3조 8월 3주차 스터디 <LIST> 팀원 : 안정석, 이예나, 배나영

발표 형식 <전 팀원 발표>

공통 문제 : <달팽이 리스트>

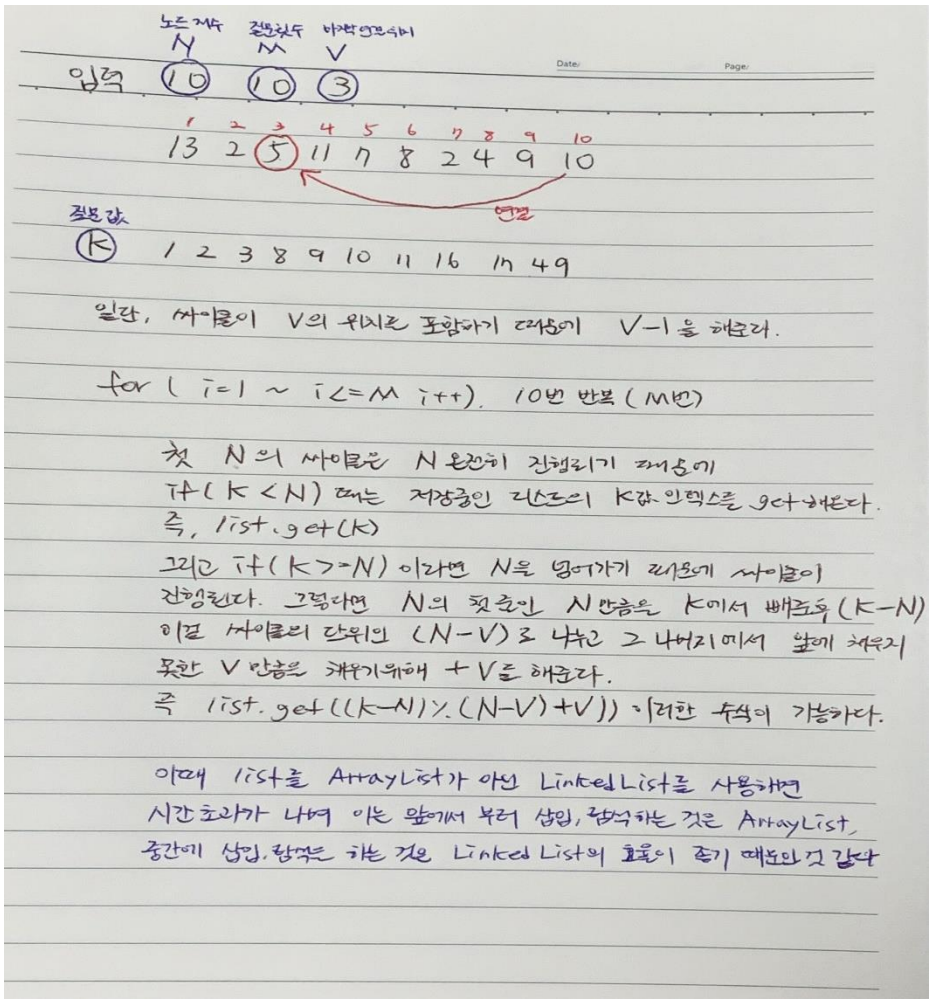
팀 논의 내용 :

1. ArrayList 와 LinkedList 의 사용 : 달팽이 리스트를 수식과 리스트를 활용해서 풀었다면 ArrayList로는 문제를 통과하고 LinkedList를 사용했다면 시간초과가 났을 것이다. 이는 ArrayList는 인덱스의 흐름 순서의 삽입, 삭제 시에 LinkedList보다 훨씬 빠른 처리 속도를 보여주기 때문이다.

