 分析题意，我们知道这是一道排列计数问题。而且，题意的要求是对于给定字符串长度n，给出对应的方案数m。我很容易联想到“f(n) = m”这样的函数关系。并且，题目中的限制条件只有“两个O不能相邻”。计数 + 简单限制 = 递推。接下来的问题就是求出递推公式了。

\* 第n格取“O”:

----------------------------------  
|   |   |   | …… |     |     |  O  |  
----------------------------------  
 1   2   3          n-2 n-1   n

    -----------------------------------  
    |   |   |   | …… |     |  E  |  O  |  
    -----------------------------------  
      1   2   3         n-2  n-1   n

    -----------------------------------  
    |   |   |   | …… |     |  F  |  O  |  
    -----------------------------------  
      1   2   3         n-2  n-1   n

      对于第n格取“O”的情况，为了保证两个“O”不相邻，n-1格有两种可能，即“E”、“F”。对于余下的n-2格，由于第n-1格不取“O”，所以第n-2格不受n-1格的限制。其排列数等于f(n-2)。

\* 第n格不取“O”:  
----------------------------------  
|   |   |   | …… |     |     |  E  |  
----------------------------------  
  1   2   3         n-2  n-1  n

----------------------------------  
|   |   |   | …… |     |     |  F  |  
----------------------------------  
  1   2   3         n-2  n-1  n

      对于第n格不取“O”的情况，即取“E”、“F”。对于余下的n-1格，由于第n格不取“O”，所以，第n-1格不受n格的限制。其排列数等于f(n-1)。

      综上，f(n) = 2\*f(n-2) + 2\*f(n-1)  
           = 2\*(f(n-2) + f(n-1))

      这里，再说明一下“第n-1格不受n格的限制”这样一个条件。例如，n=4。如果，第4格取“O”，那么剩下的3格的方案数是多少呢？？肯定不是f(3)。因为，当n=3时，即只有3格的时候，第3格是可以取“O”的。而例子中的3格中，第3格很明显不能取“O”。所以，剩下的3格方案数不是f(3)。如果，第4格取“E”或者“F”，那么剩下的3格的方案数又是多少呢？？肯定是f(3)。这就是，是否受限制的差别。这是在递归中很重要的一个概念——什么是子结构。大家在日常的训练中要多加注意，不能盲目的识别子结构。

#include<iostream>

#include<cstdio>

#include<algorithm>

#include<cstring>

#include<cstring>

#include<cstdlib>

#define ll long long

using namespace std;

ll f[50];

int main()

{

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

f[1]=3;f[2]=8;

for(int i=3;i<=40;i++)

f[i]=2\*(f[i-1]+f[i-2]);

int n;

while(cin>>n)

cout<<f[n]<<"\n";

return 0;

}