**★★★★☆**

**題組：Problem Set Archive** **with Online Judge**

**題號：10254: The Priest Mathematician**

**解題者：周承業**

**解題日期：2013年5月29日**

**題意：**

原本的Hanoi tower問題，3根柱子增加到4根 ,用以下策略，搬動盤子

問N（0<=N<=10000）個盤子，需要移動的次數是多少？

搬動盤子策略

1. 把最上面的 n-r 個盤子搬到備用的柱子。
2. 然後用原本的三個柱子的策略去把剩下的 r的盤子搬到他們的目標。
3. 最後用四個柱子把最上面的 n-r個盤子搬到目標。

**題意範例：**

輸入

1

2

28

64

輸出

1

3

769

18433

**解法：**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| N | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| F(n) | 1 | 3 | 5 | 9 | 13 | 17 | 25 | 33 | 41 | 49 | 65 |
| R(n) | 1 | 2 | 2 | 4 | 4 | 4 | 8 | 8 | 8 | 8 | 16 |
| r | 0 | 1 | 1 | 2 | 2 | 2 | 3 | 3 | 3 | 3 | 4 |

從差分規律R(n)觀察得知

20 出現1次

21 出現2次

22 出現3次

23 出現4次

2n 出現n+1次

利用大數建表1～10000

1）差分每次成長累計乘與2 (大數乘法)

2）f(n) = f(n-1) + 差分 (大數加法)

詳細公式為

R(n) = $\left⌊\frac{\sqrt{8n+1} -1}{2}\right⌋$

F(n) = $min\_{r}$F(n,r) = $min\_{r} $[2F(n-r) + $2^{r}$- 1]

**解法範例：**

$$ F\left(7\right)= F\left(7-1\right)+ R(n)$$

 $= F\left(6\right)+R(7)$

 $ = 17+8$

 $ =25$

**討論：**

公式證明過程複雜，可參考可下列文獻

 參考資料：四柱漢諾塔之初步探究-楊楷，徐川

**程式：**

#include <iostream>

#define MAX\_DISK 10001

#define MAX\_LEN 50

using namespace std;

int hanoi\_tower[MAX\_DISK][MAX\_LEN] = {0} ;

void copy(int destination[],int source[]){

 for(int i = 0; i < MAX\_LEN; i++)

 destination[i] = source[i];

}

void power\_2(int add\_num[]){

 int i;

 for (i = 0; i < MAX\_LEN; i ++)

 add\_num[i] \*= 2;

 for (i = 0; i < MAX\_LEN; i ++)

 if (add\_num[i] >= 10)

 add\_num[i + 1] ++, add\_num[i] %= 10;

}

void add(int hanoi\_tower[],int add\_num[]){

 int i;

 for (i = 0; i < MAX\_LEN; i++){

 hanoi\_tower[i] += add\_num[i];

 }

 for (i = 0; i < MAX\_LEN; i ++){

 if (hanoi\_tower[i] >= 10){

 hanoi\_tower[ i + 1 ] += (hanoi\_tower[i]/10);

 hanoi\_tower[ i ] %= 10;

 }

 }

}

void create\_table(){

 int add\_num[MAX\_LEN] = {0};

 hanoi\_tower[1][0] = 1;

 add\_num[0] = 2;

 int i, j, count = 1,sum = 0;

 for(i = 2; i < MAX\_DISK;){

 for(j = 0; j <= count; i++, j++){

 if(i < MAX\_DISK){

 copy( hanoi\_tower[ i ] , hanoi\_tower[ i - 1 ] );

 add(hanoi\_tower[i],add\_num);

 }

 else{

 break;

 }

 }

 power\_2(add\_num);

 count++;

 }

}

int main(){

 int i,j,N;

 create\_table();

 while(cin >> N){

 if(N == 0){

 cout<<"0"<<endl;

 continue;

 }

 for (i = MAX\_LEN - 1; i >= 0; i --)

 if (hanoi\_tower[N][i] > 0) break;

 for (j = i; j >= 0; j --)

 cout<<hanoi\_tower[N][j];

 cout<<endl;

 }

}