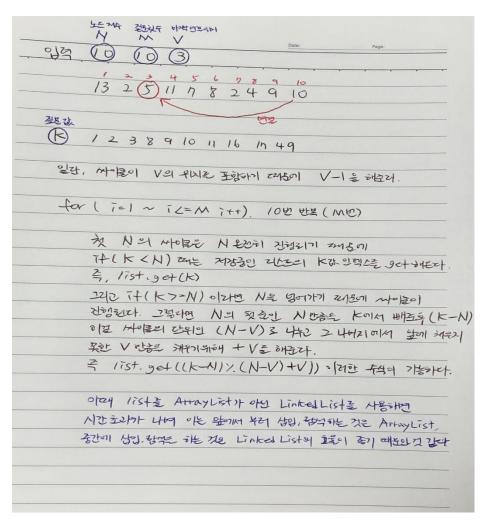
초급자를 골랐지만 팀선정문제를 풀고 스터디를 진행하면서 숙련자 문제를 풀었습니다.

Malloc은 문제 이해를 못하여 문제풀이를 실패하였고, 달팽이 리스트, 에디터 문제로 진행하였습니다.

에디터 문제는 스터디를 통해 해결책을 Iterator라는 내용을 알게 되어 풀 수 있었습니다. 또 Iterator를 몰랐다면 스택과 같은 자료구조를 이용하여 접근했어도 가능하단 것을 알 수 있었습니다.

그리고 이번 스터디로 가장 중요하게 배운 점은 ArrayList와 LinkedList의 차이입니다. ArrayList는 인덱스의 흐름 순서의 삽입, 삭제 시에 LinkedList보다 훨씬 빠른 처리 속도를 보여주고, 중간의 인덱스에 값을 삽입, 삭제시에는 LinkedList의 속도가 훨씬 빠릅니다. 이처럼 상황별로 활용할 수 있는 환경이 따로 있다는 것에 대해서 배울 수 있었습니다.



```
package day0819;
Dimport java.io.BufferedReader;
 public class SWEA_달팽이_리스트 [
               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.
        public static void main(String[] args)
                                                                           throws IOException {
               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++) {</pre>
                      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) {</pre>
                            System.out.println(list.get(K));
                      }
                      if(K>=N) {
                             System.out.println(list.get((K-N)%(N-V)+V));
               }
        }
  }
    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.eadLine());
List<character> lists = new ArrayList>();
for(int i=0;i<s.length();i+) {
lists.add(s.charAt(i));</pre>
                                                                              Arraylist AB
           for(int i=0;i<n;i++) {
    s = bf.readLine();
    StringTokenizer st=new StringTokenizer(s);
    char order = st.nextToken().charAt(0);
    if(order=='P') { //삼입
        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(new OutputStreamWriter(System.out));
while(iter.hasNext()) {
                  bw.write(iter.next());
            bw.flush();
           bw.close();
```