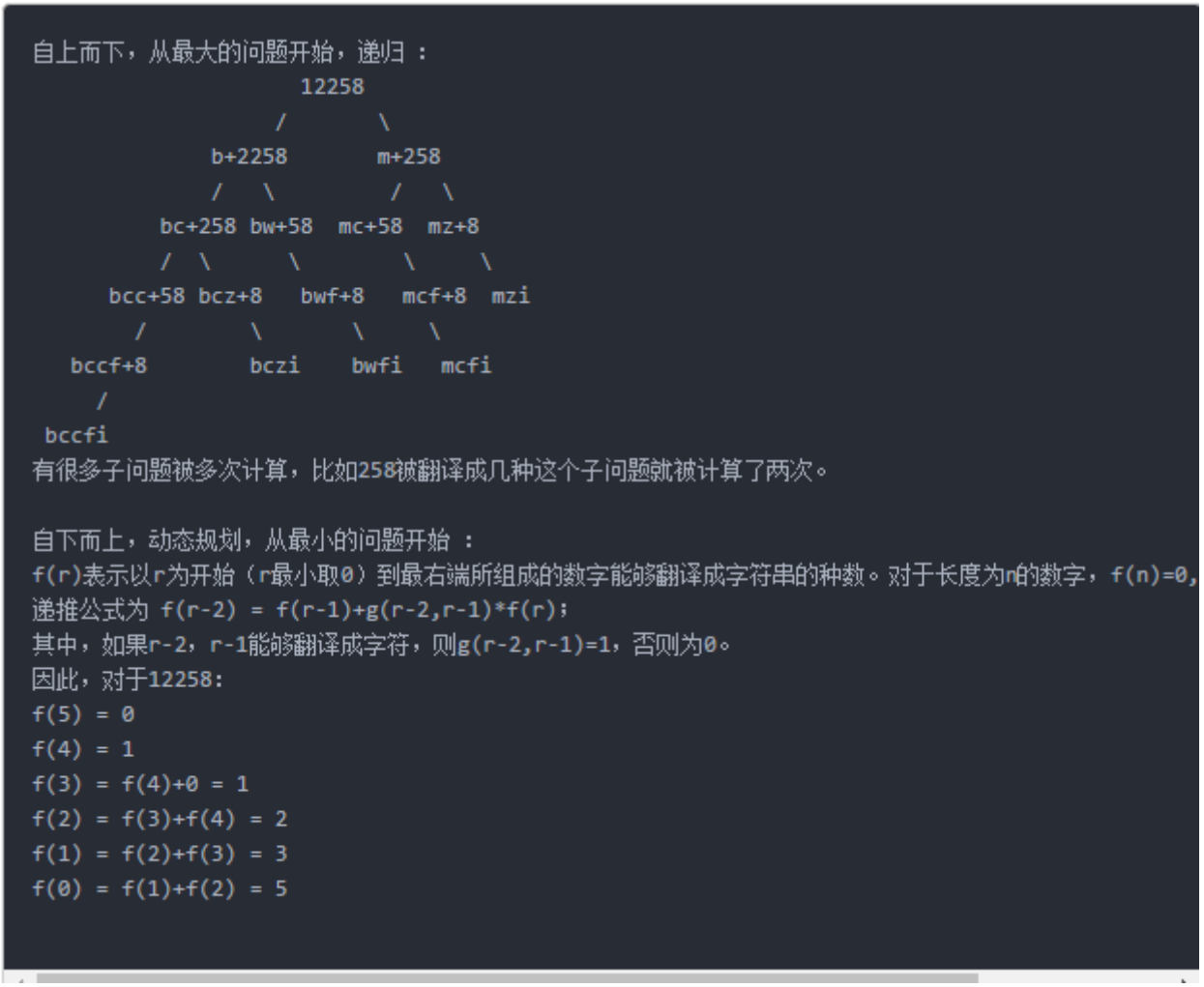
给定一个数字，按照如下规则翻译成字符串：0翻译成“a”，1翻译成“b”...25翻译成“z”，一个数字有多种翻译可能，例如12258一共有5种，分别是bccfi，bwfi，bczi，mcfi，mzi。实现一个函数，用来计算一个数字有多少种不同的翻译。

自上而下，从最大的问题开始，递归 ：



有很多子问题被多次计算，比如258被翻译成几种这个子问题就被计算了两次。

自下而上，动态规划，从最小的问题开始 ：

f(r)表示以r为开始（r最小取0）到最右端所组成的数字能够翻译成字符串的种数。对于长度为n的数字，f(n)=0,

递推公式为 f(r-2) = f(r-1)+g(r-2,r-1)*f(r);

其中，如果r-2，r-1能够翻译成字符，则g(r-2,r-1)=1，否则为0。

因此，对于12258:

f(5) = 0

f(4) = 1

f(3) = f(4)+0 = 1

f(2) = f(3)+f(4) = 2

f(1) = f(2)+f(3) = 3

f(0) = f(1)+f(2) = 5

```java
public class B46 {

    public static int getTranslationCount(int number) {
        if(number < 0)
            return 0;
        if(number == 1)
            return 1;

        return Dy(Integer.toString(number));
    }

    public static int Dy(String number) {
        int f1 = 0, f2 = 1, g = 0;
        int temp;
        for(int i = number.length() - 2; i >= 0; i --) {
            if(Integer.parseInt(number.charAt(i) + "" + number.charAt(i + 1)) < 26)
                g = 1;
            else {
                g = 0;
            }
            temp = f2;
            f2 = f2 + g * f1;
            f1 = temp;
        }
        return f2;
    }

    public static void main(String [] args) {
        System.out.println(getTranslationCount(1234));
    }
}
```