

視覚で解き明かす： ナイトの巡歴問題

コードと盤面が交差するアルゴリズムの設計図



【 ミッションの全体像 】

- Goal: $M \times N$ の盤面を、ナイトの駒で全て1回ずつ通過する経路を発見すること。
- Method: 再帰関数を用いた「深さ優先探索 (DFS)」

The Visualization

1	12	3
4	9	6
7	2	11
10	5	8

行き止まりになれば戻り、別の道を探す。この軌跡をいかにプログラムで表現するかが鍵。

物理的な動きをデータに変換する

物理空間の表現



データ配列へのマッピング

図1の移動先	①	②	③	④	⑤	⑥	⑦	⑧
● dv[]	-2	-1	1	ア	2	1	-1	-2
dh[]	1	2	2	1	-1	イ	-2	-1



[ア]のロジック解剖：移動先④への縦ベクトル



Step 1

縦方向（行）の移動を抽出。

Step 2

上方向がマイナスなら、
下方向は…？（符号の反転）

$$dv[4] = [\text{?}]$$

[イ]のロジック解剖：移動先⑥への横ベクトル



Step 1

横方向（列）の移動を抽出。

Step 2

右方向がプラスなら、左方向は・・・？（符号の反転）

$dh[6] = [イ?]]$

深さ優先探索 (DFS) の全体構造マップ

軌跡の記録

```
1 void search(int, h* oam) {  
2     int board = v h;  
3  
4     // この探索を軌跡の存在を // この状態は上断る内容を調す  
5     board[v][h] ← i  
6     if ( ... ) {  
7         printBoard()  
8         printFlag ← 1  
9     }  
10    // 次の探索の増やす  
11    for( j を 1 から 8まで 1ずつ増やす )  
12        search( ... ), board[v][h] ←, printFlag ← 1;  
13    endfor  
14    キ  
15  
16    return 0;  
17 }  
18
```

