Beautiful Summation

- Dado P, Q, N y M.

$$S_N = \sum_{k=1}^{N} P^k \times k^Q$$

Solución

- Dado que el calculo depende de N que puede variar entre $1 \leq N \leq 10^9$, se traslada la carga computacional a Q que varia entre $0 \leq Q \leq 1000$.

Method 2.

We can use divide and conquer algorithm [4][5] recursively:
if n is odd then $f(n,k) = f(n-1,k) + n^k$

if n is even then $f(n,k) = f(n/2,k) + (n/2+1)^k + (n/2+2)^k + \ldots + (n/2+n/2)^k = f(n/2,k) + \sum_{i=1}^{n/2} (n/2+i)^k = f(n/2,k) + \sum_{i=0}^{k} (\binom{k}{i} f(n/2,i)(n/2)^{k-i})^k$.
if $n = 1$ then $f(n,k) = 1$

We can precalculate binomial coefficients in O(k^2) using it's recursion formula $\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$. The recursion with different parameters should be called $klogn$ times. We will use memorization method for not solving one recursion two times. One recursion call works in O(k). So the overall complexity of this algorithm is O$(k^2 log(n))$.

Some algorithmic methods for computing the sum of powers. https://www.ijser.org/researchpaper/Some-algorithmic-methods-for-computing-the-sum-of-powers.pdf

$$f(n,k) = \sum_{i=1}^{n} i^k$$

Para asegurar que n sea par:

$$f(n,k) = f(n-1,k) + n^k$$

De esa forma:

$$f(n,k) = f(n/2,k) + (n/2+1)^k + (n/2+2)^k + ... + (n/2+n/2)^k$$

> Ej:
>
> $$\begin{aligned} f(4,2) &= 1^2 + 2^2 + 3^2 + 4^2 \\ &= f(2,2) + 3^2 + 4^2 \\ &= f(2,2) + (2+1)^2 + (2+2)^2 \end{aligned}$$

Aplicando el **teorema del binomio**:

$$\begin{aligned} x &= (n/2+1)^k + (n/2+2)^k + ... + (n/2+n/2)^k \\ &= \binom{k}{0}(n/2)^k 1^0 + \binom{k}{1}(n/2)^{k-1} 1^1 + ... + \binom{k}{k}(n/2)^0 1^k \\ &+ \binom{k}{0}(n/2)^k 2^0 + \binom{k}{1}(n/2)^{k-1} 2^1 + ... + \binom{k}{k}(n/2)^0 2^k \\ &+ \binom{k}{0}(n/2)^k 3^0 + \binom{k}{1}(n/2)^{k-1} 3^1 + ... + \binom{k}{k}(n/2)^0 3^k + ... \\ ... &+ \binom{k}{0}(n/2)^k (n/2)^0 + \binom{k}{1}(n/2)^{k-1} (n/2)^1 + ... + \binom{k}{k}(n/2)^0 (n/2)^k \end{aligned}$$

Agrupando:

$$\begin{aligned} x = &\binom{k}{0}(n/2)^k (1^0 + 2^0 + 3^0 + ... + n/2^0) \\ &+ \binom{k}{1}(n/2)^{k-1} (1^1 + 2^1 + 3^1 + .. + n/2^1) \\ &\vdots \\ &+ \binom{k}{k}(n/2)^0 (1^k + 2^k + 3^k + ... + n/2^k) \\ = &\binom{k}{0}(n/2)^k f(n/2,0) + \binom{k}{1}(n/2)^{k-1} f(n/2,1) + ... + \binom{k}{k}(n/2)^0 f(n/2,k) \end{aligned}$$

De este modo se tiene:

$$f(n,k) = f(n/2,k) + \sum_{i=0}^{k} \binom{k}{i}(n/2)^{k-i} f(n/2,i)$$

Dado que vamos a utilizar recursividad para dividir $f(n,k)$ multiples veces, definimos un caso base:

$$f(1,k)=1$$

Adaptación al problema

$$S(n,q)=\sum_{k=1}^{n}p^k\cdot k^q$$

Para asegurar que n sea par:

$$S(n,q)=S(n-1,q)+p^n\cdot n^q$$

Caso base:

$$S(1,q)=p$$

Cuando n es par:

$$S(n,q)=S(n/2,q)+p^{n/2+1}(n/2+1)^q+p^{n/2+2}(n/2+1)^q+...+p^{n/2+n/2}(n/2+n/2)^q$$

$$\begin{aligned}x&=p^{n/2+1}(n/2+1)^q+p^{n/2+2}(n/2+2)^q+...+p^{n/2+n/2}(n/2+n/2)^q\\&=p^{n/2}\sum_{i=1}^{q}\binom{q}{i}(n/2)^{q-i}S(n/2,i)\end{aligned}$$

Así se tiene la expresión para reducir:

$$S(n,q)=S(n/2,q)+p^{n/2}\sum_{i=1}^{q}\binom{q}{i}(n/2)^{q-i}S(n/2,i)$$

Consideraciones

- Calculo de binomiales mediante recursividad:

$$\binom{n}{k}=\binom{n-1}{k-1}+\binom{n-1}{k}\quad,\begin{cases}\text{if} & k=n \text{ or } k=0 & \text{then } \binom{n}{k}=1\\ \text{if} & k>n & \text{then } \binom{n}{k}=0\end{cases}$$

- Calculo de potencias (p^n) mediante recursividad: https://en.wikipedia.org/wiki/Exponentiation_by_squaring

- Uso de Programación Dinámica para almacenar valores de $S(n, q)$ y $\binom{n}{k}$ que ya han sido calculados y se usen mas tarde.