C2 E XIAO7和兔子

21371274

张展浩

题目描述

即使上周才下了猫粮雨，XIAO7 的猫粮还是不够吃了，于是她决定在猫娘乐园里养一些兔子吃。

众所周知，兔子的繁殖速率很快，能很好地解决猫粮不够的问题，本题中**假设兔子不会死亡。**

第一天，XIAO7 有一对出生一天的幼年小兔子，小兔子会在**出生后的第二天**起变得成熟，并生育一对新的兔子（即在第 x 天出生的兔子，从 x+2 天起开始生育）。

XIAO7懒得给兔子们起名字，因此她决定按兔子的出生顺序给每对兔子编号，假设最初的第一对兔子的编号为 1，如果有两对兔子在同一天出生，那么**父母编号越小的兔子编号越大。**

XIAO7 想知道在 998244353998244353 天后编号为 a 和编号为 b 的兔子的**最近公共祖先**的编号是多少。最近公共祖先是指两对兔子所共有的祖先中，离他们的距离之和最近的一对兔子。

解题思路

本题是寻找两只兔子的最近公共祖先：

寻找公共祖先的方法是，对于 编号 **a**和**b**的两只兔子， 通过以下步骤寻找最近公共祖先。

1.将 a 和 b中编号较大的兔子编号更新为该兔子的父结点编号。

2.若a和b编号相等，则此时a和b到达二者最近公共祖先的编号，ans = a = b；

若a和b编号不等，重复步骤1，直至相等。

所以本题关键思路是如何确定兔子的直接父亲结点的编号。

根据**题目描述**

兔子在两天之后成熟，之后可以繁殖新兔子，

每一天的兔子总数 = 前一天的兔子总数 + 当天新增兔子数

设第`i`天兔子数目为 `f(i)`,可以繁殖新兔子的兔子即为`f(i-2)`

可得 **f(i) = f(i-1) + f(i-2)** f(1) = f(2) = 1;

经典的斐波那契数列；

以下均用f(i)表示斐波那契数列

而本题兔子的编号逻辑是：按兔子的出生顺序给每对兔子编号，假设最初的第一对兔子的编号为 1，如果有两对兔子在同一天出生，那么**父母编号越小的兔子编号越大。**

对于第i天，新兔子加入前的兔子总数为f(i-1), 且当天会增加f(i-2)只新兔子所以，当天**新增兔子编号**为：

> f(i-1)+1, f(i-1)+2, ... , f(i-1)+f(i-2)

当天有繁殖能力的兔子是编号 1 到 编号f(i-2)的兔子，且父母编号越小的兔子编号越大，可一一映射，建立关系。

子兔子编号	父兔子编号
f(i-1) + 1	f(i-2)
f(i-1) + 2	f(i-2) -1
f(i-1) + 3	f(i-2) - 2
.....	.....
f(i-1) + f(i-2) - 1	2
f(i-1) + f(i-2)	1

所以，对于编号为n的兔子，给出以下寻找父亲编号的方法

找出**最大的小于n [f(k) < n]**的斐波那契数： 即 `n = f(k) + m //1 <= m <= f(k-1)`

编号n的兔子父亲编号: **father(n) = f(k-1) + 1 -m**

即 **father(n) = f(k-1) + 1 - [n - f(k)]**

查找最大的小于n的斐波那契数可以使用二分查找，以下为AC代码

代码

```
#include <bits/stdc++.h>
#define LL long long
LL fb[100] = {0,1,1};  //斐波那契数列
LL find_father(LL i){ //返回i的父亲结点
    int l = 1, r = 90;
    while(l + 1 < r){  //二分查找最大的小于i的斐波那契数
        int mid = (l + r)>>1;
        if(fb[mid] < i){
            l = mid;
        }
        else{
            r = mid - 1;
        }
    }
    int k = -1;
    if(fb[r] < i) k = r;
    else if(fb[l] < i) k = l;
    return fb[k-1] + 1 - (i - fb[k]);
}
int main(){
    for(int i = 3;i<=90;i++){ //预处理，获得long long范围内的斐波那契数列
        fb[i] = fb[i-1] + fb[i-2];
    }
```

```
    int t;
    scanf("%d ",&t);
    while(t--){
        LL a,b;
        scanf("%lld%lld ",&a,&b);
        while(a != b){
            if(a > b) a = find_father(a);
            else b = find_father(b);
        }
        printf("%lld\n",a);
    }

}
```