Fibonacci　2014 暑期培训第 1 讲 --林大陈宇

Fibonacci 的基本递推公式：

$$F_n = \begin{cases} 0 & \text{if } n = 0; \\ 1 & \text{if } n = 1; \\ F_{n-1} + F_{n-2} & \text{if } n > 1. \end{cases}$$

求解递推方程就可以解出如下的通项公式：

封闭形式的通项公式：$f_n = \frac{1}{\sqrt{5}}\left[\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n\right]$

$$F(n) = \left[\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n\right](n >= 1)$$

$$\begin{pmatrix} F(n) \\ F(n-1) \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}^{n-1} \begin{pmatrix} F(1) \\ F(0) \end{pmatrix}(n >= 1)$$

$\begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}^1 \begin{bmatrix} f(0) \\ f(1) \end{bmatrix} = \begin{bmatrix} f(1) \\ f(2) \end{bmatrix}$，可以推出 $\begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}^n \begin{bmatrix} f(0) \\ f(1) \end{bmatrix} = \begin{bmatrix} f(n) \\ f(n+1) \end{bmatrix}$

1. 求解 Fibonacci 的某一项（这个范围一般在 45 之内)
   这类题目 long long 就可以了！
2. 求解 Fibonacci 的某一项模 K(这个一般是大数),通用解决方法是构造矩阵求幂次
   这得使用矩阵连乘可以解决。见上面我推导的公式。要会使用模板。
3. 求解 Fibonacci 的前多少位 (这个一般是大数), 通用解法是使用通项公式
   下面举例来说明计算前 4 位

123456.32=1234.56*10^2
s=d.xxx*10^(len-4)
log10(s)=log10(d.xxxxx)+log10(10^(len-4))=log10(d.xxxx)+len-4;
log10(s)+4-len=log10(d.xxxx)
d.xxxx=10^(log10(s)+4-len)

s=(1/sqrt(5))*[(1+sqrt(5))/2.0]^i;
len=(int)log10(s)+1;
d.xxxx=10^(log10(s)+4-((int)log10(s)+1))=10^(log10(s)-(int)log10(s)+3);
例如：HDU 的 3117 题。

4. 求解 Fibonacci 的后多少位，这个和取模类似
   和 2 相同
5. 求解 Fibonacci 的前 n 项和，利用推导式, S(n) = F(n + 2) - F(2)即可。
   最好不要记住公式，要自己会推导。本题用分块矩阵可以实现，还可以构造矩阵来实现，我会专门说一下构造矩阵。
6. 更多的是基于 Fibonacci 的综合题,包括 DP,构造,等等.
   这样的题目较多，亚洲赛，多校联合等都会看到这样的题目。

接着说一下知识点：a^b%mod , A^N%mod, 2 分查找，大数加法运算，字典数等知识，矩阵连乘算是 DP 的一种。

谁还不会二分呢？

$a^b\%m =$

要会费尔马小定理和欧拉定理：离散和数论的知识

（a+b）%c=(a%c+b%c)%c

(a*b)%c=((a%c)*(b%c))%c

费马小定理：如果 m 是素数，a 与 m 互素，则 a^(p-1)%p=1;

欧拉万能公式：$a^b\%m = a^{b\%\varphi(m)+\varphi(m)}\%m$ 不管 m 是不是素数；

二分：是指数级的收敛，特快，是 logn 的算法；是二分的快速幂不是二分查找；
例：

$$7^{83}\%5 = 7*7^{82}\%5 = 7*(7*7)^{41}\%5$$
$$= 7*4^{41}\%5 = 7*4*(4)^{40}\%5$$
$$= 3*(4*4)^{20}\%5 = 3*16^{20}\%5$$
$$= 3*(16\%5)^{20}\%5 = 3$$

如果感觉自己的二分不过关的话，可以做 HDU 的 1316：How Many Fibs?

