Số học 5 - Các kiến thức cơ bản về Tổ hợp (Combinatorics)

Số học 5 - Các kiến thức cơ bản về Tổ hợp (Combinatorics)

Nguồn: HackerEarth

Người dịch: Bùi Việt Dũng

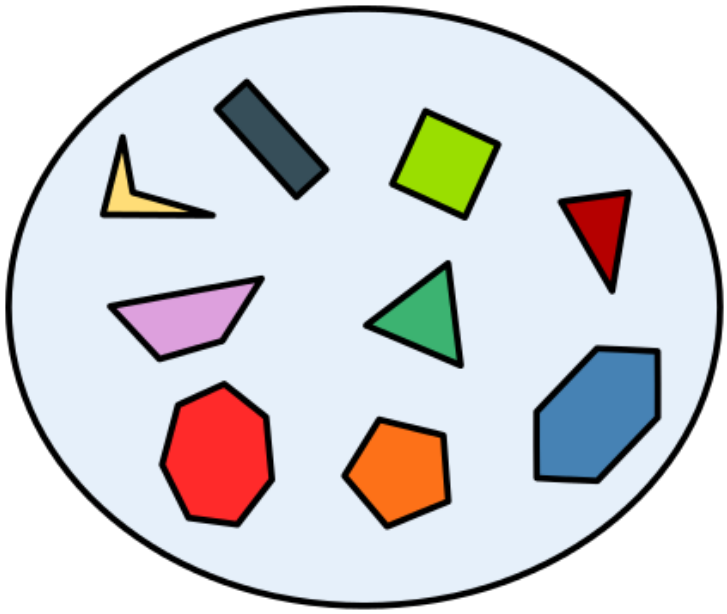
Tập hợp (Set)

Tập hợp (Set) là gì?

Trong toán học, tập hợp là một nhóm các phần tử, mỗi phần tử phân biệt với nhau.

Ví dụ, 2, 4, 6 được coi là các phần tử phân biệt khi ta xét từng số một, nhưng khi chúng ta nhóm ba số ấy thì ta được một tập hợp gồm ba phần tử được kí hiệu là {2,4,6}.

Tập hợp là một trong những khái niệm cơ bản trong Toán học.



Tập hợp các hình đa giác được biểu diễn trong biểu đồ Venn.

Tập con (Subset)

Nếu mọi phần tử thuộc tập A cũng thuộc tập hợp B, thì tập A là **tập con** của tập B, kí hiệu là $A \subset B$.

Tương tự, ta có thể viết $B \supset A$, đọc là B là **tập cha (superset)** của tập A.

Quan hệ cha-con giữa các tập hợp còn được gọi là **quan hệ chứa nhau (containment)** hay **quan hệ bao hàm (inclusion)**.

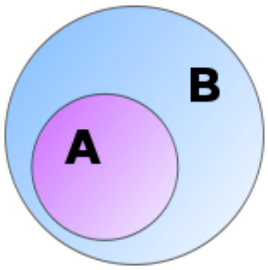
Nếu A là tập con của tập B nhưng không bằng tập B, thì A được gọi là **tập con không tầm thường (proper subset)** của tập B, kí hiệu là $A \subsetneq B$, hay $B \supsetneq A$ (đọc là B là **tập cha không tầm thường (proper superset)** của tập A).

Ví dụ:

- {1,3} $\subset$ {1,2,3,4}
- {1,2,3,4} $\subset$ {1,2,3,4}

Tập rỗng (empty set, kí hiệu $\varnothing$) là tập con của tất cả các tập và tất cả các tập là tập con của chính nó:

- $\varnothing \subset A$.
- $A \subset A$.



A là tập con của tập B.

