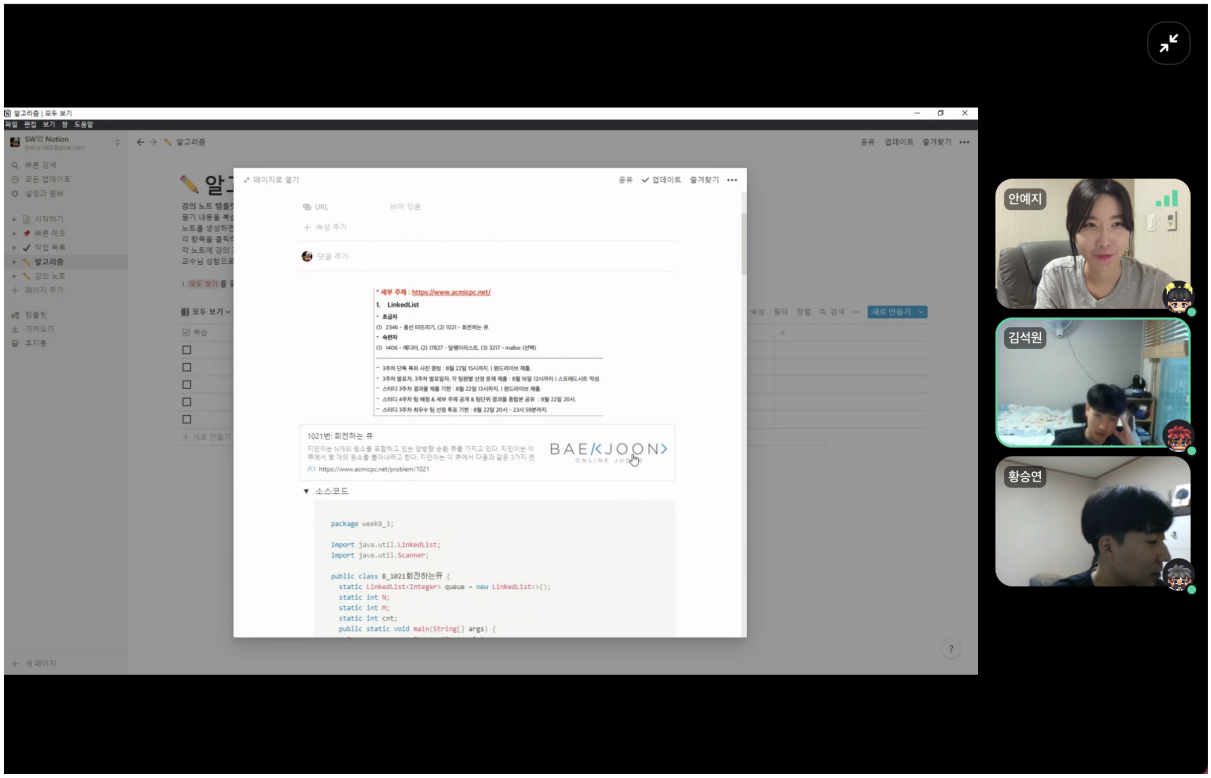
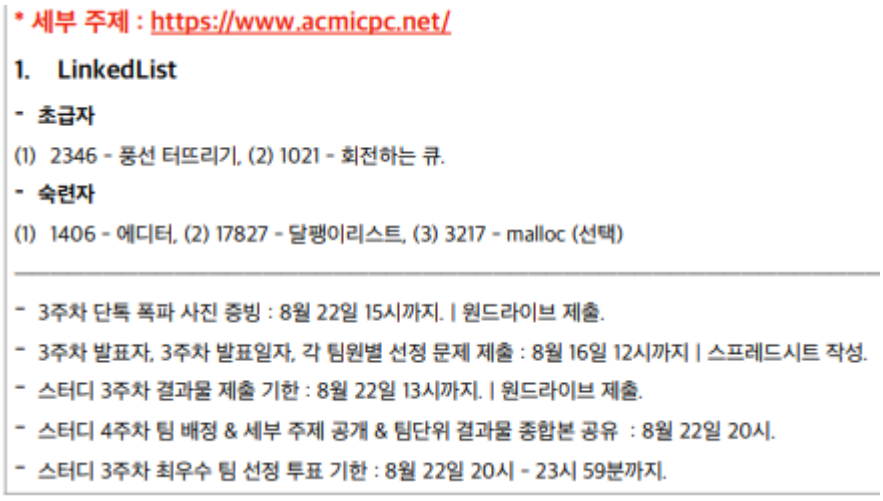
# 8월 3주차 스터디



