**Subtask 1**: Implement

**Subtask 2**: Sử dụng kỹ thuật đệ quy để sinh ra tất cả trường hợp.

**Subtask 3**: Quy hoạch động

Nhận thấy ô (i, j) chỉ có thể được đi tới từ ô:

* (i – 1, j) nếu a[i - 1][j] = ‘D’ hoặc a[i - 1][j] = ‘+’.
* (i + 1, j) nếu a[i + 1][j] = ‘U’ hoặc a[i + 1][j] = ‘+’.
* (i, j - 1) nếu a[i][j - 1] = ‘R’ hoặc a[i][j - 1] = ‘+’.
* (i, j + 1) nếu a[i][j + 1] = ‘L’ hoặc a[i][j + 1] = ‘+’.
* Sử dụng mảng dp[i][j][k] với k chạy từ 1 -> t.
* Công thức:

Nếu a[i][j] != ‘#’:

dp[i][j][k] = dp[i - 1][j][k - 1] \* (a[i - 1][j] = ‘D’ || a[i - 1][j] = ‘+’)

+ dp[i + 1][j][k - 1] \* (a[i + 1][j] = ‘U’ || a[i + 1][j] = ‘+’)

+ dp[i][j - 1][k - 1] \* (a[i][j - 1] = ‘R’ || a[i][j - 1] = ‘+’)

+ dp[i][j + 1][k - 1] \* (a[i][j + 1] = ‘L’ || a[i][j + 1] = ‘+’).

* Kết quả:

**Subtask 4**: Sử dụng nhân ma trận để tối ưu quy hoạch động.

* Chuyển mảng 2 chiều a[h][w] về mảng 1 chiều a[h \* w].
* Ma trận mũ có dạng base[h \* w][h \* w].
* Ma trận kết quả res = a \* baset.
* Kết quả =