Aritmética y Teoría de Numeros

Ariel Zylber

Training Camp Argentina

Agosto 2014

Contenidos

Lo básico

Decimos que dos enteros a y b son congruentes módulo n si n divide a la diferencia, es decir, $n \mid a - b$.
En ese caso notamos $a \equiv b \pmod{n}$.

Llamamos resto de a módulo n al número r entre 0 y $n - 1$ tal que $a \equiv r \pmod{n}$.
Existe una única forma de escribir $a = n \cdot q + r$, con $0 \leq r < n$.
En este caso notamos $a\%n = r$.

Lo básico

Decimos que dos enteros a y b son congruentes módulo n si n divide a la diferencia, es decir, $n \mid a - b$.
En ese caso notamos $a \equiv b \pmod{n}$.

Llamamos resto de a módulo n al número r entre 0 y $n - 1$ tal que $a \equiv r \pmod{n}$.
Existe una única forma de escribir $a = n \cdot q + r$, con $0 \leq r < n$.
En este caso notamos $a \% n = r$.

Lo básico

Decimos que dos enteros a y b son congruentes módulo n si n divide a la diferencia, es decir, $n \mid a - b$.
En ese caso notamos $a \equiv b \pmod{n}$.

Llamamos resto de a módulo n al número r entre 0 y $n - 1$ tal que $a \equiv r \pmod{n}$.
Existe una única forma de escribir $a = n \cdot q + r$, con $0 \leq r < n$.
En este caso notamos $a \% n = r$.

Operaciones

Lo interesante es que para realizar un cuenta donde sólo nos interesa el resto del resultado, para algunas operaciones podemos tomar resto en pasos intermedios.

Si $a + b \equiv r \pmod{n} \Rightarrow a\%n + b\%n \equiv r \pmod{n}$.
Si $a - b \equiv r \pmod{n} \Rightarrow a\%n - b\%n \equiv r \pmod{n}$.
Si $a \cdot b \equiv r \pmod{n} \Rightarrow a\%n \cdot b\%n \equiv r \pmod{n}$.
El problema surge cuando queremos dividir.

Operaciones

Lo interesante es que para realizar un cuenta donde sólo nos interesa el resto del resultado, para algunas operaciones podemos tomar resto en pasos intermedios.

Si $a + b \equiv r \pmod{n} \Rightarrow a\%n + b\%n \equiv r \pmod{n}$.
Si $a - b \equiv r \pmod{n} \Rightarrow a\%n - b\%n \equiv r \pmod{n}$.
Si $a \cdot b \equiv r \pmod{n} \Rightarrow a\%n \cdot b\%n \equiv r \pmod{n}$.
El problema surge cuando queremos dividir.

Contenidos

Elevar módulo p

Queremos calcular a^n.
El algoritmo obvio es hacer $a \cdot a \cdot a \cdot \ldots \cdot a$, n veces. El tiempo de ejecución es $O(n)$.

Queremos algo mejor. Notemos que si n es par, $n = 2k$ tenemos:

$$a^n = a^{2k} = (a^k)^2 = a^k \cdot a^k.$$

Y si n es impar, $n = 2k + 1$.

$$a^n = a^{2k+1} = a \cdot (a^{2k}) = a \cdot a^k \cdot a^k.$$

En ambos casos reducimos el problema a la mitad con $O(1)$ multiplicaciones, luego podemos elevar en $O(log(n))$.

Contenidos

Inverso Módular

Consideramos ahora un número primo p, y un entero $a \not\equiv 0$.
Miremos la siguiente secuencia módulo p:

$$a, 2 \cdot a, 3 \cdot a, \ldots, (p-1) \cdot a.$$

Son todos distintos módulo p, luego hay un m tal que $m \cdot a \equiv 1$.
Lo notamos $m = a^{-1}$.

Ahora dividir por a es como multiplicar por a^{-1}.

Inverso Módular

Consideramos ahora un número primo p, y un entero $a \neq 0$.
Miremos la siguiente secuencia módulo p:

$$a, 2 \cdot a, 3 \cdot a, \ldots, (p-1) \cdot a.$$

Son todos distintos módulo p, luego hay un m tal que $m \cdot a \equiv 1$.
Lo notamos $m = a^{-1}$.

Ahora dividir por a es como multiplicar por a^{-1}.

Inverso Módular

Consideramos ahora un número primo p, y un entero $a \not\equiv 0$.
Miremos la siguiente secuencia módulo p:

$$a, 2 \cdot a, 3 \cdot a, \ldots, (p-1) \cdot a.$$

Son todos distintos módulo p, luego hay un m tal que $m \cdot a \equiv 1$.
Lo notamos $m = a^{-1}$.

Ahora dividir por a es como multiplicar por a^{-1}.

Inverso Módular

Consideramos ahora un número primo p, y un entero $a \not\equiv 0$.
Miremos la siguiente secuencia módulo p:

$$a, 2 \cdot a, 3 \cdot a, \ldots, (p-1) \cdot a.$$

Son todos distintos módulo p, luego hay un m tal que $m \cdot a \equiv 1$.
Lo notamos $m = a^{-1}$.

Ahora dividir por a es como multiplicar por a^{-1}.

Hallando el inverso

¿Cómo hallamos el inverso de a módulo p?

Con ayuda del siguiente teorema:

Pequeño Teorema de Fermat

Dado p primo y $a \not\equiv 0$, $a^{p-1} \equiv 1 \pmod{p}$.

