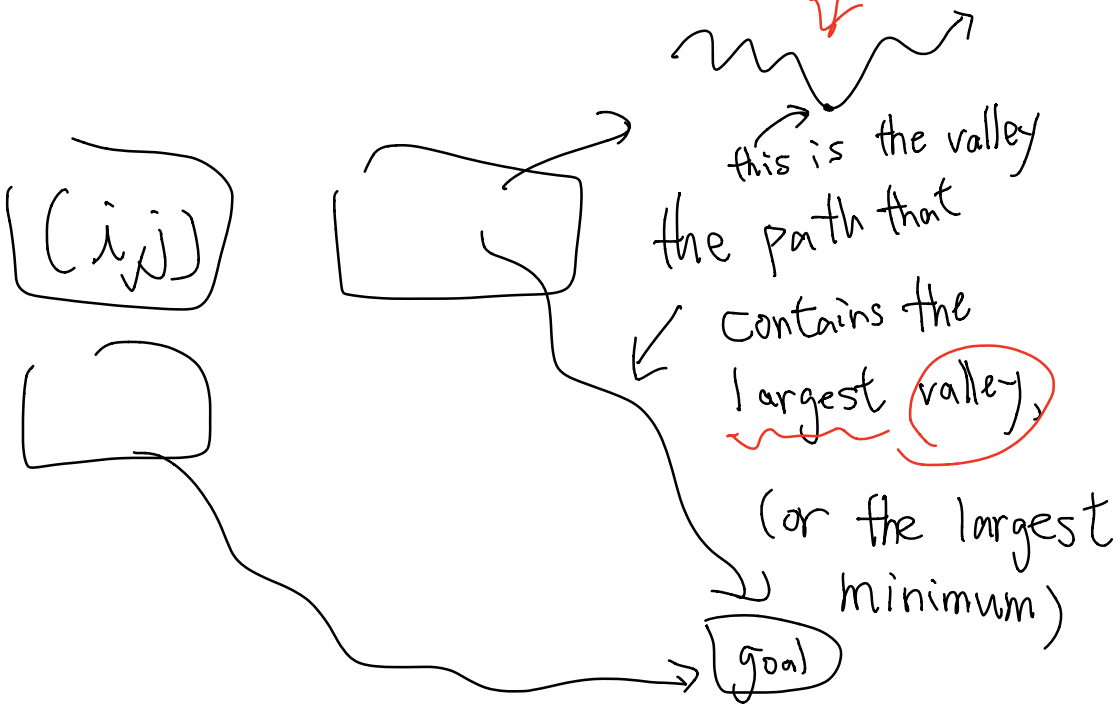
Our goal is to maximize the valley of a path starting from (0,0)



Define: valley(path) = min(each step's sum in the path)

dp[i][j] (among all paths from (i,j) toward the goal.)

= max(valley(path))

path ∈ paths toward goal

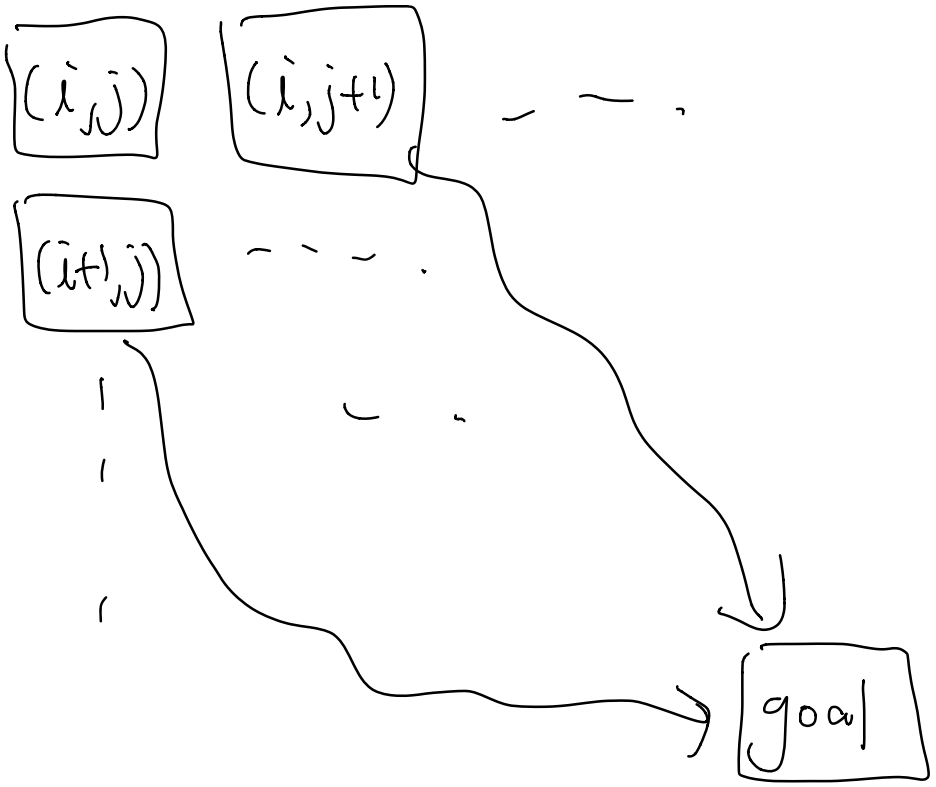
dp[i][j] = min(map[i][j], map[i][j] + max(dp[i+1][j], dp[i][j+1]))

