Algoritm PA1
서준표

로봇의 처음 상태는 (0,0)에서 동쪽을 향하고 있으며, User Input에 따른 (M,M)의 Matrix에서 명령에 따라 움직인다. TURN, MOVE가 존재하며, TURN 0은 현재에서 왼쪽으로 90도, 1은 오른쪽으로 90도, MOVE는 바라보고있는 방향으로 움직이는 것을 의미한다.

TURN의 인자는 0, 1 중 하나이며 MOVE는 1000이하의 양수이다.

Facts :

enum dir : 현재의 방향을 나타내는 스트럭쳐
int matrix, counter : 행렬과 명령횟수
int row, column : 현재의 위치
int errored : 에러확인
int moveDir : 현재의 방향

void calMove(int) : 움직이는 함수
void calDir(int) : 방향을 바꿔주는 함수
void printErr() : 에러를 출력하는 함수

Over View :

방향이 바뀌는 데에는 규칙성이 있다.
이 규칙성을 찾으면 간단한 함수들을
코드를 효율적으로 짤 수 있을듯.

알고리즘의 흐름 :

수를 입력 받는다 -> 명령의 수만큼 while문 안에서 MOVE인지 TURN인지 구분한다 -> 각각의 함수를 숫자와 함께 호출한다. -> 방향을 계산하거나 로봇을 움직인다. -> 최종적으로 에러 체크를 해서 에러가 한번이라도 발생했으면 printErr함수를 출력하고 에러가 없으면 현재의 좌표를 출력한다.

Algorithm :

enum으로 방향(Dir)을

right 0
up 1
left 2
down 3

으로 설정 해 놓은 뒤,

방향 :

TURN 0이면 Dir++,
TURN 1이면 Dir- - 해준다.

오른쪽에서 turn 0 -> up , turn 1 -> down //0보다 작으면 3으로

위쪽에서 turn 0 -> left, turn 1-> right

왼쪽에서 turn 0 -> down , turn 1 -> up

아래쪽에서 turn 0 ->right, turn 1 -> left // 3넘어가면 0으로

오른쪽과 아래쪽만 예외처리해주는 코드 작성하면 공식을 만족함.

이동 :

현재의 moveDir을 체크해서,

현재 방향이 오른쪽을 향하면 column(x) ++

현재 방향이 왼쪽을 향하면 column(x) - -

현재 방향이 위쪽을 향하면 row(y) ++

현재 방향이 밑쪽을 향하면 row(y) - -