O lo que es lo mismo $a^{p-2} \cdot a \equiv 1 \pmod{p}$.

Podemos hallar a^{p-2} en $O(log(p))$.

Hallando el inverso

¿Cómo hallamos el inverso de a módulo p?
Con ayuda del siguiente teorema:

Pequeño Teorema de Fermat

Dado p primo y $a \not\equiv 0$, $a^{p-1} \equiv 1 \pmod{p}$.

O lo que es lo mismo $a^{p-2} \cdot a \equiv 1 \pmod{p}$.

Podemos hallar a^{p-2} en $O(log(p))$.

Hallando el inverso

¿Cómo hallamos el inverso de a módulo p?
Con ayuda del siguiente teorema:

Pequeño Teorema de Fermat

Dado p primo y $a \not\equiv 0$, $a^{p-1} \equiv 1 \pmod p$.

O lo que es lo mismo $a^{p-2} \cdot a \equiv 1 \pmod p$.

Podemos hallar a^{p-2} en $O(log(p))$.

Hallando el inverso

¿Cómo hallamos el inverso de a módulo p?
Con ayuda del siguiente teorema:

Pequeño Teorema de Fermat

Dado p primo y $a \not\equiv 0$, $a^{p-1} \equiv 1 \pmod{p}$.

O lo que es lo mismo $a^{p-2} \cdot a \equiv 1 \pmod{p}$.

Podemos hallar a^{p-2} en $O(log(p))$.

Si el módulo no es primo

Ahora dado un n cualquiera, el inverso de a módulo n existe si y sólo si a y n son coprimos.

Esto nos induce a definir la función $\phi(n)$ que nos da la cantidad de números menores a n coprimos con n.

Por ejemplo, $\phi(p) = p - 1$ para p primo.
Tenemos que el inverso de a módulo n es $a^{\phi(n)-1}$.

Si el módulo no es primo

Ahora dado un n cualquiera, el inverso de a módulo n existe si y sólo si a y n son coprimos.

Esto nos induce a definir la función $\phi(n)$ que nos da la cantidad de números menores a n coprimos con n.

Por ejemplo, $\phi(p) = p - 1$ para p primo.
Tenemos que el inverso de a módulo n es $a^{\phi(n)-1}$.

Si el módulo no es primo

Ahora dado un n cualquiera, el inverso de a módulo n existe si y sólo si a y n son coprimos.

Esto nos induce a definir la función $\phi(n)$ que nos da la cantidad de números menores a n coprimos con n.

Por ejemplo, $\phi(p) = p - 1$ para p primo.
Tenemos que el inverso de a módulo n es $a^{\phi(n)-1}$.

Si el módulo no es primo

Ahora dado un n cualquiera, el inverso de a módulo n existe si y sólo si a y n son coprimos.

Esto nos induce a definir la función $\phi(n)$ que nos da la cantidad de números menores a n coprimos con n.

Por ejemplo, $\phi(p) = p - 1$ para p primo.
Tenemos que el inverso de a módulo n es $a^{\phi(n)-1}$.

Todos los inversos

Hay un simple algoritmo que permite hallar todos los inversos módulo p en tiempo lineal ($O(p)$).

Los calcularemos inductivamente. El inverso de 1 siempre es 1. Para calcular el inverso de $n > 1$, la clave es la siguiente igualdad:

$$n \cdot \left\lfloor \frac{p}{n} \right\rfloor + p \,\%n = p.$$

De esto se deduce que el inverso de n es:

$$p - \left\lfloor \frac{p}{n} \right\rfloor \cdot (p \,\%n)^{-1}.$$

Y $0 < (p \,\%n) < n$, luego ya lo tenemos calculado.

Todos los inversos

Hay un simple algoritmo que permite hallar todos los inversos módulo p en tiempo lineal ($O(p)$).

Los calcularemos inductivamente. El inverso de 1 siempre es 1. Para calcular el inverso de $n > 1$, la clave es la siguiente igualdad:

$$n \cdot \left\lfloor \frac{p}{n} \right\rfloor + p \% n = p.$$

De esto se deduce que el inverso de n es:

$$p - \left\lfloor \frac{p}{n} \right\rfloor \cdot (p \% n)^{-1}.$$

Y $0 < (p \% n) < n$, luego ya lo tenemos calculado.

Todos los inversos

Hay un simple algoritmo que permite hallar todos los inversos módulo p en tiempo lineal ($O(p)$).

Los calcularemos inductivamente. El inverso de 1 siempre es 1. Para calcular el inverso de $n > 1$, la clave es la siguiente igualdad:

$$n \cdot \left\lfloor \frac{p}{n} \right\rfloor + p \% n = p.$$

De esto se deduce que el inverso de n es:

$$p - \left\lfloor \frac{p}{n} \right\rfloor \cdot (p \% n)^{-1}.$$

Y $0 < (p \% n) < n$, luego ya lo tenemos calculado.

Todos los inversos

Hay un simple algoritmo que permite hallar todos los inversos módulo p en tiempo lineal ($O(p)$).

Los calcularemos inductivamente. El inverso de 1 siempre es 1. Para calcular el inverso de $n > 1$, la clave es la siguiente igualdad:

$$n \cdot \left\lfloor \frac{p}{n} \right\rfloor + p\,\%n = p.$$

De esto se deduce que el inverso de n es:

$$p - \left\lfloor \frac{p}{n} \right\rfloor \cdot (p\,\%n)^{-1}.$$

Y $0 < (p\,\%n) < n$, luego ya lo tenemos calculado.