* 세부 주제 : https://www.acmicpc.net/

1. LinkedList

- 초급자

(1) 2346 - 풍선 터뜨리기, (2) 1021 - 회전하는 큐.

- 숙련자

(1) 1406 - 에디터, (2) 17827 - 달팽이리스트, (3) 3217 - malloc (선택)

---

- 3주차 단톡 폭파 사진 증빙 : 8월 22일 15시까지. | 원드라이브 제출.
- 3주차 발표자, 3주차 발표일자, 각 팀원별 선정 문제 제출 : 8월 16일 12시까지 | 스프레드시트 작성.
- 스터디 3주차 결과물 제출 기한 : 8월 22일 13시까지. | 원드라이브 제출.
- 스터디 4주차 팀 배정 & 세부 주제 공개 & 팀단위 결과물 종합본 공유 : 8월 22일 20시.
- 스터디 3주차 최우수 팀 선정 투표 기한 : 8월 22일 20시 - 23시 59분까지.

1021번: 회전하는 큐

지민이는 N개의 원소를 포함하고 있는 양방향 순환 큐를 가지고 있다. 지민이는 이 큐에서 몇 개의 원소를 뽑아내려고 한다. 지민이는 이 큐에서 다음과 같은 3가지 연산을 수행할 수 있다. 큐에 처음에 포함되어