保证你顺利熟练二分和 fibonacci 数列，我的代码如下，可以参考：

```
#include <iostream>
#include <cstdlib>
#include <cstring>
#define M 105
using namespace std;
char data[1000][M+2];
char a[M+2],b[M+2];
int cmp_ab(char *s1,char *s2)
{
    for(int k=0;k<=105;k++)
      {
         if (k==105)
             return s1[k]-s2[k];
          if (s1[k]!=s2[k])  return s1[k]-s2[k];

      }

}


int find1(int i,char *x)
{
    int low=0,high=i,mid;
    while(low<=high)
    {
        mid=(low+high)/2;
        int value=cmp_ab(x,data[mid]);
        if (value>0)  low=mid+1;
        if (value==0) return mid-1;
        if (value<0)  high=mid-1;
    }

     return high; //这时候 high 小于 low
}
```

```
int find2(int i,char *x)
{
    int low=0,high=i,mid;
    while(low<=high)
    {
        mid=(low+high)/2;
        int value=cmp_ab(x,data[mid]);
        if (value>0)  low=mid+1;
        if (value==0) return mid+1;
        if (value<0)  high=mid-1;
    }
     return low; //这时候 low 大于 high
}



int main()
{
  int i,j,p;
  //memset(data,0,sizeof(data));
  data[0][105]=1;
  data[1][105]=2;
  i=2;p=105;
  while(data[i-1][5]<=1)
  {
    for(j=105;j>=p;j--)
    data[i][j]=data[i-1][j]+data[i-2][j];

    for(j=105;j>=p;j--)
    {
        int c=data[i][j]/10;
        if (c>=1)
        {
            data[i][j]=data[i][j]%10;
            data[i][j-1]=data[i][j-1]+c;
        }

    }
     if (data[i][p-1]>0) p--;
    i++;
  }

    while(cin>>a>>b)
```

```
{
    if (a[0]=='0'&&b[0]=='0') break;
     int len1=strlen(a)-1;
     int len2=strlen(b)-1;
     int k;
     for(int d=len1,k=105;d>=0;d--,k--)
        {
            a[k]=a[d]-'0';
            a[d]=0;
        }
      for(int d=len2,k=105;d>=0;d--,k--)
        {
            b[k]=b[d]-'0';
            b[d]=0;
        }
    /*for(int i=0;i<=105;i++)
    cout<<(int)b[i];
    cout<<endl;
    */
    int lt=find1(i-1,a);
    int rt=find2(i-1,b);
    //cout<<"lt="<<lt<<endl;
    //cout<<"rt="<<rt<<endl;
   //cout<<lt<<endl;

    cout<<rt-lt-1<<endl;
    memset(a,0,sizeof(a));
    memset(b,0,sizeof(b));

}

    return 0;
}
```

下面是 **m^n % k** 的快速幂：

```
// m^n % k
int quickpow_n(int m,int n,int k)
```

```
{
    int b = 1;
    while (n > 0)
    {
          if (n & 1)// 如果n是奇数
             b = (b*m)%k;
          n = n >> 1 ;// n=n/2;
          m = (m*m)%k;
    }
    return b;
}
```

下面是矩阵快速幂：

```
/*=====================================*/
|| 快速幂（quickpow）模板
|| P 为等比，I 为单位矩阵
|| MAX 要初始化！！！！
||
/*=====================================*/
/*******************************************************/
#include <cstdio>
const int MAX = 3;

typedef  struct{
        int  m[MAX][MAX];
}  Matrix;

Matrix P = {5,-7,4,
            1,0,0,
            0,1,0,
           };

Matrix I = {1,0,0,
            0,1,0,
            0,0,1,
           };

Matrix matrixmul(Matrix a,Matrix b) //矩阵乘法
{
       int i,j,k;
       Matrix c;
       for (i = 0 ; i < MAX; i++)
           for (j = 0; j < MAX;j++)
```

```
            {
                c.m[i][j] = 0;
                for (k = 0; k < MAX; k++)
                    c.m[i][j] += (a.m[i][k] * b.m[k][j])%9997;
                c.m[i][j] %= 9997;
            }
        return c;
}

Matrix quickpow(long long n)
{
        Matrix m = P, b = I;
        while (n >= 1)
        {
            if (n & 1)
               b = matrixmul(b,m);
            n = n >> 1;
            m = matrixmul(m,m);
        }
        return b;
}
```

使用：在 main()里直接调用 quickpow()就可以了。

下面是网上的一些帖子，写得还行，就是有些解题的方法和我的思路不一样，但效果相同。可以照着做做，但下面的浙大 OJ 题目我没都做，AC 不了的题可以互相讨论。

hdu_1021:

Problem Description

There are another kind of Fibonacci numbers: F(0) = 7, F(1) = 11, F(n) = F(n-1) + F(n-2) (n>=2).

Input

Input consists of a sequence of lines, each containing an integer n. (n < 1,000,000).

Output

Print the word "yes" if 3 divide evenly into F(n).

Print the word "no" if not.

解题报告： 观察法 找出循环节=8 然后%8=2 或 %8=6 就 YES

本题也可以用矩阵乘法。

Hdu_1250:

Problem Description

A Fibonacci sequence is calculated by adding the previous two members the sequence, with the first two members being both 1.