Contenidos

Dados $m_1, m_2, \ldots, m_n$ todos coprimos dos a dos y $m = m_1 \cdot m_2 \cdot \ldots \cdot m_n$, y $x_1, x_2, \ldots, x_n$ definimos el siguiente sistema de congruencias.

$$
\begin{gathered}
x \equiv x_1 \pmod{m_1} \\
x \equiv x_1 \pmod{m_1} \\
\ldots \\
x \equiv x_n \pmod{m_n}
\end{gathered}
$$

Queremos encontrar y tal que $x \equiv y \pmod{m}$.

Dados $m_1, m_2, \ldots, m_n$ todos coprimos dos a dos y $m = m_1 \cdot m_2 \cdot \ldots \cdot m_n$, y $x_1, x_2, \ldots, x_n$ definimos el siguiente sistema de congruencias.

$$\begin{gathered} x \equiv x_1 \pmod{m_1} \\ x \equiv x_1 \pmod{m_1} \\ \ldots \\ x \equiv x_n \pmod{m_n} \end{gathered}$$

Queremos encontrar y tal que $x \equiv y \pmod{m}$.

Dados $m_1, m_2, \ldots, m_n$ todos coprimos dos a dos y $m = m_1 \cdot m_2 \cdot \ldots \cdot m_n$, y $x_1, x_2, \ldots, x_n$ definimos el siguiente sistema de congruencias.

$$\begin{gathered} x \equiv x_1 \pmod{m_1} \\ x \equiv x_1 \pmod{m_1} \\ \ldots \\ x \equiv x_n \pmod{m_n} \end{gathered}$$

Queremos encontrar y tal que $x \equiv y \pmod{m}$.

Teorema Chino del Resto

Teorema Chino del Resto

Para todo sistema con esas condiciones existe solución y es única.

Resolvamos primero el caso donde $x_2 = x_2 = \ldots = x_n = 0$.
La solución es $y = x_1 \cdot (\prod_{i=2}^{n} m_i)_{m_1}^{-1} \cdot \prod_{i=2}^{n} m_i$.

Para el caso general basta sumar las soluciones para cada x_i,

$$y = \sum_{j=1}^{n} (x_j \cdot (\prod_{i=1, i\neq j}^{n} m_i)_{m_i}^{-1} \cdot \prod_{i=1, i\neq j}^{n} m_i).$$

Teorema Chino del Resto

Teorema Chino del Resto

Para todo sistema con esas condiciones existe solución y es única.

Resolvamos primero el caso donde $x_2 = x_2 = \ldots = x_n = 0$.
La solución es $y = x_1 \cdot (\prod_{i=2}^{n} m_i)_{m_1}^{-1} \cdot \prod_{i=2}^{n} m_i$.

Para el caso general basta sumar las soluciones para cada x_i,

$$y = \sum_{j=1}^{n} (x_j \cdot (\prod_{i=1, i \neq j}^{n} m_i)_{m_i}^{-1} \cdot \prod_{i=1, i \neq j}^{n} m_i).$$

Teorema Chino del Resto

Teorema Chino del Resto

Para todo sistema con esas condiciones existe solución y es única.

Resolvamos primero el caso donde $x_2 = x_2 = \ldots = x_n = 0$.
La solución es $y = x_1 \cdot (\prod_{i=2}^{n} m_i)_{m_1}^{-1} \cdot \prod_{i=2}^{n} m_i$.

Para el caso general basta sumar las soluciones para cada x_i,

$$y = \sum_{j=1}^{n} (x_j \cdot (\prod_{i=1, i \neq j}^{n} m_i)_{m_i}^{-1} \cdot \prod_{i=1, i \neq j}^{n} m_i).$$

Teorema Chino del Resto

Teorema Chino del Resto

Para todo sistema con esas condiciones existe solución y es única.

Resolvamos primero el caso donde $x_2 = x_2 = \ldots = x_n = 0$.
La solución es $y = x_1 \cdot (\prod_{i=2}^{n} m_i)_{m_1}^{-1} \cdot \prod_{i=2}^{n} m_i$.

Para el caso general basta sumar las soluciones para cada x_i,

$$y = \sum_{j=1}^{n}(x_j \cdot (\prod_{i=1,i\neq j}^{n} m_i)_{m_i}^{-1} \cdot \prod_{i=1,i\neq j}^{n} m_i).$$

Contenidos

Números combinatorios

La cantidad de elegir k objetos de un conjunto de n es:

$$\binom{n}{k} = \frac{n!}{k! \cdot (n-k)!}$$

Podemos usar lo que vimos hasta ahora para calcular este número módulo p.

Vemos que por ahora podemos calcular un combinatorio en $O(n \cdot log(p))$, de hecho, lo podemos bajar a $O(k \cdot log(p))$.

Números combinatorios

La cantidad de elegir k objetos de un conjunto de n es:

$$\binom{n}{k} = \frac{n!}{k! \cdot (n-k)!}$$

Podemos usar lo que vimos hasta ahora para calcular este número módulo p.

Vemos que por ahora podemos calcular un combinatorio en $O(n \cdot log(p))$, de hecho, lo podemos bajar a $O(k \cdot log(p))$.

