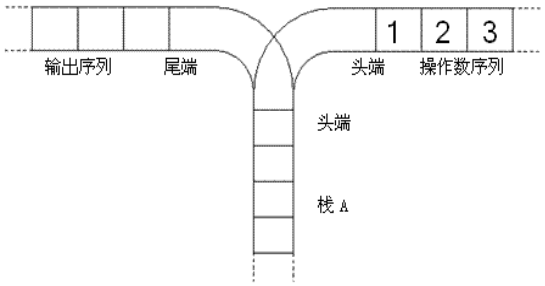
P1044 [NOIP2003 普及组] 栈

题目描述



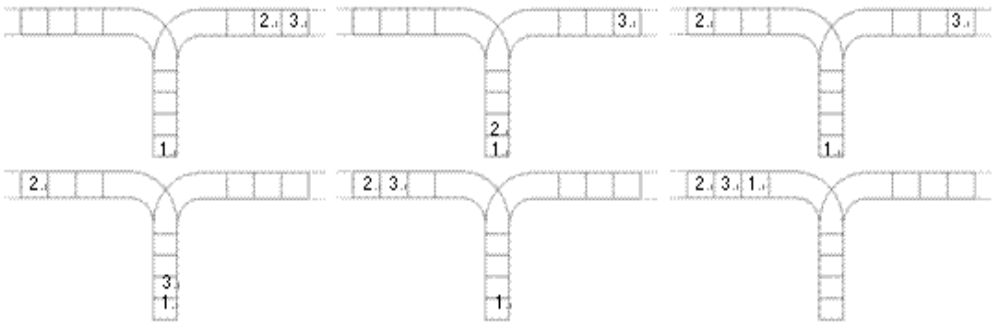
宁宁考虑的是这样一个问题：一个操作数序列，1,2,…,n（图示为 1 到 3 的情况），栈 A 的深度大于 n。

现在可以进行两种操作，

1.将一个数，从操作数序列的头端移到栈的头端（对应数据结构栈的 push 操作）

2.将一个数，从栈的头端移到输出序列的尾端（对应数据结构栈的 pop 操作）

使用这两种操作，由一个操作数序列就可以得到一系列的输出序列，下图所示为由 1 2 3 生成序列 2 3 1 的过程。



（原始状态如上图所示）

你的程序将对给定的 n，计算并输出由操作数序列 1,2,…,n 经过操作可能得到的输出序列的总数。

输入格式

输入文件只含一个整数 n（1≤n≤18）。

输出格式

输入文件只含一个整数 n（1≤n≤18）。

输入样例

3

输出样例

5

解析

先来看一个最简单的例子：aka，很明显，对于当前位置的k来说(即保持k的位置不变)，它只有一种排列方式。

接下来观察abkab，此刻对于k来说，就存在了4种排列方式，分别是：

bakab,bakba,abkab,abkba，再进一步观察，我们可以看出ab有两种排列方式，也就是说k的排列方式总数 = k前数字的排列数 * k后的数字的排列数。为什么不是加，带回第一个式子，很容易发现加法是错误的，乘法是正确的。

现在，我们稍微移动一下k的位置，使其变成kab，很明显一共有两种排列方式，即：kab和kba。

我们继续移动，使其变成13k，同样也是有两种方式：13k和31k。

因此，对于aka这个组合，我们一共有k**（2种）+ *k*(1种) + **k（2种），共计五种排列方法。我们再进一步，将k换成数字：1**（2种）+ *2*(1种) + **3（2种）,这样就得出了当n=3时的排列总数。

最终结论如下：

设 x 为当前出栈序列的最后一个，则x有n种取值,因为每一个数都可能最后出栈。

由于x是最后一个出栈的，所以可以将已经出栈的数分成两部分：比x小，比x大。

比x小的数有x-1个，所以这些数的全部出栈可能为f[x-1]

比x大的数有n-x个，所以这些数的全部出栈可能为f[n-x]

所以一个x的取值能够得到的所有可能性为f[x-1] * f[n-x]

另外，由于x有n个取值，所以：

ans = f[0]*f[n-1] + f[1]*f[n-2] + ... + f[n-1]*f[0];

这就是所谓的卡特兰数。

现在，我们来带入一组数字进行验证：1,2,3,4

首先，出栈的数字为1,1前有0个数字，我们规定f[0]=1，1后有3个数字，则对于1来说总的排列数量为f[0]*f[3]。

接下来计算出栈数字2，总排列可能性为f[1]*f[2]，因为2之前有1个数字，2之后有两个数字

继续计算3：f[2]*f[1]。

计算4：f[3]*f[0]

最终得到f[4] = f[0]*f[3] + f[1]*f[2] + f[2]*f[1] + f[3]*f[0];

f[3]如何计算？根据前文的推导，我们知道f[3] = f[0]*f[2] + f[1]*f[1] + f[2]* f[0]

我们又知道f[1]代表的是只有1个数字，那么结果必然为1，即f[1]=1。

f[2]带入公式则有：f[2]=f[0]*f[1]+f[1]*f[0] = 2。将f[2]=2带入f[3]，则有f[3]=5，事实上我们也知道f[3]就是5。

再将计算结果带入，得到f[4] = 1*5+1*2+2*1+5*1=14。

编码

```
#include<bits/stdc++.h>

#define MAX_N 20
#define ll long long
using namespace std;
int n;
ll f[MAX_N];

int main() {
    f[0] = f[1] = 1;
    scanf("%d", &n);
    //从第2项开始递推。
   for (int i = 2; i <= n; i++) {
        //cout  << "f[" << i << "]" << "=";
        for (int j = 0; j < i; j++) {
            f[i] += f[j] * f[i - j - 1];
            //cout << "(f[" << j << "]*f["
            // << i << "-" << j << "-1]) ";
        }
        //cout << endl;
    }
    printf("%lld", f[n]);
    return 0;
}
```

逻辑航线培优教育，信息学奥赛培训专家。

扫码添加作者获取更多内容。