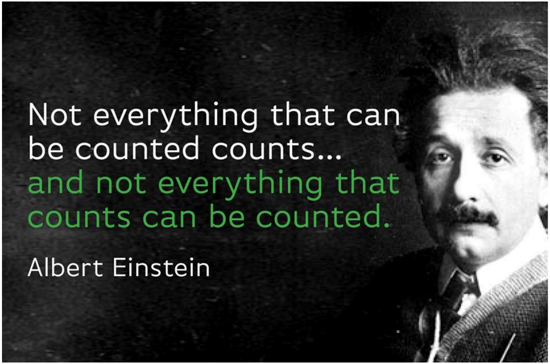
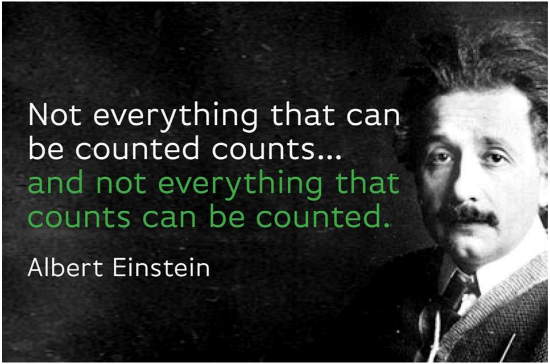
2-3 Counting

魏恒峰

hfwei@nju.edu.cn

2018 年 04 月 11 日


Not everything that can
be counted counts...
and not everything that
counts can be counted.
Albert Einstein



所以，学好 “2-3 组合与计数” 是多么重要!

Paring up (CS : $1.2 - 15$)

A tennis club has $2n$ members. We want to pair up the members by twos for singles matches.

(a) In how many ways can we pair up all the members of the club?

(b) Suppose that we also determine who serves first for each pairing. In how many ways can we specify our pairs?

Paring up (CS : $1.2-15$)

A tennis club has $2n$ members. We want to pair up the members by twos for singles matches.

(a) In how many ways can we pair up all the members of the club?

(b) Suppose that we also determine who serves first for each pairing. In how many ways can we specify our pairs?

$$\frac{1}{n!}\binom{2n}{\underbrace{2,2,\cdots,2}_{n}}$$

Paring up $(\text{CS}: 1.2 - 15)$

A tennis club has $2n$ members. We want to pair up the members by twos for singles matches.

(a) In how many ways can we pair up all the members of the club?

(b) Suppose that we also determine who serves first for each pairing. In how many ways can we specify our pairs?

$$\frac{1}{n!}\binom{2n}{\underbrace{2, 2, \cdots, 2}_{n}} = \frac{(2n)!}{\underbrace{2^n}_{\text{intra-pair}} \cdot \underbrace{n!}_{\text{inter-pair}}}$$

Paring up (CS : $1.2 - 15$)

A tennis club has $2n$ members. We want to pair up the members by twos for singles matches.

(a) In how many ways can we pair up all the members of the club?

(b) Suppose that we also determine who serves first for each pairing. In how many ways can we specify our pairs?

$$\frac{1}{n!}\binom{2n}{\underbrace{2, 2, \cdots, 2}_{n}} = \frac{(2n)!}{\underbrace{2^n}_{\text{intra-pair}} \cdot \underbrace{n!}_{\text{inter-pair}}}$$

$$\frac{(2n)!}{2^n \cdot n!} \cdot 2^n = \frac{(2n)!}{n!}$$

Passing out Apples to Children

k-Permutation (CS : $1.2 - 5$)

We need to pass out k distinct apples (pieces of fruit) to n children such that *each child may get at most one apple*.

(a) $k \leq n$?

(b) What if $k > n$?

k-Permutation $(\text{CS} : 1.2 - 5)$

We need to pass out k distinct apples (pieces of fruit) to n children such that *each child may get at most one apple*.

(a) $k \le n$?