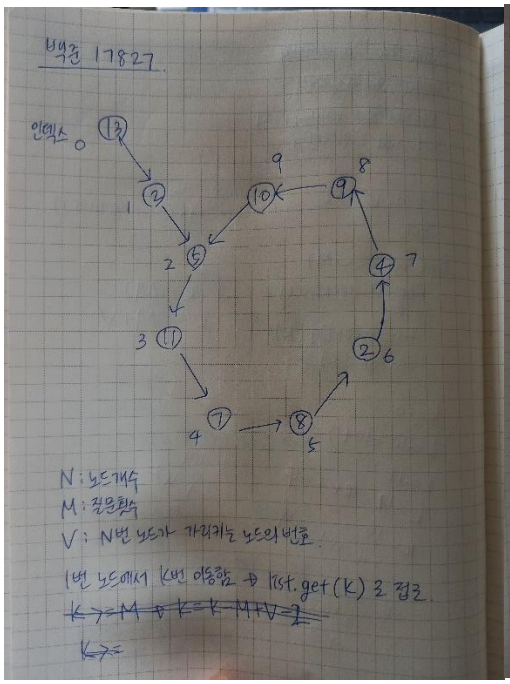
단, 중간의 인덱스에 값을 삽입, 삭제시에는 LinkedList의 속도가 훨씬 빠르다. 이처럼 상황별로 활용할 수 있는 환경이 따로 있다는 것에 대해서 배울 수 있었다.

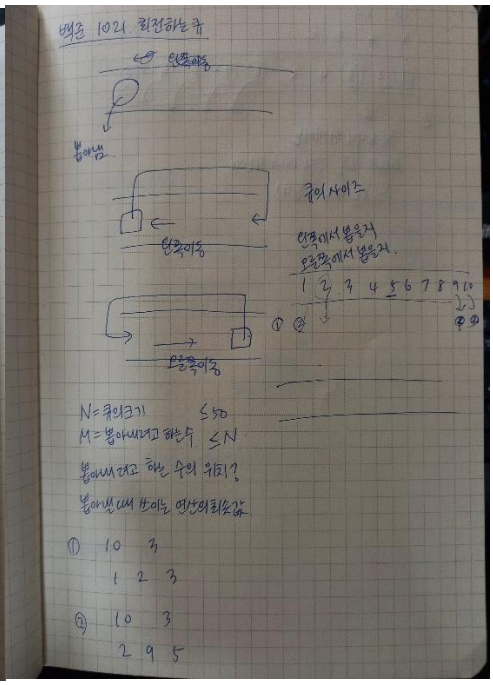
2. 나영님은 수식을 이용하지 않고 for문 안에 while문을 활용하여 무한 루프를 통해 문제를 풀었다. 이렇게 코드가 진행되면 while문으로 인해 시간초과가 일어난다. 따라서 기존의 코드를 수식을 이용하여 while을 사용하지 않고 for문 하나로 진행하도록 하였다.

<노트 스케치 - 1>



<노트 스케치 – 2>





<소스 코드 – 1>

```java
public class Snail {

    static class Node{
        int point; //가르키는 다음 인덱스
        int value; //노드의 값
        Node(int point,  int value){
            this.point = point;

            this.value = value;
        }
    }

    public static void main(String[] args) throws IOException {


        List<Node> snails = new ArrayList<>();
        BufferedReader bf = new BufferedReader(new InputStreamReader(System.in));
        String s =bf.readLine();
        StringTokenizer st =new StringTokenizer(s);
        int N =Integer.parseInt(st.nextToken());
        int M = Integer.parseInt(st.nextToken());
        int v = Integer.parseInt(st.nextToken());
        s= bf.readLine();
        st = new StringTokenizer(s);
        for(int i=0;i<N-1;i++) {
        int value = Integer.parseInt(st.nextToken());
            snails.add(new Node(i+1,value)); //마지막노드 N-1전까지 다음노드를 가리킴
        }

        snails.add(new Node(v-1,Integer.parseInt(st.nextToken()))); //마지막 노드가 v-1을 가리킴
        for(int i=0;i<M;i++) {
            int k = Integer.parseInt(bf.readLine());

            if(k>N-1) { //k가 끝노드보다 클때
                k=(k-v+1)%(N-v+1)+(v-1); // 바퀴수(k-(v-1) % (N-(v-1))) + 인덱스(v-1)
            }

            System.out.println(snails.get(k).value); //인덱스일때의 값 출력


        }


    }
}
```

<소스코드 – 2>

```java
package day0819;

import java.io.BufferedReader;

public class SWEA_달팽이_리스트 {

    public static void main(String[] args) throws IOException {

        ArrayList<Integer> list = new ArrayList<>();
        int N, M, V;

        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        StringTokenizer st = new StringTokenizer(br.readLine());

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

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

        for(int i=0; i<N; i++) {
            list.add(i, Integer.parseInt(st.nextToken()));
        }

        V = V-1;

        for(int i=1; i<=M; i++) {
            int K = Integer.parseInt(br.readLine());
            if(K<N) {
                System.out.println(list.get(K));
            }

            if(K>=N) {
                System.out.println(list.get((K-N)%(N-V)+V));
            }

        }

    }
}
```

