GUÍA DE APRENDIZAJE PARA CONCURSANTES ICPC Y IOI: COMBINATORIA, FÓRMULAS BÁSICAS

1. Introducción

La combinatoria es la rama de las matemáticas que estudia los diversos modos de agrupar los elementos de un conjunto sometidos a unas u otras condiciones. Esta rama de la matemática es aprovechada para la confección de ejercicios y problemas de concursos. Es por eso que es necesario conocer algunos elementos para poder resolver los problemas vinculados con esta área de conocimientos.

2. Desarrollo

2.1. Variaciones

Se llama **variación** de los n objetos tomados p a p, a todo conjunto ordenado formado por p objetos escogidos de cualquier modo entre los n objetos considerando distintas dos variaciones cuando difieran en algún objeto o en el orden.

Ejemplo: Con los cuatro objetos a, b, c, d las variaciones dos a dos son:

ab ba ca da

ac bc cb db

ad bd cd dc

El número de estas variaciones lo denotaremos por $V_{n,p}$

Las variaciones con cierto número de objetos dados a, b, c, d, e se pueden ir formando sucesivamente (primero las monarias, luego las binarias, después las ternarias, etc.) por un método uniforme que consiste en agregar a cada variación de cierto orden cada una de las letras (objetos) que no están en ellas. Las variaciones monarias (de primer orden) o variaciones tomadas uno a uno con las cinco letras a, b, c, d, e son evidentemente:

a b c d e

Para formar las binarias agregamos a cada una las letras restantes y se obtiene el cuadro:

ab ba ca da ea

ac bc cb db eb

ad bd cd cd ec

ae be ce de ed

Se forman ahora las ternarias agregando sucesivamente a cada binaria las letras que no están en ella, como hay 20 binarias y a cada una se le puede agregar 5 - 2 = 3 letras, resultarán 60 variaciones ternarias. Siguiendo el mismo proceso se forman las variaciones de cuarto orden y las de quinto orden.

Sea $V_{n,p}$ el número de variaciones de orden p y $V_{n,p-1}$ el número de variaciones de orden $p-1$.

Una vez formado el cuadro de variaciones de orden $p-1$, para formar el de orden p se le agrega a cada una los $n-(p-1) = n\breve{}p+1$ elementos que no están en ella: por lo tanto cada variación de orden $p-1$ produce $n\breve{}p+1$ variaciones de orden p y como hay $V_{n,p-1}$ variaciones de orden $p-1$ el número total de variaciones de orden p será:

$V_{n,p-1} = n*(n-1)*(n-2)*...*(n\breve{}p+1)$

2.1.1. Variación con repetición

En cuanto a las variaciones de n objetos tomado p a p tratado anteriormente podemos ver que p $\leq$ n. Si agrupamos k obetos de los n disponibles, siendo $k > n$, lógicamente habrá repeticiones.

Todas las posibles distribuciones con k objetos en cada una donde en cada repetición pueden aparecer objetos repetidos y se diferencian por su orden recibe el nombre de variaciones con repetición.

El número de variaciones con repetición lo denotaremos por $W_{n,p}$. Si el número de objetos es igual a n y en cada variación aparecen k objetos se pueden formar n^k variaciones con repetición.

$W_{n,p} = n^k$

Así, por ejemplo, con dos objetos (los dígitos cero y uno), las variaciones con repetición de tamaño ocho es $W_{2,8} = 256$.

2.2. Permutaciones

Se llama **permutación** de los n objetos a todo conjunto ordenado formado por dichos n elementos. Dos permutaciones se distinguen una de otra solamente por el orden de colocación de sus elementos.

Las permutaciones son un caso particular de las variaciones, cuando $p = n$. Es decir, son las variaciones de n objetos tomados n a n.

Denotaremos por P_n al número de permutaciones de n objetos.

$P_n = V_{n,n}$

Como las permutaciones de n objetos son las variaciones de orden n, pueden formarse de igual modo que las variaciones.

$P_n = V_{n,n} = n*(n-1)*(n-2)*...*(n\breve{}n+1)$

$P_n = n*(n-1)*(n-2)*...*1$

$P_n = n!$

Como de cada una de las P_{n-1} permutaciones se originan n permutaciones nuevas, se tiene la relación: $P_n = n*P_{n-1}$

2.2.1. Permutación con repetición

Hasta ahora las permutaciones que hemos formado ha sido a base de un conjunto donde todos los objetos son diferentes. Sin embargo, si algunos de los objetos son iguales se obtendrán menos permutaciones (algunas serán iguales). Por ejemplo, permutando abcd obtenemos 24 permutaciones diferentes, si en lugar de abcd tenemos que permutar las letras abab ya no son 24 porque algunas se repetirán.

El problema general se anuncia así: Se tienen k objetos diferentes. ¿Cuántas permutaciones se pueden hacer tomando n_1 elementos del primer tipo, n_2 del segundo, ..., n_k del k–ésimo?

El número total de elementos de cada permutación es igual a n.Los elementos del tipo n_1 pueden ser permutados de $n_1!$ formas, pero como son iguales, estas permutaciones no cambian nada, de forma análoga no cambian nada las $n_2!$ permutaciones de los elementos del segundo tipo, etc. Las permutaciones anteriores se pueden permutar de $n_1!$ * $n_2!$ * ... * $n_k!$ maneras de forma que no varíe, por eso el conjunto de las $n!$ permutaciones se separa en partes formadas por $n_1!$ * $n_2!$ * ... * $n_k!$ permutaciones iguales. El número de permutaciones con repetición diferentes se puede escribir:

$$P_n(n_1, n_2, ..., n_k) = \frac{n!}{n_1! * n_2! * ... * n_k!}$$

2.3. Combinaciones

Se llama **combinación** de n objetos tomados p a p, a todo conjunto de p objetos elegidos entre ellos de tal modo que dos conjuntos se diferencien al menos en un objeto.

