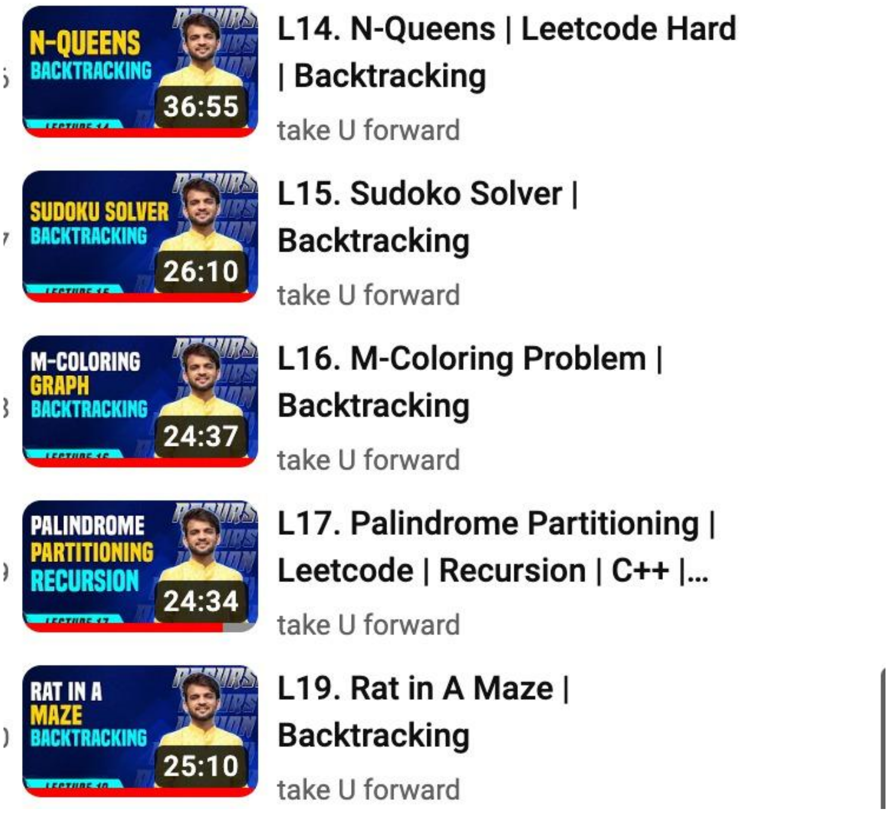
Backtracking Striver sir lecture Notes 📒

By Ramanjot Singh



L14. N-Queens | Leetcode Hard | Backtracking

take U forward

L15. Sudoko Solver | Backtracking

take U forward

L16. M-Coloring Problem | Backtracking

take U forward

L17. Palindrome Partitioning | Leetcode | Recursion | C++ |...

