3N1?M4

Team Extended Reference

Notes on Algorithms and Contents

May 11, 2019

Contents

1 Math. Teoremas

1.1 Representación de Números Enteros. Teoremas

- **Conjetura de Goldbach:** Todo número par $n \geq 4$ puede ser expresado como la suma de dos números primos.
- Todo número natural n puede representarse como la suma de números distintos menores que s, donde $s = \lceil \frac{\sqrt{1+8n}-1}{2} \rceil$, o sea, s es la base del menor número triangular que es mayor o igual que n. Es más, sea $k = (\sum_{i=1}^{s} i) - n$, tenemos que $k \in [1, s]$. Entonces n puede ser representado como la suma de todos los números desde uno hasta s, excluyendo a k.
- **Teorema de Fermat sobre la representacion como suma de cuadrados:** Un número primo p se puede expresar de la forma $p = a^2 + b^2$ si y solo si se puede expresar de la forma $p = 4c + 1$
- Un número n puede ser representado como la suma de dos cuadrados si es el producto de dos números que pueden ser expresados como la suma de dos cuadrados.
- Un entero n puede ser expresado como suma de dos cuadrados si y solo si cada divisor primo de n de la forma $4k + 3$ tiene exponente par en la representación canónica de n.
- Si p es un primo impar que divide a $a^2 + b^2$ con $gcd(a, b) = 1$, entonces $p \equiv 1 (mod\ 4)$
- Cualquier entero positivo puede ser expresado como la suma de tres cuadrados enteros positivos si y solo si no es de la forma $4^n(8k + 7)$, donde n y k son enteros no negativos.
- (Lagrange, 1770): Cada entero positivo puede ser representado como la suma de cuatro o menos cuadrados enteros.

1.2 Teoría de Números. Teoremas

- **Teorema de los Números Primos:** La cantidad de numeros primos menores o iguales a n está acotada por $O\left(\frac{n}{\ln(n)-1}\right)$
- **Teorema de Euler:** Dados $a, n \in N$ tales que $gcd(a, n) = 1$, se tiene que:

$$a^{\varphi(n)} \equiv 1 (mod\ n)$$

- **Pequeño Teorema de Fermat:** Sea p primo y a un entero no divisible por p, entonces $a^{p-1} \equiv 1 \pmod{p}$
- **Teorema de Wilson:** Si p es primo, entonces $(p-1)! \equiv -1 \pmod{p}$. El recı'proco de este teorema también es verdadero.
- Si a es un entero y p es un primo que no divide a a, entonces a^{p-2} es un inverso de a módulo p.

1.3 Función Phi de Euler

- La función $\varphi(n)$ es igual a la cantidad de números naturales menores o iguales a n que son coprimos con él. Por definición $\varphi(1) = 1$ y $\varphi(p) = p - 1$ si p es primo.

- La función φ es multiplicativa, o sea:

$$\varphi(n \cdot m) = \varphi(n) \cdot \varphi(m)$$

  si $gcd(n, m) = 1$

- $\varphi(p^k) = (p-1)p^{k-1}$, con p primo y k entero.

- $\sum_{d|n} \varphi(d) = n$

- **Producto de Euler:**

$$\varphi(n) = n\left(1 - \frac{1}{p_1}\right)\left(1 - \frac{1}{p_2}\right)\cdots\left(1 - \frac{1}{p_k}\right)$$

  si n es representado canónicamente como $n = p_1^{e_1}p_2^{e_2}\cdots p_k^{e_k}$ con $p_1, p_2, .., p_k$ primos distintos.

1.4 Otras Funciones de Teoría de Números

Sea $p_1^{e_1}p_2^{e_2}\cdots p_k^{e_k}$ la representación canónica de n:

- **Cantidad de Factores Primos:** Es igual a $e_1 + e_2 + \cdots + e_k$. Un entero n tiene hasta $\log n$ factores primos

- **Cantidad de Divisores:** n tiene $\tau(n) = (e_1 + 1)(e_2 + 1)\cdots(e_k + 1)$ divisores. El número en el rango $[1, 10^6]$ con más divisores es 720720 con 240 y para ese rango

$$\sum_{i=1}^{10^6} \tau(i) = 13970033 \approx 14M$$

- **Suma de los Divisores:** Es igual a

$$\sigma(n) = \left(\frac{p_1^{e_1+1} - 1}{p_1 - 1}\right)\left(\frac{p_2^{e_2+1} - 1}{p_2 - 1}\right)\cdots\left(\frac{p_k^{e_k+1} - 1}{p_k - 1}\right)$$

- **Producto de los Divisores:**

$$\pi(n) = n^{\frac{1}{2}\tau(n)}$$

1.5 Congruencias. Propiedades

- Si m es un entero positivo y a, b y c son enteros, entonces:
  - $a \equiv a \pmod m$
  - $a \equiv b \pmod m$ si y solo si $b \equiv a \pmod m$
  - si $a \equiv b \pmod m$ y $b \equiv c \pmod m$ entonces $a \equiv c \pmod m$

  Por consiguiente la congruencia es una relación de equivalencia.

- Si $m \in \mathbb{Z}_+$ y $a, b \in \mathbb{Z}$ con $a \equiv b \pmod m$ entonces $gcd(a, m) = gcd(b, m)$.

- Si $a \equiv b \pmod m$ entonces $a + c \equiv b + c \pmod m$, $a - c \equiv b - c \pmod m$ y $ac \equiv bc \pmod m$.

- Si $a \equiv b \pmod m$ y $c \equiv d \pmod m$ entonces $ac \equiv bd \pmod m$.

- Si $d = gcd(c, m)$, y $ac \equiv bc \pmod m$, entonces $a \equiv b \pmod{\frac{m}{d}}$. En particular, si c y m son primos relativos, entonces $a \equiv b \pmod m$.

- Si $k \in \mathbb{Z}_+$ y $a \equiv b \pmod m$ entonces $a^k \equiv b^k \pmod m$.

- Si $f(x_1, x_2, \cdots, x_n)$ es un polinomio con coeficientes enteros, y $a_1, a_2, \cdots, a_n, b_1, b_2, \cdots, b_n$ son enteros que cumplen que $a_i \equiv b_i \pmod m$ para toda $i \in [1, n]$, entonces $f(a_1, a_2, \cdots, a_n) \equiv f(b_1, b_2, \cdots, b_n) \pmod m$.

