Backtracking 02
Backtracking (Part 2)

Sudoku Solver

No repetition

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

we need to fill numbers in the blank.

allowed no → 1-9

9 Rows

9 Cols

9 Big Cells → 3x3

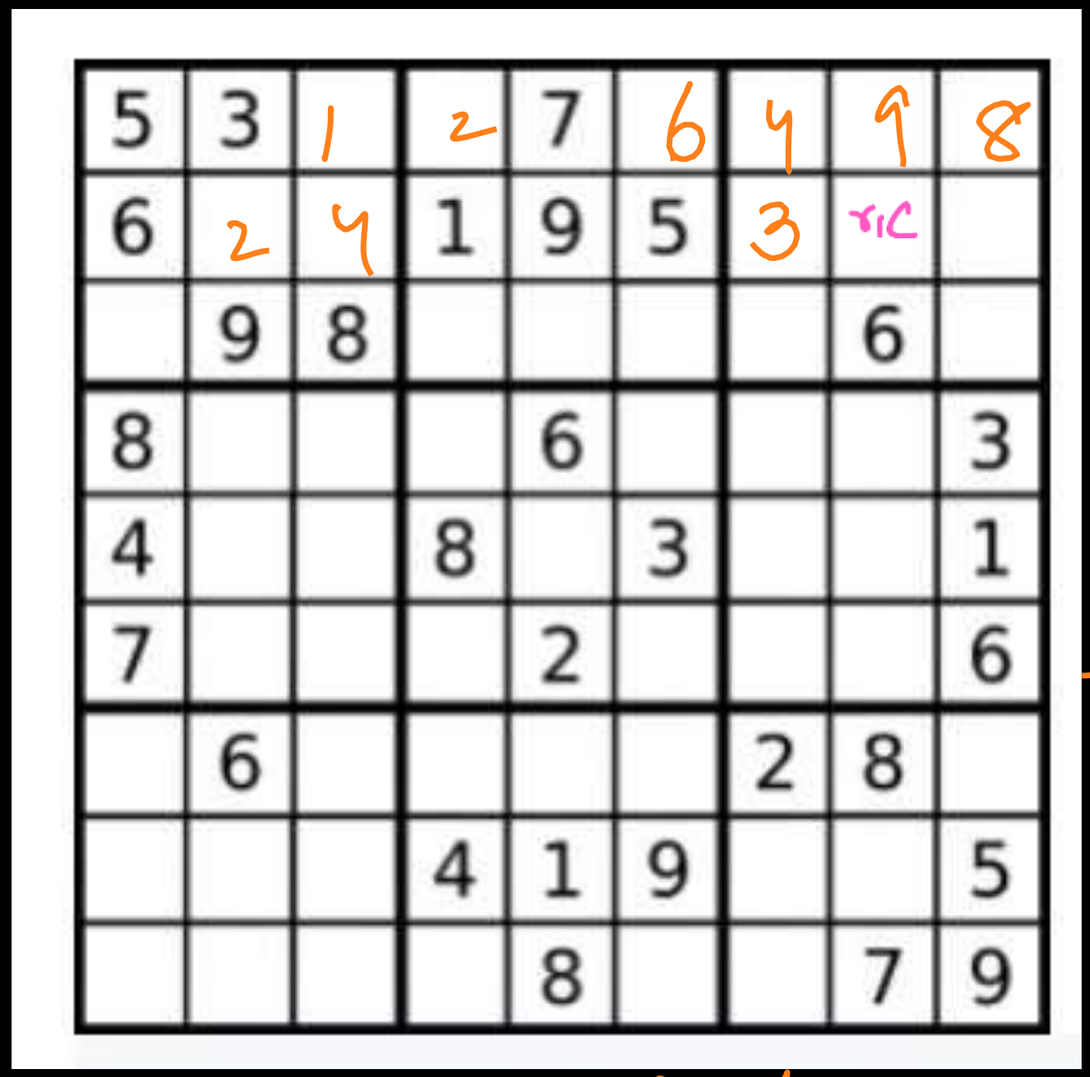
On every row or col or Big cell no number is repeated.

