Задача формата ЕГЭ, претендующая на номер 27
Сложный уровень

(автор: Кардашевский Илья Николаевич)
https://vk.com/ik4rd

Условие:

Нолику стало ужасно скучно на уроке информатики, ведь учитель по теме «Системы счисления» давал лишь муторные сложения, вычитания и переводы из одной системы в другую... Чтобы чем-то занять своего брата, Симка придумала следующую задачу: назовём натуральное число $\boldsymbol{n}$ *«вайбовым», если его можно разложить в сумму различных целых неотрицательных степеней натурального числа* $\boldsymbol{k}$.
Более формально:

$$n = k^{a_1} + k^{a_2} + k^{a_3} + \ldots + k^{a_m}$$

(при чем для любых $1 \le i, j \le m$ выполняется условие $a_i \neq a_j$)

Например, для $\boldsymbol{k} = 3$ *число* $\boldsymbol{n} = 6669$ *— «вайбовое», так как* $6669 = 3^3 + 3^4 + 3^8$
Для заданных натуральных $\boldsymbol{n}$ *и* $\boldsymbol{k}$ *($n >= k$ и $n \le 10^{20}, k \le 10^{20}$) требуется найти сумму степеней в разложении «вайбового» числа* $\boldsymbol{m}$*, которое не меньше* $\boldsymbol{n}$.

(Дисклеймер!!! Данный уровень отличается от среднего не значениями, а самой идеей)

Решение:

Алгоритм решения из предыдущего случая, к сожалению, не подойдет для таких ограничений, поэтому придумаем оптимизацию: вместо того, чтобы перебирать все числа начиная с n и проверять подходят ли они нам, мы будем из k-ичного представления числа n, путем замены и добавления разрядов получать подходящее число.

Для начала вспомним замечание: *...представление содержит только 0 и 1.* Как можно изменить или добавить несколько разрядов в числе так, чтобы оно стало не меньше и в нём были только 0 и 1?

Для этого найдем самый старший разряд больший 1. Очевидно от него нам надо избавиться, но также понятно, что для этого придется как-то менять заряды старше его (либо добавлять еще один). То есть нужно сделать первый нулевой разряд старше найденного равным 1 (либо опять же добавить к записи 1). Тогда оставшиеся разряды младше мы сделаем равными 0. Действительно, во-первых, мы избавляемся от разрядов со значением превышающим 1, во-вторых, благодаря такому трюку, мы делаем число больше исходного и при этом минимальным. Для полученной записи считаем ответ.

Благодаря такой оптимизации решение работает за $O(\log_k n)$

Код решения:

```python
def solve(n : int, k : int) -> int:
    arr = []
    while n:
        arr.append(n % k)
        n //= k
    arr.append(0)

    m = len(arr)
    for i in range(m - 1, -1, -1):
        if arr[i] > 1:
            for j in range(i + 1, m):
                if arr[j] == 0:
                    arr[j] = 1
                    for l in range(0, j):
                        arr[l] = 0
                    break
            break

    ans = 0
    for index, bit in enumerate(arr):
        if bit == 1:
            ans += index

    return ans

n, k = map(int, input().split())
print(solve(n, k))
```