BAE/KJOON
ONLINE JUDGE

/<> https://www.acmicpc.net/problem/1021

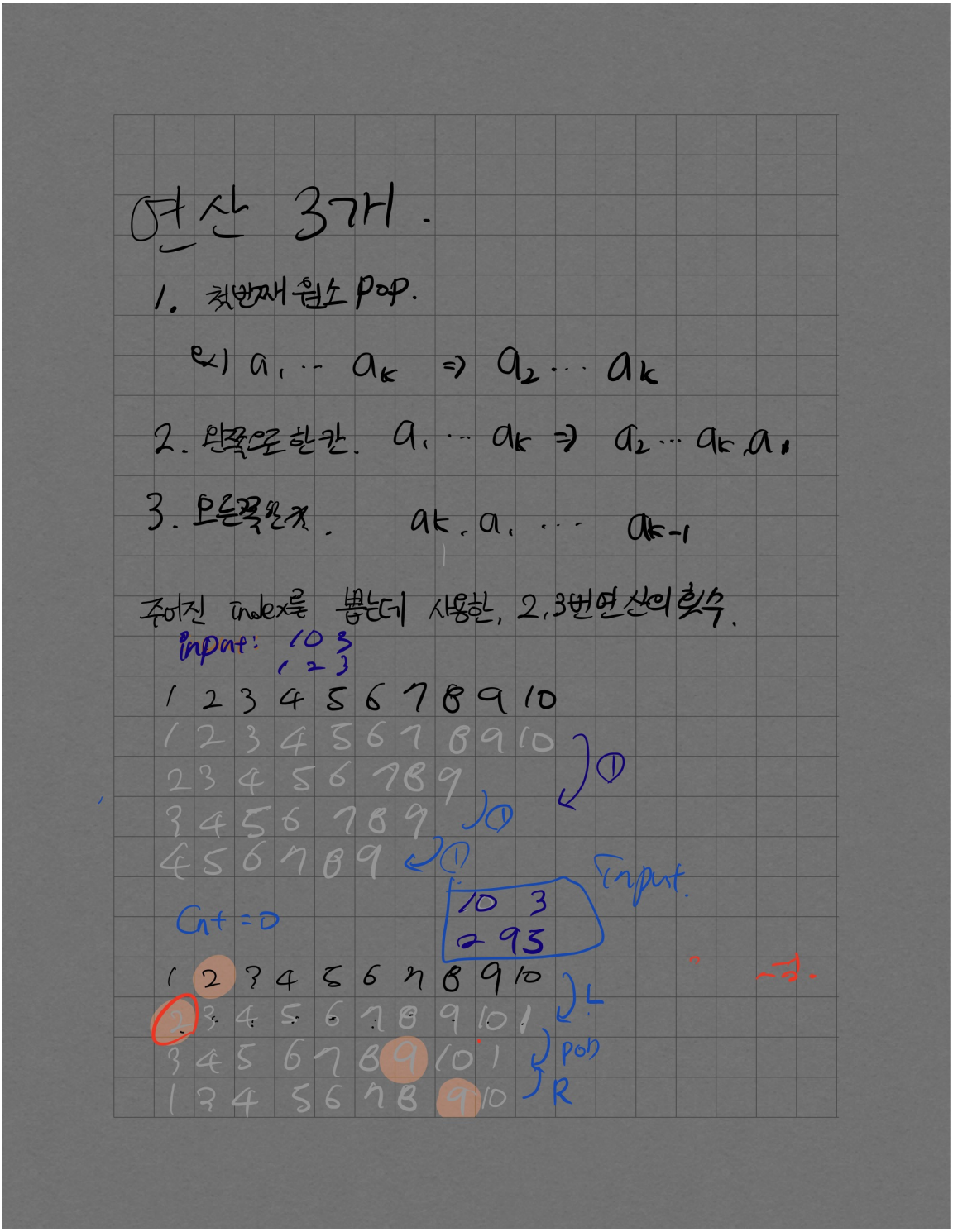
▼ 소스코드

```
package week8_3;

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

public class B_1021회전하는큐 {
  static LinkedList<Integer> queue = new LinkedList<>();
  static int N;
  static int M;
  static int cnt;
  public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    N = sc.nextInt();
    M = sc.nextInt();
    cnt = 0;
    for(int i = 1; i <= N; i++) {
      queue.offer(i);
    }
    int order[] = new int[M];
    for(int i = 0; i < M; i++) {
      order[i] = sc.nextInt();
    }
    int index = 0;
    while(true) {
      if(index == M) break;

      if(queue.peek() == order[index]) {
        popQueue();
        index++;
        continue;
      }
      if(ismiddle(order[index])) {
        leftRotate();
      }else {
        rightRotate();
      }
    }
    System.out.println(cnt);


  }
  static void popQueue() {
    queue.pop();
  }
  static void leftRotate() {
    cnt++;
    Integer tmp = queue.pollFirst();
    queue.offerLast(tmp);

  }
  static void rightRotate() {
    Integer tmp = queue.pollLast();
    cnt++;
    queue.offerFirst(tmp);
  }
  static boolean ismiddle(int num) {
```

```
    int mid = queue.size()/2;
    boolean result = false;
    for(int i  = 0; i <= mid; i++) {
      if(queue.get(i) == (Integer)num) {
        result = true;
        break;
      }
    }
    return result;
  }
}
```

▼ 설명

연산이 총 3개이다.
맨앞의 원소를 pop해주는 원소
좌회전, 우회전
만약 맨앞이 원하는 원소이면 pop해주고
아닐경우는 middle을 기준으로 좌회전, 우회전을 결정해줘서
회전 쭉 시키면된다.