Números combinatorios

La cantidad de elegir k objetos de un conjunto de n es:

$$\binom{n}{k} = \frac{n!}{k! \cdot (n-k)!}$$

Podemos usar lo que vimos hasta ahora para calcular este número módulo p.

Vemos que por ahora podemos calcular un combinatorio en $O(n \cdot log(p))$, de hecho, lo podemos bajar a $O(k \cdot log(p))$.

Algunas propiedades

Valen muchas igualdades para los números combinatorios:

- $\binom{n}{k} = \binom{n}{n-k}$

  Luego podemos elegir calcular el que requiera menos operaciones.

- $\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$

  Esto nos va a ayudar a calcular recursivamente todos los combinatorios.

- $\sum_{i=0}^{n} \binom{n}{i} = 2^n$

  Ambas cuentan la cantidad de subconjuntos de un conjunto de n elementos.

- $\sum_{i=0}^{n} i\binom{n}{i} = n \cdot 2^{n-1}$

  Ambas cuentan la suma de la cantidad de elementos de todos los subconjuntos de un conjunto de n elementos.

Algunas propiedades

Valen muchas igualdades para los números combinatorios:

- $\binom{n}{k} = \binom{n}{n-k}$

  Luego podemos elegir calcular el que requiera menos operaciones.

- $\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$

  Esto nos va a ayudar a calcular recursivamente todos los combinatorios.

- $\sum_{i=0}^{n} \binom{n}{i} = 2^n$

  Ambas cuentan la cantidad de subconjuntos de un conjunto de n elementos.

- $\sum_{i=0}^{n} i\binom{n}{i} = n \cdot 2^{n-1}$

  Ambas cuentan la suma de la cantidad de elementos de todos los subconjuntos de un conjunto de n elementos.

Algunas propiedades

Valen muchas igualdades para los números combinatorios:

- $\binom{n}{k} = \binom{n}{n-k}$

  Luego podemos elegir calcular el que requiera menos operaciones.

- $\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$

  Esto nos va a ayudar a calcular recursivamente todos los combinatorios.

- $\sum_{i=0}^{n} \binom{n}{i} = 2^n$

  Ambas cuentan la cantidad de subconjuntos de un conjunto de n elementos.

- $\sum_{i=0}^{n} i\binom{n}{i} = n \cdot 2^{n-1}$

  Ambas cuentan la suma de la cantidad de elementos de todos los subconjuntos de un conjunto de n elementos.

Algunas propiedades

Valen muchas igualdades para los números combinatorios:

- $\binom{n}{k} = \binom{n}{n-k}$

  Luego podemos elegir calcular el que requiera menos operaciones.

- $\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$

  Esto nos va a ayudar a calcular recursivamente todos los combinatorios.

- $\sum_{i=0}^{n} \binom{n}{i} = 2^n$

  Ambas cuentan la cantidad de subconjuntos de un conjunto de n elementos.

- $\sum_{i=0}^{n} i\binom{n}{i} = n \cdot 2^{n-1}$

  Ambas cuentan la suma de la cantidad de elementos de todos los subconjuntos de un conjunto de n elementos.

Algunas propiedades

Valen muchas igualdades para los números combinatorios:

- $\binom{n}{k} = \binom{n}{n-k}$

  Luego podemos elegir calcular el que requiera menos operaciones.

- $\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$

  Esto nos va a ayudar a calcular recursivamente todos los combinatorios.

- $\sum_{i=0}^{n} \binom{n}{i} = 2^n$

  Ambas cuentan la cantidad de subconjuntos de un conjunto de n elementos.

- $\sum_{i=0}^{n} i\binom{n}{i} = n \cdot 2^{n-1}$

  Ambas cuentan la suma de la cantidad de elementos de todos los subconjuntos de un conjunto de n elementos.

Algunas propiedades

Valen muchas igualdades para los números combinatorios:

- $\binom{n}{k} = \binom{n}{n-k}$

  Luego podemos elegir calcular el que requiera menos operaciones.

- $\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$

  Esto nos va a ayudar a calcular recursivamente todos los combinatorios.

- $\sum_{i=0}^{n} \binom{n}{i} = 2^n$

  Ambas cuentan la cantidad de subconjuntos de un conjunto de n elementos.

- $\sum_{i=0}^{n} i\binom{n}{i} = n \cdot 2^{n-1}$

  Ambas cuentan la suma de la cantidad de elementos de todos los subconjuntos de un conjunto de n elementos.

Algunas propiedades

Valen muchas igualdades para los números combinatorios:

- $\binom{n}{k} = \binom{n}{n-k}$

  Luego podemos elegir calcular el que requiera menos operaciones.

- $\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$

  Esto nos va a ayudar a calcular recursivamente todos los combinatorios.

- $\sum_{i=0}^{n} \binom{n}{i} = 2^n$

  Ambas cuentan la cantidad de subconjuntos de un conjunto de n elementos.

- $\sum_{i=0}^{n} i\binom{n}{i} = n \cdot 2^{n-1}$

  Ambas cuentan la suma de la cantidad de elementos de todos los subconjuntos de un conjunto de n elementos.

Algunas propiedades

Valen muchas igualdades para los números combinatorios:

- $\binom{n}{k} = \binom{n}{n-k}$
  Luego podemos elegir calcular el que requiera menos operaciones.
- $\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$
  Esto nos va a ayudar a calcular recursivamente todos los combinatorios.
- $\sum_{i=0}^{n} \binom{n}{i} = 2^n$
  Ambas cuentan la cantidad de subconjuntos de un conjunto de n elementos.
- $\sum_{i=0}^{n} i\binom{n}{i} = n \cdot 2^{n-1}$
  Ambas cuentan la suma de la cantidad de elementos de todos los subconjuntos de un conjunto de n elementos.