(b) What if $k > n$?

$$n^{\underline{k}} \triangleq n(n-1)\cdots(n-k+1)$$

k-Permutation $(\text{CS} : 1.2 - 5)$

We need to pass out k distinct apples (pieces of fruit) to n children such that *each child may get at most one apple*.

(a) $k \le n$?

(b) What if $k > n$?

$$n^{\underline{k}} \triangleq n(n-1)\cdots(n-k+1)$$

$$0$$

Multisets (CS : $1.5 - 4$)

Use multisets to determine the number of ways to pass out k **identical** apples to n children. Assume that a child may get more than one apple.

Multisets (CS : $1.5 - 4$)

Use multisets to determine the number of ways to pass out k **identical** apples to n children. Assume that a child may get more than one apple.

x_i : the # of apples the i-th child gets

$$x_1 + x_2 + \cdots + x_n = k, \quad x_i \geq 0$$

Multisets (CS : $1.5 - 4$)

Use multisets to determine the number of ways to pass out k **identical** apples to n children. Assume that a child may get more than one apple.

x_i : the # of apples the i-th child gets

$$x_1 + x_2 + \cdots + x_n = k, \quad x_i \geq 0$$

Integer composition

Multisets (CS : $1.5-4$)

Use multisets to determine the number of ways to pass out k **identical** apples to n children. Assume that a child may get more than one apple.

x_i : the # of apples the i-th child gets

$$x_1 + x_2 + \cdots + x_n = k, \quad x_i \geq 0$$

Integer composition (The order matters!)

Multisets (CS : $1.5-4$)

Use multisets to determine the number of ways to pass out k **identical** apples to n children. Assume that a child may get more than one apple.

x_i : the # of apples the i-th child gets

$$x_1 + x_2 + \cdots + x_n = k, \quad x_i \geq 0$$

Integer composition (The order matters!)

$$y_i \triangleq x_i + 1$$

Multisets (CS : $1.5 - 4$)

Use multisets to determine the number of ways to pass out k **identical** apples to n children. Assume that a child may get more than one apple.

x_i : the # of apples the i-th child gets

$$x_1 + x_2 + \cdots + x_n = k, \quad x_i \geq 0$$

Integer composition (The order matters!)

$$y_i \triangleq x_i + 1$$

$$y_1 + y_2 + \cdots + y_n = n + k, \qquad y_i \geq 1$$

Multisets (CS : $1.5 - 4$)

Use multisets to determine the number of ways to pass out k **identical** apples to n children. Assume that a child may get more than one apple.

x_i : the # of apples the i-th child gets

$$x_1 + x_2 + \cdots + x_n = k, \quad x_i \geq 0$$

Integer composition (The order matters!)

$$y_i \triangleq x_i + 1$$

$$y_1 + y_2 + \cdots + y_n = n + k, \qquad y_i \geq 1$$

$$\binom{n+k-1}{n-1} = \binom{n+k-1}{k}$$

Multisets $(\text{CS} : 1.5 - 4)$

Use **multisets** to determine the number of ways to pass out k identical apples to n children. Assume that a child may get more than one apple.

Multisets ($\text{CS} : 1.5 - 4$)

Use **multisets** to determine the number of ways to pass out k identical apples to n children. Assume that a child may get more than one apple.

Q : k-multiset of $[1 \cdots n]$ *vs.* n-multiset of $[1 \cdots k]$

Multisets (CS : $1.5-4$)

Use **multisets** to determine the number of ways to pass out k identical apples to n children. Assume that a child may get more than one apple.

Q : k-multiset of $[1 \cdots n]$ *vs.* n-multiset of $[1 \cdots k]$

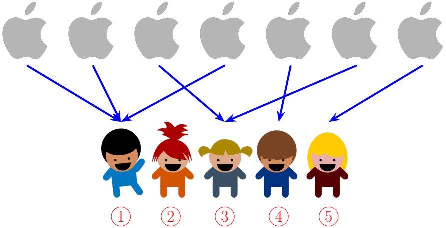
$$k = 7 \qquad n = 5$$

Multisets (CS : $1.5-4$)

Use **multisets** to determine the number of ways to pass out k identical apples to n children. Assume that a child may get more than one apple.