Linked List면 시간 초과!

for 문

<소스코드 – 3>

```java
package com.study.week03;

import java.util.LinkedList;

public class Main17827_달팽이리스트 {

    public static void main(String[] args) {

        Scanner sc=new Scanner(System.in);
        int N=sc.nextInt();
        int M=sc.nextInt();
        int V=sc.nextInt();

        LinkedList<Integer> list=new LinkedList<>();
        int[] inputK=new int[M];

        for(int i=0;i<N;i++) {
            list.add(sc.nextInt());
        }

        for(int i=0;i<M;i++) {
            inputK[i]=sc.nextInt();
        }

        for(int i=0;i<M;i++) {

            while(inputK[i]>=list.size())
                inputK[i]=inputK[i]-list.size()+V-1;

            System.out.println(list.get(inputK[i]));
        }
    }
}
```

시간초과의 범인!

문제 : <에디터>

팀 논의 내용 :

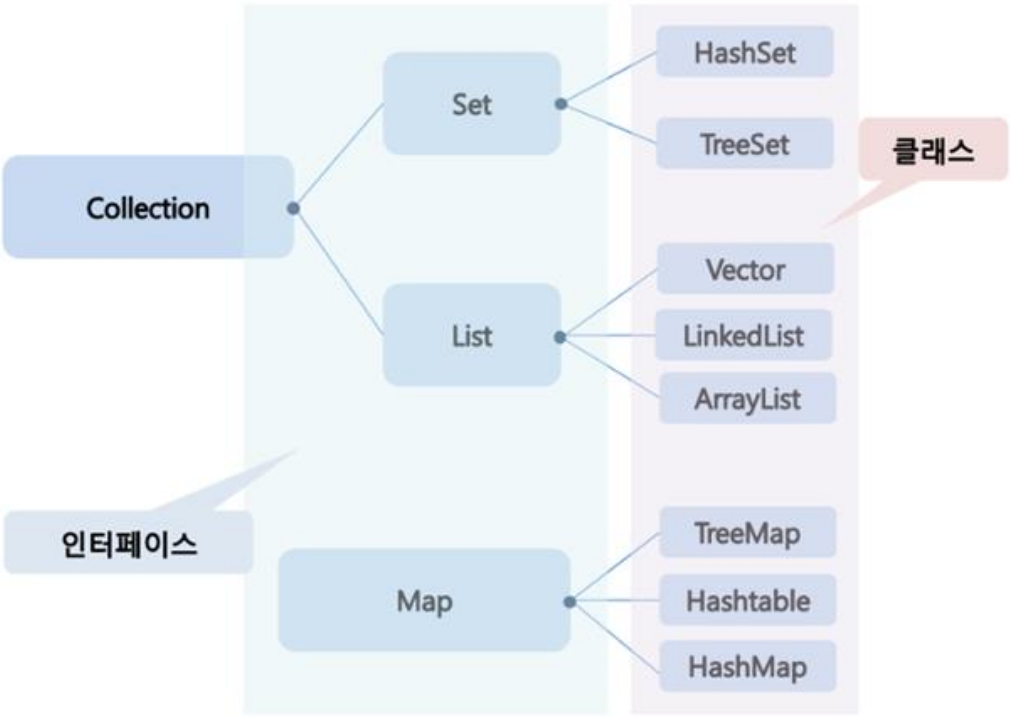
1. 에디터는 보기에는 명령에 맞춰서 시키는 대로만 하면 될 것 같은 간단한 문제로 보였지만, 시간초과라는 복병이 기다리고 있던 문제였다.

시간 제한

0.3 초 (하단 참고)

시간 초과를 피하기 위해서는 Iterator 가 필요하단 걸 알게 되었고, Iterator에 대해 알아보는 시간을 가졌다.

Iterator는 자바의 컬렉션 프레임워크에서 컬렉션에 저장되어 있는 요소들을 읽어오는 방법을 표준화한 것이다. Iterator는 Set, List와 같은 집합체로부터 정보를 얻어낸다고 볼 수 있다.



