14. Mathematics #2

2017010698
수학과 오서영

목차

1. binomial coefficient
-> Pascal's triangle
2. Catalan Number
3. Euler's phi function
4. Fermat's little theorem
5. Gaussian elimination
6. Discrete Mathematics
-> The Pigeonhole Principle
7. Stirling numbers of the second kind

1. binomial coefficient

binomial coefficient

: 이항계수 (binomial coefficient)는 주어진 크기의 (순서 없는) 조합의 가짓수

$$ {}_{n}C_{k} = \frac{n!}{k!(n-k)!} \qquad {}_{n}C_{k} = \begin{cases} {}_{n-1}C_{k} + {}_{n-1}C_{k-1} & (0<k<n) \\ 1 & (k=0 \text{ or } k=n) \end{cases} $$

binomial coefficient

```
int recursiveBinomial(int n, int k)
{
    if (k == 0 || k == n)
        return 1;
    else
        return recursiveBinomial(n-1, k-1) + bin(n-1, k);
}
# 더 나은 코드 : 동적계획법 – 메모이제이션 사용하기
```

1. binomial coefficient

파스칼의 삼각형 (Pascal's triangle)

: 이항계수를 삼각형 모양의 기하학적 형태로 배열한 것

```
                                    1                        -> n=0 일때
                                 1     1                     -> n=1 일때
                              1     2     1                  -> n=2 일때
                           1     3     3     1                   .
                        1     4     6     4     1                .
                     1     5    10    10     5     1             .
                  1     6    15    20    15     6     1
               1     7    21    35    35    21     7     1
            1     8    28    56    70    56    28     8     1
         1     9    36    84   126   126    84    36     9     1
      1    10    45   120   210   252   210   120    45    10     1
    1    11    55   165   330   462   462   330   165    55    11     1
  1    12    66   220   495   792   924   792   495   220    66    12     1
 1   13    78   286   715  1287  1716  1716  1287   715   286    78    13    1
1  14   91   364  1001  2002  3003  3432  3003  2002  1001  364   91   14   1
1  15  105  455  1365  3003  5005  6435  6435  5005  3003  1365  455  105  15  1
1  16  120  560  1820  4368  8008  11440  12870  11440  8008  4368  1820  560  120  16  1
1  17  136  680  2380  6188  12376  19448  24310  24310  19448  12376  6188  2380  680  136  17  1
1  18 153  816  3060  8568  18564  31824  43758  48620  43758  31824  18564  8568  3060  816  153  18  1
```

2. Catalan Number

Catalan Number

: 쌍을 이루는 것들을 나열하는 모든 경우의 수.

(ex) 잘 짜인 괄호

괄호 ()는 열고 닫았을 때, 잘 짜였다고 한다.
예를 들어 (())은 잘 짜인 것이지만
()))(은 잘못 짜인 것이다.
n쌍의 ()를 잘 짜인 모양으로 늘어 놓는 방법 수는?

$n = 0$:	*	1 way
$n = 1$:	()	1 way
$n = 2$:	()(), (())	2 ways
$n = 3$:	()()(), ()(()), (())(), (()()), ((()))	5 ways
$n = 4$:	()()()(), ()()(()), ()(())(), ()(()()), ()((())), (())()(), (())(()), (()())(), ((()))(), (()()()), (()(())), ((())()), ((()())), (((())))	14 ways
$n = 5$:	()()()()(), ()()()(()), ()()(())(), ()()(()()), ()()((())), ()(())()(), ()(())(()), ()(()())(), ()((()))(), ()(()()()), ()(()(())), ()((())()), ()((()())), ()(((()))), (())()()(), (())()(()), (())(())(), (())(()()), (())((())), (()())()(), (()())(()), ((()))()(), ((()))(()), (()()())(), (()(()))(), ((())())(), ((()()))(), (((())))(), (()()()()), (()()(())), (()(())()), (()(()())), (()((()))), ((())()()), ((())(())), ((()())()), (((()))()), ((()()())), ((()(()))), (((())())), (((()()))), ((((()))))	42 ways

2. Catalan Number

점화식 찾기

n 쌍의 괄호를 잘 짜는 방법의 수 : Cn
C0,C1,C2,⋯,Cn−1으로 Cn을 나타내는 방법은?.
n−1쌍의 괄호가 잘 짜여진 것에
()를 알맞은 곳에 넣어주는 방법을 세면 된다.
(A)B와 같이 넣는다고 하면 A와 B도 잘 짜여져 있어야 하고 만일 A에 괄호가 k쌍이 있다면 B에는 n−1−k쌍이 있어야 한다.
이제 문제는 A와 B로 나누는 방법을 세면 된다.