연산 3개.
1. 첫번째 원소 pop.
ex) $a_1 \cdots a_k \Rightarrow a_2 \cdots a_k$
2. 맨 뒤로 보내기. $a_1 \cdots a_k \Rightarrow a_2 \cdots a_k a_1$
3. 모든 수 제거. $a_k, a_1, \cdots a_{k-1}$
주어진 index를 뽑는데 사용한, 2,3번 연산의 횟수.
input: 10 3
1 2 3
1 2 3 4 5 6 7 8 9 10
2 3 4 5 6 7 8 9
3 4 5 6 7 8 9
4 5 6 7 8 9
Cnt=0
10 3
2 9 3
input.
pop
R

2346번: 풍선 터뜨리기

1번부터 N번까지 N개의 풍선이 원형으로 놓여 있고. i번 풍선의 오른쪽에는 i+1번 풍선이 있고, 왼쪽에는 i-1번 풍선이 있다. 단, 1번 풍선의 왼쪽에 N번 풍선이 있고, N번 풍선의 오른쪽에 1번 풍선이 있다.

https://www.acmicpc.net/problem/2346

BAEKJOON ONLINE JUDGE

▼ 소스코드

```
package day0819;

import java.util.ArrayDeque;
import java.util.Arrays;
import java.util.Deque;
import java.util.LinkedList;
import java.util.Scanner;

public class Main_2346_풍선터뜨리기 {
  public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    Deque<Integer> de = new ArrayDeque<>();
    Deque<Integer> de2 = new ArrayDeque<>();
    StringBuilder sb = new StringBuilder();
    int N = sc.nextInt();

    for (int i = 1; i < N + 1; i++) {
      de.add(sc.nextInt());
      de2.add(i);
    }
    int num = de.pollFirst();
    sb.append(de2.pollFirst());
    sb.append(" ");
    while (!de.isEmpty()) {

      if (num > 0) {
        for (int i = 0; i < num - 1; i++) {
          de.addLast(de.pollFirst());
          de2.addLast(de2.pollFirst());
        }
        num = de.pollFirst();
        sb.append(de2.pollFirst());
        sb.append(" ");
        continue;

      } else if (num < 0) {
        for (int i = 0; i < Math.abs(num) - 1; i++) {
          // System.out.println(Math.abs(num) - 1);
          de.addFirst(de.pollLast());
          de2.addFirst(de2.pollLast());
        }
        num = de.pollLast();
        sb.append(de2.pollLast());
        sb.append(" ");
      }
```

```
      continue;
    }
    System.out.println(sb);
  }

}
```

1406번: 에디터

한 줄로 된 간단한 에디터를 구현하려고 한다. 이 편집기는 영어 소문자만을 기록할 수 있는 편집기로, 최대 600,000글자까지 입력할 수 있다. 이 편집기에는 '커서'라는 것이 있는데, 커서는 문장의 맨 앞(첫 번

https://www.acmicpc.net/problem/1406

▼ 소스코드

```
package week8_3;

import java.awt.List;
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.util.LinkedList;
import java.util.ListIterator;
import java.util.StringTokenizer;

public class B_1406_에디터2 {
  static LinkedList<Character> buffer;
  static List buffer1;
  static int N;
  static int cur;
  static ListIterator<Character> iter;

  public static void main(String[] args) throws IOException {
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

    buffer = new LinkedList<>();
    for (char c : br.readLine().toCharArray()) {
      buffer.add(c);
    }
    N = Integer.parseInt(br.readLine());
    iter = buffer.listIterator(buffer.size());
    for (int i = 0; i < N; i++) {
      StringTokenizer st = new StringTokenizer(br.readLine());
      char arr = st.nextToken().charAt(0);
//      System.out.println(Arrays.toString(arr));
      switch (arr) {
      case 'L':
```

```
        L();
        break;
      case 'D':
        D();
        break;
      case 'B':
        B();
        break;
      case 'P':
        P(st.nextToken().charAt(0));
        break;
      default:
        break;
      }
//      printbuf();
    }
    printbuf();
  }

  static void printbuf() throws IOException {
    StringBuilder sb = new StringBuilder();
    for (Character c : buffer) {
      sb.append(c);
    }
    BufferedWriter bw = new BufferedWriter(new OutputStreamWriter(System.out));
    bw.write(sb.toString());
    bw.flush();
    bw.close();
  }

  static void P(char key) {
    iter.add(key);

//    buffer.add(cur, key);
//    cur++;
  }

  static void L() {
    if (iter.hasPrevious())
      iter.previous();
  }

  static void D() {
    if (iter.hasNext())
      iter.next();
  }

  static void B() {
    if (iter.hasPrevious()) {
      L();
      iter.remove();
    }
  }

}
```