Algunas propiedades

Valen muchas igualdades para los números combinatorios:

- $\binom{n}{k} = \binom{n}{n-k}$

  Luego podemos elegir calcular el que requiera menos operaciones.

- $\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$

  Esto nos va a ayudar a calcular recursivamente todos los combinatorios.

- $\sum_{i=0}^{n} \binom{n}{i} = 2^n$

  Ambas cuentan la cantidad de subconjuntos de un conjunto de n elementos.

- $\sum_{i=0}^{n} i\binom{n}{i} = n \cdot 2^{n-1}$

  Ambas cuentan la suma de la cantidad de elementos de todos los subconjuntos de un conjunto de n elementos.

Triángulo de Pascal

Este es el triángulo de Pascal:

1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1
...

El k-ésimo número de la fila n representa el número $\binom{n}{k}$.

Con esto podemos precalcular todos los combinatorios en tiempo lineal.

Triángulo de Pascal

Este es el triángulo de Pascal:

$$\begin{array}{c} 1 \\ 1\ 1 \\ 1\ 2\ 1 \\ 1\ 3\ 3\ 1 \\ 1\ 4\ 6\ 4\ 1 \\ 1\ 5\ 10\ 10\ 5\ 1 \\ \cdots \end{array}$$

El k-ésimo número de la fila n representa el número $\binom{n}{k}$.

Con esto podemos precalcular todos los combinatorios en tiempo lineal.

Triángulo de Pascal

Este es el triángulo de Pascal:

$$
\begin{array}{c}
1 \\
1\ 1 \\
1\ 2\ 1 \\
1\ 3\ 3\ 1 \\
1\ 4\ 6\ 4\ 1 \\
1\ 5\ 10\ 10\ 5\ 1 \\
\cdots
\end{array}
$$

El k-ésimo número de la fila n representa el número $\binom{n}{k}$.

Con esto podemos precalcular todos los combinatorios en tiempo lineal.

Contenidos

También puede sernos útil este resultado:

Teorema de Lucas

Si $n = n_m \cdot p^m + \ldots + n_2 \cdot p^2 + n_1 \cdot p + n_0$ y
$k = k_m \cdot p^m + \ldots + k_2 \cdot p^2 + k_1 \cdot p + k_0$ entonces:

$$\binom{n}{k} \equiv \prod_{i=0}^{m} \binom{n_i}{k_i} \pmod{p}.$$

Miremos el caso $p = 2$:

- ¿Cuándo un combinatorio es impar?.
- ¿Cuántos combinatorios de la fila n del triángulo son impares?.

También puede sernos útil este resultado:

Teorema de Lucas

Si $n = n_m \cdot p^m + \ldots + n_2 \cdot p^2 + n_1 \cdot p + n_0$ y
$k = k_m \cdot p^m + \ldots + k_2 \cdot p^2 + k_1 \cdot p + k_0$ entonces:

$$\binom{n}{k} \equiv \prod_{i=0}^{m} \binom{n_i}{k_i} \pmod{p}.$$

Miremos el caso $p = 2$:

- ¿Cuándo un combinatorio es impar?.
- ¿Cuántos combinatorios de la fila n del triángulo son impares?.

También puede sernos útil este resultado:

Teorema de Lucas

Si $n = n_m \cdot p^m + \ldots + n_2 \cdot p^2 + n_1 \cdot p + n_0$ y
$k = k_m \cdot p^m + \ldots + k_2 \cdot p^2 + k_1 \cdot p + k_0$ entonces:

$$\binom{n}{k} \equiv \prod_{i=0}^{m} \binom{n_i}{k_i} \pmod{p}.$$

Miremos el caso $p = 2$:

- ¿Cuándo un combinatorio es impar?.
- ¿Cuántos combinatorios de la fila n del triángulo son impares?.

También puede sernos útil este resultado:

Teorema de Lucas

Si $n = n_m \cdot p^m + \ldots + n_2 \cdot p^2 + n_1 \cdot p + n_0$ y
$k = k_m \cdot p^m + \ldots + k_2 \cdot p^2 + k_1 \cdot p + k_0$ entonces:

$$\binom{n}{k} \equiv \prod_{i=0}^{m} \binom{n_i}{k_i} \pmod{p}.$$

Miremos el caso $p = 2$:

- ¿Cuándo un combinatorio es impar?.
- ¿Cuántos combinatorios de la fila n del triángulo son impares?.

Contenidos

Máximo Común Divisor

El MCD de dos enteros *a* y *b* es el mayor entero positivo que divide a ambos.

También es el menor entero positivo *d* que se puede escribir de la forma:

$$d = ax + by$$

con *x* e *y* enteros.

Se tiene que $mcd(a, b) = d \Leftrightarrow mcd(a, b - a) = d$

Máximo Común Divisor

El MCD de dos enteros *a* y *b* es el mayor entero positivo que divide a ambos.

También es el menor entero positivo *d* que se puede escribir de la forma:

$$d=ax+by$$

con *x* e *y* enteros.

Se tiene que $mcd(a, b) = d \Leftrightarrow mcd(a, b - a) = d$

Máximo Común Divisor

El MCD de dos enteros a y b es el mayor entero positivo que divide a ambos.