$$C_1 = C_0C_0$$

$$C_2 = C_0C_1 + C_1C_0$$

$$C_3 = C_0C_2 + C_1C_1 + C_2C_0$$

$$C_4 = C_0C_3 + C_1C_2 + C_2C_1 + C_3C_0$$

3. Euler's phi function

Euler's phi function

: 1부터 n까지의 양의 정수 중에
n과 서로소인 것의 개수를 나타내는 함수

$$\varphi(n) = n \prod_{p|n} (1 - 1/p)$$

Euler's phi function

```
pi[i] = i;
for (int i=2;i<=N;i++) {
    if (pi[i] == i)  { // 소수인 경우
        pi[i] = i - 1;
        for (int j=i+i; j<=N; j+=i)
            pi[j] = pi[j] * (i-1) / i;
}
}
```

4. Fermat's little theorem

Fermat's little theorem

: p 가 소수이고 a 가 p 로 나누어지지 않는 정수이면,
Then a^(p−1) ≡ 1 (mod p)

모든 a 에 대하여,
we have a^p ≡ a (mod p)

(ex) $7^{222} \bmod 11$ 을 계산하라.

$$\begin{aligned} 7^{11-1} &\equiv 1(\bmod 11) \\ 7^{10} &\equiv 1(\bmod 11) \end{aligned}$$

$$7^{222} = 7^{22\times 10} \times 7^2$$

$$\begin{aligned} 7^{222} &\equiv 1^{22} \times 7^2(\bmod 11) \\ &\equiv 49(\bmod 11) \\ &\equiv 5(\bmod 11) \end{aligned}$$

5. Gaussian elimination

Gaussian elimination

: 행렬을 사용해 선형 연립 방정식을 푸는 방법

(ex)
$$\begin{aligned} 2x + 4y + 6z &= 18 \\ 2x - y + z &= 8 \\ 3x - z &= 3 \end{aligned}$$

1. 첫 번째 식에 적절한 수를 곱해 두 번째 식에 더하거나 빼준다.

[2] - [1]

결과: [2] : -5y -5z = -10

2. 두 번째 식에 적절한 수를 곱해준다

[2] * (-1/5)

결과: [2] : y + z = 2

3. 첫 번째 식에 적절한 수를 곱해준다

[1] * (1/2)

결과: [1] : x+ 2y + 3z = 9

4. 첫 번째 식에 적절한 수를 곱해 세 번째 식에 더하거나 빼준다.

[3] – 3*[1]

결과: -6y -10z = 24

5. Gaussian elimination

5. 세 번째 식에 적절한 수를 곱해준다.

[3] * (-2)

결과: [3] : 3y + 5z = 12

6. 두 번째 식에 적절한 수를 곱해 세 번째 식에 더하거나 빼준다.

[3] – 3*[2]

결과: 2z = 6

…

$$\begin{matrix} [\,1 & 0 & 0 & 2\,] \\ [\,0 & 1 & 0 & -1\,] \\ [\,0 & 0 & 1 & 3\,] \end{matrix}$$

대각선 아래의 값들만 0

-> 각 변수들에 대한 Unique한 해를 구할 수 있다.

6. Discrete Mathematics -> Pigeonhole Principle

비둘기집의 원리

: n개의 비둘기집과
n+1마리의 비둘기가 있다고 가정
-> 어느 비둘기집에는
두 마리 이상의 비둘기가 있다

(ex) 뉴욕시의 모든 주민들은 그의 머리카락 수가
뉴욕시 전체 주민수보다 적다고 한다.
뉴욕시 주민들 중 대머리가 없다고 가정할 때,
머리카락 수가 같은 주민이 적어도 두 사람 있음을 보여라.

7. Stirling numbers of the second kind

집합의 분할 (제2종 스털링 수) = S(n, k)

: n 개의 원소를 가진 집합을
k 개의 부분집합으로 나누는 경우의 수

- 각 부분집합은 공집합이어서는 안되며,
각 부분집합끼리 교집합이 있어서는 안된다.

(ex) $S(3,2) = 3$

점화식

$$S(n,k) = S(n-1, k-1) + kS(n-1, k)$$

7. Stirling numbers of the second kind

Stirling numbers

```
def stirling(i,j) :
    global array
    global col

if j == 0 or j == i:
        array[i][j] = 1
    else :
        array[i][j] = (j + 1) * array[i-1][j] + array[i-1][j-1];

    for k in range(0, j+1) :
        array[i-1][k] = 0;
```

Reference

[1] [알고리즘/Algorithm] 이항계수(Binomial Coefficient),
https://m.blog.naver.com/gywhd8928/220513756901

[2] 카탈란 수(catalan number),
https://suhak.tistory.com/77

[3] 오일러 파이 함수,
https://makesource.tistory.com/entry/%EC%98%A4%EC%9D%BC%EB%9F%AC-%ED%8C%8C%EC%9D%B4-%ED%95%A8%EC%88%98

[4] 페르마의 소정리,
https://johngrib.github.io/wiki/Fermat-s-little-theorem/

[5] https://github.com/jtjisgod/stirling-number-of-the-second-kind---python/blob/master/getS.py