Các phép toán với tập hợp

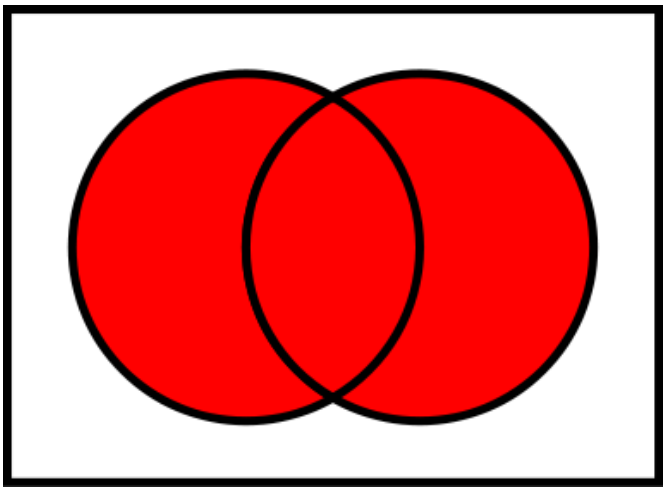
Có nhiều phép toán có khả năng xây dựng một tập hợp mới dựa trên các tập hợp đã cho.

Phép hợp (Union)

Hai tập hợp có thể được ghép vào nhau. Hợp của hai tập hợp A và B, kí hiệu là $A \cup B$, là một tập hợp gồm các phần tử thuộc tập A hoặc thuộc tập B.

Ví dụ:

- {1,2} $\cup$ {1,2} = {1,2}
- {1,2} $\cup$ {2,3} = {1,2,3}
- {1,2,3} $\cup$ {3,4,5} = {1,2,3,4,5}



Hợp của hai tập hợp A và B, kí hiệu là $A \cup B$.

Một vài tính chất cơ bản của phép hợp:

- $A \cup B = B \cup A$.
- $(A \cup B) \cup C = A \cup (B \cup C)$.
- $A \subset (A \cup B)$.

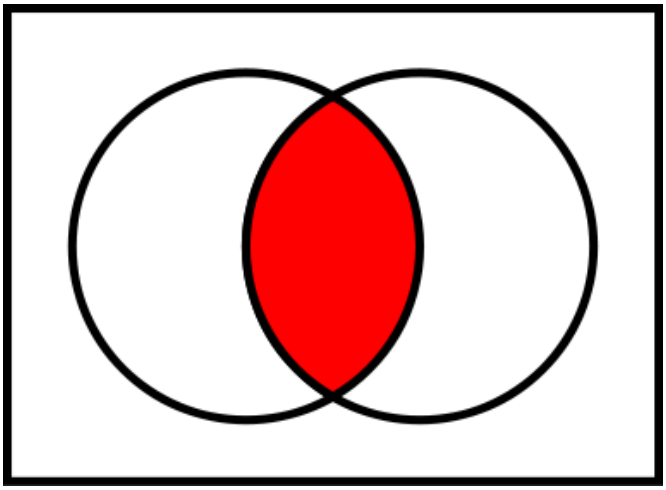
- $A \cup A = A$.
- $A \subset B$ khi và chỉ khi $A \cup B = B$.

Phép giao (Intersection)

Một tập hợp mới có thể được xây dựng từ các phần tử mà cả hai tập đều có. Giao cuả hai tập hợp A và B, kí hiệu $A \cap B$, là tập hợp các phần tử cùng thuộc tập A và tập B. Nếu $A \cap B = \varnothing$, tập A và tập B là hai **tập rời nhau (disjoint)**.

Ví dụ:

- {1,2} ∩ {1,2} = {1,2}.
- {1,2} ∩ {2,3} = {2}.



Giao của hai tập hợp A và B, kí hiệu là $A \cap B$.

Một vài tính chất cơ bản của phép hợp:

- $A \cap B = B \cap A$.
- $(A \cap B) \cap C = A \cap (B \cap C)$.
- $A \cap B \subset A$.
- $A \cap A = A$.
- $A \cap \varnothing = \varnothing$.
- $A \subset B$ khi và chỉ khi $A \cap B = A$.

