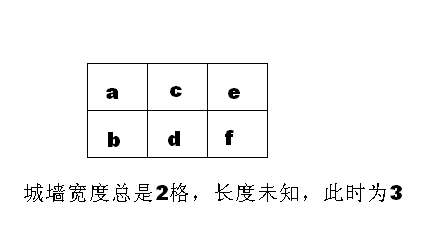
**格子刷油漆**

问题描述

　　X国的一段古城墙的顶端可以看成 2\*N个格子组成的矩形（如下图所示），现需要把这些格子刷上保护漆。  
  
  
　　你可以从任意一个格子刷起，刷完一格，可以移动到和它相邻的格子（对角相邻也算数），但不能移动到较远的格子（因为油漆未干不能踩！）  
　　比如：a d b c e f 就是合格的刷漆顺序。  
　　c e f d a b 是另一种合适的方案。  
　　当已知 N 时，求总的方案数。当N较大时，结果会迅速增大，请把结果对 1000000007 (十亿零七) 取模。

输入格式

　　输入数据为一个正整数（不大于1000）

输出格式

　　输出数据为一个正整数。

样例输入

2

样例输出

24

样例输入

3

样例输出

96

样例输入

22

样例输出

359635897

锦囊1

动态规划。

本题的C++参考代码如下：

#include<cstdio>

#define MAXN 1010

#define MOD 1000000007

int main()

{

long long DP[5][MAXN],ans=2;

int n;

scanf("%d",&n);

if(n>1){

DP[0][2]=DP[2][2]=DP[3][2]=DP[4][2]=4;

DP[1][2]=8;

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

DP[0][i]=(DP[0][i-1]\*2+DP[1][i-1]\*2)%MOD;

DP[1][i]=(DP[1][i-1]\*2+DP[2][i-1]\*4+DP[3][i-1]\*2+DP[4][i-1]\*2)%MOD;

DP[2][i]=DP[2][i-1]\*2%MOD;

DP[3][i]=DP[1][i-1];

DP[4][i]=DP[1][i-1];

}

ans=0;

for(int i=0;i<5;i++)ans+=DP[i][n];

ans%=MOD;

}

printf("%I64d\n",ans);

return 0;

}

本题的C参考代码如下：

#include <stdio.h>

long long a[1001],b[1001],sum;

#define NUM 1000000007

int main()

{

int i,n;

scanf("%d",&n);

b[1]=1;

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

b[i]=(b[i-1]\*2%NUM);

a[1]=1;a[2]=6;

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

a[i]=(2\*a[i-1]+b[i]+4\*a[i-2])%NUM;

sum=4\*a[n];

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

sum=((sum+8\*b[n-i]\*a[i-1]%NUM)%NUM+(8\*a[n-i]\*b[i-1])%NUM)%NUM;

printf("%I64d\n",sum);

return 0;

}

本题的Java参考代码如下：

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

public class Main {

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

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

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

long sum=0,mod=1000000007;

long[] A=new long[n+1];

long[] B=new long[n+1];

A[1]=1; A[2]=2;

B[1]=1; B[2]=6;

int i,j;

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

{

A[i]=2\*A[i-1]%mod;

B[i]=(2\*B[i-1]%mod+2\*A[i-2]%mod+B[i-2]\*2%mod+2\*A[i-2]%mod+2\*B[i-2]%mod)%mod;

}

sum=4\*B[n]%mod;

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

{

sum+=(2\*(A[j-1]\*2\*B[n-j]\*2%mod+2\*A[n-j]\*2\*B[j-1]%mod));

sum%=mod;

}

if(n>1) System.out.println(sum);

else System.out.println(2);

}

}