- Si todos los m_i, con $i \in [1, n]$, son coprimos par a par, y $a \equiv b \pmod{m_i}$ para toda i, entonces $a \equiv b \pmod{m_1 m_2 \cdots m_n}$.

- Un inverso de a módulo m, donde $gcd(a, m) = 1$, puede ser encontrado usando el algoritmo extendido de Euclides para encontrar los enteros x, y tales que $ax + my = 1$. lo que implica que x es un inverso de a módulo m.

2 Math. Métodos

2.1 Josephus Problem

- El problema se trata de n personas sentadas en cículo. Comenzando por el primer jugador, se comienza a contar hasta k. El k-ésimo jugador es eliminado. Se comienza a contar por el siguiente jugador, hasta k con los jugadores que quedan. Así hasta un solo jugador se mantenga dentro del juego. Hay que calcular, dados n y k, quién queda de último.

- **Caso Especial:** Cuando $k = 2$, usando la representación binaria del número $n = \overline{1b_1b_2\cdots b_k}$, entonces la respuesta es $\overline{b_1b_2\cdots b_k1}$, es decir, movemos el bit más significativo de n para hacerlo el menos significativo. Esto puede hacerse en $O(1)$.

- Sea $f(n,k)$ la posición del sobreviviente para un círculo de tamanño n y saltando de k en k. Después de que la k-ésima persona es eliminada, el tamanño de círculo disminuye a $n-1$, y la posición del sobreviviente es $f(n-1,k)$. La relación recurrente es

$$f(n,k) = (f(n-1,k)+k)\%n$$

El caso base es cuando $n=1$, donde tenemos que $f(1,k)=0$.

2.2 Teorema Chino del Resto

- **Teorema:** Sean $m_1, m_2, \cdots m_n$ enteros mayores que 1 y primos relativos entre sí, y $a_1, a_2, \cdots a_n$ enteros cualesquiera, entonces existe una solución x para las congruencias simultáneas

$$\begin{cases} x \equiv a_1 (mod\ m_1) \\ x \equiv a_2 (mod\ m_2) \\ \cdots\cdots\cdots\cdots\cdots \\ x \equiv a_n (mod\ m_n) \end{cases}$$

si x y x' son dos soluciones, estonces $x \equiv x' (mod\ M)$ donde $M = m_1 m_2 \cdots m_n$

- **Métodos con Módulos No Necesariamente Coprimos:**
Para dos ecuaciones tenemos

$$\begin{cases} x \equiv a_1 (mod\ m_1) \\ x \equiv a_2 (mod\ m_2) \end{cases}$$

Aquí m_1 y m_2 no necesariamente son coprimos.
Este sistema de congruencias implica que

$$\begin{cases} x = a_1 + m_1 k_1 \\ x = a_2 + m_2 k_2 \end{cases}$$

para unos enteros k_1, k_2.
Igualando los miembros derechos de la ecuaciones, tenemos que

$$a_1 + m_1 k_1 = a_2 + m_2 k_2$$

$$m_1(-k_1) + m_2 k_2 = a_1 - a_2$$

Ya que conocemos los valores de a_1, a_2, m_1 y m_2, solo hay que resolver una ecuación diofántica. Sea $d = gcd(m_1, m_2)$, sabemos que divide el primer miembro de la ecuación, así que para que haya soluciones, debe dividir también al segundo.
Usando el Algoritmo Extendido de Euclides, podemos encontrar un par (x', y') tal que $m_1 x' + m_2 y' = d$. Luego multiplicamos ambos miembros por $\frac{a_1 - a_2}{d}$, obteniendo

$$m_1 x' \frac{a_1 - a_2}{d} + m_2 y' \frac{a_1 - a_2}{d} = a_1 - a_2$$

Entonces $k_1 = -x' \frac{a_1 - a_2}{d}$ y $k_2 = y' \frac{a_1 - a_2}{d}$ Ahora podemos sustituir k_1 en el segundo sistema para obtener

$$x = a_1 + x' \frac{a_2 - a_1}{d} m_1$$

- **Infinitas Soluciones**
Digamos que tenemos dos soluciones x_1 y x_2. Ahora tenemos que $x_1 \equiv a_1 (mod\ m_1)$ y que $x_2 \equiv a_1 (mod\ m_1)$. Entonces por transitividad de congruencias tenemos que $x_1 \equiv x_2 (mod\ m_1)$ y haciendo lo mismo para m_2, $x_1 \equiv x_2 (mod\ m_2)$.
Esas dos congruencias son equivalentes a $x_1 \equiv x_2 (mod\ lcm(m_1, m_2))$ así que las infinitas soluciones son $x = x_0 + k \cdot lcm(m_1, m_2)$ donde x_0 es la solución original del sistema y $k \in \mathbb{Z}$

- **Implementación:**
Ya que los números pueden llegar a ser un poco grandes, debemos hacer los cálculos módulo $l = lcm(m_1, m_2)$. Entonces

$$x = \left(a_1 + \left(\left(x' \cdot \frac{a_1 - a_1}{d} \right) mod\ l \right) \cdot m_1\ mod\ l \right) mod\ l$$

Si $m_1, m_2 \leq 10^9$, entonces $lcm(m_1, m_2) \leq 10^{18}$, $x' \leq 10^9$, $\frac{a_2 - a_1}{d} \leq 10^9$ Podemos calcular $x' \cdot \frac{a_2 - a_1}{d}$, pero ya que eĺ y l pueden ser hasta 10^{18}, la multiplicación final por m_1 causará un desbordamiento. Para manejar este problema usaremos la propiedad:

$$ca \bmod cb = c(a \bmod b)$$

Sabiendo que $lcm(m_1, m_2) = \frac{m_2}{d} \cdot m_1$ y tomando en la propiedad anterior $a = x' \cdot \frac{a_2 - a_1}{d}$, $b = \frac{m_2}{d}$ y $c = m_1$, tenemos que $\left(x' \cdot \frac{a_2 - a_1}{d} \bmod l\right) \cdot m_1 \bmod l$ es igual a:

$$\left(x' \cdot \frac{a_2 - a_1}{d} \bmod \frac{m_2}{d} \right) \cdot m_1 \bmod l$$

Ya que $\frac{m_2}{d} \leq 10^9$, es problema de desbordamiento está solucionado.