▼ 설명

Linkedlist는 삽입, 삭제가 O(1)시간에 이루어진다고 알고있다. 단,,, 삽입, 삭제할 iter를 알경우에,,,

만약 index값으로 찾을려고하면 O(N)시간이 걸리므로 iter를 사용해야한다고한다.

17827번: 달팽이 리스트

문제 영진이는 달팽이를 좋아한다. 달팽이를 너무너무 좋아하기 때문에 특정한 모양의 단방향 연결리스트에 달팽이 리스트라는 이름을 붙여주었다. 일반적인 선형 단방향 연결리스트의 각 노드 번호를 연결된

https://www.acmicpc.net/problem/17827

BAEKJOON ONLINE JUDGE

▼ 소스코드

```
package practice;

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

public class Boj_17827_달팽이리스트4_찐마지막 {
public static void main(String[] args) throws IOException {
  BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
  StringTokenizer st = new StringTokenizer(br.readLine());
  StringBuilder sb = new StringBuilder();

  int N = Integer.parseInt(st.nextToken());
  int M = Integer.parseInt(st.nextToken());
  int V = Integer.parseInt(st.nextToken());

  List<Integer> list = new ArrayList<>();

  StringTokenizer str = new StringTokenizer(br.readLine());

  // 마지막 노드까지 추가
  for(int i=0;i<N;i++) {
    list.add(Integer.parseInt(str.nextToken()));

  }


  //명령어 저장할 배열
  int[] arr = new int[M];
  for(int i=0;i<M;i++) {
```

```
    arr[i] = Integer.parseInt(br.readLine());
  }


  for (int i = 0; i < M; i++) {
    if(arr[i]<N) {
      System.out.println(list.get(arr[i]));
      continue;
    }
      System.out.println(list.get(((arr[i]-N)%(N-V+1))+V-1));
  }

}

static class Node {
  public int data;
  public Node link;
  public Node() {
    super();
    // TODO Auto-generated constructor stub
  }
  public Node(int data, Node link) {
    super();
    this.data = data;
    this.link = link;
  }

}
```

}

▼ 설명

LinkedList를 사용했을 때 계속 시간초과가 나서 수학적으로 접근했다. K가 N보다 작다면 그냥 K번째 노드의 데이터를 가져오면 되고(인덱스는 -1이므로) N보다 크다면 K-N 이후에는 N-V+1만큼 사이클이 반복되므로 K-N을 N-V+1로 나눈 나머지에서 V를 더해주면 K번 이후의 노드에 접근하게 된다. 리스트의 인덱스는 -1이 되므로 마지막에 -1을 해주어서 데이터를 가져오면 K번 이후의 노드의 데이터를 가져오게 된다.

3217번: malloc

메모리 할당 명령을 시뮬레이팅하는 프로그램을 작성하시오. 메모리는 100,000개의 연속된 공간이고, 1번부터 100,000번까지 번호가 매겨져 있다. 초기에 모든 공간은 할당되지 않은 상태이다. 명령어의 종

https://www.acmicpc.net/problem/3217