还记得汉诺塔III吗？他的规则是这样的：不允许直接从最左(右)边移到最右(左)边(每次移动一定是移到中间杆或从中间移出)，也不允许大盘放到小盘的上面。xhd在想如果我们允许最大的盘子放到最上面会怎么样呢？（只允许最大的放在最上面）当然最后需要的结果是盘子从小到大排在最右边。

**Input**

输入数据的第一行是一个数据T，表示有T组数据。   
每组数据有一个正整数n(1 <= n <= 20)，表示有n个盘子。

**Output**

对于每组输入数据，最少需要的摆放次数。

**Sample Input**

2

1

10

**Sample Output**

2

19684

**如果有n个盘子，则需要前n-1个挪到中间的盘子上，然后最大的盘子挪到最右面，需要两步，把前（n-1）个盘子从左边挪到中间是和从中间挪到右边需要相同的次数。而a数组中存放的就是那个前n-1个盘子挪动到相同位置需要的次数。结果即为a[i-1]\*2+2。**

**而求a数组需要用到递推。公式为第i个为前n-1个移动次数的三倍加一，简化到两个盘子，小的先移动两次到最右边，大的移到中间，然后小的在移回中间，小的移动了三次，而大的移动了一次，就使他们全部挪动了一个位置**

推得公式和输出结果时所用的公式不一样了

#include <stdio.h>

#include <math.h>

#include<iostream>

using namespace std;

int main()

{

//freopen("input.txt","r",stdin);

\_\_int64 f[25]={0,1};

int n,x;

for(int i=2;i<=21;i++)

f[i]=f[i-1]\*3+1;

cin>>n;

while(n--)

{

cin>>x;

cout<<f[x-1]\*2+2<<endl;

}

return 0;

}