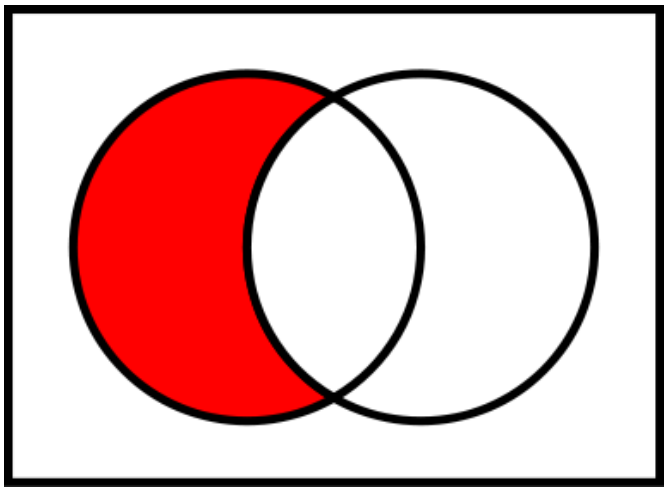
Phép lấy hiệu (Relative complement)

Ta có thể thực hiên phép trừ với hai tập hợp. Hiệu của hai tập hợp A và B, kí hiệu là $A \setminus B$, là tập hợp bao gồm tất cả các phần tử thuộc A nhưng không thuộc B. Lưu ý rằng ta có thể trừ phần tử mà không thuộc tập hợp, ví dụ như bỏ phần tử 'xanh' khỏi tập hợp {1,2,3}, khi đó tập hợp {1,2,3} không bị thay đổi.

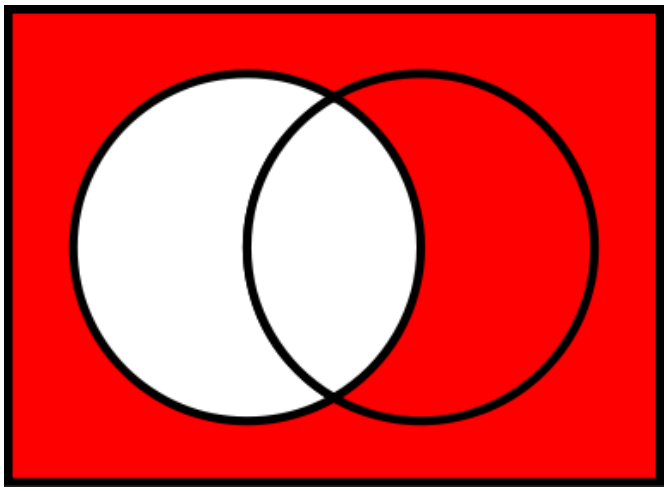
Trong một số trường hợp tập A được coi là tập con của một tập U lớn hơn. Trong trường hợp đó, $U \setminus A$ được gọi là **phần bù hoàn toàn (absolute complement)** của tập A, và được kí hiệu là A'.

Ví dụ:

- {1,2} \ {1,2} = ∅.
- {1,2,3,4} \ {1,3} = {2,4}.
- Nếu U là tập hợp các số nguyên, E là tập hợp các số nguyên chẵn, O là tập hợp các số nguyên lẻ, khi đó $U \setminus E = E' = O$.



Hiệu của hai tập hợp A và B.



Phần bù của A trong U.

Một vài tính chất cơ bản của phép lấy hiệu

- $A \setminus B \neq B \setminus A$ với $A \neq B$.
- $A \cup A' = U$.
- $A \cap A' = \varnothing$.
- $(A')' = A$.
- $A \setminus A = \varnothing$.
- $A \setminus B = A \cap B'$.
- $U' = \varnothing$ và $\varnothing' = U$.

Các quy tắc cơ bản

Kí hiệu $||A||$ là số phần tử của tập A (hay còn được gọi là lực lượng của tập A).

