// CPP program to count the number

// of numbers formed by digits a

// and b exactly of a length N such

// that the sum of the digits of the

// number thus formed is of digits a and b.

#include <bits/stdc++.h>

using namespace std;

const int mod = 1e9 + 7;

const int N = 1000005;

int fact[N], invfact[N];

// function to check if sum of

// digits is made of a and b

int check(int x, int a, int b)

{

    // sum of digits is 0

    if (x == 0)

        return 0;

    while (x) {

        // if any of digits in sum is

        // other than a and b

        if (x % 10 != a and x % 10 != b)

            return 0;

        x /= 10;

    }

    return 1;

}

// calculate the modInverse V / of a number in O(log n)

int modInverse(int a, int m)

{

    int m0 = m;

    int y = 0, x = 1;

    if (m == 1)

        return 0;

    while (a > 1) {

        // q is quotient

        int q = a / m;

        int t = m;

        // m is remainder now, process

        // same as Euclid's algo

        m = a % m, a = t;

        t = y;

        // Update y and x

        y = x - q \* y;

        x = t;

    }

    // Make x positive

    if (x < 0)

        x += m0;

    return x;

}

// function to pregenerate factorials

void pregenFact()

{

    fact[0] = fact[1] = 1;

    for (int i = 1; i <= 1000000; ++i)

        fact[i] = (long long)fact[i - 1] \* i % mod;

}

// function to pre calculate the

// modInverse of factorials

void pregenInverse()

{

    invfact[0] = invfact[1] = 1;

    // calculates the modInverse of the last factorial

    invfact[1000000] = modInverse(fact[1000000], mod);

    // precalculates the modInverse of all factorials

    // by formulae

    for (int i = 999999; i > 1; --i)

        invfact[i] = ((long long)invfact[i + 1] \*

                      (long long)(i + 1)) % mod;

}

int comb(int big, int smal)

{

    return (long long)fact[big] \* invfact[smal] % mod \*

                              invfact[big - smal] % mod;

}

// function that returns the count of numbers

int count(int a, int b, int n)

{

    // function call to pre-calculate the

    // factorials and modInverse of factorials

    pregenFact();

    pregenInverse();

    // if a and b are same

    if (a == b)

        return (check(a \* n, a, b));

    int ans = 0;

    for (int i = 0; i <= n; ++i)

        if (check(i \* a + (n - i) \* b, a, b))

            ans = (ans + comb(n, i)) % mod;

    return ans;

}

int main()

{

    int a = 3, b = 4, n = 11028;

    cout << count(a, b, n);

    return 0;

}