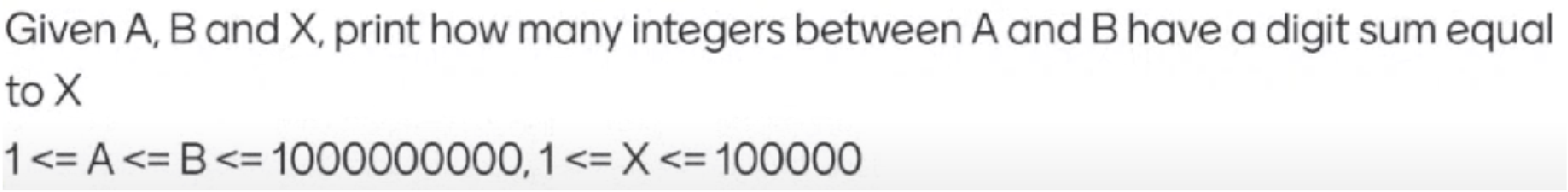
Digit DP



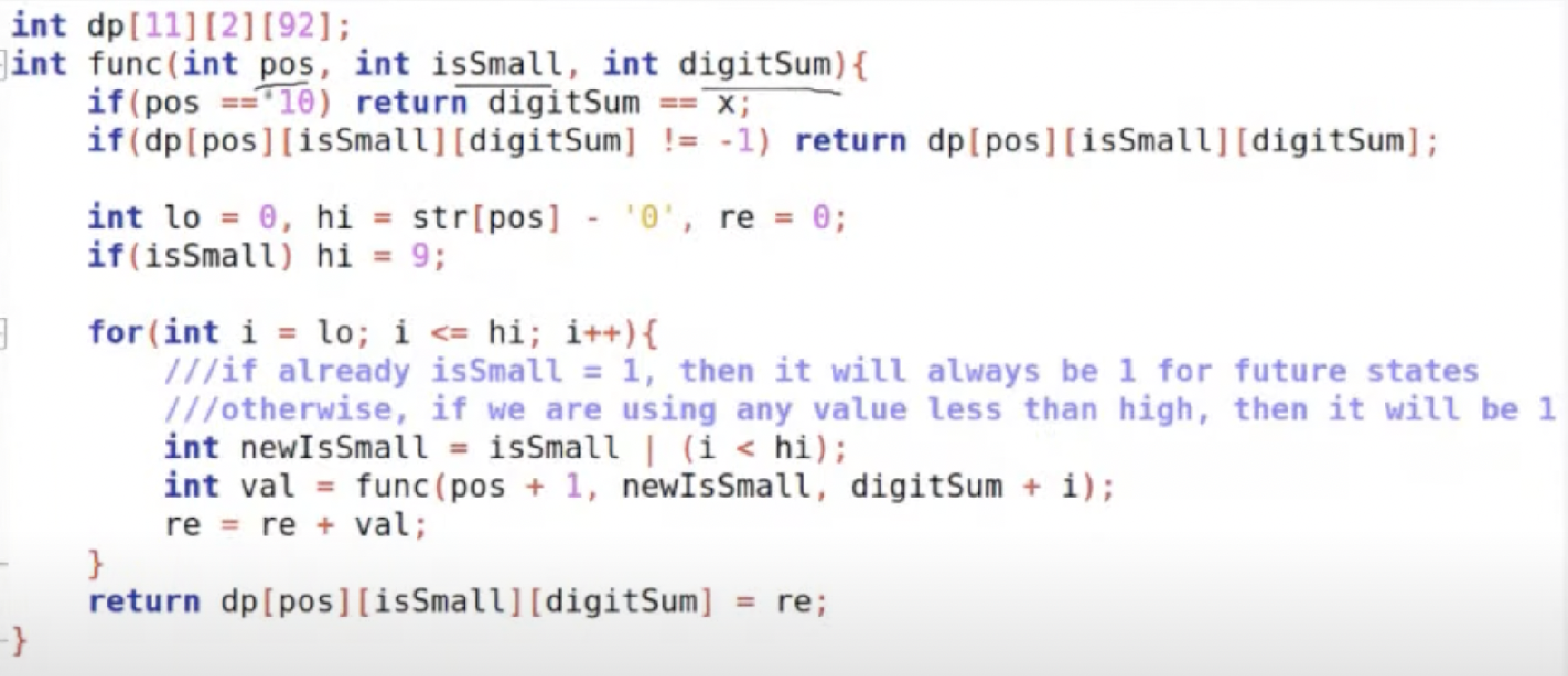
Given A, B and X, print how many integers between A and B have a digit sum equal to X

1 <= A <= B <= 1000000000, 1 <= X <= 100000

$$\mathrm{Solve}(L,R) = \mathrm{Solve}(0,R) - \mathrm{Solve}(0,L-1)$$

Observations

dp [position] [is_small] [digit_sum]



```cpp
int dp[11][2][92];
int func(int pos, int isSmall, int digitSum){
    if(pos == 10) return digitSum == x;
    if(dp[pos][isSmall][digitSum] != -1) return dp[pos][isSmall][digitSum];

    int lo = 0, hi = str[pos] - '0', re = 0;
    if(isSmall) hi = 9;

    for(int i = lo; i <= hi; i++){
        ///if already isSmall = 1, then it will always be 1 for future states
        ///otherwise, if we are using any value less than high, then it will be 1
        int newIsSmall = isSmall | (i < hi);
        int val = func(pos + 1, newIsSmall, digitSum + i);
        re = re + val;
    }
    return dp[pos][isSmall][digitSum] = re;
}
```