Một vài quy tắc về tổ hợp cần nhớ:

1. **Quy tắc nhân (The Rule of Product):**
   Giả sử có hai tập hợp A và B. Khi đó số cách chọn cặp gồm một phần tử thuộc tập A và một phần tử thuộc tập B là $||A||.||B||$
2. **Quy tắc cộng (The Rule of Sum):** Giả sử có hai tập hợp A và B. Khi đó số cách chọn một phần tử thuộc tập A hoặc thuộc tập B là $||A|| + ||B||$ nếu hai tập A và B rời nhau.

3. **Quy tắc cộng mở rộng (sieve principle)** (còn gọi là **công thức bao hàm - loại trừ (Inclusion-Exclusion Formula)**): $\|A \cup B\| = \|A\| + \|B\| - \|A \cap B\|$.

Trong trường hợp tổng quát, ta có:

$\| \bigcup_{i=1}^{n} A_i\| = \sum_{i=1}^{n} \|A_i\| - \sum_{i \neq j} \|A_i \cap A_j\| + \|A_1 \cap A_2 \cap A_3\| + \|A_1 \cap A_2 \cap A_4\| + \ldots + \|A_{n-2} \cap A_{n-1} \cap A_n\| - \ldots - (-1)^n \|A_1 \cap A_2 \cap \ldots \cap A_n\|$

Lí do ta phải cộng trừ giao của một số tập hợp vì nếu ta không làm như vậy, ta có thể đếm nhiều lần các phần tử xuất hiện tại nhiều tập hợp khác nhau.

Các quy tắc trên cũng đúng khi ta có ba hay nhiều tập hợp.

Các kiến thức cơ bản về Chỉnh hợp và Hoán vị (Permutation)

Chỉnh hợp không lặp (Permutation of Distinct Objects)

Cho tập hợp A gồm n phần tử. Mỗi bộ gồm k ($0 \leq k \leq n$) phần tử được sắp thứ tự của tập hợp A được gọi là một chỉnh hợp chập k của n phần tử thuộc A.

Ví dụ: Trong trận chung kết bóng đá phải phân định thắng thua bằng đá luân lưu 11 mét. Huấn luyện viên của mỗi đội cần trình với trọng tài một danh sách sắp thứ tự 5 cầu thủ trong số 11 cầu thủ để đá luân lưu 5 quả 11 mét.

Mỗi danh sách có xếp thứ tự 5 cầu thủ được gọi là một chỉnh hợp chập 5 của 11 cầu thủ.

Kí hiệu số chỉnh hợp chập k của n phần tử là A_n^k.

Số chỉnh hợp chập k của n phần tử được tính bởi công thức

$A_n^k = n(n-1)\ldots(n-k+1) = \frac{n!}{(n-k)!}$.

với $n! = 1.2.3\ldots n$ và $0! = 1$.

Hoán vị không lặp

Mỗi một chỉnh hợp chập n của n phần tử là một hoán vị của n phần tử đó.

Kí hiệu số hoán vị của n phần tử là P_n.

Số hoán vị của n được tính bởi công thức:

$P_n = n!$.

Hoán vị lặp (Permutation with Repetition)

Hoán vị trong đó mỗi phần tử xuất hiện ít nhất một lần được gọi là hoán vị lặp.

Số hoán vị lặp của n phần tử thuộc k loại, mà các phần tử loại i ($1 \leq i \leq k$) xuất hiện n_i lần được kí hiệu là $P(n_1, n_2, \ldots, n_k)$ và được tính bằng công thức

$P(n_1, n_2, \ldots, n_k) = \frac{n!}{n_1!n_2!\ldots n_k!}$

Các đối tượng tổ hợp (Combinatorial Objects)

Một **song ánh (bijection)** là tương ứng một-một giữa hai tập hợp, ví dụ tập hợp những người chồng và tập hợp những người vợ (một chồng chỉ có một vợ, và một vợ chỉ có đúng một chồng). Do đó, nếu bạn biết được lực lượng của một tập hợp, bạn có thể biết được lực lượng của tập kia. Ta có thể sử dụng tính chất này trong nhiều bài toán Tổ hợp, đặc biệt là các bài toán đếm, nhưng trước tiên, ta phải biết tính lực lượng của một tập các đối tượng tổ hợp.

