그렇지 않으면 본 문제의 입출력 예시와 다른 결과가 나올 수 있음).



○ **입출력 형식:**

1) 첫 번째 라인 : 양의 정수 **q** (원형 큐의 크기)

2) 두 번째 라인 : 양의 정수 **n** (연산의 개수)

3) 세 번째 이후 라인 : 연산이 차례로 입력됨.

※ 연산의 종류는 **I** (삽입), **D** (삭제), **P** (출력)

- **I 10** : 원형 큐에 원소 **10**을 삽입 (큐 원소는 양의 정수).
- **D** : 원형 큐에서 원소를 삭제한 후 해당 배열 원소 값을 **0**으로 치환.
- **P** : **배열 원소 전체**를 차례로 화면에 출력 (입출력 예시 참조).

※ **overflow 발생 시** (즉, 포화 상태에서 삽입을 시도한 경우),
화면에 overflow와 배열 값들을 모두 출력하고 프로그램 종료.

※ **underflow 발생 시** (즉, 공백 상태에서 삭제를 시도한 경우),
화면에 underflow를 출력하고 프로그램 종료.

입력 예시 1

```
6              ↦ q = 6
10             ↦ n = 10
I 10           ↦ 삽입
I 20           ↦ 삽입
P              ↦ 화면출력
I 30           ↦ 삽입
I 40           ↦ 삽입
D              ↦ 삭제
P              ↦ 화면출력
I 50           ↦ 삽입
I 60           ↦ 삽입
I 70           ↦ 삽입
```

출력 예시 1

```
□0 10 20 0 0 0        ↦ 3번째 연산(P)에 의한 출력
□0 0 20 30 40 0       ↦ 7번째 연산(P)에 의한 출력
overflow□60 0 20 30 40 50
                ↦ 10번째 연산(I 70)에서 overflow 발생
```