終了条件・ゴール判定

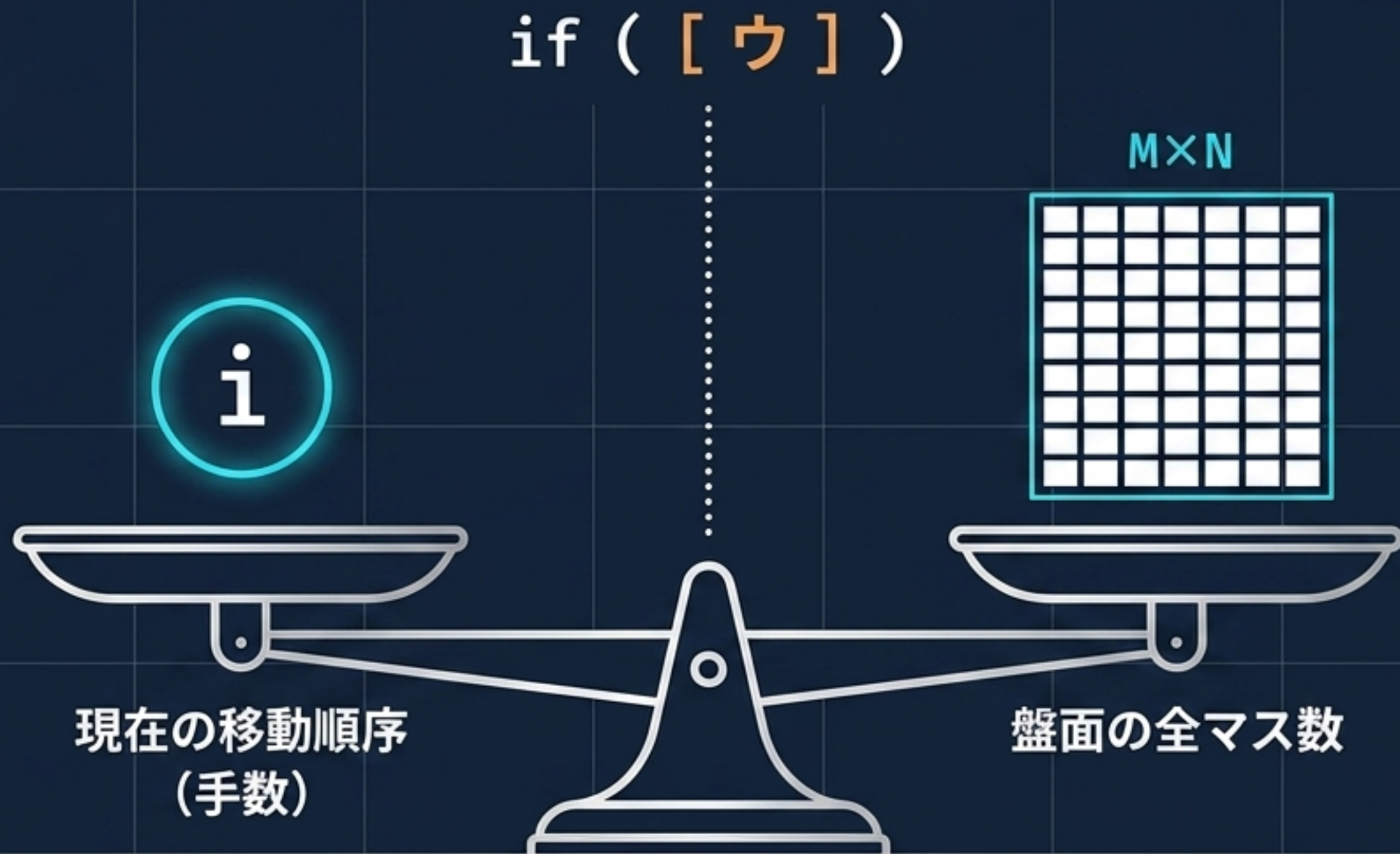
次の探索・再帰呼び出し

取り消し・バックトラック

まずは全体像を把握。
この4つのブロックが連動して迷路を解く。

[ウ]のロジック解剖：ゴール到達の判定

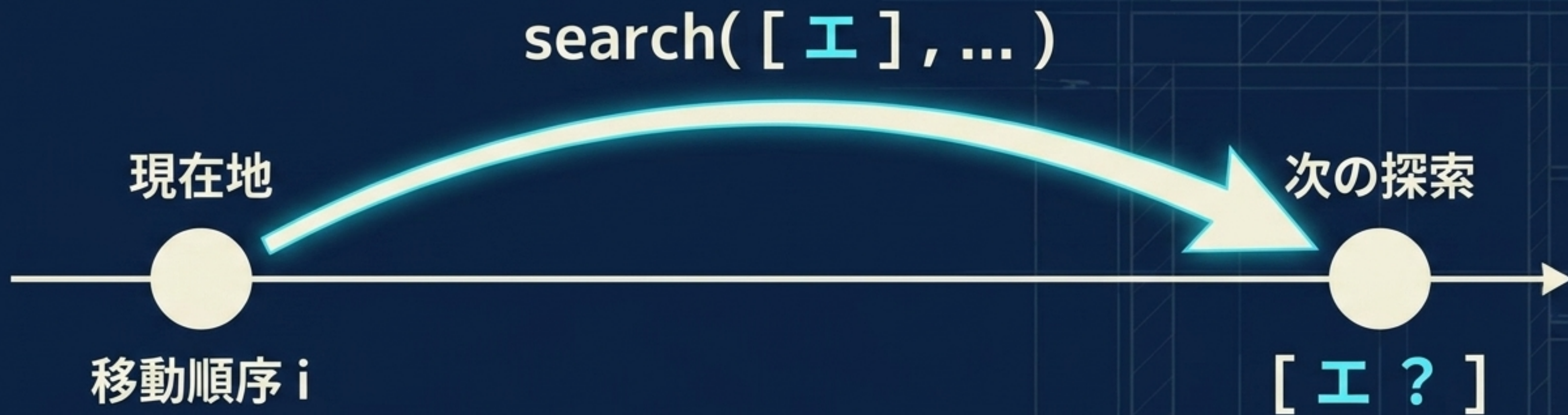
『記録した移動順序が $M \times N$ に等しい
場合...解の一つとして出力する』



盤面の全マス数を、変数 m と n を使った式で表すと天秤が釣り合う？

ウ = ??