Tổ hợp không lặp (Combinations without repetition)

Trong tổ hợp, ta thường phải chọn một tập các phần tử nào đó và không quan tâm đến thứ tự của chúng. Số lượng tập con k phần tử của một tập n phần tử (còn gọi là số tổ hợp chập k của n phần tử) là:

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

Tổ hợp có lặp (Combinations with repetition)

Giả sử ta cần chọn k phần tử từ một tập n phần tử, không quan trọng thứ tự và một phần tử có thể được chọn nhiều lần. Khi đó, số cách chọn là số tổ hợp lặp chập k của n phần tử và có giá trị là:

$\overline{C_n^k} = \binom{n+k-1}{k} = \frac{(n+k-1)!}{k!(n-1)!}$

Một tính chất thú vị về số tổ hợp có lặp: $\overline{C_n^k}$ là số nghiệm nguyên không âm của phương trình: $x_1 + x_2 + \ldots + x_n = k$ với k là hằng số nguyên dương.

Vector nhị phân (Binary Vectors)

Vector nhị phân là kiểu dữ liệu `<bitset>` trong C++ STL.

Ngoài ra, các tính chất về tổ hợp của vector nhị phân cũng rất quan trọng. Sau đây là một số tính chất hay được sử dụng của vector nhị phân.

1. Số lượng vector nhị phân độ dài n là 2^n.
2. Số lượng vector nhị phân độ dài n có k số 1 là $\binom{n}{k}$, vì ta chọn k vị trí có số 1 trong n vị trí.
3. Số lượng cặp vector nhị phân $(a; b)$ (có quan tâm đến thứ tự) thỏa mãn điều kiện khoảng cách giữa a và b là k là $\binom{n}{k}.2^n$.

Khoảng cách giữa hai vector nhị phân a và b là số lượng giá trị i nguyên không âm thỏa mãn $a_i \neq b_i$

Hệ thức truy hồi (Recurrence Relations)

Hệ thức truy hồi là công cụ hỗ trợ đắc lực trong các bài toán đếm. Truy hồi còn giúp ta định nghĩa được nhiều cấu trúc như cây, danh sách, công thức quy hoạch động hay các thuật toán "chia để trị", nên truy hồi được sử dụng rất nhiều trong tin học.

Hệ thức truy hồi là một phương trình dùng để xác định dãy số hoặc hàm số bằng cách dùng các số hạng trước để định nghĩa số hạng sau. Nó rất hữu dụng vì nhiều dãy số có thể dễ dàng được định nghĩa bằng hệ thức truy hồi:

- **Hàm đa thức (Polynomials)**: $a_n = a_{n-1} + 1, a_1 = 1 \rightarrow a_n = n$.
- **Hàm mũ (Exponentials)**: $a_n = 2a_{n-1}, a_1 = 2 \rightarrow a_n = 2^n$.
- Giai thừa: $a_n = n.\, a_{n-1}, a_1 = 1 \rightarrow a_n = n!$.

Ta thường dễ dàng tìm được hệ thức truy hồi để giải các bài toán đếm. Giải hệ thức truy hồi để có được dạng công thức cần tìm là cả một nghệ thuật, tuy vậy ta có thể sử dụng trực tiếp hệ thức truy hồi để giải một số bài toán đơn giản.

Hệ số nhị thức (Binomial Coefficients)

Hệ số nhị thức $\binom{n}{k}$ được sử dụng để đếm số cách chọn k vật trong số n vật.

Đếm số đường đi trên lưới (Paths Across a Grid)

Có bao nhiêu cách để đi từ góc trái trên của một bảng $n * m$ ô đến góc phải dưới của ô đó, nếu ta chỉ được phép đi về bên phải hoặc đi xuống dưới. Ta thấy mọi đường đi hợp lệ có $n + m$ bước, và hai đường đi khác nhau nếu và chỉ nếu chúng có một bước đi xuống dưới khác nhau, vậy ta có $\binom{n+m}{n}$ cách đi.