take U forward

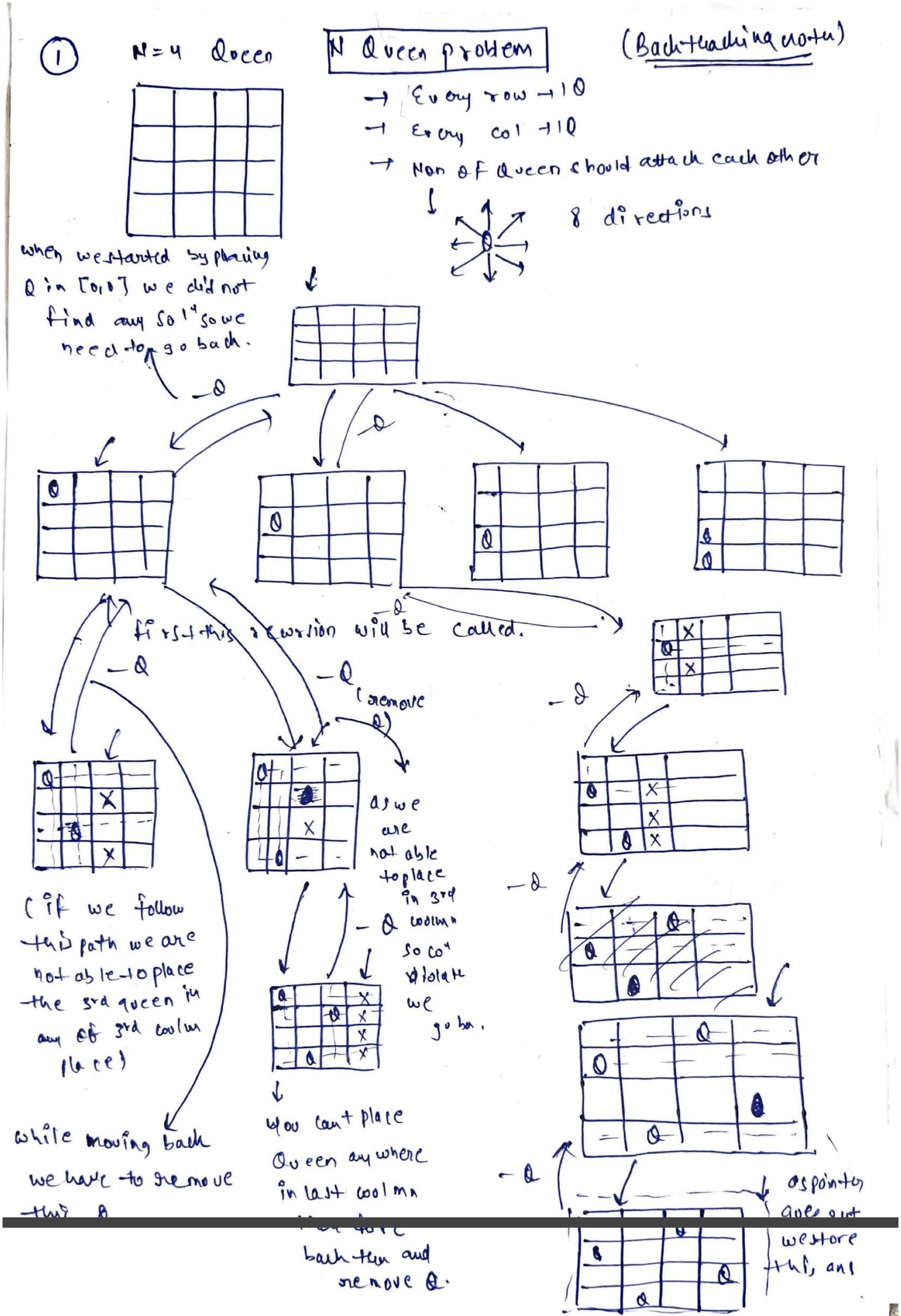
L19. Rat in A Maze | Backtracking

take U forward

~Thank you, Striver Sir, for your exceptional guidance in explaining the complex topic of backtracking. Your teaching has been incredibly effective and I would like to express my gratitude❤

Playlist link

Backtracking(L14-L19



① N=4 Queen

N Queen problem

(Backtracking note)

→ Every row → 1Q

→ Every col → 1Q

→ Non of Queen should attach each other

8 directions

when we started by placing Q in [0,0] we did not find any solⁿ so we need to go back.

first this recursion will be called.

(if we follow this path we are not able to place the 3rd queen in any of 3rd coulm place)

while moving back we have to remove this Q

as we are not able to place in 3rd coolmn so coⁿ violate we go ba.

you can't place Queen anywhere in last coolmn we have back then and remove Q.

as pointer goes out we store this ans

Structure of code

```
f( col)
{   for ( i=0 --> n-1)
    if ( fill ✓ )
      { make [new] [col] = Q
        f( col +1 ).
        empty
      }
}
```

checking is it possible to fill or not

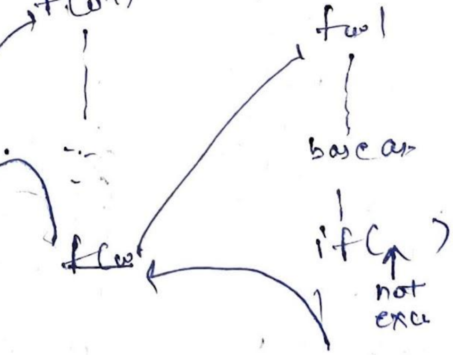
(Q को डाल देना, Matrix में)

उसको fill कर दिया तो next col को move कर गए

when this recursion will be complete remove Q

agar woh safe na hua तो back track करते हुए piche jaya जहाँ से call करना shuru hua tha.

