energy = 3. S = starting, collect all L's (n=20, m=20)

.	.	R	L	L	L
X	.	.	X	X	X
X	(S)	.	L	R	X
.	.	.	X	X	X
R	X	X	X	X	R
X	X	R	L	L	X
X	L	L	X	L	X

The first thought that I had was using BFS.

But then I thought minimum number of moves is required.

So I feel like I have to do multiple dfs each time with lesser cost.

Cost minimization is goal. → How can we minimize cost?

(Cost → {collecting all L's together}) minimize this.

Somehow need to figure out infinite looping. Stop condition.

Since we might have to revisit cells hence that condition is necessary

dfs(i, j, countL, energy) { → 20 * 20 * 50 * (20 or 4)

dp[i][j][countL][energy] = 20 * 20 * 50

= 2^4 * 10^4 * 50

= 80 * 10^5

= (8 * 10^6) memory

(min)

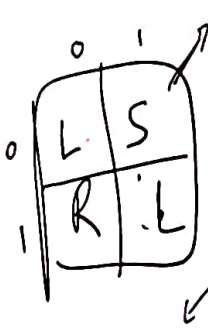
dfs(i+1, j, newCountL, newEnergy)

dfs(i, j+1, newCountL, new-)

;

dfs(i, j-1, ...)

dfs(i-1, j, ...)



e = 4.

f(0, 1, 2, 4)

directions

(i, j)

[L, S
R, .]

f(1, 1, 1, 3)

callvisit[ ][ ] = Set(pair)