출처: https://thefif19wlsvy.tistory.com/41 [FIF's 코딩팩토리]

2. 그렇다면 'Iterator를 모르는 상태에서 어떻게 시간초과를 피해서 풀까?' 하고 접근하는 방법도 생각해 오셔서 Stack으로 구현하는 에디터도 함께 코드 리뷰를 진행하였다.

<소스코드 – 1 List 버전>

```java
public class Editor2_list {

    public static void main(String[] args) throws IOException {


        BufferedReader bf = new BufferedReader(new InputStreamReader(System.in));
        String s = bf.readLine();
        int curser = s.length();
        String rev="";
        int n = Integer.parseInt(bf.readLine());
        List<Character> lists = new ArrayList<>();
        for(int i=0;i<s.length();i++) {
        lists.add(s.charAt(i));

        }

        for(int i=0;i<n;i++) {
            s = bf.readLine();
            StringTokenizer st=new StringTokenizer(s);
            char order = st.nextToken().charAt(0);
            if(order=='P') { //삽입
                char ch = st.nextToken().charAt(0);
                lists.add(curser,ch);
                curser++;
            }
            else if(order=='L') {//커서 왼쪽으로 한칸 옮김
                if(curser!=0) curser--;
            }
            else if(order=='D') {//커서 오른쪽으로 한칸 옮김
                if(curser!=s.length()+1) curser++;
            }
            else if(order=='B') {//커서 왼쪽 문자 하나 삭제
                if(lists.size()==0) continue;
                if(curser!=0) lists.remove(curser-1);
                curser--;
                if(curser<=0) curser=0;
            }


        }

        Iterator<Character> iter = lists.iterator();
        BufferedWriter bw= new BufferedWriter(new OutputStreamWriter(System.out));
        while(iter.hasNext()) {
            bw.write(iter.next());
        }
        bw.flush();
        bw.close();
    }

}
```

Iterator 사용

<소스코드 – 2 Stack 버전>

```java
    public static void main(String[] args) throws IOException {

        Stack<Character> left = new Stack<>();
        Stack<Character> right = new Stack<>();
        BufferedReader bf = new BufferedReader(new InputStreamReader(System.in));
        String s = bf.readLine();
//      int curser = s.length();
        String rev="";
        int n = Integer.parseInt(bf.readLine());
//      List<Character> lists = new ArrayList<>();
        for(int i=0;i<s.length();i++) {
//          lists.add(s.charAt(i));
            left.push(s.charAt(i));
        }

        for(int i=0;i<n;i++) {
            s = bf.readLine();
            StringTokenizer st=new StringTokenizer(s);
            char order = st.nextToken().charAt(0);
            if(order=='P') { //삽입
                char ch = st.nextToken().charAt(0);
//              lists.add(curser,ch);
//              curser++;
                left.push(ch);
            }
            else if(order=='L') {//커서 왼쪽으로 한칸 옮김
//              if(curser!=0) curser--;
                if(!left.empty())
                right.add(left.pop());
            }
            else if(order=='D') {//커서 오른쪽으로 한칸 옮김
//              if(curser!=s.length()+1) curser++;
                if(!right.empty())
                left.push(right.pop());
            }
            else if(order=='B') {//커서 왼쪽 문자 하나 삭제
//              if(lists.size()==0) continue;
//              if(curser!=0) lists.remove(curser-1);
//              curser--;
//              if(curser<=0) curser=0;
                if(!left.empty())left.pop();
            }

            //System.out.println("order: "+order+" curser: "+curser);
        //

        }
        while(!left.empty()) {
            right.push(left.pop());
        }
        BufferedWriter bw= new BufferedWriter(new OutputStreamWriter(System.out));
        while(!right.empty() ) {
            bw.write(right.pop());
        }
        bw.flush();
        bw.close();
```

