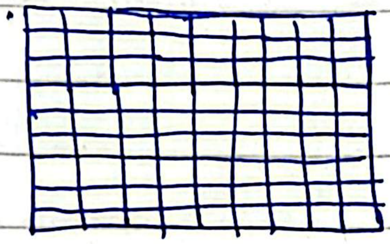
Sudoku Solver.

It is a 9x9 board



Rules ① each digit 1-9 should appear exactly once in any row.

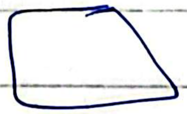
② each digit 1-9 should appear exactly once in every column.

③ digits 1-9 should appear exactly once in a 3x3 board. which is part of 9x9 board

Solution

On every insertion check

↕ column ⟷ row



↓

Subbox

then add. but it exceeds.

OPTIMIZED Solution.

Sets to keep track of each row , cols , sub_boxes.

each rowhave set
like

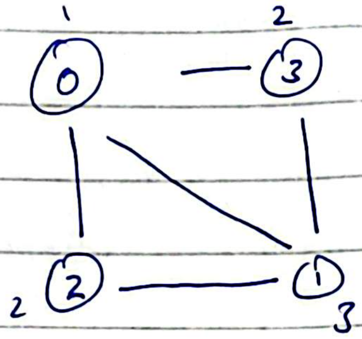
```
rows = [0: set(),
        1: set(),
        set(),
        set(),....]
```

also for columns.

first O(N) traverse full
sodoku

```
    if board not empty like not= "."
    then add it into thr
            rows[r].add(num)
            cols[c].add (num)
            box_indexes = (r//3)*3 + (c//3]
            boxes[box_indexes].add(num)
```

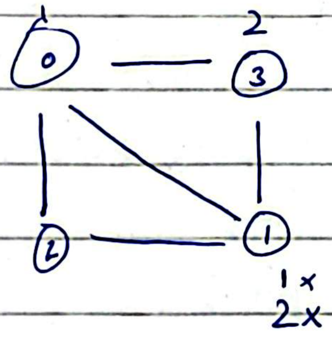
M-Coloring Problem.



M=3

we were able to do 3 color of graph

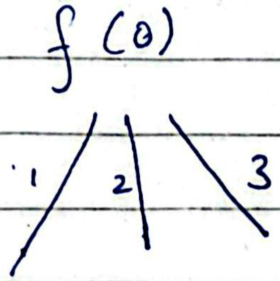
what if M=2



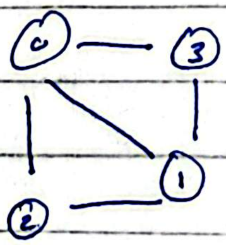
cuz 0—1 is adjacent

we cannot do this.

Recursion Solution

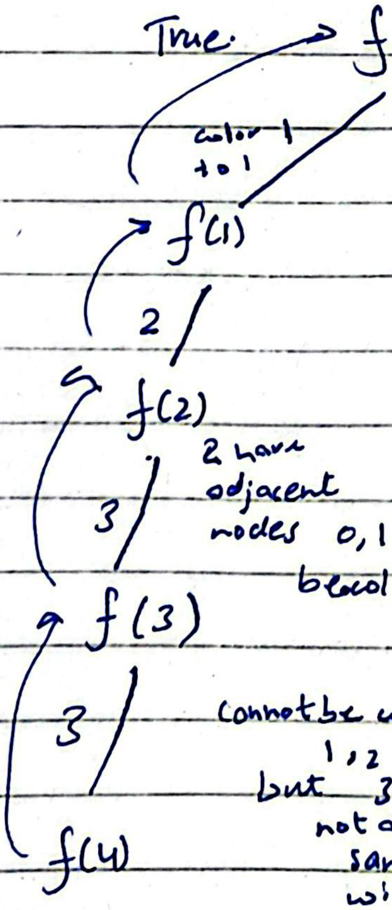


different colors



Now

lets start coloring



2 have odjacent nodes 0,1 it cannot be colored 1 or 2

connot be colored 1,2 but 3 cuz it is not adj to any same node with this no

Code:

```
 f (node)
if node == N
    return T
for (col 1-M)
        if (possible)
   {
      color [node] = color
      if (f (node +1) == T
          { return T }

      else :
            back track

return F
```