- **Con Más de Dos Relaciones de Congruencias:**
Digamos que tenemos t congruencias de la forma

$$x \equiv a_i \bmod m_i, \text{ para } i = 1, 2, \cdots, t$$

Entonces podemos mezclar las relaciones una por una. Luego de resolver el sistema formado por las primeras dos relaciones, obtenemos una nueva de la forma

$$x \equiv s \bmod lcm(m_1, m_2)$$

Ahora podemos mezclarla con la tercera, y así sucesivamente... Este algoritmo tiene una complejidad de $O(t \cdot \log lcm(m_1, m_2, \cdots, m_t))$

2.3 Resolución de Ecuaciones Recursivas

Recurrencia Lineal Homogénea:

- La forma general es $a_0 t_n + a_1 t_{n1} + \cdots + a_k t_{nk} = 0$
- Su ecuación característica es $a_0 x^k + a_1 x^{k1} + \cdots + a^k = 0$
- Suponiendo que $r_1, r_2, r_3, .., r_k$ son k raices distintas de la ecuación característica, la solución será $t_n = \sum_{i=1}^{k} c_i r_i^n$ donde las k constantes $c_1, c_2, c_3, .., c_k$ se pueden hallar sustituyendo t_n por los casos base y hallando las soluciones del sistema resultante.
- Cuando las raíces no son todas diferentes, si m_i es la multiplicidad de la raíz r_i, entonces la solución tiene la foma $t_n =$ donde todas las c, son potencialmente diferentes, y pueden ser indexadas como $c_1, c_2, .., c_k$

Rec. Lineal No Homogenea y Miembro Derecho de la Forma $b^n P(n)$

- La forma general es $a_0 t_n + a_1 t_{n1} + \cdots + a_k t_{nk} = b^n P(n)$ donde b es una constante y $P(n)$ es un polinomio de grado d.
- La ecuacion característica es $a_0 x^k + a_1 x^{k1} + \cdots + a^k = 0$

2.4 Resolución de Ecuaciones Recursivas con Matrices

- **Definición:** Este método resuelve relaciones de recurrencias lineares, o sea, de la forma

$$f(i) = c_1 \cdot f(i-1) + c_2 \cdot f(i-2) + \cdots + c_k \cdot f(i-k) + d$$

donde k es el mayor entero tal que $f(i)$ depende de $f(i-k)$ y d es una constante.

- **Notación:**
  - F_i es el vector columna de tamaño $k+1$, formado por los valores $f(i), f(i+1), \cdots, f(i+k-1), d$.
- **Método de Resolución:**
  - Determinar k y los valores iniciales de la recursión. Si el número de estos es menor que k hay que calcular los restantes a partir de los que ya se tienen. Se forma el vector F_1, de tamaño $k+1$:

$$F_1 = \begin{pmatrix} f(1) \\ f(2) \\ \vdots \\ f(k) \\ d \end{pmatrix}$$

  - Construir la matriz de transformación T: una matriz cuadrada de lado $k+1$, tal que $T \cdot F_i = F_{i+1}$
    Sean $c_1, c_2, \cdots, c_k$ los coeficientes de la ecuación recursiva, la matriz T tiene la forma:

$$T = \begin{pmatrix} 0 & 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ c_k & c_{k-1} & c_{k-2} & c_{k-3} & \cdots & c_1 & 1 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 1 \end{pmatrix}$$

- Calculamos F_n de la siguiente forma:

$$F_n = T^{n-1} F_1$$

- **Condición Dependiente de la Paridad:** La función se comporta de forma diferente de acuerdo a la paridad del argumento. Construimos dos matrices T_{par} y T_{impar} tales que:

$$T_{par} \cdot F_i = F_{i+1}, \text{ para i par}$$

$$T_{impar} \cdot F_i = F_{i+1}, \text{ para i impar}$$

Ahora construimos también la matriz $T = T_{par} \cdot T_{impar}$. Esta es tal que podemos calcular F_n como:

$$F_n = T^{\lfloor n/2 \rfloor} \cdot F_1, \text{ si n es impar}$$

$$F_n = T_{impar} \cdot T^{\lfloor (n-1)/2 \rfloor} \cdot F_1, \text{ de otra forma}$$

Nótese que T_{par} y T_{impar} deben ser del mismo tamaño. Si no lo son, se deben agregar coeficientes cero donde corresponda.

3 Math. Fórmulas

3.1 Fómulas de Sumas

- $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$
- $\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$
- $\sum_{i=1}^{n} i^3 = \left(\sum_{i=1}^{n} i\right)^2 = \left(\frac{n(n+1)}{2}\right)^2$
- $\sum_{i=1}^{n} i^4 = \frac{6n^5+15n^4+10n^3-n}{30}$
- $\sum_{i=1}^{n} i^5 = \frac{2n^6+6n^5+5n^4-n^2}{12}$
- $\sum_{i=1}^{n} i_{par} = n(n+1)$
- $\sum_{i=1}^{n} i_{impar} = \left(\frac{n+1}{2}\right)^2$
- $\sum_{i=1}^{n} r^i = \frac{r^{n+1}-1}{r-1}$
- $\sum_{i=1}^{n} \frac{1}{i^k} = H_n^k$ donde H_n^k es un número armónico generalizado
- Las siguientes son dos fórmulas para la suma de potencias con igual exponente y diferente base:

$$\sum_{i=1}^{n} i^p = \frac{(n+1)^{p+1}}{p+1} + \sum_{k=1}^{p} \frac{B_k}{p-k+1} C_k^p (n+1)^{p-k+1}$$

donde $B_k = -\sum_{i=0}^{k-1} C_i^k \frac{B_i}{k+1-i}$ y $B_0 = 1$

Esta otra usa números de Stirling de segundo tipo:

$$\sum_{i=1}^{n} i^p = \sum_{r=1}^{p} \left\{ \begin{matrix} p \\ r \end{matrix} \right\} r! \binom{n+1}{r+1}$$

