# 4조 - LinkedList

## 초급자

### 1. 허범

#### 2346 - 풍선 터뜨리기

- 메모리 초과가 계속 발생하는 원인을 못 찾았는데, 팀원들과 같이 코드를 보며 큐로 처리를 하는 과정에서 문제가 있을거 같다는 조언을 듣고 해결할 수 있었습니다.

```
package week_03;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.LinkedList;
import java.util.Queue;
import java.util.StringTokenizer;

public class 풍선터트리기 {
  public static void main(String[] args) throws NumberFormatException, IOException {
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    StringBuilder sb = new StringBuilder();
    StringTokenizer st;

    Queue<balloon> queue = new LinkedList<>();//풍선을 저장할 큐 생성.

    int N = Integer.parseInt(br.readLine());//풍선의 개수 입력받기.

    st = new StringTokenizer(br.readLine());//풍선안에 적힌 수 받기위해 자름.

    for(int i = 1; i <= N; i++) {
      queue.offer(new balloon(i, Integer.parseInt(st.nextToken())));//하나씩 큐에 저장.
    }
    int num = queue.poll().val;//맨 앞 값을 빼고 값을 변수에 저장.
    sb.append(1 + " ");//빌더에 1과 공백을 더해줌.

    while (!queue.isEmpty()) {//큐가 빌때까지 반복.
      if(num > 1) {//앞에서 받은 풍선의 값이 양수이면
        num = (num-1)%queue.size();//이미 풍선을 뺐으니 -1.
        while(num-->0)//값이 0이 될때까지 큐에서 빼서 다시 넣어줌.
          queue.offer(queue.poll());
      }else if(num<0){//값이 음수일 경우.
        num %= queue.size();
        num += queue.size();
        while(num-->0)//값이 0이 될때까지 큐에서 빼서 다시 넣어줌.
          queue.offer(queue.poll());
      }
      num = queue.peek().val;//다음 터뜨릴 풍선의 값을 변수에 저장하고.
      sb.append(queue.poll().index + " ");//빌더에 위치를 붙이고 풍선 제거
    }
    System.out.println(sb);//빌더를 출력.
  }

  static class balloon{
    int index, val;//풍선 클래스에 위치와 풍선에 넣어있는 값을 저장.

    balloon(int index, int val){
      this.index = index;
      this.val = val;
    }
  }
}
```

## 2. 김응철

### 1021 - 회전하는 큐

- Deque<Integer>선언하려했는데 (제 한계로는)indexof를 써야할 것 같아서 LinkedList<Integer>로 선언했습니다. N의 갯수가 짝수일 경우 2로 나누었을때 시작하는 인덱스가 정가운데가 아니어서 -1을 해줘야하나 싶었지만 계산해보니 나눌 필요가 없을것 같았고 실제로 코드에 적용시켜봐도 그랬습니다.

```
import java.util.LinkedList;
import java.util.Scanner;

public class RollingQueue_1021 {

   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);

      int cnt = 0;
      int N = sc.nextInt();// 큐의 크기
      int M = sc.nextInt();// 뽑아낼 숫자 갯수
      int[] index = new int[M];// 뽑아내고 싶은 숫자의 인덱스 저장

      // 1 = removeFirst() 한개
      // 2 = pollFirst()하고 addLast()
      // 3 = pollLast()하고 addFirst()

      LinkedList<Integer> deque = new LinkedList<>();// deque 구현 = indexof를 사용하기 위해 LinkedList로 선언

      for (int n = 1; n <= N; n++) {
         deque.addLast(n);// deque에 숫자를 순서대로 채움. 인덱스와 일치시키기 위함.
      }

      for (int m = 0; m < M; m++) {
         index[m] = sc.nextInt();// 뽑아낼 숫자의 인덱스를 index 배열에 저장.
      }//입력

      int m = 0;
      while (m < M) {// m개의 숫자를 뽑을 때까지 while문으로 반복
         if (deque.peekFirst() == index[m]) {//만약 제일 앞의 숫자가 뽑을 숫자와 일치하면 숫자를 뽑는다.
            deque.removeFirst();
            m++;// 숫자를 뽑앗을 시 m++. 숫자는 제일 앞에서 밖에 뽑지 않기 때문.
         } else {//제일 앞의 숫자가 뽑을 숫자와 일치하지 않으면 2번이나 3번 실시.

            if (deque.size() / 2 < deque.indexOf(index[m])) {//뽑을 숫자의 현재 위치가 뒤쪽인지 앞쪽인지 판단하는 if문.
               //뽑을 숫자의 현재 인덱스가 deque의 현재 사이즈의 반쪽보다 크면 뒤쪽, 작으면 앞쪽임.
               while (!(deque.peekFirst() == index[m])) {//deque의 제일 앞쪽 숫자를 체크하여 뽑을 숫자와 같을때 까지 while문 돌기
                  deque.addFirst(deque.pollLast());// 만약 뽑을 숫자의 인덱스가 더 크면 3번실행
                  cnt++;//2번 실행했으니 count 세주기
               }
            } else {//뽑을 숫자의 현재 위치가가 현재 deque의 정중앙보다 앞쪽일때.
               while (!(deque.peekFirst() == index[m])) {//deque의 제일 앞쪽 숫자를 체크하여 뽑을 숫자와 같을때 까지 while문 돌기
                  deque.addLast(deque.pollFirst());// 만약 뽑을 숫자의 인덱스가 더 작으면 2번실행
                  cnt++;//3번 실행했으니 count 세주기.
               }
            }
         }
      }//while문 반복하면서 cnt를 계속 더해주기
      System.out.println(cnt);//cnt 정답 출력.
   }
}
```