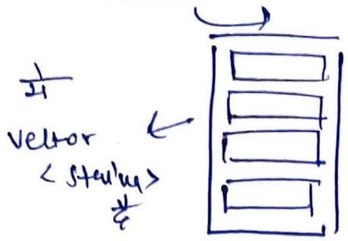
```
f(col)
{  for (i = 0 --> n-1)
     if (fill)
     { make [ ] = Q
       f(col+1)
       emp
     }
}
```

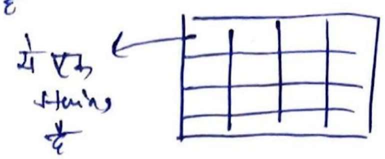


it will return

or Phir for loop next के लिए करेगी

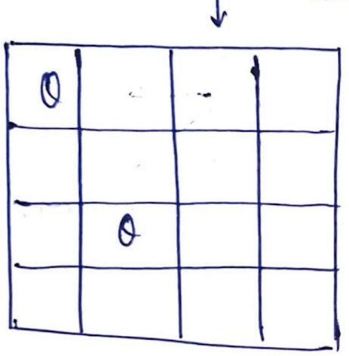
vector <vector <string>> matlab है



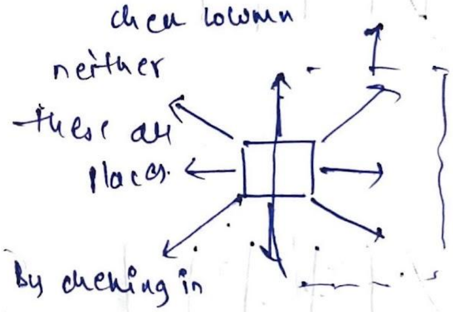


ab inको store करने wale vector <vector<string>> honge जो possible ans store करेंगे।

now we need to write the fuction to chck whether it is safe to fill up or not.

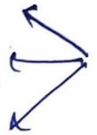


as we are placing queen once in a column so we don't need to chech column neither these all places

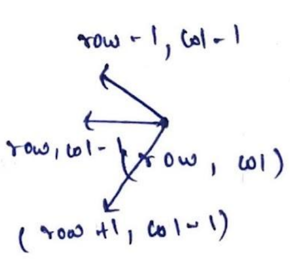


By cheking in these three direction we can be sure about placing queen

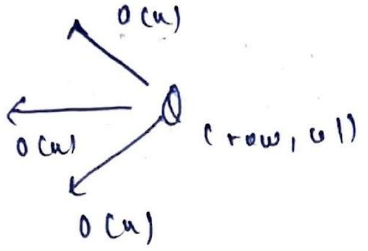
* Queen can attack any in this direction



So we won't just chech in adjacent rather we will run a loop till the end to check.

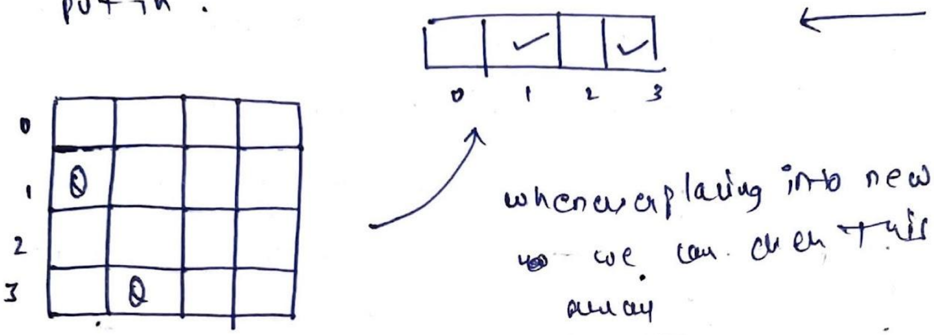


जो भी index आ रहा है उन सबके liya we will run the loop.



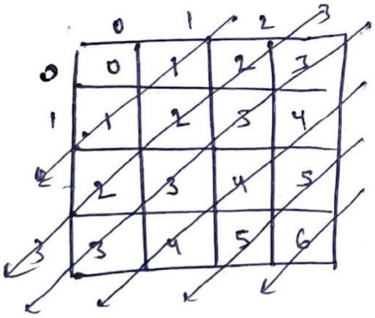
· We can optimise this checking code using hashing

Now for case of row checking we can maintain a array mark it for all the row Q have been put in.



whenever placing into new row we can check this array

for diagonal we need to observe a pattern.