Q : k-multiset of $[1\cdots n]$ *vs.* n-multiset of $[1\cdots k]$

$$k=7 \qquad n=5$$

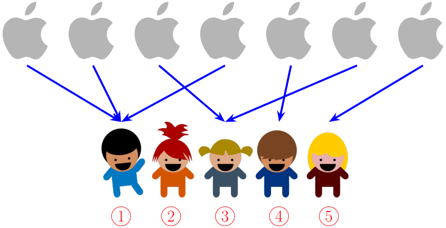


Multisets (CS : $1.5-4$)

Use **multisets** to determine the number of ways to pass out k identical apples to n children. Assume that a child may get more than one apple.

Q : k-multiset of $[1 \cdots n]$ *vs.* n-multiset of $[1 \cdots k]$

$$k = 7 \qquad n = 5$$



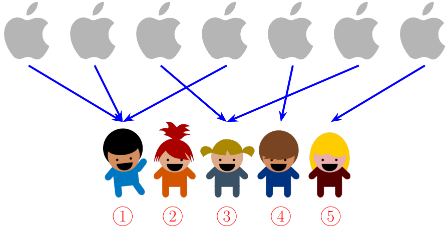
$$\{1, 1, 1, 3, 3, 4, 5\}$$

Multisets (CS : $1.5 - 4$)

Use **multisets** to determine the number of ways to pass out k identical apples to n children. Assume that a child may get more than one apple.

Q : k-multiset of $[1 \cdots n]$ *vs.* n-multiset of $[1 \cdots k]$

$$k = 7 \qquad n = 5$$



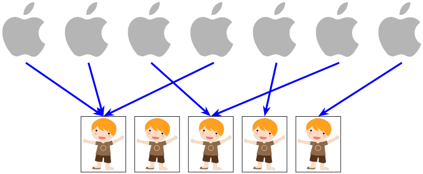
$$\{1, 1, 1, 3, 3, 4, 5\}$$

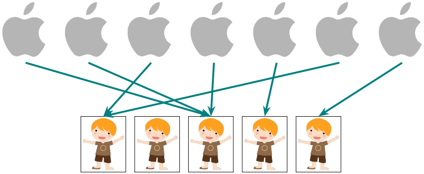
Integer Partition (CS : $1.5 - 4$ Extended)

What is the number of ways to pass out k identical apples to n-胞胎.
Assume that a child may get more than one apple.

Integer Partition (CS : $1.5 - 4$ Extended)

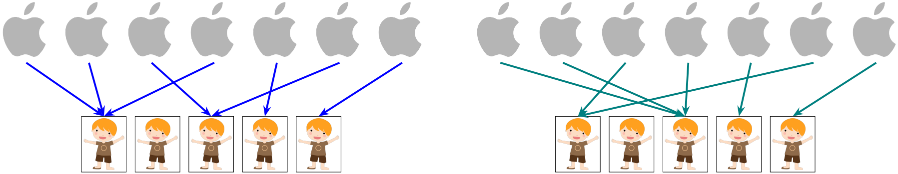
What is the number of ways to pass out k identical apples to n-胞胎.
Assume that a child may get more than one apple.





Integer Partition (CS : $1.5-4$ Extended)

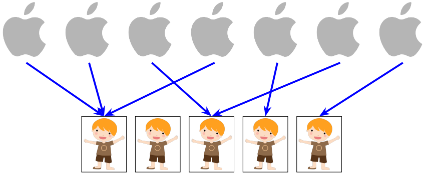
What is the number of ways to pass out k identical apples to n-胞胎.
Assume that a child may get more than one apple.

