susuk Algo#1 : 백준 3190

Date	@Oct 14, 2020
Tags	

백준 3190번 - 뱀

3190번: 뱀

'Dummy' 라는 도스게임이 있다. 이 게임에는 뱀이 나와서 기어다니는데, 사과를 먹으면 뱀 길이가 늘어난다. 뱀이 이리저리 기어다니다가 벽 또는 자기자신의 몸과 부딪히면 게임이 끝난다.

https://www.acmicpc.net/problem/3190

BAEKJOON ONLINE JUDGE

삼성 역테 예상문제!!

지렁이게임이랑 비슷한 시뮬레이션 문제이다.

시뮬레이션인 만큼

제일 중요한게 **로직 순서**!!

1. 다음칸으로 이동 가능한지 체크
2. 불가능하면 종료
3. 가능하면 다음칸에 사과 유무에 따라 로직 처리
4. 방향 변환 로직 처리
5. 그 다음칸으로 좌표 갱신
6. 1부터 다시 반복

어려운 내용은 없는데 구현 방법이 한번에 안떠올라서 50분쯤 걸림....

```
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 BJ_3190 {
  private static int N;
  private static int[][] directions = {{-1,0},{0,1},{1,0},{0,-1}};

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

    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    N = Integer.parseInt(br.readLine());
    int K = Integer.parseInt(br.readLine());
    int[][] apple = new int[K][2];
    for (int i = 0; i < K; i++) {
      StringTokenizer st = new StringTokenizer(br.readLine()," ");
      apple[i][0]=Integer.parseInt(st.nextToken());
      apple[i][1]=Integer.parseInt(st.nextToken());
    }
    int L = Integer.parseInt(br.readLine());
    int chgFlag = 0;
    int[] chgTime = new int[L];
    char[] chgDir = new char[L];
    for (int i = 0; i < L; i++) {
      StringTokenizer st = new StringTokenizer(br.readLine()," ");
      chgTime[i]=Integer.parseInt(st.nextToken());
      chgDir[i]=st.nextToken().charAt(0);
    }
    // 입력 끝


    int len = 1, curX = 1, curY = 2, dir = 1;// dir> 0 : 상, 1: 우, 2: 하, 3: 좌
    int time = 0;

    Queue<int[]> queue = new LinkedList<>();
    queue.add(new int[] {1, 1});

    while(true) {
      ++time;
      //다음 이동 체크
      if(!isValid(curX, curY, len, dir,queue))  break;
```

```
      queue.add(new int[] {curX,curY});

      //사과 먹었는지 체크
      int flag=0;
      for (int i = 0; i < K; i++) {
        if(curX == apple[i][0] && curY == apple[i][1]) {
          apple[i][0]=-1;
          apple[i][1]=-1;
          flag=1;
          break;
        }
      }
      //사과 먹음
      if(flag==1) {
        ++len;
      }
      else {
        queue.poll();
      }

      // 방향 전환
      if(chgFlag<L && chgTime[chgFlag]==time) {
        if(chgDir[chgFlag]=='L') {
          --dir;
          if(dir==-1) dir=3;
        }
        else {
          ++dir;
          if(dir==4)  dir=0;
        }
        ++chgFlag;
      }

      //다음으로 몸 움직일꺼야
      curX+=directions[dir][0];
      curY+=directions[dir][1];
    }
    System.out.println(time);

  }

  //머리 내밀었을 때 check
  private static boolean isValid(int x, int y, int len, int dir, Queue<int[]> queue) {

    if(x<1 || y<1 || x>N || y >N) return false;

    for (int i = 0; i < queue.size(); i++) {
      int[] tmp = queue.poll();
      if(tmp[0] == x && tmp[1]==y)  return false;
      queue.add(tmp);
    }

    return true;
```

```
  }
}
```

💡 ***자료구조 순회가 필요한 경우 Queue보다는 Deque 사용!***

To Do Later

- [ ] 스택을 덱으로 바꿔보기
- [ ] int[][] map 사용해서 풀어보기
- [ ] apple 자료구조 바꿔서 시간 효율 높이기