**有效的数独**

判断一个 9x9 的数独是否有效。只需要**根据以下规则**，验证已经填入的数字是否有效即可。

1. 数字 1-9 在每一行只能出现一次。
2. 数字 1-9 在每一列只能出现一次。
3. 数字 1-9 在每一个以粗实线分隔的 3x3 宫内只能出现一次。



上图是一个部分填充的有效的数独。

数独部分空格内已填入了数字，空白格用 '.' 表示。

**示例 1:**

**输入:**

[

["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"]

]

**输出:** true

**示例 2:**

**输入:**

[

  ["8","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"]

]

**输出:** false

**解释:** 除了第一行的第一个数字从 **5** 改为 **8** 以外，空格内其他数字均与 示例1 相同。

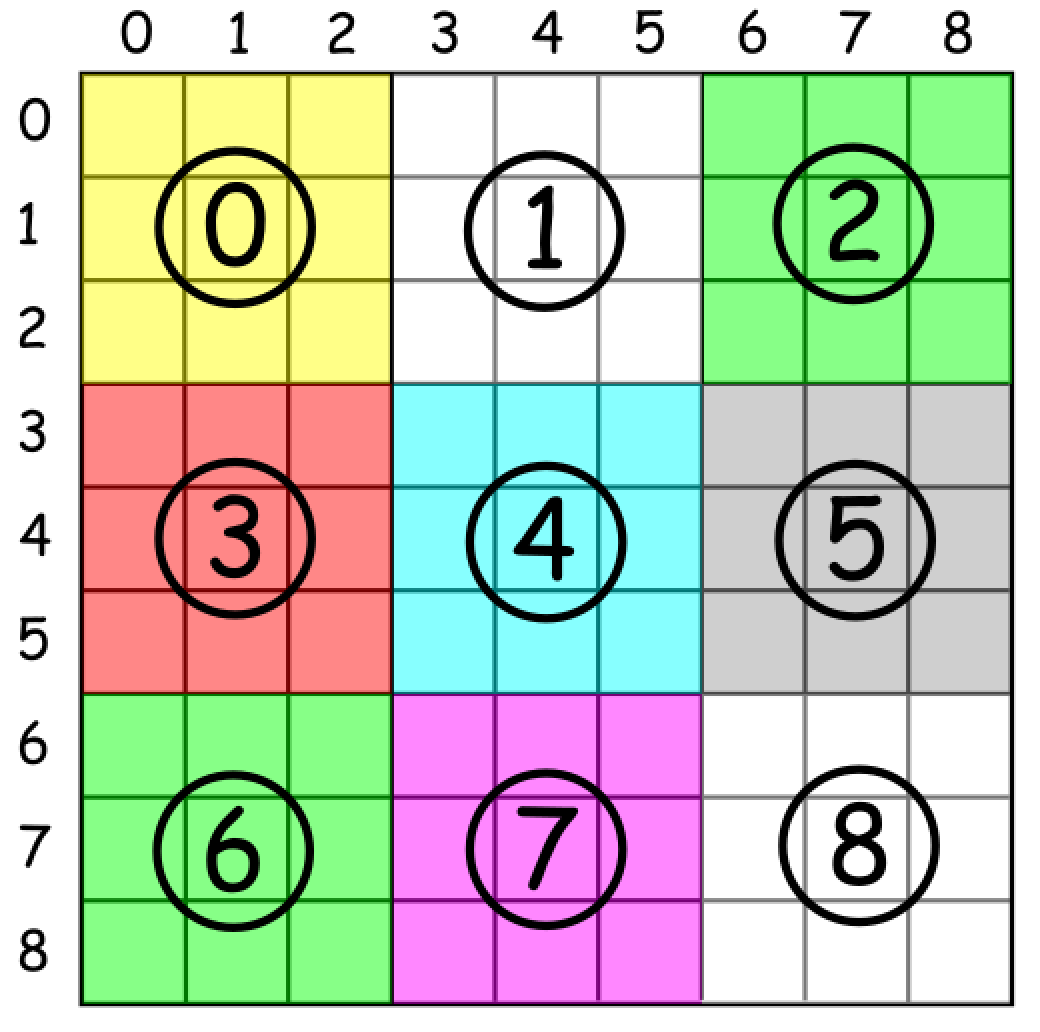
但由于位于左上角的 3x3 宫内有两个 8 存在, 因此这个数独是无效的。

**说明:**

* 一个有效的数独（部分已被填充）不一定是可解的。
* 只需要根据以上规则，验证已经填入的数字是否有效即可。
* 给定数独序列只包含数字 1-9 和字符 '.' 。
* 给定数独永远是 9x9 形式的。

Solution

其实就是在判断行，列，3x3的子数独中，是否出现了重复的数字，复杂的问题拆解开就是基本的问题。判断重复数字一般大多使用哈希表。



* 如何枚举子数独？

可以使用 box\_index = (row / 3) \* 3 + columns / 3，其中 / 是整数除法。

* Python

class Solution:

def isValidSudoku(self, board):

"""

:type board: List[List[str]]

:rtype: bool

"""

# init data

rows = [{} for i in range(9)]

columns = [{} for i in range(9)]

boxes = [{} for i in range(9)] #为三个要判别的新建哈希表

# validate a board

for i in range(9):

for j in range(9):

num = board[i][j]

if num != '.':

num = int(num) # 类型转换

box\_index = (i // 3 ) \* 3 + j // 3 # 子数独序号

# keep the current cell value

rows[i][num] = rows[i].get(num, 0) + 1

columns[j][num] = columns[j].get(num, 0) + 1

boxes[box\_index][num] = boxes[box\_index].get(num, 0) + 1

# check if this value has been already seen before

if rows[i][num] > 1 or columns[j][num] > 1 or boxes[box\_index][num] > 1:

return False

return True

Python 中 dictionary get函数需要注意 get(key,value) 如果字典中存在key的value 则返回对应的value，如果不存在，则赋值value。

Java

class Solution {

public boolean isValidSudoku(char[][] board) {

// init data

HashMap<Integer, Integer> [] rows = new HashMap[9];

HashMap<Integer, Integer> [] columns = new HashMap[9];

HashMap<Integer, Integer> [] boxes = new HashMap[9];

for (int i = 0; i < 9; i++) {

rows[i] = new HashMap<Integer, Integer>();

columns[i] = new HashMap<Integer, Integer>();

boxes[i] = new HashMap<Integer, Integer>();

}

// validate a board

for (int i = 0; i < 9; i++) {

for (int j = 0; j < 9; j++) {

char num = board[i][j];

if (num != '.') {

int n = (int)num;

int box\_index = (i / 3 ) \* 3 + j / 3;

// keep the current cell value

rows[i].put(n, rows[i].getOrDefault(n, 0) + 1);

columns[j].put(n, columns[j].getOrDefault(n, 0) + 1);

boxes[box\_index].put(n, boxes[box\_index].getOrDefault(n, 0) + 1);

// check if this value has been already seen before

if (rows[i].get(n) > 1 || columns[j].get(n) > 1 || boxes[box\_index].get(n) > 1)

return false;

}

}

}

return true;

}

}