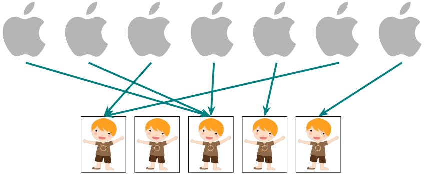


Integer partition of k into $\leq n$ parts

Integer Partition (CS : $1.5 - 4$ Extended)

What is the number of ways to pass out k identical apples to n-胞胎.
Assume that a child may get more than one apple.

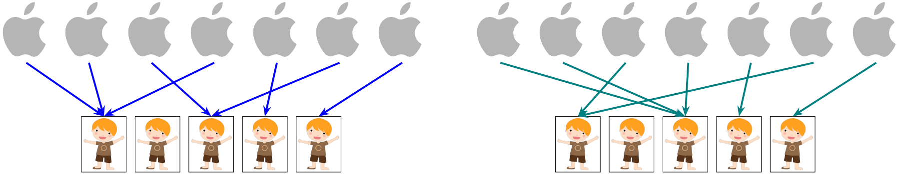




Integer partition of k into $\leq n$ parts (The order does not matter!)

Integer Partition (CS : $1.5-4$ Extended)

What is the number of ways to pass out k identical apples to n-胞胎.
Assume that a child may get more than one apple.



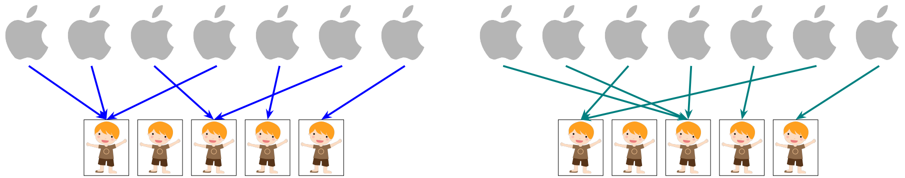
Integer partition of k into $\leq n$ parts (The order does not matter!)

Theorem ( )

$$p(k) \triangleq \sum_{x=1}^{x=k} p_x(k) \sim \frac{1}{4\sqrt{3}k} \exp\left(\pi\sqrt{\frac{2k}{3}}\right)$$

Integer Partition (CS : $1.5-4$ Extended)

What is the number of ways to pass out k identical apples to n-胞胎.
Assume that a child may get more than one apple.



Integer partition of k into $\leq n$ parts (The order does not matter!)

Theorem (G. H. Hardy, Ramanujan (1918))

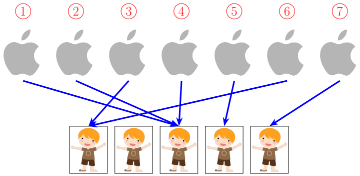
$$p(k) \triangleq \sum_{x=1}^{x=k} p_x(k) \sim \frac{1}{4\sqrt{3}k} \exp\left(\pi\sqrt{\frac{2k}{3}}\right)$$

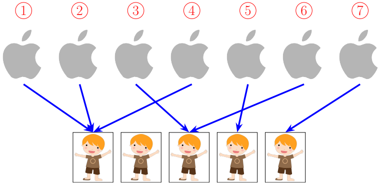
Set Partition (CS : $1.5 - 4$ Extended)

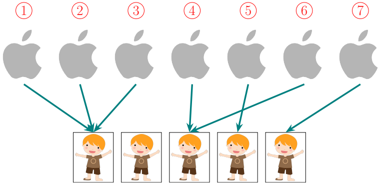
What is the number of ways to pass out k distinct apples to n-胞胎. Assume that a child may get more than one apple.

Set Partition (CS : $1.5-4$ Extended)

What is the number of ways to pass out k distinct apples to n-胞胎.
Assume that a child may get more than one apple.

