{Count Numbers with Unique Digits II.}

Well its a digit dp qustion, so we already know we have to play around with

pos, tigit & variable

variable → This depends on the problem statement.

in this qustion our variable is a mask

mask → $\begin{bmatrix} - & - & - & - & - & - & - & - & - & - \\ 9 & 8 & 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \end{bmatrix}$

whichever bit has already occured we set that bit as 1.

All bits set as 1 will be like first 10 bits set as 1 so that is $(2^{10}-1)$

what code I had intitially written.

assume I had done caching already

```
int func(int pos, int tight, int mask, String nums){
    if(pos == nums.length()){
        return 1;
    }
    limit = tight==1 ? nums[pos] : 9;
    ans = 0
    for(i = 0 -> limit){
        if((mask & (1<<i)) == 1)){
            continue;
        }
        newmask = mask | (1<<i);
        newtight = (tight==1 && i==limit) ? 1 : 0;
        ans += func(pos+1, newtight, newmask, nums);
    }
    return ans.
}
```

1st mistake, mask = 01000 & (1<<i) will give 01000 and not

digit is used

we are doing left shift of 1 and not right shift of mask

((mask>>i) & 1 == 1) would give 1.

Another mistake which I did,

In my code I have not taken into account the fact that leading zeros will be there, but they should not be counted as of mask should not mark "0" as used for that.

My code will do that,

(Has Number Started)

So we have to use another state called hasNumberStarted.

what does it mean?

It means if a non-zero number has already appeared before pos. ~~If yes then mask~~

if (No)

and the current number we are choosing is

nonzero → then mark numberhasStarted as true

else send the old mask

to next state.

Correct Code

```
f( pos, tight, mask, nums, hasNumberStarted ) {
    if( pos == nums.length() ) {
        only return 1
            if number has Started   → else all are leading zeros.
    }
    limit = (tight == 1) ? nums[pos] : 9;
    for( i = 0 → limit ){
        if( ( mask & (1<<i) ) != 0 ) { continue }   // correct condition → digit picked
        newMask = mask | (1<<i)
        newtight = (tight == 1 && i == limit) ? 1 : 0;
        if( hasNumberStarted == 0 ){
            i == 0 ? → ans += f(pos+1, newtight, 0, 0);   // [this one is also so number has not started]
            i ≠ 0 ? ans += f(pos+1, newtight, newmask, 1);   // this i≠0, Non-zero leading number so number has started
        }
        else {
            ans += f(pos+1, newtight, newmask, nums, hasNumberStarted)
        }   // already means number has started.
    }
    return ans;
}
```