find any invalid s.t

9x9

2d array

Why not try all possibilities ?:



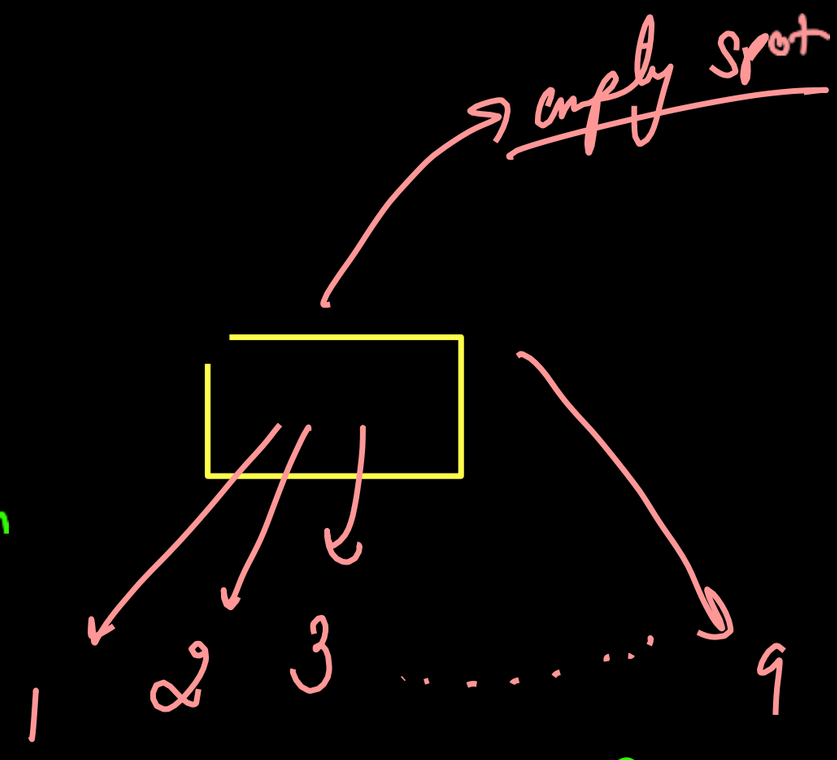
5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

0 1 2 3 4 5 6 7 8

0 1 2 → 0th Big Row

3 4 5 → 1st Big Row

6 7 8 → 2nd Big Row



$\left(\frac{col}{3}\right) \times 3$

$\left(\frac{row}{3}\right) \times 3$

$\left(\frac{0}{3}\right)\times 3 \to 0 \quad \left(\frac{1}{3}\times 3\right) \to 0$

$\left(\frac{4}{3}\right)\times 3 \to 0$

$\left(\frac{3}{3}\right)\times 3 \quad \left(\frac{4}{3}\times 3\right) \quad \left(\frac{5}{3}\right)\times 3$

3 3 3

we will go row by row :-

B.C → r == 9 →

$f(grid, r, c)$ =

if (isSafe (grid, r, c, j)) {

grid [r][c] = j;

$f(grid, r, c+1)$

grid [r][c] = '.';

}

$f(grid, r+1, 0)$

$f(grid, r, c+1)$

if (grid[r][c] == '.')

$\forall j \in [1, 9]$

if (c >= 9)

if (grid[r][c] != '.')