- Familia de fórmulas de suma ia^i:

$$\sum_{i=0}^{n-1} ia^i = \frac{a - na^n + a^{n+1}(n-1)}{(1-a)^2}$$

$$\sum_{i=0}^{n-1} i2^i = 2^n(n-2) + 2$$

que es un caso especial para $a = 2$

- Propiedades de los logaritmos y las potencias generalizadas:

$$\sum_{i=s}^{t} \ln f(n) = \ln \prod_{n=s}^{t} f(n)$$

$$c^{\left(\sum_{i=s}^{t} f(n)\right)} = \prod_{n=s}^{t} c^{f(n)}$$

3.2 Trabajo con Bits

- Apaga el 1-bit más a la derecha, produciendo 0 si no hay (ej. 01011000 se convierte en 01010000):

$$x \wedge (x-1)$$

- Deja encendido solo el 0-bit más a la derecha, produciendo 0 si no hay (ej. 10100111 se convierte en 00001000):

$$\neg x \wedge (x+1)$$

- Propaga hacia la derecha el 1-bit más a la derecha, produciendo todos 1's si no hay (ej., 01011000 se convierte en 01011111):

$$x \vee (x-1)$$

- Apaga la cadena más a la derecha de 1-bits (ej. 01011000 se convierte en 01000000):

$$((x \vee (x-1)) + 1) \wedge x$$

4 Combinatoria

4.1 Conceptos Básicas

- **Variaciones sin Repeticion:** Se dan n objetos. Se forman todas las distribuciones ordenadas de los k objetos. El número de distribuciones es

$$V_k^n = \frac{n!}{(n-k)!}$$

- **Variaciones con Repetición:** Se dan objetos que pertenecen a n formas distintas. Se forman todas las distribuciones ordenadas de k objetos en las cuales pueden figurar objetos repetidos. Dos distribuciones se consideran iguales si en cada posición tienen objetos iguales respectivamente. El nuḿero de estas distribuciones es

$$\overline{V}_k^n = n^k$$

- **Permutaciones sin Repetición:** Se dan n objetos. Se forman todas las distribuciones ordenadas de los n objetos. El número de distribuciones es

$$P_n = n!$$

- **Permutaciones con Repetición:** Se dan n objetos que pertenecen a k clases. Se forman todas las distribuciones ordenadas de los n objetos. Los objetos de una misma clase son iguales y su cantidad es $n_1, n_2, \cdots, n_k$ respectivamente, de tal forma que $n_1 + n_2 + \cdots + n_k = n$. El número de distribuciones es

$$P_(n_1, n_2, \cdots n_k) = \frac{n!}{n_1! n_2! \cdots n_k!}$$

- **Combinaciones sin Repetición:** Se dan n objetos y se quieren separar en dos grupos, uno de k objetos y el otro de $n-k$. El número de maneras de hacerlo es

$$C_k^n = \frac{n!}{k!(n-k)!}$$

- **Combinaciones con Repetición:** Se tienen objetos de n tipos diferentes. De cuántas formas podemos tomar k objetos sin importar que tomemos objetos de un mismo tipo.

$$\overline{C}_k^n = C_k^{n+k-1}$$

4.2 Fórmulas

- **Propiedades de los Coeficientes Binomiales**
  - $C_k^n = C_{n-k}^n$
  - $C_0^n + C_1^n + C_2^n + \cdots + C_n^n = 2^n$
  - $C_0^n - C_1^n + C_2^n - \cdots + (-1)^n C_n^n = 0$
  - $\sum_{i=0}^{n} -1 = C_{k+1}^n$
  - $C_n^n + C_n^n + 1 + C_n^n + 2 + \cdots + C_n^n + m - 1 = C_{n+1}^{n+m}$
- **Propiedades de los Factoriales y las Permutaciones**
  - $n! = (n-1)\big[(n-1)! + (n-2)!\big]$
  - $\sum_{n_1, n_2, \cdots n_k} P_(n_1, n_2, \cdots n_k) = k^n$
- **Subfactoriales**
  - $D_n = P_n - C_1^n P_{n-1} + C_2^n P_{n-2} - \cdots + (-1)^n C_n^n$
  - $D_n = n!\big[1 - \frac{1}{1!} + \frac{1}{2!} - \cdots + \frac{(-1)^n}{n!}\big]$
  - $D_n = (n-1)[D_{n-1} + D_{n-2}]$ con $D_0 = 1$ y $D_1 = 0$
  - $D_n = n \cdot D_{n-1} + (-1)^n$

5 Secuencias

5.1 Números de Catalan

- $cat(n)$ cuenta:
  - el número de expresiones que contienen n pares de paréntesis que están correctamente parentizados.
  - la cantidad de formas en las que $n+1$ factores pueden ser parentizados.
  - el número de formas en las que un polígono de $n+2$ lados puede ser triangulado
  - la cantidad de caminos monótonicos a través de los bordes de las casillas de una matriz de $n \times n$ y que no van a través de su diagonal.
  - el número de formas de unir $2n$ puntos sobre una circunferencia a la misma distancia, de forma que las cuerdas no se intersecten.
  - la de formas de colocar los números $1, 2, .., 2n$ en una matriz monotona de $2 \times n$. Una matriz es monotona si los valores crecen dentro de cada columna y dentro de cada fila.
  - Supóngase que se tira una moneda $2n$ veces y que el resultado es "head" exactamente n veces y "tail" exactamente n veces. El número de secuencias de lances en las que el número acumulado de heads es siempre al menos tan grande como el número acumulado de tails es $cat(n)$. Por ejemplo, para $n = 3$ tenemos HTHTHT, HTHHTT, HHTTHT, HHTHTT, HHHTTT
  - En el ejemplo anterior, la cantidad de secuencias de lances en las que el número acumulado de heads siempre excede el número acumulado de tails hasta el último lance es $cat(n-1)$. Para $n = 3$ tenemos HHTHTT, HHHTTT.
- Dos fórmulas para calcularlos: una cerrada y la otra recursiva:

$$cat(n) = \frac{C_n^{2n}}{n+1}$$

$$cat(n+1) = \frac{4n+2}{n+2} cat(n), cat(0) = 1$$

5.2 Números de Fibonacci

- **Teorema de Zeckendorf:** Cada entero positivo puede ser escrito de forma única como la suma de uno o más números de Fibonnacci tal que la suma no incluya ningún par de números de Fibonacci consecutivos.
- **Período de Pisano:** la(s) ultima(s) una/dos/tres/cuatro cifra(s) de un número de Fibonacci se repiten con un período de 60/300/1500/15000, respectivamente.

