Задача формата ЕГЭ, претендующая на номер 27
Средний уровень

(автор: Кардашевский Илья Николаевич)
https://vk.com/id321975104

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 ***n*** *«вайбовым», если его можно разложить в сумму различных целых неотрицательных степеней натурального числа* ***k****.*
Более формально:

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

(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 ***n*** *и* ***k*** *(*$n >= k$ и $n \le 10^6, k \le 100$*) требуется найти сумму степеней в разложении минимального «вайбового» числа* ***m****, которое не меньше* ***n****.*

Решение*:*

Заметим, что «вайбовым» число будет тогда, когда при переводе в систему счисления с основанием k, представление будет состоять из 0 и 1. Для решения задачи будем перебирать числа от n, переводя их в систему счисления с основанием k и проверяя их на «вайбовость». Для первого найденного числа считаем ответ.

Замечание: код будет работать достаточно быстро, поскольку между n и nk найдется хотя бы число k^a *(для некоторого a), подходящее под ответ.*

Код решения:

```
def check(n : int, k : int) -> bool: # функция проверки на «вайбовость»
	while n:
		if n % k > 1: # если встречается разряд > 1 – не подходит
			return False
		n //= k
	return True

def solve(n : int, k : int) -> int: # функция решения
	s = ''
	while n: # перевод n в k-ичную систему счисления
		s += str(n % k)
		n //= k

	ans = 0
	for index, bit in enumerate(s):
		if bit == '1':
			ans += index # суммируем степени

	return ans

n, k = map(int, input().split())

m = n
while True:
	if check(m, k): # если число «вайбовое»
		print(solve(m, k)) # выводим ответ
		break
	m += 1
```

Этапы решения:

1) Заведем функцию `check`, принимающую в качестве аргументов исходное число n и k — основание системы. Условие `while n` тождественно `while n != 0`, то есть мы будем получать очередной разряд и делить нацело число n на k, пока оно не станет равным 0.
2) Объявим функцию `solve`, решающую нашу задачу. Она будет принимать значение n и k и возвращать нам сумму степеней в разложении.
3) Переменная s — разложение числа n. При помощи цикла аналогичного пункту 1, мы будем добавлять в эту строку разряды и делить нацело число n на k.
4) Переменная ans — наш ответ. При помощи цикла `enumerate(s)`, мы поддерживаем, как и значение степени для разряда, так и само его значение. Таким образом, проверяя является ли очередной разряд равным 1, мы можем спокойно добавлять значение степени в ответ.
5) Далее идет основная часть программы: мы вводим значения n и k. Потом заводим переменную m — стартовую для поиска ответа и при помощи цикла перебираем значения с шагом 1, и, если очередное значение переменной m нас устраивает, мы выводим и ответ и выходим из цикла.