↳ not

it will fill the grid for sudoku from (r,c) cell in row by row form

main → $f(grid, 0, 0)$

```
bool sudoku ( grid, r, c) {
    if ( r == 9) return true;
    if ( c == 9) return f(grid, r+1, 0);
    if ( grid[r][c] != '.') return f(grid, r, c+1);

    for ( d = 1; d <= 9 ; d++) {
        if ( isSafe ( grid, r, c, d)) {
            grid[r][c] = '0' + d
            r = f(grid, r, c+1);
            if ( r == true) return true;
            grid[r][c] = '.';
        }
    }
    return false;
}
```

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

$\left(\frac{row}{3}\right) \times 3 \rightarrow x$

$\left(\frac{col}{3}\right) \times 3 \rightarrow y$

row, col → x, y

for (max ; m < x+3 ; m++)

[2, 5, 2, 1, 2] target = 5

[5]

[2, 2, 1]

[1, 2, 2, 2, 5]

combination → subset

Set S = { _____ } n

How many subsets → 2^n

Brute force

(generate all possible subsets) → 2^n

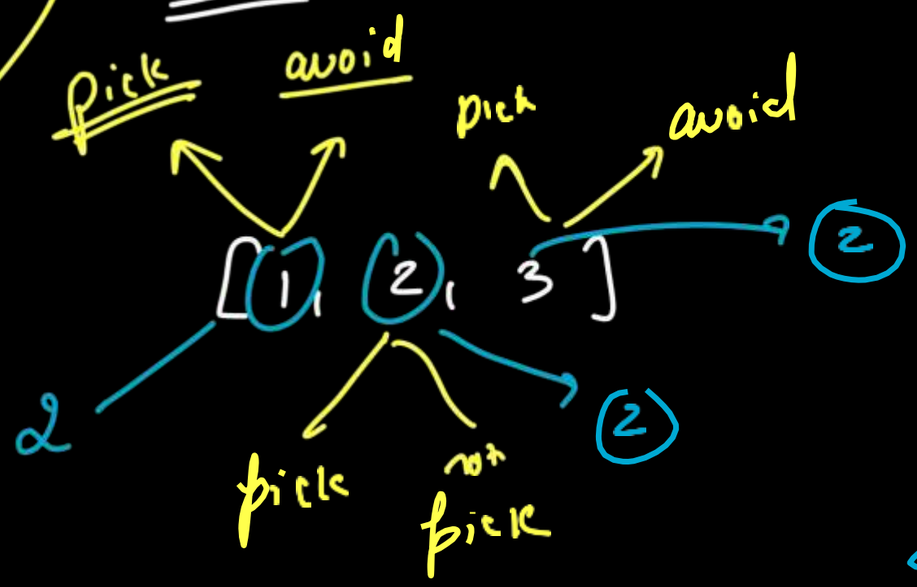
{1, 2, 3} → 3

{1} {2} {3}

{1, 2} {1, 3} {2, 3}

{1, 2, 3} {ϕ}

2^3



<2 * 2 * 2 = 8 2^3

[1 2 3]

	x	x	x	→ { }
→	x	x	✓	{ 3 }
	x	✓	x	{ 2 }
	x	✓	✓	{ 2, 3 }
	✓	x	x	{ 1 }
	✓	x	✓	{ 1, 3 }
	✓	✓	x	{ 1, 2 }
	✓	✓	✓	{ 1, 2, 3 }

Instead of generating all subsets we can form some of them which are not necessary:

$f(cand, t, idx, subset) =$ (vector)

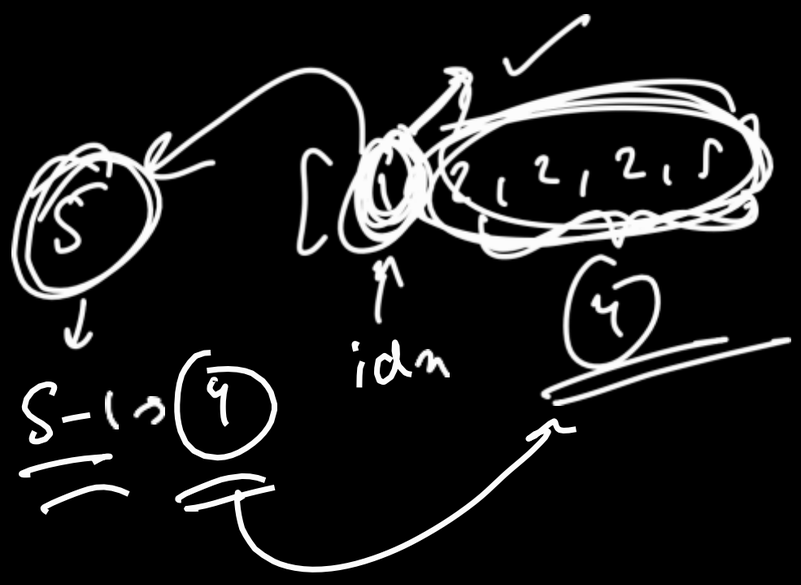
$f(cand, t, idx+1, subset)$ if $(cand[idx] > t)$ ↓ not pick