Stack 사용

문제 : <풍선 터뜨리기>

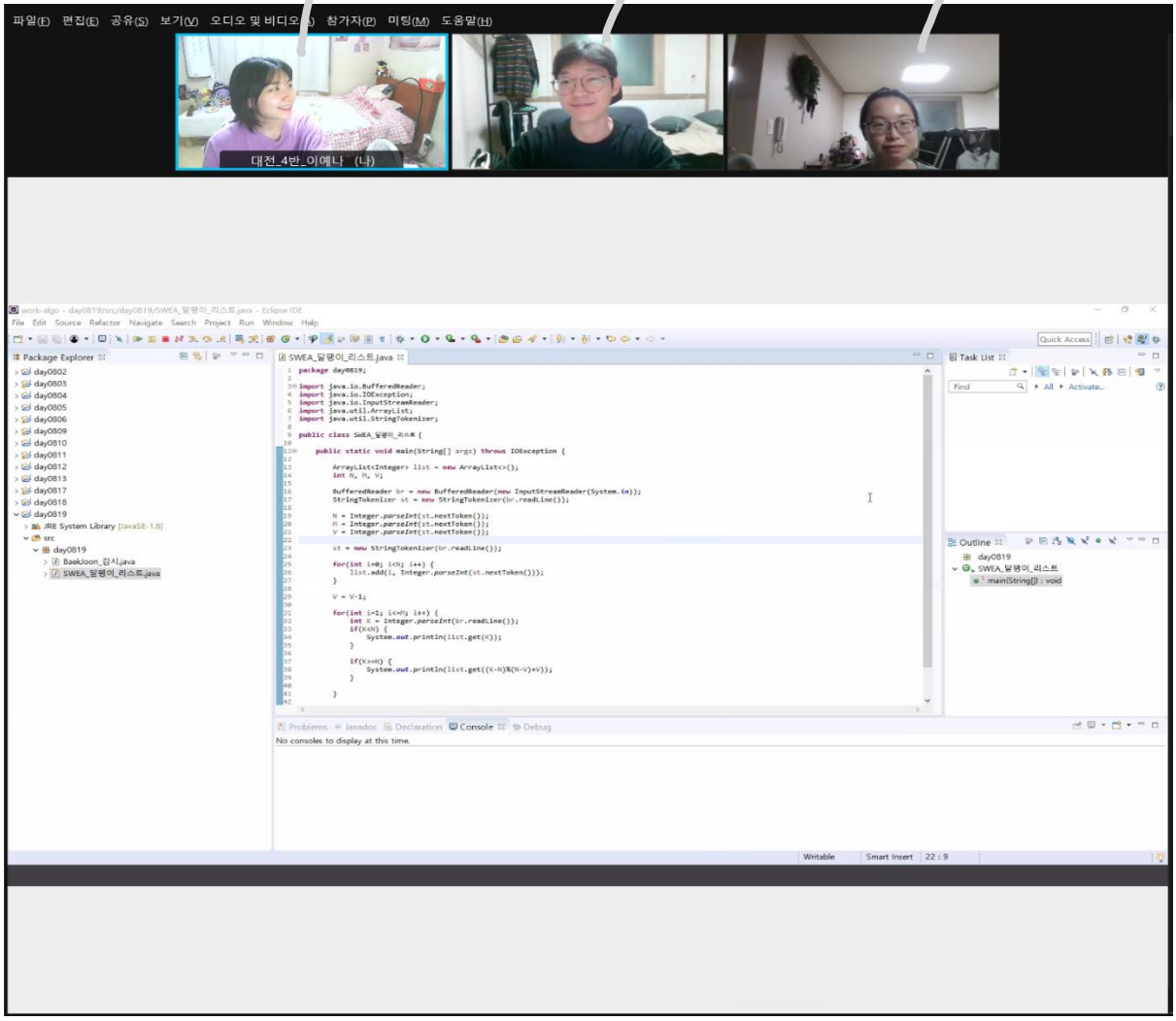
팀 논의 내용 :