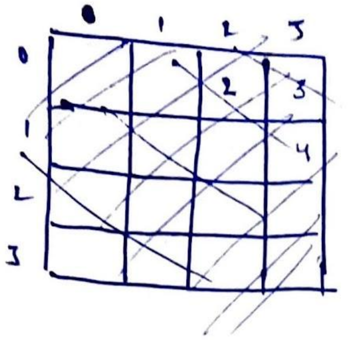
	0	1	2	3
0	0	1	2	3
1	1	2	3	4
2	2	3	4	5
3	3	4	5	6

we can observe this pattern which will there be in an N×N matrix

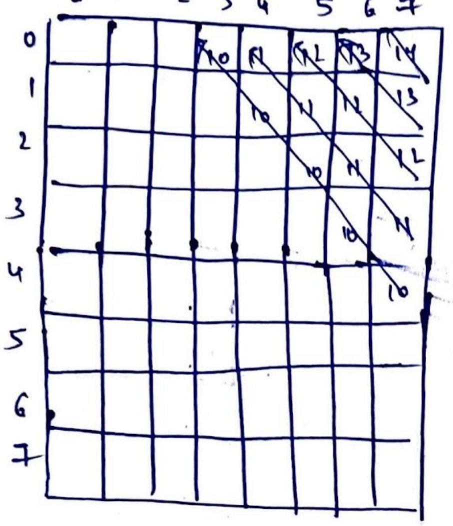
$i + j = K$ for a diagonal,

bcoz of yn.

* So we can make an arr. of 2n-1 size and in that if at any index we place a Q we will mar i+j wala inde→



n = 8



now we need to fill it for

this diagonal

for that we will use

a formula

n-1 + col - row

8-1 + 7 - 0 = 14

8 - 1 + 6 - 0 = 13

8 - 1 + 7 - 1 = 13

So again this we will make an array of 2n-1 size and hash the values.

(n - 1 + j - i) should be marked 1 for any index we place.

So now in recursion we pass left row, upper Diagonal. and lower Diagonal and check if they are zero put 'Q' in specified row and column and mark them 1 and then call recursion once recursion returns un marking all and removing Q or replacing it with blank.

Sudoku Solver!

① The digit 1-9 only appears once in any row or col and in 3×3 matrix

(1) Traverse the matrix to figure ~~of~~ out empty place

(2) when empty place found try all possible combination for it for each number call is valid function

(3)

```
if ( is valid ( board, i, j, c))
{
        board [i] [j] = c ;
        if (solve (board) == true)
            return true;
        else
            board [i][j] = '.';
}
return false.
```

अगर यही ~~स्थ~~ वाला aya true varna backtrack करते हुए replace Kardo जितनी बार है

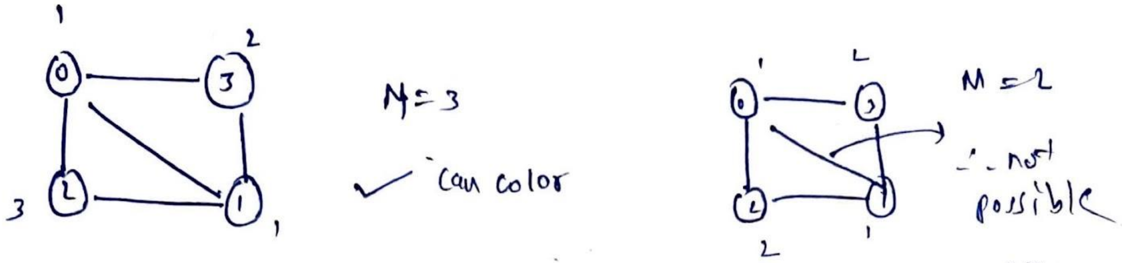
(4) if there are never an empty cell ∴ at last true is returned.