Denotaremos por $C_{n,p}$ al número de combinaciones de n objetos tomados p a p.

Las combinaciones de n objetos tomados p a p son los distintos conjuntos que pueden formarse con p objetos elegidos entre n dados, de modo que un conjunto se diferencie de otro al menos en uno de los elementos.

En las variaciones, *abc* y *bca* son variaciones distintas, pero es la misma combinación, sólo un conjunto que contenga un nuevo elemento se considera una nueva combinación. ej: *abd*

Si imaginamos formadas $C_{n,p}$ combinaciones de orden p que se pueden formar con n objetos, ejemplo, las combinaciones ternarias de las cuatro letras a, b, c, d

abc abd acd bcd

En cada combinación permutamos las letras de todas las maneras posibles y obtenemos el cuadro:

abc abd acd bcd

acb adb adc bdc

bac bad cad cbd

bca bda dac dbc

cba dba dca dcb

El cual contiene las variaciones ternarias de las cuatro letras a, b, c, d pues las que proceden de la misma combinación difieren en el orden de las letras y las que proceden de combinaciones distintas difieren al menos en una letra. Como cada combinación de orden p da lugar a P_p combinaciones distintas, entre los números $C_{n,p}$, P_p y $V_{p,n}$ existe la relación:

$C_{n,p} * P_p = V_{p,n}$

de donde

$C_{n,p} = \frac{V_{p,n}}{P_p}$

sustituyendo P_p y $V_{p,n}$

$C_{n,p} = \frac{n*(n-1)*(n-2)*...*(n-p+1)}{p!}$

Con el objetivo de completar n! en el numerador, multiplicamos el numerador y el denominador por $(n-p)!$

$C_{n,p} = \frac{n*(n-1)*(n-2)*...*(n-p+1)*(n-p)!}{p!*(n-p)!}$

$C_{n,p} = \frac{n!}{p!*(n-p)!}$

Las expresiones de la forma $C_{n,p}$ reciben el nombre de números combinatorios.

3. Implementación

3.1. C++

```
#define ULL unsigned long long
ULL variants( ULL n,  ULL  k) {
    ULL res = 1;
   for ( ULL i = 0; i < k; i++) {
      res = res * (n - i) / (i + 1);
   }
   return res;
}

ULL variantsWithRepetions(ULL n,  ULL  k) {
   long res = 1;
   while (k > 0) {
      if ((k & 1)==1 )
         res = res * n;
      n = n * n;
      k >>= 1;
        }
}

ULL permutations(ULL n) {
```

```
   ULL res = 1;
   for (ULL i = 1; i <= n; i++) {
      res = res * i;
   }
   return res;
}

ULL permutationsWithRepetions(ULL n, int k, ULL  * ns) {
   ULL res =1;
   ULL * fact = new ULL [n+1];
   for(ULL i =1;i<=n;i++){
      res= res*i;
      fact[i] = res;
   }
   for(int i=0;i<k;i++)
      res/=fact[ns[i]];
   return res;
}

ULL combination(ULL n,ULL  k) {
   ULL res = 1;
   k = min(k, n - k);
   for (ULL i = 0; i < k; i++) {
      res = res * (n - i) / (i + 1);
   }
   return res;
}
```

3.2. Java

```
public long variants(long n, long k) {
  long res = 1;
  for (long i = 0; i < k; i++) {
     res = res * (n - i);
  }
  return res;
}

public long variantsWithRepetions(long n, long k) {
   long res = 1;
   while (k > 0) {
      if ((k & 1)==1 )
         res = res * n;
      n = n * n;
      k >>= 1;
   }
   return res;
}
```

```java
public long permutions(long n) {
    long res = 1;
    for (long i = 1; i <= n; i++) {
        res = res * i;
    }
    return res;
}

public long permutationsWithRepetions(long n, int k, long [] ns) {
    long res =1;
    long [] fact = new long [n+1];
    for(long i =1;i<=n;i++){
        res= res*i;
        fact[i] = res;
    }
    for(int i=0;i<k;i++)
        res/=fact[ns[i]];
    return res;
}

public long combination(long n, long k) {
    k = Math.min(k, n - k);
    long res = 1;
    for (long i = 0; i < k; i++) {
        res = res * (n - i) / (i + 1);
    }
    return res;
}
```

4. Complejidad

Una vez visto una primera implementación de cada una de estas funciones de conteo es evidente que exceptuando la de variación con repetición la complejidad del resto es $O(n)$ en el peor de los casos. En caso de la variación con repetión se implemento una exponención binaria por tanto su complejidad es $O(\log n)$, en el caso de permutación con repetición tiene un costo en memoria de $O(n)$. Pero tranquilo a medida que avancemos con más profundidad en cada una de ellas veremos formas más eficientes de calcular según la situación.

5. Aplicaciones

Entre los ejercicios y problemas de concursos existen un gran número que su solución radica en saber particionar o agrupar los elementos bajo cierta condiciones y es donde es útil las fórmulas básicas de la combinatoria. Por lo general los ejercicios que requieren el uso de estas para su solución lo hacen de forma combinadas. Otro elemento caracteristicos es que por lo general la solución debe ser modulada ya que estás funciones sus valores crecen rapidamente.

6. Ejercicios propuestos

A continuación una lista de problemas que se pueden usar aplicando las fórmulas básicas de la combinatoria.

- DMOJ - Dreamoon y Wi-Fi
- DMOJ - Cantidad de Números. Tener en cuenta el caso espacial de 0 dígitos impares con 1 dígito par.