map[i][j] → itself can become a valley.

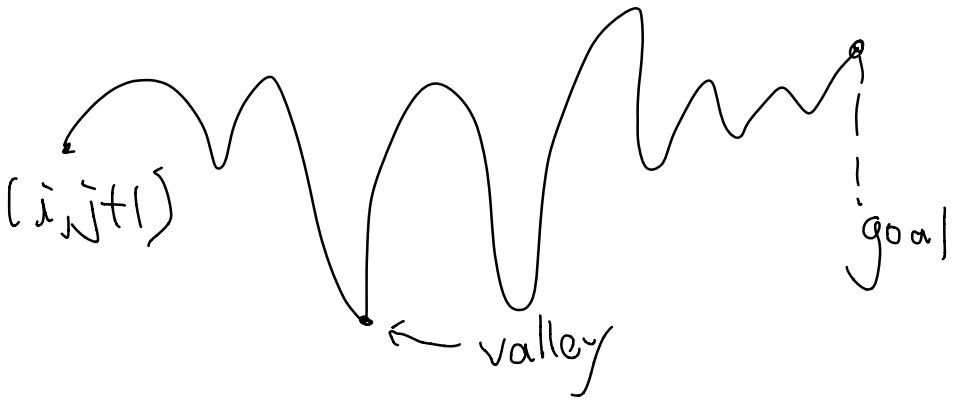
return if dp[0][0] ≥ 0, output 1

else output (-dp[0][0] + 1)

Illustration:

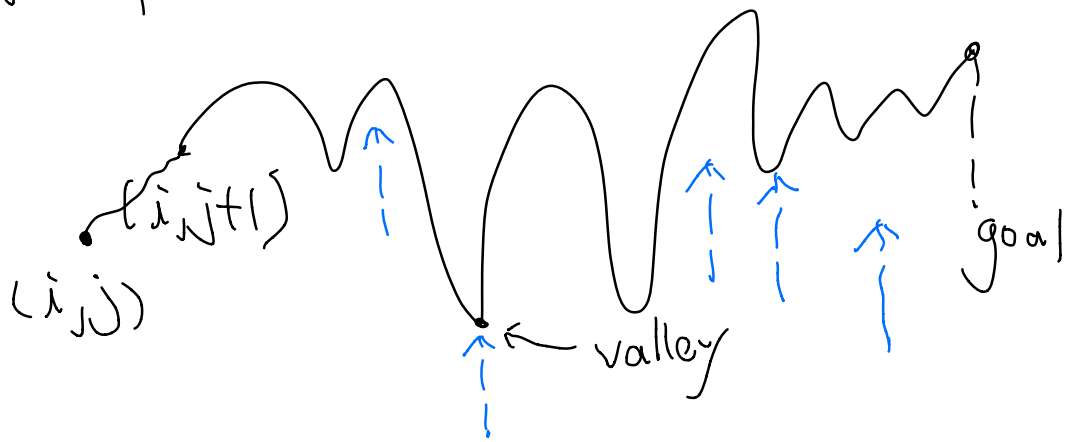


(i,j+1) ⇝ goal



adding map[i][j] will look like:

If map[i][j] ≥ 0



↑ means the whole path is shifted up by map[i][j]

If map[i][j] < 0, the whole path will be shifted down

By adding (i,j), the new valley can only be

① map[i][j]

or

② max(map[i][j] + dp[i][j+1], map[i][j] + dp[i+1][j])

↳ shift the path's value by map[i][j]