**Sum of all substrings of a number**

Given an integer **s** represented as a string, the task is to get the sum of all possible sub-strings of this string.  
As the answer will be large, return answer **modulo 109+7**.

**Note:** The number may have leading zeros.

**Example 1:**

**Input:**

s = "1234"

**Output:**1670

**Explanation:**Sum = 1 + 2 + 3 + 4 + 12 + 23 + 34 + 123 + 234 + 1234 = 1670

**Example 2:**

**Input:**

s = "421"

**Output:**491

**Explanation:**Sum = 4 + 2 + 1 + 42 + 21 + 421 = 491

**Your Task:**  
You only need to complete the function **sumSubstrings** that takes **s** as an argument and returns the answer **modulo 109+7**.

**Expected Time Complexity:**O(|s|).  
**Expected Auxiliary Space:**O(|s|).

**Constraints:**  
1 <= |s| <= 105

class Solution

{

public:

int mod = 1e9+7;

long long sumSubstrings(string s)

{

int n= s.size();

vector<long long> dp(n,0);

dp[0] = s[0]-'0';

long long ans = dp[0];

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

{

dp[i] = ((dp[i-1]\*10)%mod + ((s[i]-'0')\*(i+1))%mod)%mod;

ans = (ans + dp[i])%mod;

}

return ans;

}

};

Link : <https://www.geeksforgeeks.org/problems/sum-of-all-substrings-of-a-number-1587115621/1>