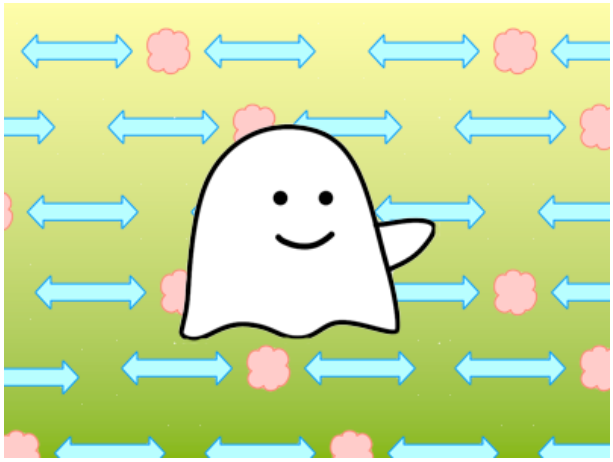
マヨイドーロ問題
（解説編）

http://www.hyuki.com/codeiq/



マヨイドーロ

Y から出るのは反転回数が奇数のときで、Z から出るのは反転回数が偶数のときです。

Y から出るか、Z から出るかはさておき、「ちょうど n 回反転してマヨイドーロから出るルートの種類の数」を a_n とすると、N が与えられたときの P は以下の式で得られます。

$$P = a_1 + a_3 + a_5 + \cdots + a_M$$

ただし、M は「N 以下の最大奇数」を表します。したがって、数列 a_n がわかれば、N から P が求められることになります。そこで、a_n についての漸化式が作れないか考えていきます。

ちょうど n 回反転して Y から出るルートの場合の数が a_n 通りあるとします。いま、X から Y に至るまでのルートで「最後に反転したマヨイ」が何であるかに注目します。Y から出るルートでは、最後に反転したマヨイは B の場合と、C の場合のいずれかです。最後に反転したマヨイが A になることはありません。最後に反転したマヨイが A なら、出口 Z から出てしまうからです

「最後に B で反転して Y から出るルート」というのは、X → B → A → Y の場合を除き、必ず B の反転前に A で反転していることがわかります。すなわち、X → B →……→ A → B → A → Y というルートになるということです。このルートの最後の「B → A →」を削除して考えればわかりますが、この形のルートは a_{n-2} 通りあります。形式的に $a_{-1} = 1$ と定めれば、先ほど除いた X → B → A → Y の場合（これは $n = 1$ の場合）も含めて a_{n-2} 通りと書けます。

「最後に C で反転して Y から出るルート」というのは、C で反転せず直進したことを想像すればわかるように、ちょうど a_{n-1} 通りあります。

ここまでをまとめると、Y から出るルートの場合には、

$$\begin{cases} a_{-1} &= 1 \\ a_n &= a_{n-2} + a_{n-1} \end{cases} \qquad (\text{ここでの } a_{n-1} \text{ は Z から出るルートの数})$$

という漸化式が得られました。

さて、同じように Z から出るルートを考えます。Z から出るルートが a_n 通りあるとしましょう。Z から出るルートは、1 度も反転せずに出た場合と、最後に反転したマヨイが B である場合と、最後に反転したマヨイが A の場合とがあります（C の場合はありません）。

- 1 度も反転せずに出るというのは $a_0 = 1$ の 1 通りです。
- 最後に B で反転して Z から出るルートというのは、a_{n-2} 通りあります。
- 最後に A で反転して Z から出るルートは、a_{n-1} 通りあります。

したがって、Z から出るルートの場合には、

$$\begin{cases} a_0 &= 1 \\ a_n &= a_{n-2} + a_{n-1} \end{cases} \qquad (\text{ここでの } a_{n-1} \text{ は Y から出るルートの数})$$

という漸化式が得られました。

結局、数列 a_n の漸化式は、

$$\begin{cases} a_{-1} & = 1 \\ a_0 & = 1 \\ a_n & = a_{n-2} + a_{n-1} \qquad n \geq 1 \end{cases}$$

になります。

この漸化式から、数列 a_n はフィボナッチ数列 F_n の添字を 2 つ分ずらした数列であることがわかりました。

n	(-1)	0	1	2	3	4	5	6	7	8
F_n		0	1	1	2	3	5	8	13	21
a_n	(1)	1	2	3	5	8	13	21	34	55

私たちが求めている P は、この数列 a_n の「添字が 1 以上の奇数で、かつ添字が N 以下」である項を加えた数になります。

N	P
0	0
1	$a_1 = 2$
2	$a_1 = 2$
3	$a_1 + a_3 = 2 + 5 = 7$
4	$a_1 + a_3 = 2 + 5 = 7$
5	$a_1 + a_3 + a_5 = 2 + 5 + 13 = 20$
6	$a_1 + a_3 + a_5 = 2 + 5 + 13 = 20$

言い換えるなら、フィボナッチ数列 F_n の「添字が 3 以上の奇数で、かつ添字が $N + 2$ 以下」である項を加えた数を計算すれば、P が求められます。

N	P
0	0
1	$F_3 = 2$
2	$F_3 = 2$
3	$F_3 + F_5 = 2 + 5 = 7$
4	$F_3 + F_5 = 2 + 5 = 7$
5	$F_3 + F_5 + F_7 = 2 + 5 + 13 = 20$
6	$F_3 + F_5 + F_7 = 2 + 5 + 13 = 20$

ところで、フィボナッチ数列には、

$$F_1 + F_3 + \cdots + F_{2k-1} = F_{2k} \qquad (k \text{ は } 1 \text{ 以上の整数})$$

という性質がありますので、私たちの求める P は、

$$P = F_S - 1$$

で求められます。S は $N + 2$ 以上の最小偶数です。

N	P
0	$F_2 - 1 = 0$
1	$F_4 - 1 = 2$
2	$F_4 - 1 = 2$
3	$F_6 - 1 = 7$
4	$F_6 - 1 = 7$
5	$F_8 - 1 = 20$
6	$F_8 - 1 = 20$

以上で、与えられた N に対する P を求める方法がわかりました。
フィボナッチ数列 を計算すればいいのですね！

コード例（Ruby）

```ruby
#! /usr/bin/ruby

FIB = Hash.new(nil)

def fib(n)
  if not FIB[n]
    FIB[n] = case n
             when 0
               0
             when 1
               1
             else
               fib(n-1) + fib(n-2)
             end
  end
  FIB[n]
end

def solve(n)
  fib((n + 3) / 2 * 2) - 1
end

puts solve(STDIN.gets.to_i)
```


Enjoy!