- **Fórmula de Binet:**

$$F_n = \frac{1}{\sqrt{5}}\left(\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n\right)$$

- **Identidades:**
  - $\sum_{i=1}^{n} F_i = F_{n+1} - 1$
  - $F_{n+m} = F_{m+1}F_n + F_m F_{n-1}$
  - $F_m F_{n+1} - F_{m+1}F_n = (-1)^n F_{m-n}$
  - $F_{2n} = F_n^2 + 2F_n F_{n-1}$
  - $F_0^2 + F_1^2 + F_2^2 + \cdots + F_n^2 = F_n F_{n+1}$
  - $gcd(F_n, F_m) = F_{gcd(n,m)}$
  - Para calcular el n-ésimo término en tiempo logarítmico

$$\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}^n = \begin{bmatrix} F_{n+1} & F_n \\ F_n & F_{n-1} \end{bmatrix}$$

5.3 Stirling Numbers de Primer Tipo

- El número de Stirling de primer tipo $\left[{n \atop k}\right]$ cuenta la cantidad de formas de organizar n objetos en k ciclos. Estos ciclos son permutaciones cíclicas, como los collares. O sea, si $[A, B, C, D]$ es un ciclo, entonces $[A, B, C, D] = [B, C, D, A] = [C, D, A, B] = [D, A, B, C]$
- Se cumple que:
  - $\left[{n \atop 1}\right] = (n-1)!$ para $n > 0$
  - $\left[{n \atop k}\right] \geq \left\{{n \atop k}\right\}$ para enteros $n, k \geq 0$
- Cada organización de n objetos en k ciclos, o coloca el último objeto en un ciclo él solo (de $\left[{n-1 \atop k-1}\right]$ maneras), o inserta este en una de los $\left[{n-1 \atop k}\right]$ configuraciones de los primeros $n-1$ objetos. En el segundo caso hay $n-1$ maneras de hacer la inserción. (No es difícil verificar que hay j formas de hacer una inserción en un ciclo de tamanño j; y sumando todas las jotas da un total de $n-1$ formas) De aquí:

$$\left[{n \atop k}\right] = (n-1)\left[{n-1 \atop k}\right] + \left[{n-1 \atop k-1}\right]$$

5.4 Stirling Numbers de Segundo Tipo

- El número de Striling de segundo tipo $\left\{{n \atop k}\right\}$ cuenta el número de formas de particionar un conjunto de n objetos en k subconjuntos no vacíos. Hay solo una forma de poner n elementos en un solo conjunto no vacío, y solo una forma de distribuirlos en n subconjuntos, por tanto, $\left\{{n \atop 1}\right\} = \left\{{n \atop n}\right\} = 1$ para toda $n > 0$.

  Por otro lado $\left\{{0 \atop 1}\right\} = 0$ porque tal subconjunto está vacío.

  Por acuerdo se toma $\left\{{0 \atop 0}\right\} = 1$; y como un conjunto no vacío necesita al menos una parte $\left\{{n \atop 0}\right\} = 0$ para todo $n > 0$.

- Dado un conjunto de $n > 0$ elementos para ser particionados en k subconjuntos no vacíos de $\left\{{n \atop k}\right\}$ maneras, podemos, o poner el último objeto en un nuevo subconjunto formado por él mismo (esto se puede hacer de $\left\{{n-1 \atop k-1}\right\}$ formas), o lo colocamos en alguno de los conjuntos formados con los $n-1$ primeros objetos. Esto último se puede hacer de $k\left\{{n-1 \atop k}\right\}$ maneras, porque cada una de las $\left\{{n-1 \atop k}\right\}$ formas de distribuir los $n-1$ objetos dan k subconjunto en los que podemos colocar el n-ésimo objeto. De ahí que:

$$\left\{{n \atop k}\right\} = k\left\{{n-1 \atop k}\right\} + \left\{{n-1 \atop k-1}\right\}$$

- $\left[{n \atop k}\right]$ representa igualmente el número de permutaciones de n elementos que contienen exactamente k ciclos. Por tanto

$$\sum_{k=0}^{n} \left[{n \atop k}\right] = n!$$

  para todo $n \geq 0$

6 Grafos. Conectividad

6.1 Clasificación de las Aristas

Esta es la clasificación de las aristas de un grafo según su posición respecto al árbol implícito del DFS.

- **Tree edges:** son las aristas que pertenecen al árbol del DFS
- **Back edges:** son aquellas aristas que conectan un vértice con su ancestro en el árbol del DFS
- **Forward edges:** son aquellas que no pertenecen al árbol pero conectan a un vértice con su decendiente dentro de éste.
- **Cross edges:** son todas las otras aristas. Estas pueden ir desde un vértice a otro de tal forma que ninguno de los vértices es ancestro ni descendiente de otro, o pueden ir entre vértices de diferentes árboles de búsqueda en profundidad dentro del bosque de búsqueda en profundidad.

6.2 Puntos de Articulación

Definición

- es un vértice tal, que si se elimina junto a todas sus aristas adyacentes, causa que el grafo se desconecte.
- un grafo que no tiene puntos de articulación es una componente biconexa
- si un nodo v tiene un decendiente en el árbol del DFS, del que sale un back edge hacia un ancestro de v (o sea, se forma un ciclo), este nodo no puede ser un punto de articulación, porque no es obligatorio pasar por él para llegar a sus hijos .

Algoritmo Hopcroft-Tarjan que Encuentra Puntos de Articulación

- a medida que corremos el DFS, llenamos dos arreglos que nos ayudarán a clasificar las aristas: $dt[]$ y $low[]$. $dt[u]$ guarda el número de orden del nodo u en el DFS y $low[u]$ guarda el menor $dt[]$ de un nodo en el subárbol del DFS de u, al que se puede llegar desde u.
- Al principio $low[u] = dt[u]$. Entonces si hay un ciclo, o sea un back edge, $low[u]$ se puede hacer más pequeño. Nótese que si hay una arista hacia atrás (back edge) hacia el padre p_u de u no se actualiza $low[u]$ con $dt[pu]$. O sea que los ciclos que cuentan son los que tienen 3 aristas o más.
- Cuando se está en el vértice u que tiene a v como un vértice adyacente y $low[v] \geq dt[u]$ entonces u es un punto de articulación. Esto es porque el hecho de que $low[v]$ no es menor que el $dt[u]$ implica que no hay back edge desde v que pueda alcanzar a otro vértice w con menor $dt[w]$ que $dt[u]$. Un nodo w con menor $dt[w]$ que u implica que w es ancestro de u en el rbol del DFS. Esto significa que para llegar a un ancestro(s) de u desde v hay que pasar por el vértice u. Entonces, quitarlo causa que el grafo se desconecte.
- Caso especial: la raíz del árbol del DFS, es un punto de articulación si y solo si tiene más de un hijo.

