Valid Sudoku

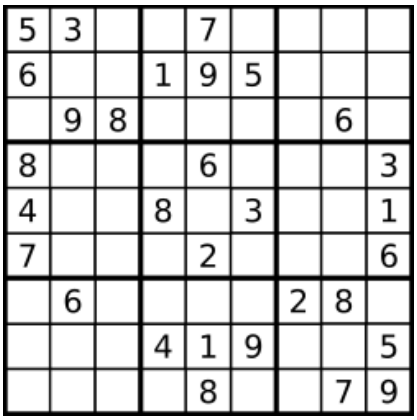
Determine if a `9 x 9` Sudoku board is valid. Only the filled cells need to be validated **according to the following rules**:

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 nine `3 x 3` sub-boxes of the grid must contain the digits `1-9` without repetition.

Note:

- A Sudoku board (partially filled) could be valid but is not necessarily solvable.
- Only the filled cells need to be validated according to the mentioned rules.

Example 1:



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: board =

```
[["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"]]
```

Output: true

Example 2:

Input: board =

```
[["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"]]
```

Output: false

Explanation: Same as Example 1, except with the **5** in the top left corner being modified to **8**. Since there are two 8's in the top left 3x3 sub-box, it is invalid.

```
public class Solution {
    public bool IsValidSudoku(char[][] board) {
        char val;
        bool retVal = true;
        for(int i=0;i<board.Length;i++)
        {
            for(int j=0;j<board[i].Length;j++)
            {
                val = board[i][j];

                if(val != '.')
                {
                    //Check current Row all element
                    for(int c=j+1;c<board[i].Length;c++)
                    {
                        if(board[i][c] == val)
                        {
                            retVal = false;
                            break;
                        }
                    }

                    if(retVal)
                    {
                        //Check current Col all element
                        for(int r=i+1;r<board.Length;r++)
                        {
                            if(board[r][j] == val)
                            {
                                retVal = false;
                                break;
                            }
                        }
                    }

                    if(retVal)
                    {
                        //Check 3X3 Sub array
                        int rMax = ((i/3)*3)+3;
                        int cMax = ((j/3)*3)+3;
                        for(int r=i+1;r<rMax;r++)
                        {
                            for(int c=(j/3)*3;c<cMax;c++)
                            {
                                if(board[r][c] == val)
                                {
                                    retVal = false;
                                    break;
                                }
                            }
                        }
                    }

                    if(!retVal)
                    {
                        break;
                    }
                }
            }

            if(!retVal)
            {
                break;
            }
        }
        return retVal;
    }
}
```