Tính hệ số nhị thức có thể gây tràn số ở các bước trung gian, vì vậy ta nên tính hệ số nhị thức bằng công thức: $\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}$

```
/*
Sử dụng công thức truy hồi:
    nCr = (n-1)Cr + (n-1)C(r-1)

ta dễ dàng khởi tạo tất cả các giá trị nCr trong O(N^2).
```

```
6
7   Code có thể chạy được với n <= 5000 và mod bất kỳ (không cần nguyên tố).
8   */
9   //by Tanmay Chaudhari
10  #include <bits/stdc++.h>
11  using namespace std;
12
13  const int MOD = 1e9 + 7;
14  long long ncr[5005][5005];
15
16  void precompute()
17  {
18      int k;
19      for (int i = 0; i < 5001; i++)
20      {
21          ncr[i][0] = ncr[i][i] = 1;
22          k = i >> 1;
23          for (int j = 1; j < k + 1; j++)
24              ncr[i][j] = ncr[i][i - j] = (ncr[i - 1][j] + ncr[i - 1][j - 1]) % MOD;
25      }
26  }
27
28  int main()
29  {
30      precompute();
31      cout << ncr[4892][231] << endl;
32      return 0;
33  }
```

Chương trình trên chỉ tính được $\binom{n}{k}$ với n nhỏ. Bạn có thể tham khảo chương trình sau để tính $\binom{n}{k} \% p$ với p là một số nguyên tố và n lớn.

Chú ý: Code sau sử dụng nghịch đảo modulo, đã được giới thiệu ở bài viết Số học 4.5

```
1   /*
2   Tính nCr trong O(N) với mod P nguyên tố.
3
4   Ta sử dụng công thức nCr = n! / r! / (n-r)!
5
6   Khởi tạo trước fac[i] = i! mod P
7   Khởi tạo trước ifac[i] = i!^-1 mod P (nghịch đảo modulo P của i!).
8
9   Từ đó dễ dàng tính được nCr trong O(1).
10  */
11  //by Tanmay Chaudhari
12  #include <bits/stdc++.h>
13  using namespace std;
14
15  const int MOD = 1e9 + 7;
16  #define N 2123456
17  #define LL long long
18
19  LL fac[N], ifac[N];
20
21  LL PowerMod(LL a, LL n){
22      LL ret = 1;
23      while (n){
24          if (n & 1){
25
```

```
            ret *= a;
            ret %= MOD;
        }
        a *= a;
        a %= MOD;
        n /= 2;
    }
    return ret;
}

inline void precompute(){
    int i;
    fac[0] = 1;
    for (i = 1; i < N; i++){
        fac[i] = (i * fac[i - 1]) % MOD;
    }
    ifac[N - 1] = PowerMod(fac[N - 1], MOD - 2);
    for (i = N - 2; i >= 0; i--){
        ifac[i] = ((i + 1) * ifac[i + 1]) % MOD;
    }
}

LL com(int n, int r){
    LL ret = fac[n];
    ret *= ifac[r];
    ret %= MOD;
    ret *= ifac[n - r];
    ret %= MOD;
    return ret;
}

int main()
{
    precompute();
    cout << com(4892,231) << endl;
    return 0;
}
```

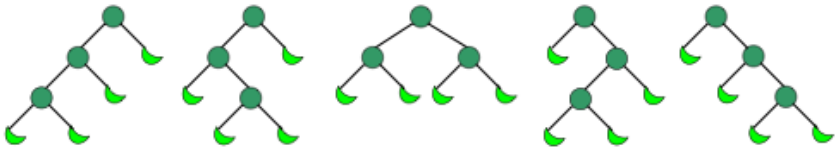
Một vài dãy số cơ bản (Counting Sequences)