Imp board [3* (row/3) + i/ 3] [3* (col/3) + i%3] = c

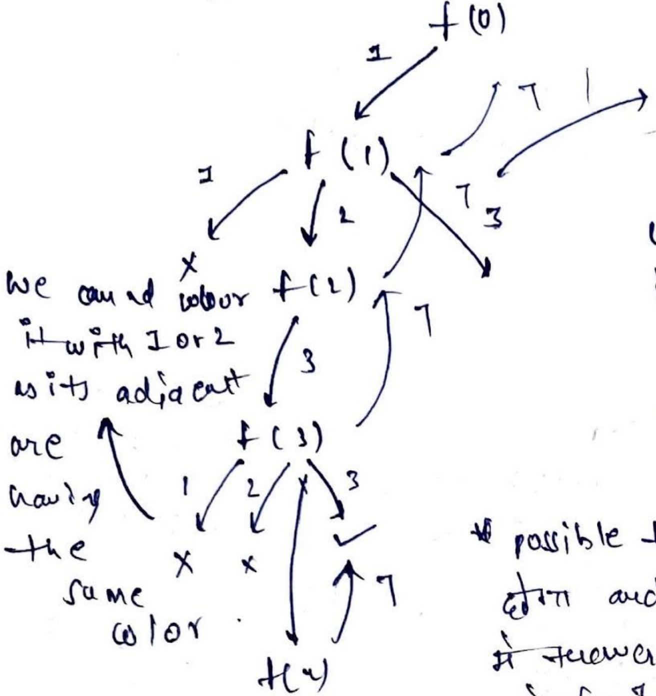
to traverse in small matrix.

M Coloring of Graph!

You have to use max of m colours such a way that no two nodes in graph are having same colour.



* lets try colouring every node with all the possible way.



f(0)
1
f(1)
1
2
3
7
7
x
f(2)
3
7
f(3)
1
2
x
3
x
x
✓
7
f(4)
We can't colour it with 1 or 2 as its adjacent are having the same color.

we have also posibilty at various nodes of colouring but as the question is can we so ∴ when we reach at node N = 4 the we keep on returning true.

* possible fuction में कुछ नहीं node diya होगा and color ad ushe adjacent! में traverse karo and dekho r adjace में किसी का color voh नहीं है

```
f(node)
{   if (node == N) return True;
    for (col = (1 ⟶ m))
    {  if (possible)
       {  color[node] = col;
          if (f(node+1) == T)
             return T
          else  color[node] = 0
       }
    }
    return false.
}
```

a fuction to check if col that node with that particular color possible or not

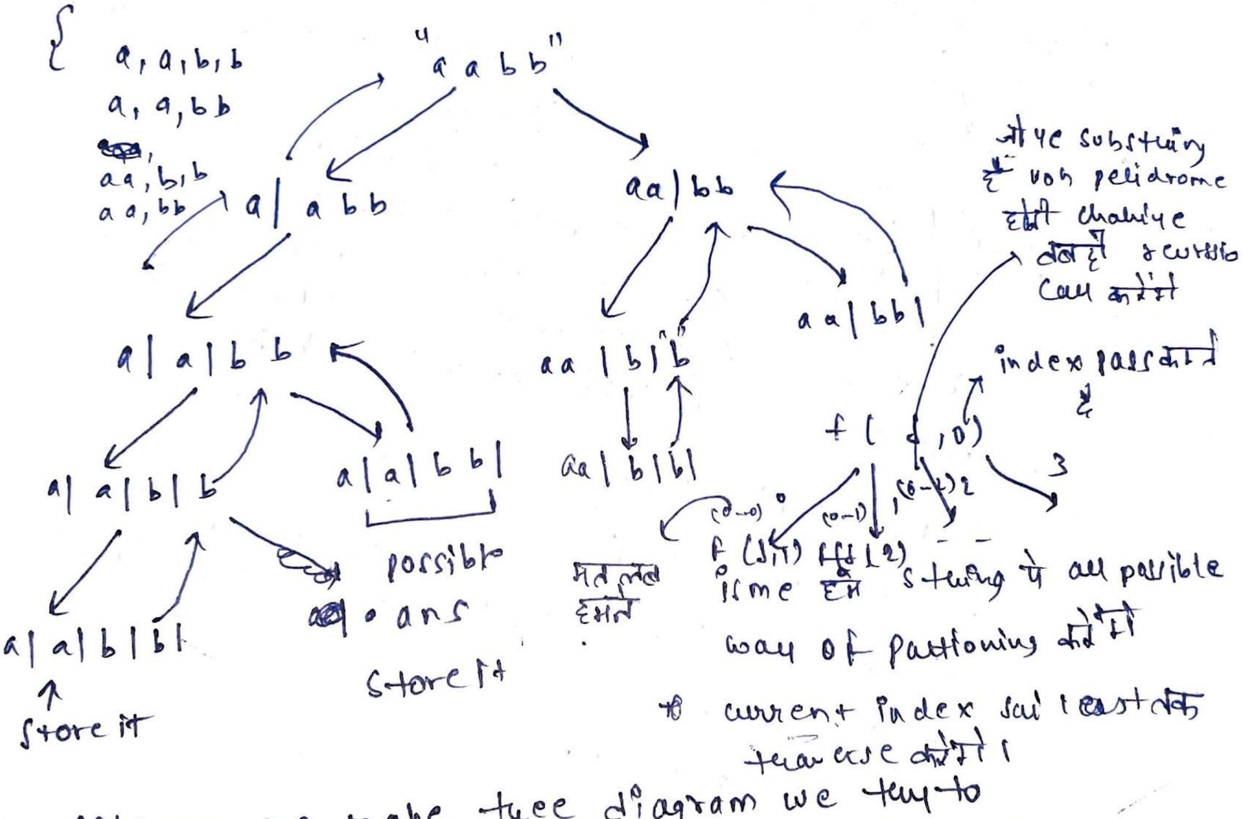
after trying for all the color if it won't be possible then we return false.

