**算法训练 摆动序列**

时间限制：1.0s   内存限制：512.0MB

问题描述

　　如果一个序列满足下面的性质，我们就将它称为摆动序列：  
　　1. 序列中的所有数都是不大于*k*的正整数；  
　　2. 序列中至少有两个数。  
　　3. 序列中的数两两不相等；  
　　4. 如果第*i* – 1个数比第*i* – 2个数大，则第*i*个数比第*i* – 2个数小；如果第*i* – 1个数比第*i* – 2个数小，则第*i*个数比第*i* – 2个数大。  
　　比如，当*k* = 3时，有下面几个这样的序列：  
　　1 2  
　　1 3  
　　2 1  
　　2 1 3  
　　2 3  
　　2 3 1  
　　3 1  
　　3 2  
　　一共有8种，给定*k*，请求出满足上面要求的序列的个数。

输入格式

　　输入包含了一个整数*k*。（*k*<=20）

输出格式

　　输出一个整数，表示满足要求的序列个数。

样例输入

3

样例输出

8

锦囊1

状态压缩的动态规划。

锦囊2

用F[S, i]表示以i结尾的序列，序列中已包含S中的元素的方案数，则F[S, i]=\sum F[S/{i},j]，其中j是S/{i}中的元素。 最终求所有的F[T, i]的和，其中T中包含至少两个元素。

本题的C++参考代码如下：

#include<stdio.h>

int main()

{

int n,i,a[30],sum=0;

scanf("%d",&n);

a[2]=2;

sum+=a[2];

for(i=3;i<=n;i++)

{

a[i]=sum+i\*(i-1);

sum+=a[i];

}

printf("%d",a[n]);

return 0;

}

本题的C参考代码如下：

//天农

#include<stdio.h>

int f[21][21][21]; //f[i][j][k] i表示数的长度，j表示倒数第2位的取值，k表示最后一位的取值

int main()

{

int n,i,j,k,p;

int sum=0;

scanf("%d",&n);

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

for(j=1;j<=n;j++)

if(i!=j)

f[2][i][j]=1;

for(i=3;i<=n;i++)//从长度为3开始

{

for(j=1;j<=n;j++)

{

for(k=1;k<=n;k++)

{

for(p=1;p<=n;p++)

{

if(j>p&&k<p||j<p&&k>p)

f[i][j][k]+=f[i-1][p][j];

}

}

}

}

for(i=2;i<=n;i++)

for(j=1;j<=n;j++)

for(k=1;k<=n;k++)

sum+=f[i][j][k];

printf("%d",sum);

return 0;

}

本题的Java参考代码如下：

import java.io.\*;

public class Main{

public static void main(String args[])throws IOException{

BufferedReader bf=new BufferedReader(new InputStreamReader(System.in));

int n=Integer.parseInt(bf.readLine());

System.out.println((int)(Math.pow(2,n)-n-1)\*2);

}

}