También es el menor entero positivo d que se puede escribir de la forma:

$$d=ax+by$$

con x e y enteros.

Se tiene que $mcd(a,b) = d \Leftrightarrow mcd(a, b-a) = d$

Algoritmo de Euclides

Si $a > b > 0$ aplicando sucesivamente la propiedad se tiene que $mcd(b, a\%b) = mcd(a, b)$.
Y ahora el máximo de ambos números decreció y $a\%b < b$.

Luego podemos repetir esto hasta que $b = 0$.
En este caso $mcd(a, 0) = a$.

Este proceso se conoce como algoritmo de Euclides.
La complejidad es $O(log(n))$, con $n = min(a, b)$.

Algoritmo de Euclides

Si $a > b > 0$ aplicando sucesivamente la propiedad se tiene que $mcd(b, a\%b) = mcd(a, b)$.
Y ahora el máximo de ambos números decreció y $a\%b < b$.

Luego podemos repetir esto hasta que $b = 0$.
En este caso $mcd(a, 0) = a$.

Este proceso se conoce como algoritmo de Euclides.
La complejidad es $O(log(n))$, con $n = min(a, b)$.

Algoritmo de Euclides

Si $a > b > 0$ aplicando sucesivamente la propiedad se tiene que $mcd(b, a\%b) = mcd(a, b)$.
Y ahora el máximo de ambos números decreció y $a\%b < b$.

Luego podemos repetir esto hasta que $b = 0$.
En este caso $mcd(a, 0) = a$.

Este proceso se conoce como algoritmo de Euclides.
La complejidad es $O(log(n))$, con $n = min(a, b)$.

Algoritmo de Euclides

Si $a > b > 0$ aplicando sucesivamente la propiedad se tiene que $mcd(b, a\%b) = mcd(a, b)$.
Y ahora el máximo de ambos números decreció y $a\%b < b$.

Luego podemos repetir esto hasta que $b = 0$.
En este caso $mcd(a, 0) = a$.

Este proceso se conoce como algoritmo de Euclides.
La complejidad es $O(log(n))$, con $n = min(a, b)$.

Algoritmo de Euclides

Si $a > b > 0$ aplicando sucesivamente la propiedad se tiene que $mcd(b, a\%b) = mcd(a, b)$.
Y ahora el máximo de ambos números decreció y $a\%b < b$.

Luego podemos repetir esto hasta que $b = 0$.
En este caso $mcd(a, 0) = a$.

Este proceso se conoce como algoritmo de Euclides.
La complejidad es $O(log(n))$, con $n = min(a, b)$.

Contenidos

La combinación lineal

También con este algoritmo podemos hallar un par x e y que cumplen con $mcd(a, b) = ax + by$.

Recursivamente, $mcd(a, 0) = a \cdot 1 + 0 \cdot 0$. Y si $mcd(b, a\%b) = bx + (a\%b)y$, llamamos $q = a \div b$.
Entonces
$mcd(a, b) = bx + (a\%b)y = bx + (a - qb)y = ay + b(x - qy)$.

Luego en $O(log(n))$ podemos hallar los x e y.

La combinación lineal

También con este algoritmo podemos hallar un par x e y que cumplen con $mcd(a, b) = ax + by$.

Recursivamente, $mcd(a, 0) = a \cdot 1 + 0 \cdot 0$. Y si $mcd(b, a\%b) = bx + (a\%b)y$, llamamos $q = a \div b$. Entonces $mcd(a, b) = bx + (a\%b)y = bx + (a - qb)y = ay + b(x - qy)$.

Luego en $O(log(n))$ podemos hallar los x e y.

La combinación lineal

También con este algoritmo podemos hallar un par x e y que cumplen con $mcd(a, b) = ax + by$.

Recursivamente, $mcd(a, 0) = a \cdot 1 + 0 \cdot 0$. Y si $mcd(b, a\%b) = bx + (a\%b)y$, llamamos $q = a \div b$.
Entonces
$mcd(a, b) = bx + (a\%b)y = bx + (a - qb)y = ay + b(x - qy)$.

Luego en $O(log(n))$ podemos hallar los x e y.

La combinación lineal

También con este algoritmo podemos hallar un par x e y que cumplen con $mcd(a, b) = ax + by$.

Recursivamente, $mcd(a, 0) = a \cdot 1 + 0 \cdot 0$. Y si $mcd(b, a\%b) = bx + (a\%b)y$, llamamos $q = a \div b$.
Entonces
$mcd(a, b) = bx + (a\%b)y = bx + (a - qb)y = ay + b(x - qy)$.

Luego en $O(log(n))$ podemos hallar los x e y.

La combinación lineal

También con este algoritmo podemos hallar un par x e y que cumplen con $mcd(a, b) = ax + by$.

Recursivamente, $mcd(a, 0) = a \cdot 1 + 0 \cdot 0$. Y si $mcd(b, a\%b) = bx + (a\%b)y$, llamamos $q = a \div b$.
Entonces
$mcd(a, b) = bx + (a\%b)y = bx + (a - qb)y = ay + b(x - qy)$.

Luego en $O(log(n))$ podemos hallar los x e y.

Contenidos

Primalidad

Un número entero positivo p es primo si tiene exactamente dos divisores, ó equivalentemente si $p > 1$ y los divisores de p son 1 y p.

Para verificar si un número p es primo basta buscar si hay un divisor de p entre 2 y $\sqrt{p}$ inclusive.