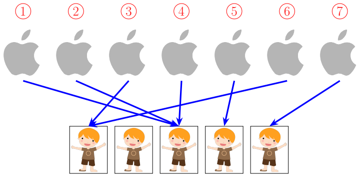


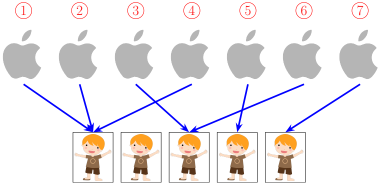


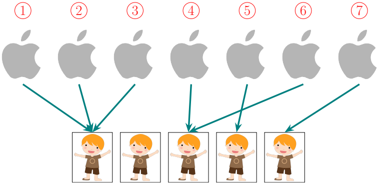


Set Partition (CS : $1.5-4$ Extended)

What is the number of ways to pass out k distinct apples to n-胞胎.
Assume that a child may get more than one apple.







Set partition of $[1 \cdots k]$ into $\leq n$ parts

Set Partition (CS : $1.5 - 12$)

$S(n,k)$ ($\left\{ n \atop k \right\}$) : # of set partitions of $[1 \cdots n]$ into k classes

Set Partition ($\mathrm{CS}: 1.5-12$)

$S(n,k)$ ($\left\{ n \atop k \right\}$) : # of set partitions of $[1 \cdots n]$ into k classes

Stirling number of the second kind

Set Partition $(\text{CS} : 1.5 - 12)$

$$S(n,k) \ (\left\{ {n \atop k} \right\}) : \# \text{ of set partitions of } [1 \cdots n] \text{ into } k \text{ classes}$$

Stirling number of the second kind

Theorem (Recurrence for $S(n,k)$)

$$S(0,0) = 1, \quad S(n,0) = S(0,n) = 0 \ (n > 0)$$

$$S(n,k) = S(n-1,k-1) + kS(n-1,k), \quad n > 0, k > 0$$

Set Partition $(\text{CS}: 1.5-12)$

$$S(n,k)\ \left(\left\{ {n \atop k} \right\}\right): \#\text{ of set partitions of } [1\cdots n] \text{ into } k \text{ classes}$$

Stirling number of the second kind

Theorem (Recurrence for $S(n,k)$)

$$S(0,0)=1, \quad S(n,0)=S(0,n)=0\ (n>0)$$

$$S(n,k)=S(n-1,k-1)+kS(n-1,k), \quad n>0, k>0$$

Proof.

$$S(n,k)=\underbrace{S(n-1,k-1)}_{n \text{ is alone}}+\underbrace{kS(n-1,k)}_{n \text{ is not alone}}$$

□

$$\text{Bell number: } B_n = \sum_{k=0}^{k=n} \left\{ {n \atop k} \right\}$$

$$\text{Bell number: } B_n = \sum_{k=0}^{k=n} \left\{ {n \atop k} \right\}$$

Theorem (Berend & Tassa (2010))

$$B_n < \left(\frac{0.792n}{\ln(n+1)} \right)^n, n \in \mathbb{Z}^+$$

$$\text{Bell number: } B_n = \sum_{k=0}^{k=n} \left\{ {n \atop k} \right\}$$

Theorem (Berend & Tassa (2010))

$$B_n < \left(\frac{0.792n}{\ln(n+1)} \right)^n, n \in \mathbb{Z}^+$$

Theorem (de Bruijn (1981))

As $n \to \infty$,

$$\frac{\ln B_n}{n} = \ln n - \ln\ln n - 1 + \frac{\ln\ln n}{\ln n} + \frac{1}{\ln n} + \frac{1}{2}\left(\frac{\ln\ln n}{\ln n}\right)^2 + O\left(\frac{\ln\ln n}{(\ln n)^2}\right)$$

THE TWELVEFOLD WAY

balls per urn	unrestricted	≤ 1	≥ 1
n labeled balls, m labeled urns	n-tuples of m things	n-permutations of m things	partitions of $\{1,\ldots,n\}$ into m ordered parts
n unlabeled balls, m labeled urns	n-multicombinations of m things	n-combinations of m things	compositions of n into m parts
n labeled balls, m unlabeled urns	partitions of $\{1,\ldots,n\}$ into $\leq m$ parts	n pigeons into m holes	partitions of $\{1,\ldots,n\}$ into m parts
n unlabeled balls, m unlabeled urns	partitions of n into $\leq m$ parts	n pigeons into m holes	partitions of n into m parts