TC → (N^m)

SC → O(N) + O(N)

Palindrome Partioning

* wheuor you reach the end your rcursion call is over it means that you are able to partion then we back track.



जो ये substring है non pelidrome होनी chahiye तब ही recursion call करेंगे

index pass करते है

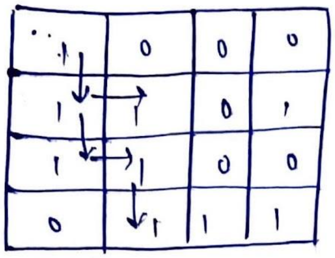
मतलब इसमें

isme EH string पे all possible way of partitioning करेंगे

* current index sai last तक traverse करेंगे।

** wheuer we make tree diagram we try to make all to our given constrains and based on conusion of recursion tree we tell how the rewrsion would be called.

Rat in A Maze :



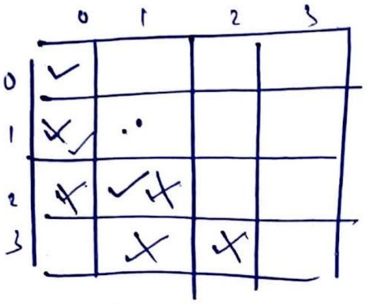
1	0	0	0
1	1	0	1
1	1	0	0
0	1	1	1

→ we will maintain an string in which we would keep adding the order

→ we would start travousing in D L R U fasion as we want laxographical order

→ we also maintain a visited array bcoz we don't want to visit same again and have row and col.

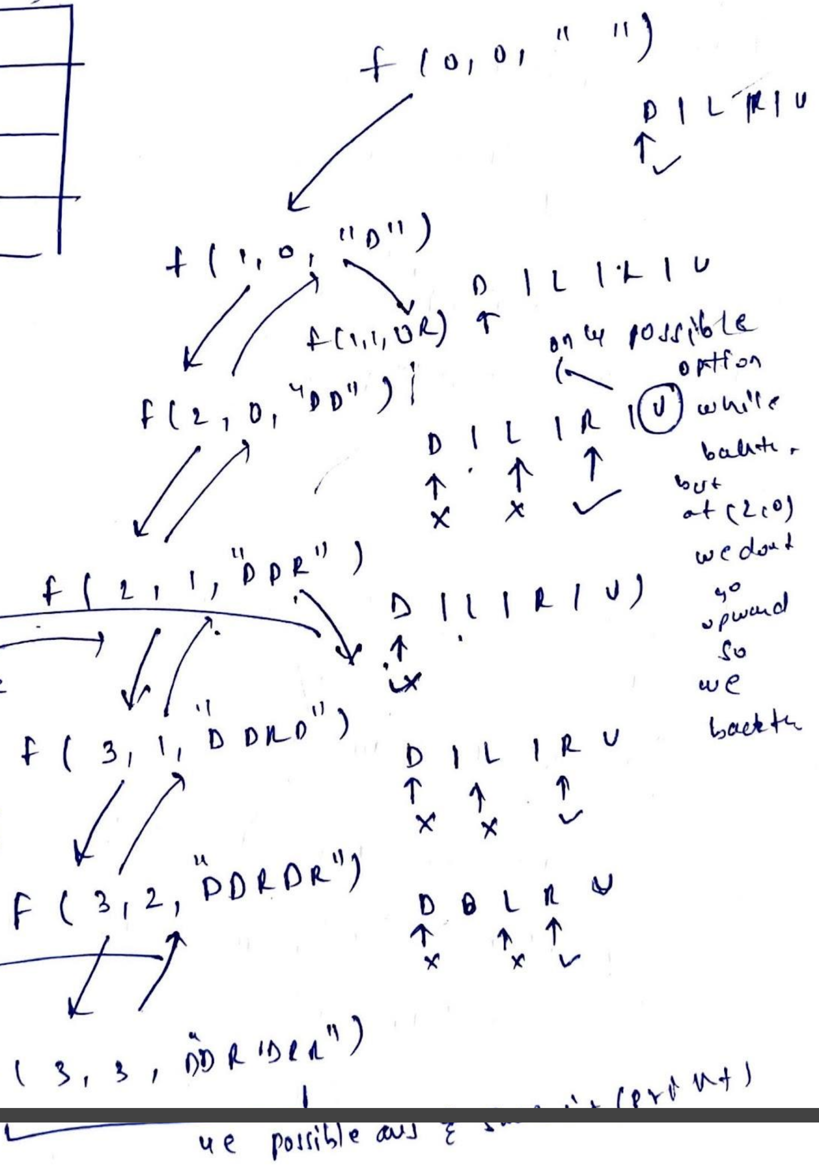
f (0, 0, " ")