Esto es porque si d es un divisor de p, $\frac{p}{d}$ también lo es. Y alguno de los dos es menor a $\sqrt{p}$.
Luego podemos chequear si un número es primo en $O(\sqrt{p})$.

Primalidad

Un número entero positivo p es primo si tiene exactamente dos divisores, ó equivalentemente si $p > 1$ y los divisores de p son 1 y p.

Para verificar si un número p es primo basta buscar si hay un divisor de p entre 2 y $\sqrt{p}$ inclusive.

Esto es porque si d es un divisor de p, $\frac{p}{d}$ también lo es. Y alguno de los dos es menor a $\sqrt{p}$.
Luego podemos chequear si un número es primo en $O(\sqrt{p})$.

Primalidad

Un número entero positivo p es primo si tiene exactamente dos divisores, ó equivalentemente si $p > 1$ y los divisores de p son 1 y p.

Para verificar si un número p es primo basta buscar si hay un divisor de p entre 2 y $\sqrt{p}$ inclusive.

Esto es porque si d es un divisor de p, $\frac{p}{d}$ también lo es. Y alguno de los dos es menor a $\sqrt{p}$.
Luego podemos chequear si un número es primo en $O(\sqrt{p})$.

Primalidad

Un número entero positivo p es primo si tiene exactamente dos divisores, ó equivalentemente si $p > 1$ y los divisores de p son 1 y p.

Para verificar si un número p es primo basta buscar si hay un divisor de p entre 2 y $\sqrt{p}$ inclusive.

Esto es porque si d es un divisor de p, $\frac{p}{d}$ también lo es. Y alguno de los dos es menor a $\sqrt{p}$.
Luego podemos chequear si un número es primo en $O(\sqrt{p})$.

Criba de Eratóstenes

Si queremos verificar muchos números, podemos hacer una criba para calcular los primos.

El procedimiento es el siguiente:

- Creamos un arreglo con los números de 1 a N.
- Inicialmente todos son primos menos el 1.
- Recorremos los números en orden creciente desde el 2.
- Para cada número que sea primo marcamos todos sus múltiplos como no primos.

Con esto encontramos todos los primos entre 1 y N en $O(Nlog(N))$.

Criba de Eratóstenes

Si queremos verificar muchos números, podemos hacer una criba para calcular los primos.

El procedimiento es el siguiente:

- Creamos un arreglo con los números de 1 a N.
- Inicialmente todos son primos menos el 1.
- Recorremos los números en orden creciente desde el 2.
- Para cada número que sea primo marcamos todos sus múltiplos como no primos.

Con esto encontramos todos los primos entre 1 y N en $O(Nlog(N))$.

Criba de Eratóstenes

Si queremos verificar muchos números, podemos hacer una criba para calcular los primos.

El procedimiento es el siguiente:

- Creamos un arreglo con los números de 1 a N.
- Inicialmente todos son primos menos el 1.
- Recorremos los números en orden creciente desde el 2.
- Para cada número que sea primo marcamos todos sus múltiplos como no primos.

Con esto encontramos todos los primos entre 1 y N en $O(Nlog(N))$.

Criba de Eratóstenes

Si queremos verificar muchos números, podemos hacer una criba para calcular los primos.

El procedimiento es el siguiente:

- Creamos un arreglo con los números de 1 a N.
- Inicialmente todos son primos menos el 1.
- Recorremos los números en orden creciente desde el 2.
- Para cada número que sea primo marcamos todos sus múltiplos como no primos.

Con esto encontramos todos los primos entre 1 y N en $O(Nlog(N))$.

Criba de Eratóstenes

Si queremos verificar muchos números, podemos hacer una criba para calcular los primos.

El procedimiento es el siguiente:

- Creamos un arreglo con los números de 1 a N.
- Inicialmente todos son primos menos el 1.
- Recorremos los números en orden creciente desde el 2.
- Para cada número que sea primo marcamos todos sus múltiplos como no primos.

Con esto encontramos todos los primos entre 1 y N en $O(Nlog(N))$.

Criba de Eratóstenes

Si queremos verificar muchos números, podemos hacer una criba para calcular los primos.

El procedimiento es el siguiente:

- Creamos un arreglo con los números de 1 a N.
- Inicialmente todos son primos menos el 1.
- Recorremos los números en orden creciente desde el 2.
- Para cada número que sea primo marcamos todos sus múltiplos como no primos.

Con esto encontramos todos los primos entre 1 y N en $O(Nlog(N))$.

Contenidos

Extendiendo la criba

Podemos obtener más información de la criba que si un número es primo ó no.
Por ejemplo, en lugar de marcar un 0 a los números que no son primos, podemos guardar un primo que los divide.

Recordemos que un entero positivo n se escribe de una única forma como:

$$n = p_1^{e_1} \cdot p_2^{e_2} \cdot \ldots \cdot p_k^{e_k}.$$

De esta forma podemos encontrar esta factorización para cualquier número n entre 1 y N dividiendo sucesivamente por el primo que me da la criba.

Y la complejidad de esto es $O(log(n))$.

Extendiendo la criba

Podemos obtener más información de la criba que si un número es primo ó no.
Por ejemplo, en lugar de marcar un 0 a los números que no son primos, podemos guardar un primo que los divide.

