**★★★☆☆**

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

**題號：10120: Gift?!**

**解題者：張碩文**

**解題日期：2023年5月4日**

**題意：**

有 n 個石頭在小溪上，石頭編號依序由 1....n，而編號 1 的石頭左邊有岸，編號 n 的石頭右邊也有岸，石頭間皆相距一公尺，而左岸與 1 號石頭、右岸與 n 號石頭也都相距一公尺，有隻青蛙從左岸起步，第 i 次跳躍只能往左或右跳 (i\*2-1) 公尺，若青蛙能夠到達 m 號石頭則輸出 Let me try! ，無法則輸出 Don't make fun of me!，若青蛙跳到左岸或右岸則停止。

**題意範例：**

Case1:

|  |  |
| --- | --- |
| 石頭數量 | 禮物所在位置 |
| 12 | 2 |

Output: Let me try!

Case2:

|  |  |
| --- | --- |
| 石頭數量 | 禮物所在位置 |
| 9 | 5 |

Output: Don't make fun of me!

**解法：**當N < 49的時候，使用DFS直接暴力搜尋，當N>=49時必定會有解，直接輸出Let me try!

證明大致流程如下:

1. 由於每次走的步伐是2k-1，假設一直往前走的話則總距離為 $\sum\_{k=1}^{t}(2k-1) $= $t^{2}$，假設M為禮物位置，當M為$t^{2}時$則必定有辦法走到此位置
2. 設青蛙目前的所在位置是x，下一步要跳的距離為(2k+1)，如果x-(2k+1)>=1，則青蛙一定有辦法到位置x+2

Proof:

假設青蛙的位置在x，且先往後跳 2k+1，再往前跳2k+3，則青蛙後來的位置會在x-(2k+1)+(2k+3)=x+2 => 得證

1. 設青蛙目前的所在位置是x，下一步要跳的距離為(2k+1)，如果x-((2k+1)+(y-1)\*2)>=1，則青蛙一定有辦法走到位置x+2y

****

1. 設M為禮物位置，由數論的某個定理可以得知必定能存在一個 k 使得$k^{2}$ <= M <= $(k+2)^{2}$，且 M – $k^{2}$ 必定是偶數
2. 上述的定理推得 M – $k^{2}$ = 2r ，整理後可得 M = $k^{2}$ + 2r，此時令 x = $k^{2}$、y = (M – $k^{2}$) / 2，整理後可得M = x + 2y，由第3點可得知 x-((2k+1)+(y-1)\*2)>=1 必成立，現在將x、y分別以$ k^{2}$、 (M – $k^{2}$) / 2代入，可得$k^{2}$-((2k+1)+2(M – $k^{2}$-1))，再繼續簡化會得出2$ k^{2}$ -2k + 1 - M >= 1 …(1)
3. 由於$(k+2)^{2}$>=M必成立，因此將(1)的M改成$(k+2)^{2}$，相當於是減去了更大的數字，如果要證明仍會 >=1的話，就要去解k的2次不等式，將(1)的M改成$(k+2)^{2}$之後可得到 => $k^{2}$ - 6k – 4 >= 0，要滿足上述不等式，經由公式解的計算後會得到 k >= 6.7時才會滿足，由於k必定是正整數，因此範圍會變成k >= 7
4. 知道k= 7，會滿足之後，再用第4點的性質知道當M >= 49時必定有解

8.

For M < 49

If N >= 49必定有解，因為當N = 49的時候，1-49都有辦法透過DFS找出來

If N < 49 情況則是用上述提到的DFS搜尋解即可

**解法範例：**已於解法的解釋過程中合併說明

**討論：**

* + 由於本題的N最大可以到$10^{6}$因此單純用DFS去解會有TLE的問題
	+ 但是本題當N >= 49時必定有解的數學推導非常的複雜，因此認為本題的難度可能不只3顆星
	+ 詳細的證明過程是以下網址https://algorithmist.com/wiki/UVa\_10120\_-\_Gift%3F!

**程式：**

#include<bits/stdc++.h>

using namespace std;

bool dfs(int pos,int count,int N,int M)

{

    //bool solution = false;

    if(pos < 1 || pos > N)

        return false;

    if(pos == M)

        return true;

    else

    {

        int step = count\*2-1;

        bool backward = dfs(pos-step,count+1,N,M);//2

        bool forward = dfs(pos+step,count+1,N,M);//16

        return forward || backward;

    }

}

int main()

{

    int count = 1;

    int N,M;

    while(cin>>N>>M)

    {

        if(N == 0)

            break;

        int position = 1;

        bool check = false;

        if(N>=50)

            cout<<"Let me try!"<<endl;

        else

        {

            check = dfs(position,count+1,N,M);

            if(check)

                cout<<"Let me try!"<<endl;

            else

                cout<<"Don't make fun of me!"<<endl;

        }

    }

    return 0;

}