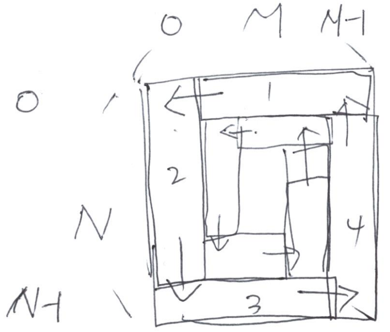
Baekjoon 16926 배열 돌리기1

날짜	@2021년 8월 11일 → 2021년 8월 11일
링크	https://www.acmicpc.net/problem/16926
열	
태그	Baekjoon Silver3 matrix

Baekjoon 16926번 실버3



풀이법 2가지

① Linked List 테두리마다 삽입 하나씩 돌려서 빼내면 된다.

회전이 반복되면

② 행렬 좌표 직접 계산

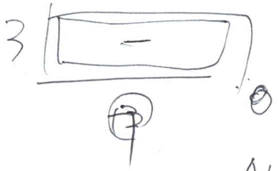
2번으로

(0, 1~M-1) ⇓ (0, 0~M-2)

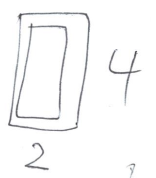
(0, ~N-2), 0) ⇓ (1~N-1), 0)

(1, (2~M-2)) ↓ (1, (1~M-3))

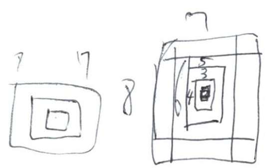
((1~N-3), 1) ((2~N-2), 1)



N-1, (0~M-2)   (0~N-1), M-1   N-2, (1~M-3)   (2~N-2) M-2



N-1, (1~M-1)   (0~N-2), M-1   N-2, (2~M-2)   (1~N-3) M-2



연산 오래걸릴듯 그냥

Linked 하자

```
import java.io.BufferedReader;
import java.io.FileInputStream;
```

```
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.ArrayList;
import java.util.LinkedList;
import java.util.StringTokenizer;

//public class Main {
public class Baek16926_1 {
  static int N, M, R, listSize;
  static int[][] num;
  static ArrayList<LinkedList> a = new ArrayList<>();;
  static int[][] ans;

  public static void main(String[] args) throws IOException {
    System.setIn(new FileInputStream("C:/CodingStudy/Baekjoon/Silver3/16926_input.tx
t"));
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    StringTokenizer st = new StringTokenizer(br.readLine());
    N = Integer.parseInt(st.nextToken());
    M = Integer.parseInt(st.nextToken());
    R = Integer.parseInt(st.nextToken());
    listSize = (Math.min(N, M) % 2 == 0) ? Math.min(N, M) / 2 : Math.min(N, M) / 2 +
 1;

    for (int i = 0; i < listSize; i++) {
      a.add(new LinkedList<Integer>());
    } // end 각 테두리를 관리할 링크드 리스트 생성

    num = new int[N][M];
    ans = new int[N][M];
    for (int n = 0; n < N; n++) {
      st = new StringTokenizer(br.readLine());
      for (int m = 0; m < M; m++) {
        num[n][m] = Integer.parseInt(st.nextToken());
      }
    } // end 초기 맵 입력

    readLine(0); // 테두리 별로 링크드 리스트에 넣기

    for (int idx = 0; idx < a.size(); idx++) {
      for (int i = 0; i < ((a.get(idx).size() > R) ?R: ( R%a.get(idx).size()) ); i++)
 {
        a.get(idx).add(a.get(idx).poll()); // 돌리기
      }
    }
    outLine(0);
    for (int col = 0; col < N; col++) {
      for (int row = 0; row < M; row++) {
        System.out.print(ans[col][row] + " ");
      }
      System.out.println();
    }
    br.close();

  }// end main

  public static void outLine(int depth) {// 테두리 별로 링크드 리스트에서 불러오기
    for (int row = depth; row <= M - 2 - depth; row++) {
```

```
      ans[depth][row] = (int) a.get(depth).pop();
    }
    for (int col = depth; col <= N - 2 - depth; col++) {
      ans[col][M - 1 - depth] = (int) a.get(depth).pop();
    }
    for (int row = M - 1 - depth; row > 0 + depth; row--) {
      ans[N - 1 - depth][row] = (int) a.get(depth).pop();
    }
    for (int col = N - 1 - depth; col > 0 + depth; col--) {
      ans[col][depth] = (int) a.get(depth).pop();
    }
    if (depth == (listSize - 1))
      return;
    outLine(depth + 1);
  }

  public static void readLine(int depth) {// 테두리 별로 링크드 리스트에 넣기
    for (int row = depth; row <= M - 2 - depth; row++) {
      a.get(depth).add(num[depth][row]);
    }
    for (int col = depth; col <= N - 2 - depth; col++) {
      a.get(depth).add(num[col][M - 1 - depth]);
    }
    for (int row = M - 1 - depth; row > 0 + depth; row--) {
      a.get(depth).add(num[N - 1 - depth][row]);
    }
    for (int col = N - 1 - depth; col > 0 + depth; col--) {
      a.get(depth).add(num[col][depth]);
    }
    if (depth == (listSize - 1))
      return;
    readLine(depth + 1);
  }
}
```