	0	1	2	3
0	✓			
1	✗✓			
2	✗	✓✗		
3		✗	✗	

visited.

~~D~~ | L | R | U



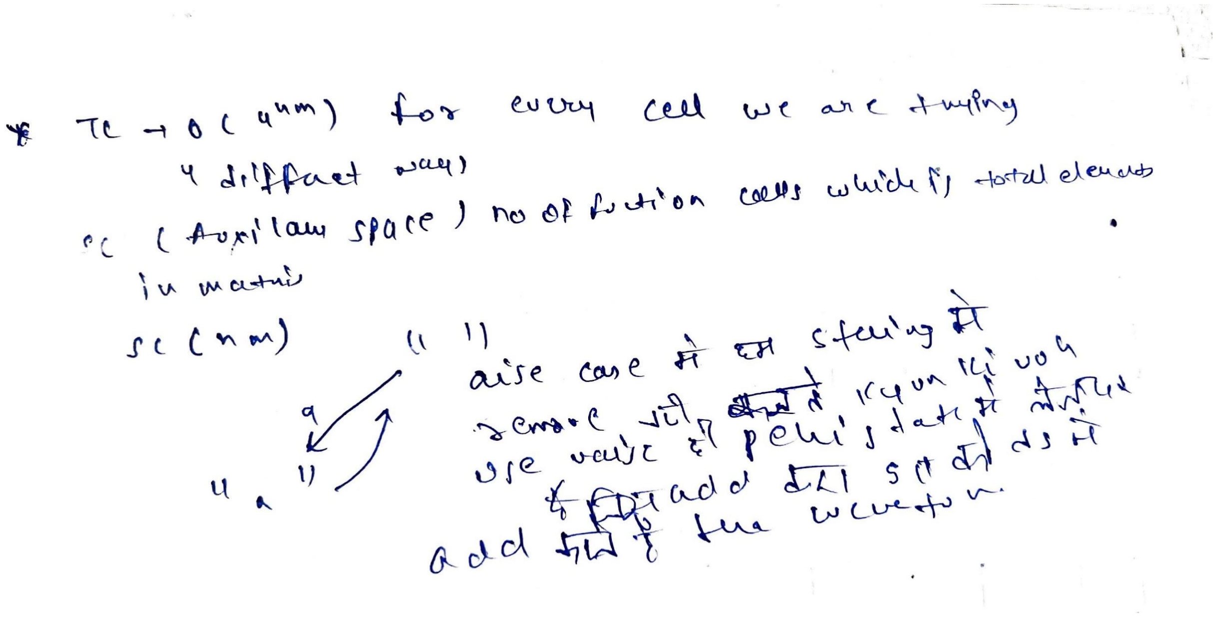
now for this as we have gone through Down so we will tery from L, R, U

always remember well continue from whe we left D possib

F (1, 1, DDRU)

non possible we get 1.

[whenever u back-track pls unmark the visited.]



* TC → O(4^{nm}) for every cell we are trying 4 different ways

SC (Auxilary space) no of fuction calls which is total elements in matrix

SC(nm)

aise case में हम string से remove जो ~~करते~~ है kyunki voh use value है pehli state में नहीं add करो s() को ds में add किया है then recursion.

Follow Ramanjot Singh for more such content