7 Árboles

7.1 Centroid Decomposition

- **Definición:** Dado un árbol con n nodos, un centroide es un nodo cuya eliminación separa al árbol dado en un bosque, donde cada árbol resultante no contiene más de $n/2$ vértices. Para un árbol cualquiera siempre existe al menos un centroide.
- **Propiedades de los Centroides**
  - Si llamamos peso de un nodo al número de vértices en el mayor subárbol de dicho nodo, un nodo con mínimo peso es un centroide del árbol.
  - Cada árbol tiene uno o dos centroides. Si un árbol tiene dos centroides, entonces estos están unidos por una arista y la aliminación de dicha arista deja dos árboles de igual cantidad de nodos.
  - Un vértice es el único centroide de un árbol de n vértices si y solo si su peso es a lo más $\frac{n-1}{2}$.
  - Definimos $d(u, v)$ como el largo del camino más corto entre u y v. La distancia del vértice u, denotado $L(u)$, es la suma de todas las $d(u, v)$ para cada v, vértice del árbol.

    $$L(u) = \sum_{v \in T} d(u, v)$$

    Un centroide del árbol es tal que $L(u)$ es mínimo.
  - Si modelamos con el árbol, un sistema de líneas telefónicas, donde cada arista representa un sector de línea y los caminos llamadas entre los vértices finales. Se asume que, al mismo tiempo, un vértice puede estar envuelto en solo una llamada.
    Definimos el "switchboard number" del vértice v, $sb(v)$, como el máximo número de llamadas a través de v en un momento determinado. Los centroides de del árbol en tal modelo, cumplen que $sb(v)$ es máximo.
- **Encontrar el Centroide de un Árbol:** Una manera de encontrar el centroide es elegir una raíz arbitraria, entonces hacer un BFS calculando la cantidad de nodos en cada subárbol. Luego nos movemos, comenzando desde la raíz, al subárbol de mayor tamaño hasta que lleguemos a un vértice cuyo subárbol no tenga más de $n/2$ nodos. Este nodo es el centroide del árbol.
- **Descomponiendo el Árbol para Formar un Árbol de Centroides:** Cuando se elimina el centroide, el árbol original de descompone en un número de árboles diferentes, cada uno con menos de $n/2$ nodos. Tomamos este centroide como la raíz de un nuevo árbol al que llamaremos "centroid tree" o árbol de centroides. Entonces hacemos lo mismo recursivamente para los árboles que se formaron e incluimos los centroides de estos como hijos de nuestra raíz. La recursión para cuando el árbol está formado por un solo vértice. Así un nuevo centroid tree se forma a partir de nuestro árbol original.
- **Propiedades de los Árboles de Centroides**
  - El ábol formado contiene los n nodos del árbol original.
  - La altura del árbol de centroides es $O(\log n)$
  - Consider any two arbitrary vertices A and B and the path between them (in the original tree) can be broken down into A–¿C and C–¿B where C is LCA of A and B in the centroid tree.

Considérese cualesquiera dos vértices u y v y el camino entre ellos en el árbol original. Dicho camino $u \to v$, puede ser dividido en dos caminos $u \to lca(u,v)$ y $lca(u,v) \to v$, donde $lca(u,v)$ es el ancestro común más bajo entre u y v en el centroid tree.

- Se descompone el árbol original en $O(n \log n)$ caminos diferentes (desde cada centroide a todos los vertices en su subárbol en el centroid tree) tales que cualquier camino en el árbol original es la concatenación de dos caminos diferentes de este conjunto.

8 Programación Dinámica

8.1 Usando Matrix Exponentiation

- Esta técnica puede ser usada en recurrencias de la forma

$$f(n, s_i) = \sum_{j=1}^{k} c_{i,j} \cdot f(n-1, s_j)$$

donde $s_1, s_2, \cdots s_k$ son los k estados donde puede escontrarse nuestro sistema y n representa una condicion del estado adicional, que no influye en las transiciones entre los k estados s_i.

- Para resolver tal recursión definimos una matriz de trnasiciones M donde cada elemento $m_{i,j} = c_{i,j}$ o sea el "peso" de la transición desde el estado i al j.

- Ahora considérese la matriz M^{n-1}. La suma de todos los elemetos el la fila i nos dará $f(n, s_i)$, asumiendo que $\forall i : f(0, i) = 0$

- Este proceso es similar a calcular la cantidad de caminos de largo n en un grafo. Para ello elevamos la matriz de adyacencia al exponente n y el valor en la fila i columna j de la matriz resultante es el número de caminos de largo n desde el vértice i al j.

8.2 Optimización Usando la Desigualdad Cuadrangular

- **El Problema:** Tenemos una matriz de costos $w_{i,j}$ y tamaño $n \times n$. Hay que calcular los valores de la función f tal que:
  - $f(i,i) = w_{i,i} = 0$
  - $f(i,j) = w_{i,j} + \min_{i<k\leq j}\{f(i,k-1) + f(k,j)\}$

  para $i < j$, porque no consideramos los valores de $f(i,j)$ para $i > j$. Usualmente $f(1,n)$ es el resultado deseado. El algoritmo en $O(n^3)$ puede ser optimizado a $O(n^2)$

- **Quadrangle Inequalitiy (QI):** Si w es tal que $w_{a,c} + w_{b,d} \leq w_{b,c} + w_{a,d}$ con $a \leq b \leq c \leq d$, se dice que satisface la desigualdad cuadrangular.

- **Monotonía:** La matriz w es monótona si $w_{b,c} \leq w_{a,d}$ con $a \leq b \leq c \leq d$.

