การคำนวณจำนวนฟิโบนักชีโดยใช้การยกกำลังเมทริกซ์อย่างรวดเร็ว

0, 1, 1, 2, 3, 5, 8, ... เป็นลำดับของจำนวนฟิโบนักชี ($F_0 = 0, F_1 = 1, F_2 = 1, ...$) วิธีหนึ่งในการหา F_n คือคำนวณ $\begin{bmatrix}0 & 1\\ 1 & 1\end{bmatrix}^n$ ได้ผลเป็นเมทริกซ์ขนาด 2×2 มี F_n ที่มุมขวาบนของเมทริกซ์ เช่น $\begin{bmatrix}0 & 1\\ 1 & 1\end{bmatrix}^3 = \begin{bmatrix}1 & 2\\ 2 & 3\end{bmatrix}$,$\begin{bmatrix}0 & 1\\ 1 & 1\end{bmatrix}^4 = \begin{bmatrix}2 & 3\\ 3 & 5\end{bmatrix}$

ถ้าคิดดูดี ๆ จะพบว่าการหาด้วยวิธีข้างต้นนี้คือการหาค่ายกกำลัง ซึ่งเราก็ไม่น่าหาแบบค่อยๆ คูณไปทีละครั้ง คือถ้า A เป็นเมทริกซ์ การหา A^{10} ก็ไม่น่าใช้วิธีที่เริ่มด้วยเมทริกซ์เอกลักษณ์ I = $\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$ แล้วคูณด้วย A ไป 10 ครั้ง น่าจะใช้วิธีการหา A^5 แล้วจับมาคูณกับตัวเอง ก็จะได้ A^{10} เหมือนกับที่เรียนเรื่อง power mod นั่นคือ

$$A^n = \begin{cases} \mathrm{I} & n = 0 \\ \left(A^{\lfloor n/2 \rfloor}\right)^2 & n \text{ is even} \\ A\left(A^{\lfloor n/2 \rfloor}\right)^2 & n \text{ is odd} \end{cases}$$

จงเขียนฟังก์ชัน **fib(n, k)** เพื่อคำนวณ F_n **% k** ด้วยวิธีข้างต้นนี้ โดยใช้ คำสั่งใช้ numpy เพื่อการคูณเมทริกซ์ (หมายเหตุ : หลังการคูณเมทริกซ์ทุกครั้ง ให้นำผลที่ได้มา % k numpy จะทำ % k แบบ element-wise ในเมทริกซ์)

```
import numpy as np

def fib(n, k):

   ???

n,k = [int(e) for e in input().split()]
print( fib(n,k) )
```

ข้อมูลนำเข้า

จำนวนเต็ม 2 ค่า **n** กับ **k** (**0 ≤ n ≤ 10000000000000, 0 ≤ k ≤ 100000**)

ข้อมูลส่งออก

แสดงค่า F_n **% k**

ตัวอย่าง

input (จากแป้นพิมพ์)	output (ทางจอภาพ)
0 10	0
1 10	1
2 10	1
89 10	9
11111 111	55
1111111111 111	76
1234567890 1234	162
10000000000000 999	600