# 숙련자

## 1. 김지언

### 1406 - 에디터

- 리스트를 하나만 사용해 시간초과가 발생했었음.
- 링크드리스트의 가장 앞, 뒤에 대한 삽입/삭제의 경우, 위치를 찾지 않아 시간 복잡도가 O(1)
- 커서를 기준으로 리스트를 두개로 나눠 시간초과가 발생하지 않도록 함

```
package silver.lv3;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.LinkedList;
import java.util.List;
import java.util.StringTokenizer;

// 1406. 에디터
public class Test1406 {
  static int M, idx;
  static List<Character> pre, next;

  public static void main(String[] args) throws IOException {
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    char[] chars = br.readLine().toCharArray();
    pre = new LinkedList<>(); // 커서 왼쪽 문장
    next = new LinkedList<>(); // 커서 오른쪽 문장
    for (int i = 0; i < chars.length; i++) {
      pre.add(chars[i]);
    }
    M = Integer.parseInt(br.readLine());
    for (int i = 0; i < M; i++) {
      StringTokenizer st = new StringTokenizer(br.readLine());
      String key = st.nextToken();
      switch (key) {
      case "L": // 커서를 왼쪽으로 한 칸 옮김
        if (!pre.isEmpty()) // 왼쪽에 문장이 있다면
          next.add(0, pre.remove(pre.size() - 1)); // 왼쪽 마지막 요소를 오른쪽 첫번째 요소로 입력
        break;
      case "D": // 커서를 오른쪽으로 한 칸 옮김
        if (!next.isEmpty()) { // 오른쪽에 문장이 있다면
          pre.add(next.remove(0)); // 오른쪽 첫번째 요소를 왼쪽 마지막 요소로 입력
        }
        break;
      case "B": // 커서 왼쪽에 있는 문자를 삭제함
        if (!pre.isEmpty()) { // 왼쪽에 문장이 있다면
          pre.remove(pre.size() - 1); // 왼쪽 리스트의 마지막 요소 제거
        }
        break;
      case "P": // 커서 왼쪽에 문자 추가함
        pre.add(st.nextToken().charAt(0)); // 왼쪽 리스트에 문자 추가
        break;
      }
    }

    StringBuilder sb = new StringBuilder();
    for (char c : pre) {
      sb.append(c);
    }
    for (char c : next) {
      sb.append(c);
    }
    System.out.println(sb.toString());
  }

}
```

# 2. 현병욱

## 17827 - 달팽이리스트

- List 사용 없이 array로 해결하고자 함.

- array로 문제를 풀었던 팀원들 모두 시간초과 혹은 오답 처리되어 최대한 연산 횟수를 줄이고자 함.
- 최종적으로는 StringBuffer와 수학식 단순화를 통해 해결.

```
package day0820;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.StringTokenizer;

public class boj17827 {
  static int[] snail = new int[200001];
  static int N, M, V;
  public static void main(String[] args) throws NumberFormatException, IOException {
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    StringBuffer sb = new StringBuffer();
    StringTokenizer st = new StringTokenizer(br.readLine(), " ");

    N = Integer.parseInt(st.nextToken()); // 노드 개수
    M = Integer.parseInt(st.nextToken()); // 질문의 횟수
    V = Integer.parseInt(st.nextToken()); // 마지막 노드가 가리키는 노드의 번호

    // 달팽이 입력
    st = new StringTokenizer(br.readLine());
    for(int i = 0; i < N; i++) {
      snail[i] = Integer.parseInt(st.nextToken());
    }

    int l = N - V + 1;
    for(int i = 0; i < M; i++) {
      int tmp = Integer.parseInt(br.readLine());
      if(tmp < N) {
        sb.append(snail[tmp] + "\n");
//        System.out.println(snail.get(tmp));
      }else {
        sb.append(snail[(tmp-V+1) % l + V - 1] + "\n");
//        System.out.println(snail.get((tmp-N)%(N-V+1) + V-1));
      }
    }
    System.out.println(sb.toString());
  }

}
```