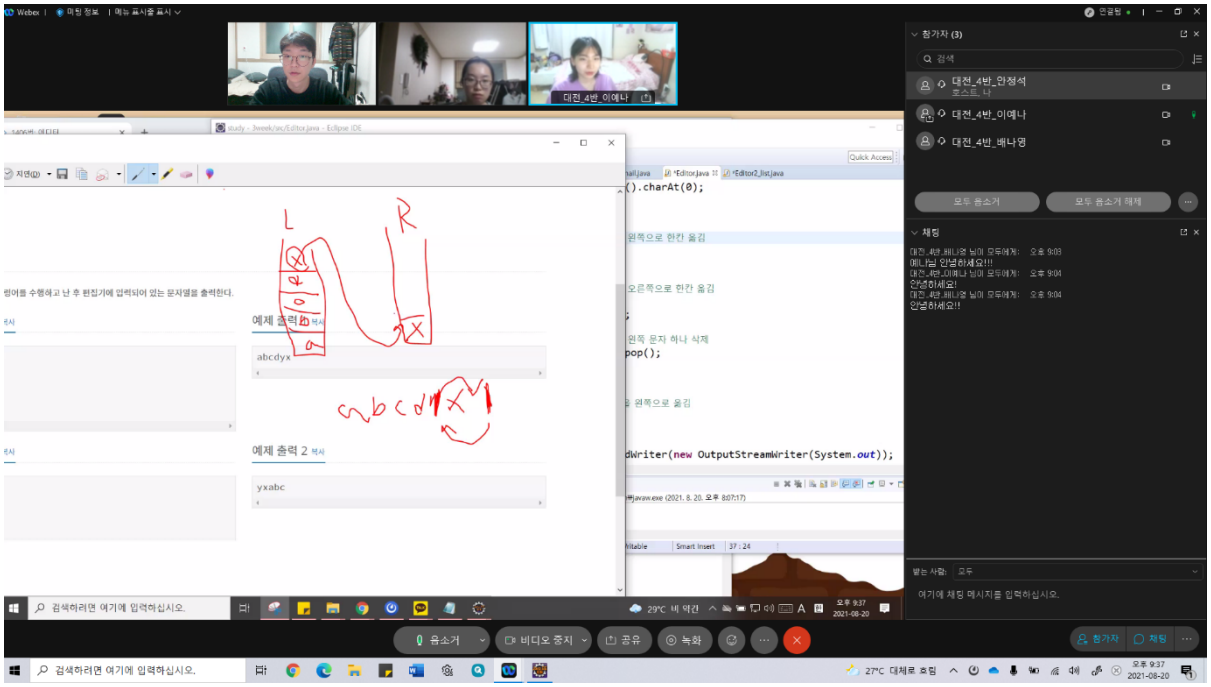
1. 팀원이 풀다가 해결하진 못한 문제를 함께 해결해보기 위해 코드 리뷰를 진행하였다. 코드에서 문제가 되는 부분은 인덱스의 범위로 인한 런타임 에러였다. 이를 해결하기 위해 다양한 의견을 나눴다. 첫 째는 수식을 이용한 해결방법이다. 수식을 이용하여 인덱스를 컨트롤하면 범위를 벗어나지 않게 할 수 있을 것이라 봤다. 두 번째는 Deque 자료구조를 사용하는 것이다. Deque는 큐를 양쪽으로 삽입, 삭제 컨트롤이 가능하기 때문에 풍선 터뜨리기를 풀 때 유용한 자료구조가 될 것이라고 토의하였다.

<소스코드 – 1>

```
import java.util.ArrayList;

public class Main2346_풍선터뜨리기 {

    static class Balloon { // 풍선
        int idx;
        int num;

        public Balloon(int idx, int num) {
            this.idx=idx;
            this.num=num;
        }
    }

    public static void main(String[] args) {

        Scanner sc=new Scanner(System.in);
        ArrayList<Balloon> list=new ArrayList<>();
        int N=sc.nextInt();

        for(int i=0;i<N;i++) {
            int num=sc.nextInt();
            list.add(new Balloon(i+1, num));
        }

        int index=0;
        StringBuilder result=new StringBuilder();

        while(!list.isEmpty()) { // 풍선이 다 터지지 않은 동안

            result.append(list.get(index).idx+" ");
            index+=list.remove(index).num; // 삭제

            // 인덱스가 리스트의 범위 밖이 되었을 경우
            if(index<0) index+=list.size();
            else if(index>list.size()) index-=list.size();

//////////////////////////////////////////////////////////////////////////
//          디버깅
//          for(int i=0;i<list.size();i++) {
//              System.out.print("idx: "+list.get(i).idx);
//              System.out.println(", num: "+list.get(i).num);
//          }
//          System.out.println("result: "+result);
//          System.out.println("-----------------");
        }
        System.out.println(result);
    }
}
```

<팀 활동 사진>


파일(F) 편집(E) 공유(S) 보기(V) 오디오 및 비디오(A) 참가자(P) 미팅(M) 도움말(H)
대전_4반_이예나 (나)


대전_4반_이예나
대전_4반_안정석
대전_4반_이예나
대전_4반_배나영
예제 출력 1
abcdyx
예제 출력 2
yxabc