Dãy số Catalan

Định nghĩa: $C_n = \sum_{k=0}^{n-1} C_k C_{n-1-k} = \frac{1}{n+1}\binom{2n}{n}$

Các ứng dụng của dãy số Catalan:

- Cho một đa giác lồi $n+2$ đỉnh, ta chia đa giác thành các tam giác bằng cách vẽ các đường chéo không cắt nhau trong đa giác. C_n là số cách chia như vậy.
- C_n là số các từ Dyck có độ dài $2n$. Từ Dyck là từ gồm n chữ X và n chữ Y sao cho với mọi tiền tố của từ đó, số lượng chữ X không ít hơn số lượng chữ Y. Ví dụ, $C_3 = 5$ từ sau đây là từ Dyck độ dài 6: XXXYYY, XYXXYY, XYXYXY, XXYYXY, XXYXYY.
- Thay mỗi chữ X trong từ Dyck thành dấu mở ngoặc đơn, mỗi chữ Y thành dấu đóng ngoặc đơn, khi đó mỗi từ Dyck trở thành một dãy ngoặc hợp lệ. Vậy C_n còn đếm số dãy ngoặc hợp lệ gồm n cặp ngoặc: ((())), ()(()), ()()(), (())(), (()()).
- C_n còn là số cách đặt ngoặc cho một biểu thức gồm $n+1$ thừa số. Ví dụ với $n = 3$, ta có 5 cách đặt ngoặc cho 4 thừa số: $((ab)c)d), (a(bc))d), (ab)(cd), a((bc)d), a(b(cd))$.
- C_n còn là số cây nhị phân đầy đủ có n là (một cây nhị phân được gọi là đầy đủ nếu mọi nút của nó có hai nút con hoặc không có nút con nào).



Và còn nhiều ứng dụng khác...

Số Euler

Số Euler $\left\langle {n \atop k} \right\rangle$ là số lượng hoán vị các số từ 1 đến n mà có đúng k phần tử lớn hơn phần tử đứng trước nó. Hệ thức truy hồi tính số Euler được lập bằng cách xét mỗi hoán vị p của 1,2,...,$n-1$. Có n vị trí để thêm số n vào hoán vị, và một cách thêm số n có thể làm tăng số phần tử lớn hơn phần tử đứng trước nó của p, hoặc bảo toàn số phần tử lớn hơn phần tử đứng trước nó. Vì vậy,

$$\left\langle {n \atop k} \right\rangle = k\left\langle {n-1 \atop k} \right\rangle + (n-k+1)\left\langle {n-1 \atop k-1} \right\rangle$$

Bài toán phân tích một số nguyên (Integer Partitions)

Ta xét bài toán sau:

Cho một số nguyên n. Hãy cho biết có bao nhiêu cách phân tích số n thành tổng của dãy các số nguyên dương, các cách phân tích là hoán vị của nhau chỉ tính là một cách.

Ví dụ: $n = 5$ có 7 cách phân tích:

1. $5 = 1 + 1 + 1 + 1 + 1$
2. $5 = 1 + 1 + 1 + 2$
3. $5 = 1 + 1 + 3$
4. $5 = 1 + 2 + 2$
5. $5 = 1 + 4$
6. $5 = 2 + 3$
7. $5 = 5$

Cách dễ nhất để đếm số cách phân tích số n là định nghĩa hàm $f(n, k)$ là số cách phân tích số n thành tổng của các số nguyên dương nhỏ hơn hoặc bằng k. Các cách phân tích số n thành tổng cách số nguyên dương nhỏ hơn hoặc bằng k có thể chia làm hai loại: chứa số k trong phép phân tích và không chứa số k trong phép phân tích, vì thế ta có $f(n, k) = f(n - k, k) + f(n, k - 1)$. Ngoài ra, ta còn dễ dàng có được $f(1, 1) = 1$ và $f(n, k) = 0$ với $k > n$.

Bài tập

- Hackerearth - Ankit and Race Team
- Hackerearth - Tic Tac Toe
- Hackerearth - Roy and Little Mario
- VNOJ - TREELINE