- 원소의 중복이 허용되는 배열을 채우는 경우의 수 https://www.acmicpc.net/problem/1234

ex) N개의 배열을 A 원소 3개, B 원소 2개, C 원소 4개로 채우는 경우의 수 = **N! / (3!\*2!\*4!)**

- M개의 원소 중 N개를 고르는 경우의 수 **nCr (n combination r) = n! / r! \* (n-r)!**

- 다익스트라 알고리즘을 이용해 최단 왕복 경로 찾기

ex) a,b,c,d,X 접점이 있고 간선 정보가 주어질 때에 원래 주어진 간선 정보로 출발점을 X로 두면 X에서 각 접점으로

이동하는 최단거리, 간선 정보의 방향을 바꾸어 출발점을 X로 두면 각 접점에서 X로 이동하는 최단거리

- ‘목적을 이루는 가장 적은 횟수를 구하시오’ 류의 문제는 상태 변환에 드는 노력을 간선으로 각 상태를 접점으로 바꾸어 생각하면

다익스트라 알고리즘 문제로 바꿀 수 있다. http://koitp.org/problem/PARTY2/read/

- 멍텅구리 덧셈의 경우 뒤에서 더해지는 분수를 모두 비교해 볼 필요 없이 분모로 묶어서 분자가 가장 큰 것들만 비교하면 덧셈을 하였을

때에 가장 커지는 경우를 구하는 탐색 수를 줄일 수 있다. 멍텅구리 덧셈 ( 1/3 + 1/2 = 2/5)

- 원의 표면의 각 점에서 이어지는 선이 있고 선들을 선택하여 겹치지 않는 선을 가장 많은 원을 만들 수 있는 경우를 구하는 경우

원을 직선으로 펼치고 각 구간 당 온전한 선을 가장 많이 만들 수 있는 경우로 문제를 바꾸어 DP로 해결

https://www.acmicpc.net/problem/2673

- 정보들이 주어지는 문제에서 N번째 단계를 생각할 때에 N-1 과의 관계만 생각해도 되는 경우 입력을 받을 때에 Priority Queue를 이용하여

간단히 해결할 수 있는 경우들이 있다. <http://koitp.org/problem/SDS_PRO_6_6/read/>

- N Queen 문제 <http://koitp.org/problem/NQUEEN/read/>

- **모듈러 연산 법칙** (http://kwony91.tistory.com/99)

( ( A % MOD ) + ( B % MOD ) ) % MOD = ( A + B ) % MOD

( ( A % MOD ) – ( B % MOD ) ) % MOD = ( A – B ) % MOD

( ( A % MOD ) \* ( B % MOD ) ) % MOD = ( A \* B ) % MOD

**( A / B ) % MOD = ( A \* POW(B, MOD -2)) % MOD** : modInv( K , MOD ) = POW ( K, MOD – 2 ) % MOD

: A / B % MOD 의 경우 A / B 를 구한 후 modular를 하지 않고 A 에다가 K의 곱셉 역원을 곱하는 방식으로 한다.

Module 연산을 고려한 빠른 POW 함수

**long long POW(long long a, long long b) {**

**if (b == 1) return a;**

**if (b == 0) return 1;**

**long long tmp = POW(a, b / 2);**

**if (b % 2) return (long long)((long long)((tmp\*tmp) % MOD)\*a) % MOD;**

**else return (long long)(tmp\*tmp) % MOD;**

**}**