

一.问题描述及规格说明

给定整数 n ，返回具有唯一数字的所有数的计数,其中
($0 \leq n < 10, 0 \leq x < 10^n$)

二.算法设计

- 注意到 $0 \leq x < 10^n$,可以把所有范围内的数字分成 n 个阶段，分别是
 $0 \leq x < 10^0; 10^0 \leq x < 10^1; 10^1 \leq x < 10^2 \dots 10^{n-1} \leq x < 10^n$ (即 $0, [1, 9], [10, 99], [100, 999] \dots [10^{n-1}, 10^n - 1]$)。
- 对于数字长度为 n 的部分来说，首位数字只能有九种选择，即1—9，剩下的 $n - 1$ 位可以任意选择，即0—10(但要排除刚才选的第一个数字)，
总共的选择数

$$9 * \overbrace{9 * 8 * 7 * 6 \dots}^{n-1 \uparrow} = 9 * A(9, n - 1) \text{ (其中 } A(m, n) \text{ 为排列数)}$$

- 对于所有部分来说，全部选择数
 $= 1 + 9 * (A(9, 0) + A(9, 1) + \dots + A(9, n - 1)) = 1 + \sum_{k=1}^n A(9, k - 1)$
。所以该问题的通项公式 $a_n = 1 + \sum_{k=1}^n A(9, k - 1)$ 。
- 因此我们有 $a_{n-1} = 1 + \sum_{k=1}^{n-1} A(9, k - 1) \therefore a_n = A(9, n - 1) + a_{n-1}$
。对于原问题来说， $0 \leq x < 10^{n-1}$ 的 a_{n-1} 就是子问题，它包括了从0到数字长为 $n-1$ 的问题求解总数，而 $9 * A(9, n)$ 是数字长为 n 的求解总数，所以两者加起来就可以构成原问题。

三.程序结构设计

既然满足最优子结构的问题，就可以用动态规划来求解。创建一个长度为 $n + 1$ 的一维数组 $dp[n + 1]$ 。

动态转移方程为 $dp[i] = dp[i - 1] + 9A(9, i - 1)$,其中 $dp[0] = 1$ 。

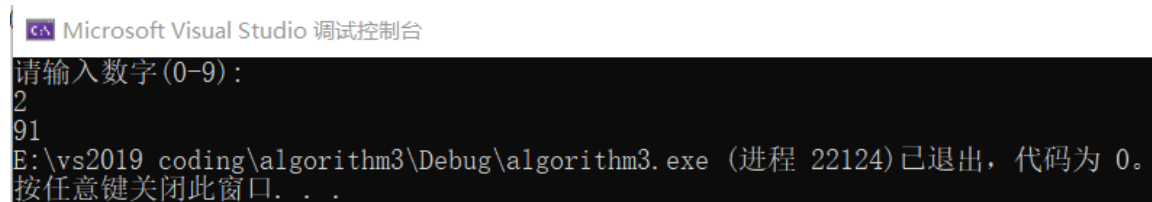
排列数 $A(m, n)$ 可以由如下的函数计算而来:

```
int Permutations(const int m, const int n)//计算排列数
{
    int ans = 1;
    for (int k = m; k > m - n; k--)
    {
        ans *= k;
    }
    return ans;
}
```

根据问题的描述，我们要求解 $dp[n]$:

```
int solutions(const int n)
{
    int* dp = new int[n + 1];
    dp[0] = 1;
    for (int i = 1; i < n + 1; i++)
    {
        dp[i] = dp[i - 1] + 9 * Permutations(9, i - 1);
    }
    return dp[n];
}
```

四.程序运行截图



Microsoft Visual Studio 调试控制台

请输入数字(0-9):
2
91
E:\vs2019 coding\algorithm3\Debug\algorithm3.exe (进程 22124)已退出, 代码为 0。
按任意键关闭此窗口. . .

请输入数字(0-9):

0

1

E:\vs2019 coding\algorithm3\Debug\algorithm3.exe (进程 26808)已退出, 代码为 0。

按任意键关闭此窗口. . .