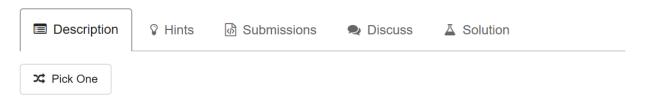
36. Valid Sudoku



Determine if a 9x9 Sudoku board is valid. Only the filled cells need to be validated according to the following rules:

- 1. Each row must contain the digits 1-9 without repetition.
- 2. Each column must contain the digits 1-9 without repetition.
- 3. Each of the 9 3x3 sub-boxes of the grid must contain the digits 1-9 without 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

Example 1:

```
Input:
[
    ["5","3",".","","7",".","","",""],
    ["6",".",".","1","9","5",".","6","."],
    ["8",".",".","6",".",".","3"],
    ["4",".",".","8",".","3",".","","1"],
    ["7",".",".","2",".",".","6"],
    [".","6",".",".",".","2","8","."],
    [".",".",".","4","1","9",".","5"],
    [".",".",".",".","8",".","7","9"]
]
Output: true
```

Example 2:

```
Input:
[
    ["8","3",".",".","7",".",".",".","."],
    ["6",".",".","1","9","5",".",""],
    ["8",".",".",".","6",".",".","3"],
    ["4",".",".","8",".","3",".","","1"],
    ["7",".",".","2",".",".","6"],
    [".","6",".",".","2","8","."],
    [".","6",".",".","8",".","5"],
    [".",".",".","8",".","7","9"]
]
Output: false
```

```
class Solution {
  public boolean isValidSudoku(char[][] board) {
         HashSet<Character> set = new HashSet<Character>();
         for(int i = 0; i < 9; i ++) {
             for(int j =0; j < 9; j ++) {
                if(board[i][j] != '.') {
                    if(!set.contains(board[i][j])) {
                         set.add(board[i][j]);
                     }else {
                        return false;
                 }
             set.clear();
         }
         for(int j = 0; j < 9; j ++) {
             for(int i = 0; i < 9; i ++) {
                 if(board[i][j] != '.') {
                     if(!set.contains(board[i][j])) {
                         set.add(board[i][j]);
                     }else {
                        return false;
                 }
             }
             set.clear();
         }
         for(int k = 0; k < 9; k ++) {</pre>
             for(int i = k / 3 * 3; i < k / 3 * 3 + 3; i ++) {
                  for (int j = (k%3)*3; j < (k%3)*3+3; j++) {
                       if (board[i][j] == '.')
                             continue;
                         if (set.contains(board[i][j]))
                             return false;
                         set.add(board[i][j]);
                  }
             }
             set.clear();
         return true;
}
}
```