- Definimos la matrix k, donde $k_{i,j} = \min\{t \mid f(i,j) = w_{i,j} + f(i,t-1) + f(t,j)\}$, donde $i < j$. En particular, se define $k_{i,i} = i$.
  En otras palabras, $k_{i,j}$ es el índice k más pequeño, tal que se logra el mínimo en la función.

- **Teorema:** Si w cumple con IQ y es monótona, entonces f también cumple con IQ. Más específicamente: $k_{i,j-1} \leq k_{i,j} \leq k_{i+1,j}$

9 Game Theory

9.1 Juegos Imparciales

- **Propiedades de los Juegos Imparciales**
  - Hay dos jugadores y el conjunto de posiciones es finito.
  - Ambos jugadores alternan turnos y el conjunto de jugadas posibles depende de la posición y no de que jugador está en turno.
  - Si es un juego normal (normal play), el juego termina cuando uno de los jugadores no puede hacer un movimiento, el jugador que no puede mover pierde; si se juega "misere", el que no puede mover gana.

- **Posiciones en los Juegos Imparciales**
  - El juego está en una P-posición si se asegura una victoria para el jugador Previo.
  - El juego está en una N-Posición si se asegura una victoria para el próximo jugador(Next).
  - Una posición A es seguidora de otra posición B si el jugador en turno puede ir de B a A con su movimiento. Al conjunto de las posiciones seguidoras de A se las denota $F(A)$.
  - Una posición es terminal si no tiene ninguna seguidora, o sea el juego termina si se llega a ella.

- **Inducción Hacia Atrás**
  - Etiquete cada posición terminal como P.
  - Etiquete cada posición que tiene como seguidora a una P como N.
  - Etiquete cada posición que solo tiene como seguidoras a N-posiciones como P-posiciones.
  - Si se juega misere, en el paso 1, cada posicion terminal es N.

- **Nim y Suma Nim**

- El Nim es un juego donde se cuenta con n montones de fichas. Dos jugadores alternan movimientos donde en cada uno pueden tomar un número de fichas de un montón. Quien no pueda tomar una ficha (cuando no quede ninguna), pierde.
- La suma nim de uno o varios números es simplemente el *xor* de estos.
- Cada juego combinatorio imparcial se comporta como un juego de nim.
- En nim, una posición es P si y solo si la suma nim de los montículos es igual a cero.

- **Poker Nim y Las Jugadas Reversibles**
  - El Poker Nim es esencialmente el mismo juego del nim pero también se puede adicionar fichas a los motículos.
  - Cada vez que el oponente adicione fichas, el jugador puede sustraer la misma cantidad. Es una jugada reversible.

- **Juegos en Grafos Dirigidos**
  - Un juego puede verse como un grafo dirigido donde los nodos son las posiciones y las aristas las jugadas.
  - Desde cada nodo, hay una arista a cada uno de los nodos correspondientes a sus seguidores.

- **Función Sprage-Grundy**
  - Es una función G que le asigna a cada nodo del grafo del juego, un número de tal forma que:

$$G(x) = min\{n \geq 0 \colon n \neq G(y), \forall y \in F(x)\}$$

    o sea, $G(x)$ es el menor entero que no figura entre los valores Grundy correspondientes a las posiciones seguidoras de x.
  - El valor de la función Grundy para las posiciones terminales es cero.
  - Una posición es perdedora para el jugador en turno si y solo si el valor Grundy es cero.

- **Suma Disyuntiva de Juegos Combinatorios**
  Dados varios juegos combinatorios, se puede formar un nuevo juego combinando estos, jugando bajo las siguientes reglas:
  - Se da una posicion inicial para cada uno de los juegos.
  - Los jugadores alternan jugadas. Una jugada consiste en seleccionar uno o varios de los subjuegos y hacer una jugada legal en ellos, dejando igual los demás.
  - Se llega a una posición terminal cuando no se puede hacer una jugada en ninguno de los subjuegos.

- **Reglas Generales de la Suma de Juegos**
  - **Si el jugador en turno elige un juego y hace una jugada en este**: Se aplica el Teorema Sprague-Grundy.
  - **Si el jugador en turno elige un conjunto no vacío de juegos (posiblemente todos) y hace una jugada en todos ellos**: Una posición es perdedora si y solo si cada juego está en una posición perdedora.
  - **Si el jugador en turno elige un subconjunto propio (no vacío, pero no todos) de juegos y hace una jugada en todos ellos**: La posición es perdedora ssi todos los valores Grundy de los juegos son iguales.
  - **Si el jugador debe jugar en todos los juegos y pierde si no puede mover en alguno**: Una posición es perdedora ssi alguno de los juegos esta en una posición perdedora.

- **Propiedades de la Suma de Juegos**
  - **Teorema Sprague-Grundy**: Si G_i es la función Grundy para cada grafo i, con $1 \leq i \leq n$, entonces el grafo de la suma de los n juegos tiene función Sprague-Grundy G tal que

$$G(x) = G_1(x) \oplus G_2(x) \oplus \cdots \oplus G_n(x)$$

    o sea, la suma nim de las funciones.
  - **Producto Paralelo**: Si el jugador en turno debe hacer una jugada legal en cada uno de los subjuegos, el valor de la función Grundy es el mínimo de las funciones Grundy de cada posición de los juegos componentes.
  - **Nim de Moore**: La suma de orden p de varios juegos, es la suma disyuntiva en la cual un jugador debe hacer una jugada legal en al menos uno de los subjuegos y en a lo mas p juegos. La suma usual es de orden 1. La suma de orden p se calcula tomando la extensión binaria de los tamaños. Se calculan los valores w_l como

$$w_l = ((n_1)_l + (n_2)_l + \cdots + (n_k)_l) mod(p+1)$$

    donde $(n_i)_l$ es el bit l-ésimo del i-ésimo tamaño de un montículo.

- **El Nim de Lasker y los Juegos Take-and-Break**
  - En el Nim de Emanuel Lasker, los jugadores tienen otra posible jugada: dividir una pila en dos pilas no vacías, sin quitar ninguna ficha del juego.
  - Se calcula la función Sprague-Grundy para este juego pero teniendo en cuenta que al hacer una jugada en un montículo, las posiciones seguidoras de la actual, serán no solamente las correspondientes a los montículos menores, sino también las de dividir un motículo en dos.