F(1) = 1, F(2) = 1, F(3) = 1,F(4) = 1, F(n>4) = F(n - 1) + F(n-2) + F(n-3) + F(n-4)
Your task is to take a number as input, and print that Fibonacci number.
解题报告： 没办法，纯大数运算

Hdu-1568:

Problem Description

2007 年到来了。经过 2006 年一年的修炼，数学神童 zouyu 终于把 0 到 100000000 的 Fibonacci 数列
(f[0]=0,f[1]=1;f[i] = f[i-1]+f[i-2](i>=2))的值全部给背了下来。
接下来，CodeStar 决定要考考他，于是每问他一个数字，他就要把答案说出来，不过有的数字太长了。所以规定超过 4 位的只要说出前 4 位就可以了，可是 CodeStar 自己又记不住。于是他决定编写一个程序来测验 zouyu 说的是否正确。

Input

输入若干数字 n(0 <= n <= 100000000)，每个数字一行。读到文件尾。

Output

输出 f[n]的前 4 个数字（若不足 4 个数字，就全部输出）。
解题报告：
因为是前 4 位，所以可以用封闭公式： 1/sqrt(5).((1+sqrt(5))/2.0)^n ，再设 f(n)=d.xxxxx*10^x,取对数，就可以了。

Hdu-3117:
For each test case, a line will contain an integer i between 0 and 10^8 inclusively, for which you must compute the ith Fibonacci number fi. Fibonacci numbers get large pretty quickly, so whenever the answer has more than 8 digits, output only the first and last 4 digits of the answer, separating the two parts with an ellipsis ("...").
There is no special way to denote the end of the of the input, simply stop when the standard input terminates (after the EOF).
解题报告：
下面举例来说明计算前 4 位
123456.32=1234.56*10^2

s=d.xxx*10^(len-4)
log10(s)=log10(d.xxxxx)+log10(10^(len-4))=log10(d.xxxx)+len-4;
log10(s)+4-len=log10(d.xxxx)
d.xxxx=10^(log10(s)+4-len)
s=(1/sqrt(5))*[(1+sqrt(5))/2.0]^i;
len=(int)log10(s)+1;
d.xxxx=10^(log10(s)+4-((int)log10(s)+1))=10^(log10(s)-(int)log10(s)+3);

计算后 4 位 矩阵乘法幂取模,注意后 4 位时 ，例如 0123 输出要占 4 位，而不是 123

先把前 8 位的 fibonacci 的值打表打出来；8 位以后的 f(i)的值用上面方法

下面见 HDU3117 具体代码：

```
#include <iostream>
#include <stdio.h>
#include <math.h>
#define Mod 10000
using namespace std;
long long f[100];
const int MAX = 2;

typedef  struct{
       long long  m[MAX][MAX];
}  Matrix;

Matrix P = {0,1,
           1,1};

Matrix I = {1,0,
           0,1};


Matrix matrixmul(Matrix a,Matrix b) //矩阵乘法
{
      int i,j,k;
      Matrix c;
      for (i = 0 ; i < MAX; i++)
          for (j = 0; j < MAX;j++)
            {
                c.m[i][j] = 0;
                for (k = 0; k < MAX; k++)
                    c.m[i][j] += (a.m[i][k] * b.m[k][j])%Mo
d;
                c.m[i][j] %= Mod;
            }
      return c;
}

Matrix quickpow(long long n)
{
      Matrix m = P, b = I;
      while (n >= 1)
      {
            if (n & 1)
               b = matrixmul(b,m);
            n = n >> 1;
```

```
            m = matrixmul(m,m);
        }
        return b;
}

int main()
{
    //int aaaa=123;
    //printf("%04d\n",aaaa);
    double log_s=0.0;
    Matrix tmp;
    int n,bit=0;
    f[0]=0;
    f[1]=1;
    for(int i=2;i<=50;i++)
    {
     f[i]=f[i-1]+f[i-2];
     if (f[i]>=100000000)
       { bit=i-1;break;      }
    }
    //cout<<f[12]<<endl;


    while(cin>>n)
    {
      if (n<=bit) cout<<f[n]<<endl;
      if (n>bit)
      {
       tmp=quickpow(n);
       int ans_e=tmp.m[0][1];


        log_s=log10(1.0/sqrt(5)) +(double)n*log10((1.0+sqrt
(5))/2.0);
        int ans_s=(int)(pow(10,log_s-(int)log_s+3));
        cout<<ans_s<<"...";
        printf("%04d\n",ans_e);
      }
    }
    //cout<<quickpow(2,3,5)<<endl;
    //cout << "Hello world!" << endl;
    return 0;
}
```