Problema escolhido:

3 - Sudoku Válido

```python
def isValidSudoku(board):
    row = [[] for _ in range(9)]
    column = [[] for _ in range(9)]
    group = [[] for _ in range(9)]
    for i in range(9):
        for j in range(9):
            n = board[i][j]
            if n!='.':
                index = (i // 3) * 3 + (j // 3)
                if n in column[j] or n in row[i] or n in group[index]:
                    return False

                row[i].append(n)
                column[j].append(n)
                group[index].append(n)

    return True


def board():
    board = [[] for i in range(9)]
    for i in range(0, 9):
        board[i] = input().split()
    return print(isValidSudoku(board))
```

A função principal é

```python
def isValidSudoku(board):
    row = [[] for _ in range(9)]
    column = [[] for _ in range(9)]
    group = [[] for _ in range(9)]
    for i in range(9):
        for j in range(9):
            n = board[i][j]
            if n!='.':
                index = (i // 3) * 3 + (j // 3)
                if n in column[j] or n in row[i] or n in group[index]:
                    return False

                row[i].append(n)
                column[j].append(n)
                group[index].append(n)

    return True
```

Ela cria 3 arrays de arrays com 9 elementos e com dois loops passa elemento por elemento do tabuleiro, comparando com os arrays já existentes e vendo se estão ou não listados, caso não estejam, o valor é adicionado ao array, caso esteja, retorna que o tabuleiro não é válido (false).

Para formatar os 9 inputs e facilitar as coisas para a função principal, criei a função board, ela transforma todos os inputs e arrays dentro de um array maior, board.

```
20 def board():
21     board = [[] for i in range(9)]
22     for i in range(0, 9):
23         board[i] = input().split()
24     return print(isValidSudoku(board))
25
```

O resultado seria:
board = [[input_1], [input_2], [input_3], [input_4], [input_5], [input_6], [input_7], [input_8], [input_9]]
E depois retorna o print da função principal.

```
26 board()
```

Chama as função board.

Input 1:

```
>>> ~ python crossbots-ps/ps.py
4 9 . . . 3 . . .
. 3 5 . . . 4 . .
. . 6 . . 8 . . 5
. 5 . 4 7 . . 3 .
7 . . . . . . . 6
. 1 . . 6 5 . 4 .
8 . . 7 . . 2 . .
. . 7 . . . 6 9 .
. . . 5 . . . 7 8
```

Output 1:

```
True
```

Input 2:

```
>>> ~ python crossbots-ps/ps.py
4 9 . . . 3 . . .
. 3 5 . . . 4 . .
. . 6 . . 8 . . 5
. 5 . 4 7 . . 3 .
7 . . . . . . . 6
. 1 . . 6 5 . 4 .
8 . . 7 . . 2 . .
. . 7 . . . 6 9 .
. . . 5 . . . 7 5
```

Output 2:

```
False
```