Digit DP

Q1 How many numbers in [a, b] are there, where the digit d occurs exactly k times.

Eg- $a = 7$ $b = 58$

$d = 7$ $k = 1$

[7, 58] ans = 6

7 17 27 37 47 57

Brute force : iterate using for loop from a to b. Check if the number is valid

for (i=a ; i≤b ; i++){

}

Iterations of for loop = $b-a+1$

Now imagine $a = 100$
$b = 10^{16}$

This will always give TLE

- We need a faster way of solving

ans $[a,b]$ = $0,1,2,\dots,a-1,$ $a,a+1,\dots\dots,b-1,b$

$[0,b] - [0,a-1]$

$0,1,2,\dots,a-1,$ $a,$ $a+1,$ $a+b,\dots\dots,b$

$0,1,2,\dots,a-1,$ $[0,b] - [0,a-1]$

$0,1,2,$ $3,4,5,6,7,8$ $[0,8]$
$- [0,2]$

$0,1,2$

- $[a,b] \Rightarrow [0,b] - [0,a-1]$

[0, x]

5 2 3 4
5 _ _ _ _

new_d < cur_d less
new_d > cur_d bigger
new_d = cur_d cant say

Digit DP Build the number digit by digit satisfying all the conditions.

	0	1	2	3	4
Let x =	5	4	3	2	1

5 4 _ _ _

↑

Info needed ⇒ 1) cur_digit_index
2) cur num vs given no
3) How many times digit d has been placed till now.

```
int dp[pos][is_smaller or bigger]
        [count_of_d]    3D
                        array

int solve(int pos, bool is_small,
              int count_of_d) {
    if(count_of_d > k)
        return 0
    if(pos == num.size()) {
        if(count_of_d == k)
            return 1
        else
            return 0
    }

    if( dp[pos][is_smaller][count_of_d]
          already filled)
        return dp[pos][is_smaller][count_of_d]
```

```
int ans = 0
int max_digit
if ( is_smaller == true)
    max_digit = 9
else
    max_digit = num[pos]

for ( i=0 ; i≤ max_digit ; i++) {
    if( i == d)
        count_of_d++
    if (is_smaller == false &&
                i < num[pos] )
        is_smaller = true
    ans += solve (pos+1, is_smaller,
                          count_of_d)
}
dp[pos][is_smaller][count_of_d]
        = ans
return ans
}
```

```
main() {
    num = b                         [0,b]
    ans = solve(0,0,0)
        // solve for [0,b]

    num = a-1                       [0,a-1]
    ans -= solve(0,0,0)
        // solve for [0,a-1]
}
```

$[0,b] - [0,a-1]$

$\Rightarrow [a,b]$

TC: #states × TC of one state.

$\log_{10} n \times 2 \times k \times 10$

$\Rightarrow 20\, k \log n$

$\Rightarrow O(k \log n)$

Q2 Investigation.

How many no.s in range [a, b] are there, whose sum of digits is divisible by K.

Eg - [10, 28] K = 3

10 11 12 13 14 15 16 17 18 19

20 21 22 23 24 25 26 27 28

ans = 6

int dp[pos][is_smaller or bigger][sum_of_digits % k]

Main logic

```
for ( i=0 ; i≤ max_digit ; i++ ){
    sum = (sum+i) % K
    if ( is_smaller == false  &&
                i < num[pos] )
          is_smaller = true
    ans += solve( pos+1, is_smaller,
                          sum )
}
```

TC: $\log_{10} n \times 2 \times k \times 10$

$20\, k \log n \Rightarrow O(k \log n)$

Q3 Hard Investigation.

How many no.s in range [a, b] are there, whose sum of digits is divisible by K and the number itself is also div by K.

Eg - [10, 28] K = 2

10 11 12 13 14 15 16 17 18 19

20 21 22 23 24 25 26 27 28

ans = 5

ans = 5

int dp[pos][is-smaller][sum of dig % K][num % K]

0

5 7 4 _ _

$0 \times 10 + 5 = 5$

$5 \times 10 + 7 = 57$

$57 \times 10 + 4 = 574$

Main logic

```
for ( i=0 ; i≤ max_digit ; i++ ){

    dig_sum = (dig_sum + i ) % K
    new_num = (10*num + i ) % K

    if ( is_smaller == false  &&
                 i < num[pos] )
          is_smaller = true
    ans += solve (pos+1, is_smaller,
                 dig_sum, new_num)
}
```

TC: $\log_{10} n \times 2 \times k \times k \times 10$

$O(k^2 \log n)$

57321

$573218 = 10 * 57321 + 8$

635

$6352 = 10 * 635 + 2$

- $new\_num = 10 * old\_num + dig$

<u>TC</u>

How many digits does n have

$\Rightarrow \log_{10} n$

Iterations $\propto \log_{10} n$

Digit DP $\quad n \rightarrow \log_{10} n$

{done}

1 2 3 4 5 6

Q sum of digits at even loc
− sum of digits at odd loc
= 1

for (i=0 ; i ≤ max_digit ; i++) {

dig_sum_even } acc to pos
dig_sum_odd

if (is_smaller == false &&
i < num [pos])
is_smaller = true
ans += solve (pos+1 , is_smaller ,
dge , dgo)
}

{done}