Recordemos que un entero positivo n se escribe de una única forma como:

$$n = p_1^{e_1} \cdot p_2^{e_2} \cdot \ldots \cdot p_k^{e_k}.$$

De esta forma podemos encontrar esta factorización para cualquier número n entre 1 y N dividiendo sucesivamente por el primo que me da la criba.

Y la complejidad de esto es $O(log(n))$.

Extendiendo la criba

Podemos obtener más información de la criba que si un número es primo ó no.
Por ejemplo, en lugar de marcar un 0 a los números que no son primos, podemos guardar un primo que los divide.

Recordemos que un entero positivo n se escribe de una única forma como:

$$n = p_1^{e_1} \cdot p_2^{e_2} \cdot \ldots \cdot p_k^{e_k}.$$

De esta forma podemos encontrar esta factorización para cualquier número n entre 1 y N dividiendo sucesivamente por el primo que me da la criba.

Y la complejidad de esto es $O(log(n))$.

Extendiendo la criba

Podemos obtener más información de la criba que si un número es primo ó no.
Por ejemplo, en lugar de marcar un 0 a los números que no son primos, podemos guardar un primo que los divide.

Recordemos que un entero positivo n se escribe de una única forma como:

$$n = p_1^{e_1} \cdot p_2^{e_2} \cdot \ldots \cdot p_k^{e_k}.$$

De esta forma podemos encontrar esta factorización para cualquier número n entre 1 y N dividiendo sucesivamente por el primo que me da la criba.

Y la complejidad de esto es $O(log(n))$.

Extendiendo la criba

Podemos obtener más información de la criba que si un número es primo ó no.
Por ejemplo, en lugar de marcar un 0 a los números que no son primos, podemos guardar un primo que los divide.

Recordemos que un entero positivo n se escribe de una única forma como:

$$n = p_1^{e_1} \cdot p_2^{e_2} \cdot \ldots \cdot p_k^{e_k}.$$

De esta forma podemos encontrar esta factorización para cualquier número n entre 1 y N dividiendo sucesivamente por el primo que me da la criba.

Y la complejidad de esto es $O(log(n))$.

Funciones multiplicativas

Las funciones multiplicativas son aquellas funciones de naturales en naturales tal que si m y n son coprimos $f(nm) = f(n)f(m)$.

Algunas de estas funciones son:
$d(n) = \prod_{i=1}^{k}(e_i + 1)$, la cantidad de divisores de un número.
$\sigma(n) = \prod_{i=1}^{k} \frac{p^{e_i+1}-1}{p-1}$, la suma de los divisores de un número.
$\phi(n) = \prod_{i=1}^{k}(p^e - p^{e-1})$, la cantidad de números menores que n coprimos con n.

Todas estas funciones se pueden calcular en $O(log(n))$ gracias a la criba extendida.

Funciones multiplicativas

Las funciones multiplicativas son aquellas funciones de naturales en naturales tal que si m y n son coprimos $f(nm) = f(n)f(m)$.

Algunas de estas funciones son:
$d(n) = \prod_{i=1}^{k}(e_i + 1)$, la cantidad de divisores de un número.
$\sigma(n) = \prod_{i=1}^{k} \frac{p^{e_i+1}-1}{p-1}$, la suma de los divisores de un número.
$\phi(n) = \prod_{i=1}^{k}(p^e - p^{e-1})$, la cantidad de números menores que n coprimos con n.

Todas estas funciones se pueden calcular en $O(log(n))$ gracias a la criba extendida.

Funciones multiplicativas

Las funciones multiplicativas son aquellas funciones de naturales en naturales tal que si m y n son coprimos $f(nm) = f(n)f(m)$.

Algunas de estas funciones son:
$d(n) = \prod_{i=1}^{k}(e_i + 1)$, la cantidad de divisores de un número.
$\sigma(n) = \prod_{i=1}^{k} \frac{p^{e_i+1}-1}{p-1}$, la suma de los divisores de un número.
$\phi(n) = \prod_{i=1}^{k}(p^e - p^{e-1})$, la cantidad de números menores que n coprimos con n.

Todas estas funciones se pueden calcular en $O(log(n))$ gracias a la criba extendida.

Funciones multiplicativas

Las funciones multiplicativas son aquellas funciones de naturales en naturales tal que si m y n son coprimos $f(nm) = f(n)f(m)$.

Algunas de estas funciones son:
$d(n) = \prod_{i=1}^{k}(e_i + 1)$, la cantidad de divisores de un número.
$\sigma(n) = \prod_{i=1}^{k} \frac{p^{e_i+1}-1}{p-1}$, la suma de los divisores de un número.
$\phi(n) = \prod_{i=1}^{k}(p^e - p^{e-1})$, la cantidad de números menores que n coprimos con n.

Todas estas funciones se pueden calcular en $O(log(n))$ gracias a la criba extendida.

Funciones multiplicativas

Las funciones multiplicativas son aquellas funciones de naturales en naturales tal que si m y n son coprimos $f(nm) = f(n)f(m)$.

Algunas de estas funciones son:
$d(n) = \prod_{i=1}^{k}(e_i + 1)$, la cantidad de divisores de un número.
$\sigma(n) = \prod_{i=1}^{k} \frac{p^{e_i+1}-1}{p-1}$, la suma de los divisores de un número.
$\phi(n) = \prod_{i=1}^{k}(p^e - p^{e-1})$, la cantidad de números menores que n coprimos con n.

Todas estas funciones se pueden calcular en $O(log(n))$ gracias a la criba extendida.