1. 题目
2. 题干

一条包含字母 A-Z 的消息通过以下方式进行了编码：

'A' -> 1

'B' -> 2

...

'Z' -> 26

给定一个只包含数字的非空字符串，请计算解码方法的总数。

题目数据保证答案肯定是一个 32 位的整数。

1. 示例

示例 1：

输入："12"

输出：2

解释：它可以解码为 "AB"（1 2）或者 "L"（12）。

示例 2：

输入："226"

输出：3

解释：它可以解码为 "BZ" (2 26), "VF" (22 6), 或者 "BBF" (2 2 6) 。

示例 3：

输入：s = "0"

输出：0

示例 4：

输入：s = "1"

输出：1

示例 5：

输入：s = "2"

输出：1

1. 题解
2. 思路

典型的动态规划问题，但是做起来会非常地麻烦。要想知道当前的字符串的解码方法数，就需要依赖于前一个字符串的解码方法数。当前字符串的最后一个数字如果是0的话，那么前一个数字只能是1或者2时才能解码，否则无法解码，返回0。若当前字符串的最后一个数字的前一个数字为1时，可以和最后一个数字结合，若前一个数字为2时，只能和1-6这6个数字结合。若前一个数字既不是1或者2，那么解码方法不会增加。

1. 代码实现

Java：

class Solution {

    public int numDecodings(String s) {

        int n = s.length();

        int[] dp = new int[n+1];

        if(s.charAt(0) == '0'){

            return 0;

        }else if(n == 1){

            return 1;

        }

        dp[0] = 1;

        dp[1] =1;

        for(int i=1; i < n; ++i){

            if(s.charAt(i) == '0'){

                if(s.charAt(i-1) == '1' || s.charAt(i-1) == '2'){

                    dp[i+1] = dp[i-1];

                }else{

                    return 0;

                }

            }else{

                if(s.charAt(i-1) == '1' || s.charAt(i-1) == '2' && s.charAt(i) >= '1' && s.charAt(i) <= '6'){

                    dp[i+1] = dp[i-1] + dp[i];

                }else{

                    dp[i+1] = dp[i];

                }

            }

        }

        return dp[n];

    }

}