// pick
subset.push(cand[idx])
$f(cand, t - cand[idx], idx+1, subset)$
subset.pop()

else

// avoid
$f(cand, t, idx+1, subset)$

↓
Calc subset, from cand[idx, n-1]
such that target sum is t



([1, 2, 2, 2, 5], 5)

(2, 2, 2, 5) → 4

(2, 2, 5) → 4

1, 2, 2, 2, 5

x

6 → ugly

[1, 2, 2, 2, 5)

(cand, 5, 0, {})

✓

X

(cand, 4, 1, {1})

✓

{1, 2}

(cand, 2, 2, {1, 2})

(cand, 4, 2, {1})

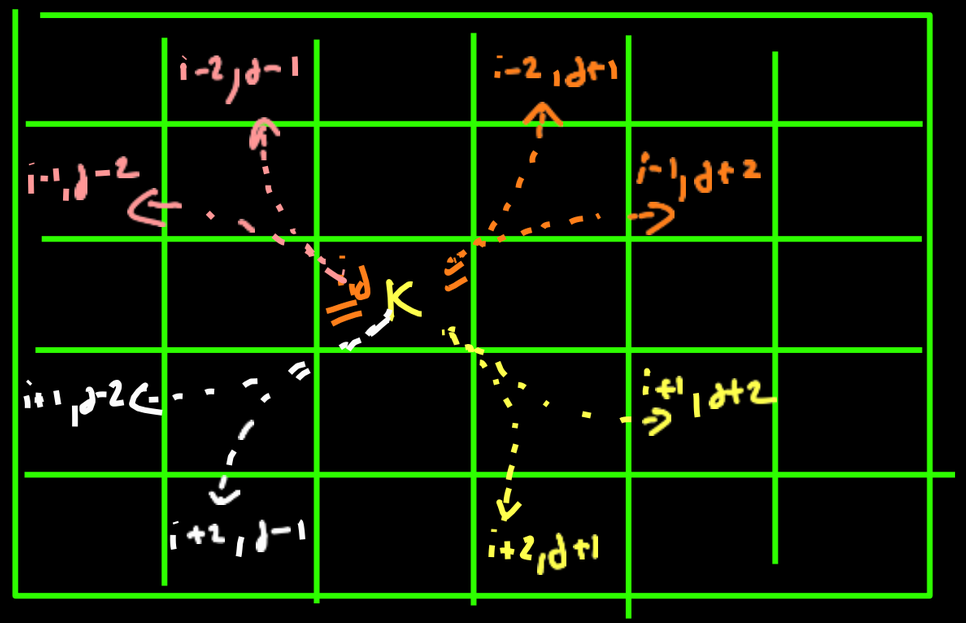
✓

X

[cand, 0, 3, {1, 2, 2})

(cand, 2, 3, {1, 2})

✓



Move for a Knight

no cell should be revisited

Bonus

K_0	K_{13}	K_{14}		K_2
	K_4	K_1	K_{12}	K_{15}
K_6	K_9	K_{19}	K_3	
		K_5	K_8	K_{11}
	K_7	K_{10}		

5×5

$n = 5$

isItSafe( )

grid( )[ ] = k Step

f( )

grid[ ][ ] = " "

PW SKILLS

THANK YOU