- Por ejemplo $G(0) = 0$ y $G(1) = 1$. Los seguidores de 2 son 0, 1, y $(1, 1)$; sus valores Grundy son 0, 1, y el valor de la suma nim de los valores Grundy: $G(1) \oplus G(1) = 0$. El valor Grundy es $G(2) = 2$. La posicion 3 tiene como seguidores a 0, 1, 2 y $(1, 2)$; $G(3) = 4$, mientras que la 4 tiene se seguidores a 0, 1, 2, $(1, 3)$, $(2, 2)$. $G(4) = 3$. Este patron se repite en todo el grafo.

10 Strings

10.1 Algoritmo Z

- Llamaremos segmentos coincidentes (SC) a aquellas subcadenas que coinciden con un prefijo de la cadena S.
- Mantenemos un intervalo $[l, r]$ que representa el segmento coincidente que termina mas a la derecha. El índice r puede verse tambien como el límite hasta el cual la cadena ha sido explorada.
- Hacemos un ciclo de 1 a n (n, largo de la cadena). Para cada posicion i tenemos dos casos:
  - Si $i > r$: La posicion actual esta fuera de los limites de lo que ya hemos procesado. Calcularemos $z[i]$ comparando los valores uno por uno. Al final, si $z[i] > 0$ tenemos que actulizar los limites de $[l, r]$ porque esta garantizado que el nuevo $r = i + z[i]1$ es mejor que el r actual. El limite l se actualiza a la posicion actual.
  - Si $i \leq r$: la posicion actual esta dentro de los limites del actual SC. Podemos usar los valores de z ya calculados.Observese que las subcadenas $s[l...r]$ y $s[0...rl]$ coinciden. Esto significa que podemos tomar como una aproximacion inicial para $z[i]$, el valor correspondiente en el segmento ya calculado. Este valor es $z[il]$, aunque puede ser demasiado largo porque cuando se aplique a la posicion i, puede exceder el indice r y no sabemos nada de los caracteres mas alla de r. Por lo tanto, tomamos $min(ri + 1, z[il])$. Luego de inicializar $z[i]$ tratamos de coincidir los caracteres en las posiciones $z[i]$ y $i + z[i]$ uno por uno.

10.2 Aho-Corasick

- **Estructura:** El Aho-Corasick está compuesto por las siguientes tres funciones:
  - **Go To:** La función $g(s, c)$ que simplemente sigue las aristas en un trie formado por todos los patrones dados. El nodo u es el padre de $g(u, c)$ en el trie y la arista entre ellos está etiquetada con el caracter c.
  - **Failure:** Esta función $f(s)$ guarda todas las aristas correspondientes a cuando el caracter actual no tiene una arista en el árbol.
  - **Salida:** La función $o(s)$ que guarda los índices de todos patrones al final del estado actual.
- **Preprocesamiento:**
  - Primero se construye un trie con todas los patrones (función g).
  - Luego construimos la función de fallo f. Para un estado encontramos el sufijo propio más largo que es también prefijo de algún patrón, o sea, que tiene un nodo dentro del trie que lo representa. Esto se hace con un BFS.
  - Para calcular $f(u)$ se busca el primer se hace w tal que $g(w,)$
  - La función de salida $o(u) = o(u) \cup o(f(u))$. Esto se hace porque los patrones reconocidos en $f(u)$, y solo esos, son sufijos propios de la cadena que representa el nodo u, y serán entonces reconocidos también en el estado u.
  -
  -

11 Geometry

11.1 Elementos Primitivos

- **Dot product:**
  - El producto escalar de dos vectores $\overrightarrow{v}$ y $\overrightarrow{w}$ puede verse como una magnitud de cuán similares son sus direcciones. Se define como
    $$\overrightarrow{v} \cdot \overrightarrow{w} = ||\overrightarrow{v}||||\overrightarrow{w}|| \cos\theta$$
    donde $||\overrightarrow{v}||$ y $||\overrightarrow{w}||$ son las normas de los vectores y θ es el ángulo entre $\overrightarrow{v}$ y $\overrightarrow{w}$
  - Ya que $\cos(-\theta) = \cos(\theta)$ el signo del ángulo no importa y es simétrico, o sea $\overrightarrow{v} \cdot \overrightarrow{w} = \overrightarrow{w} \cdot \overrightarrow{v}$
  - Tomando solo $\theta \in [0, \pi]$, si $\theta < \pi/2$ entonces $\overrightarrow{v} \cdot \overrightarrow{w} > 0$, si $\theta > \pi/2$ es negativo, y si $\theta = \pi/2$, $\overrightarrow{v} \cdot \overrightarrow{w} = 0$
- **Dot product:**
  - El producto escalar de dos vectores $\overrightarrow{v}$ y $\overrightarrow{w}$ puede verse como una magnitud de cuán perpendiculares son. Se define como
    $$\overrightarrow{v} \times \overrightarrow{w} = ||\overrightarrow{v}||||\overrightarrow{w}|| \sin\theta$$
  - Ya que $\sin(-\theta) = -\sin(\theta)$ el signo del ángulo importa y este cambia cuando los vectores son intercambiados: $\overrightarrow{w} \times \overrightarrow{v} = -\overrightarrow{v} \times \overrightarrow{w}$
  - $\overrightarrow{v} \times \overrightarrow{w}$ es positivo si $\overrightarrow{w}$ está a la izquierda de $\overrightarrow{v}$ y negativo si está a la derecha.

- Si tomamos solo $\theta \in (-\pi, \pi]$, entonces $\overrightarrow{v} \times \overrightarrow{w}$ es positivo si $0 < \theta < \pi$, negativo si $-\pi < \theta < 0$ y cero si $\theta = 0$ ó $\theta = \pi$, o sea, si los vectores son colineares.

- **Orientación:** Podemos usar el producto cruz para determinar la orientación relativa de tres puntos A, B y C. Para esto calculamos $\overrightarrow{AB} \times \overrightarrow{AC}$. Esta expresión es positiva si C está en el lado izquierdo de $\overrightarrow{AB}$, negativa si está en el lado derecho, y es igual a cero si C está sobre la recta que contiene a $\overrightarrow{AB}$.