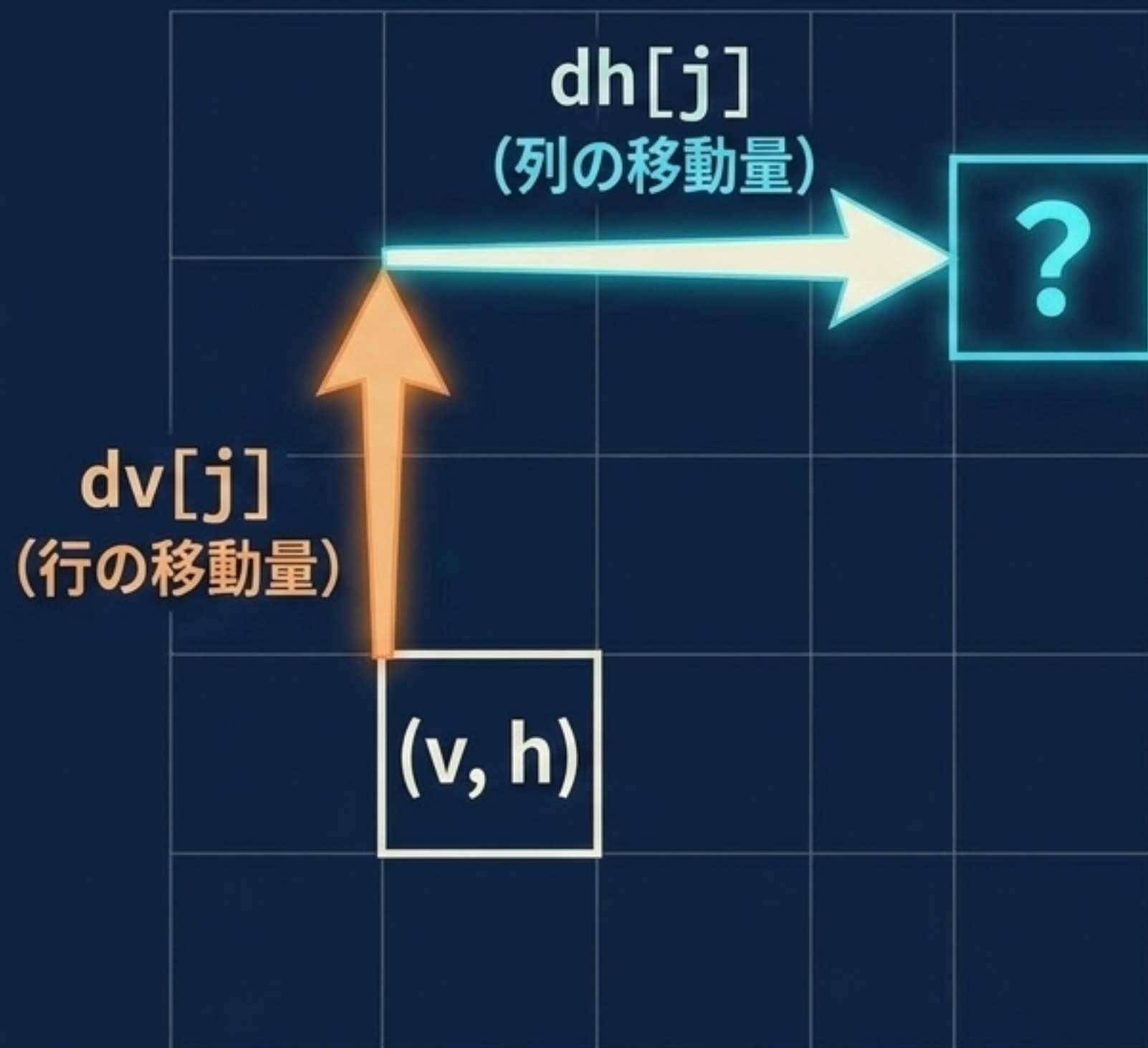
[エ] のロジック解剖：次の一手へ（時間の進行）



再帰関数の第1引数は『次の手の番号』を指す。

現在が i 手目なら、次のステップに渡すべき数は？

[オ][カ]のロジック解剖：次の一手へ（空間の移動）



次の行：現在の行 v に何を足す？

`search( ..., [オ], [カ] )`

次の列：現在の列 h に何を足す？

[キ] のロジック解剖：行き止まりからの帰還（バックトラック）

Context: forループ（8方向の探索）が全て失敗して終わった直後の処理。

『現在のマスの移動順序を取り消して，マスを通過していない状態に戻す。』



Step 1:

Line 4:

```
if( board[v][h] が 0 )
```

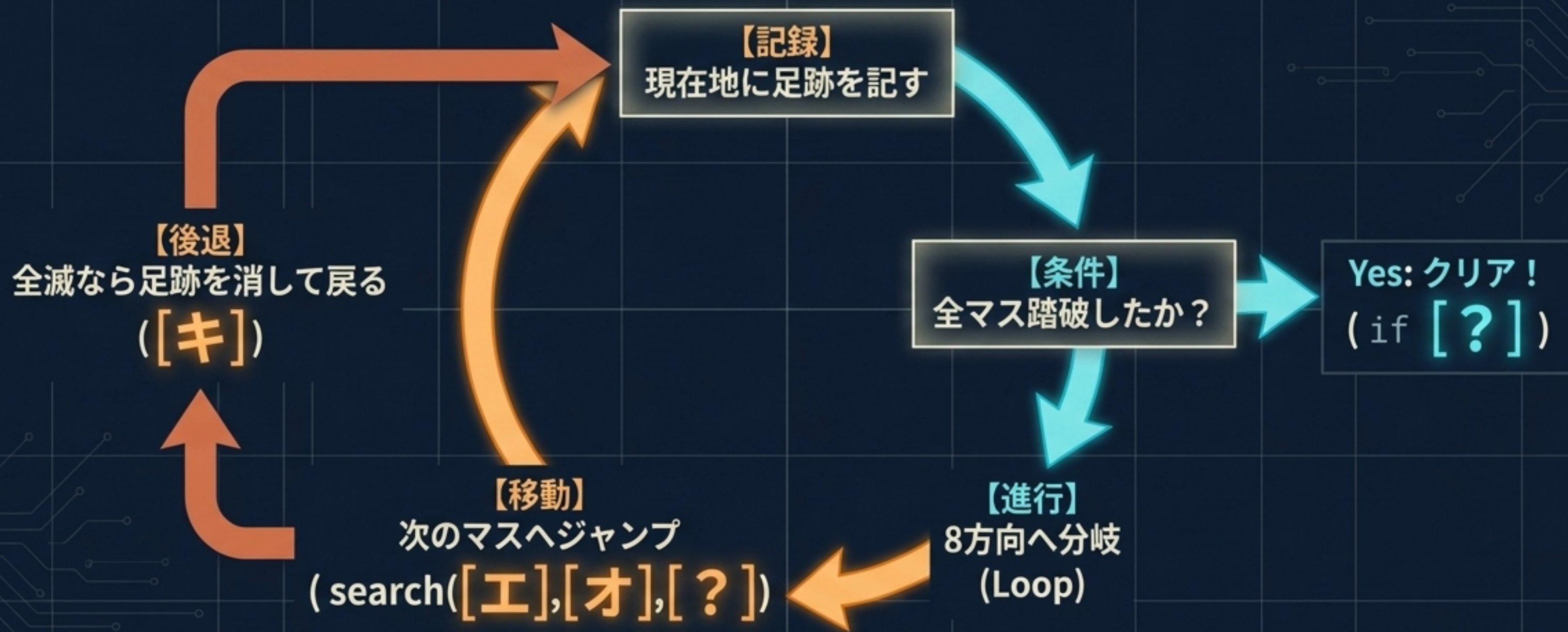
//通過したマス进行判定する

Step 2:

Line 14: [キ]

未訪問（通過していない状態）を意味する『初期値』を代入するには？

アルゴリズムの軌跡：思考のピースを統合する



移動のルール（幾何）と、状態の管理（時間・空間・後退）が交差する時、ナイトは全てのマスを網羅する。
ピースは全て揃った。あなたの頭の中で、コードを完成させよう。