GEMP UFC | CAMPUS QUIXADÁ

Sejam Bem-vindos ao:

Blog da Omma

Introdução a Digit DP

Existem vários tipos de problemas que pedem para contar a quantidade de números inteiros **x** entre dois inteiros '**a**' e '**b**' de tal forma que **x** satisfaz uma propriedade específica relacionada aos dígitos. Portanto, se tivermos uma função **G(x)** a qual nos informa a quantidade de números entre **1** e **x** (incluindo o próprio **x**) que respeitam uma determinada propriedade, então os números inteiros que respeitam essa propriedade entre dois inteiros '**a**' e '**b**' é igual a **G(b) - G(a-1).**

Vamos começar com o básico.

A Digit DP mais básica de todas.

Dado dois inteiros 'a' e 'b', calcule quantos números inteiros existem neste intervalo.

Caso de teste

10 110

Saída
100

Tricks

Preciso calcular a dp várias vezes e agora?? socorro meu deus nao deu certo

Muita calma nessa hora, vamos pensar diferente (:

Que tal fazermos do último dígito até o primeiro?

A Digit DP mais básica de todas.

Dado dois inteiros 'a' e 'b', calcule quantos números inteiros existem neste intervalo.

```
string number;
int solve(int i, int can){
	if (i == number.size()) return 1;
	if (tempo[i][can] != -1) return dp[i][can];
	int ans = 0;
	for(int digit = can ? 9 : number[i] - '0'; digit >= 0; digit- - ){
		ans += solve(i+1, can | (number[i] - '0' > digit));
	}
	return dp[i][can] = ans;
}
```

Tricks

Quantos números em um intervalo tem a soma dos dígitos divisível por 3,4,5,6,7 e 8.

Qual é o k-esimo numero que respeita uma determinada propriedade??
https://codeforces.com/contest/919/problem/B

Preciso calcular a dp várias vezes e agora??

Tricks

Reduzir pela metade a memória necessária.

```
int solve(int i, int can){
	if (i == number.size()) return 1;
	if (dp[i] != -1) return dp[i];
	int ans = 0;
	for(int digit = can ? 9 : number[i] - '0'; digit >= 0; digit- - ){
		ans += solve(i+1, can | (number[i] - '0' > digit), mask);
	}
	return can ? dp[i] = ans : ans;
}
```

Dado um L e R, conte quantos números tem a frequencia par de todos os seus digitos. 22 1122 1212 2211 e assim por diante.

Outra forma de ver uma digit dp

A palindromic number or numeral palindrome is a **'symmetrical'** number like 16461 that remains the same when its digits are reversed. In this problem you will be given two integers **i and j**, you have to find the number of palindromic numbers between **i** and **j** (inclusive)

E se por acaso eu quiser fazer o intervalo de uma vez só?

```
int solve(int i, int can_lower, int can_higher){
        if (i == number.size()) return 1;
        if (dp[i][can_lower][can_higher] != -1) return dp[i][can_lower][can_higher];
        int ans = 0;
        for(int digit = can_higher ? 9 : bigger[i] - '0'; digit >= can_lower ? 0 : smaller[i] - '0'; digit- - ){
                ans += solve(i+1, can_lower | (digit > smaller[i] - '0'), can_higher | (bigger[i] - '0' > digit));
        }
        return dp[i][can_lower][can_higher] = ans;
}
```

Problem List

1. Investigation
2. LIDS
3. Magic Numbers
4. Palindromic Numbers
5. Chef and Digits
6. Maximum Product
7. Cantor
8. Digit Count
9. Logan and DIGIT IMMUNE numbers
10. Sanvi and Magical Numbers
11. Sum of Digits
12. Digit Sum
13. Ra-One Numbers
14. LUCIFER Number
15. 369 Numbers
16. Chef and special numbers
17. Perfect Number
18. The Great Ninja War