Computing $\binom{n}{k}$ (CS $1.5 : 14$)

procedure $\textsc{Binom}(n, k)$ ▷ Required: $n \ge k \ge 0$
  if $k = 0 \lor n = k$ **then**
    return 1
  return $\textsc{Binom}(n-1, k) + \textsc{Binom}(n-1, k-1)$

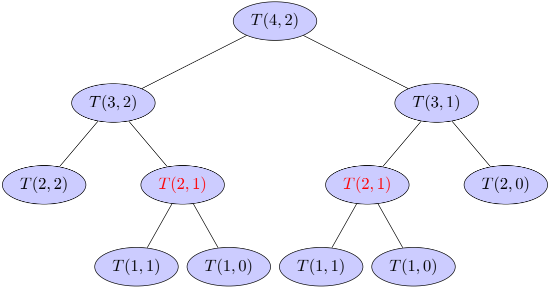
Computing $\binom{n}{k}$ (CS $1.5:14$)

procedure $\textsc{Binom}(n, k)$ ▷ Required: $n \geq k \geq 0$

  if $k = 0 \lor n = k$ **then**

    return 1

  return $\textsc{Binom}(n-1, k) + \textsc{Binom}(n-1, k-1)$



procedure $\textsc{Binom}(n, k)$ ▷ Required: $n \geq k \geq 0$
    if $k = 0 \lor n = k$ **then**
        return 1
    return $\textsc{Binom}(n-1, k) + \textsc{Binom}(n-1, k-1)$

```
procedure BINOM(n, k)                              ▷ Required: n ≥ k ≥ 0
    if k = 0 ∨ n = k then
        return 1
    return BINOM(n − 1, k) + BINOM(n − 1, k − 1)
```

(i) # of "+":

```
procedure BINOM(n, k)                          ▷ Required: n ≥ k ≥ 0
    if k = 0 ∨ n = k then
        return 1
    return BINOM(n − 1, k) + BINOM(n − 1, k − 1)
```

(i) # of "+":

$$A(n,k) = 1 + A(n-1,k) + A(n-1,k-1)$$

```
procedure BINOM(n, k)                    ▷ Required: n ≥ k ≥ 0
    if k = 0 ∨ n = k then
        return 1
    return BINOM(n − 1, k) + BINOM(n − 1, k − 1)
```

(i) # of "+":

$$A(n,k) = 1 + A(n-1,k) + A(n-1,k-1)$$

(ii) # of recursive calls of BINOM:

```
procedure Binom(n, k)                        ▷ Required: n ≥ k ≥ 0
    if k = 0 ∨ n = k then
        return 1
    return Binom(n − 1, k) + Binom(n − 1, k − 1)
```

(i) # of "$+$":

$$A(n,k) = 1 + A(n-1,k) + A(n-1,k-1)$$

(ii) # of recursive calls of $\textsc{Binom}$:

$$R(n,k) = 2 + R(n-1,k) + R(n-1,k-1)$$

---

procedure $\textsc{Binom}(n, k)$ ▷ Required: $n \geq k \geq 0$
    if $k = 0 \lor n = k$ **then**
        return 1
    return $\textsc{Binom}(n-1, k) + \textsc{Binom}(n-1, k-1)$

(i) # of "+":

$$A(n,k) = 1 + A(n-1,k) + A(n-1,k-1)$$

(ii) # of recursive calls of $\textsc{Binom}$:

$$R(n,k) = 2 + R(n-1,k) + R(n-1,k-1)$$

$$\boxed{T(n,k) = \begin{cases} \\ T(n-1,k) + T(n-1,k-1) + c, \end{cases}}$$

procedure $\textsc{Binom}(n, k)$ ▷ Required: $n \geq k \geq 0$

  if $k = 0 \vee n = k$ **then**

    return 1

  return $\textsc{Binom}(n-1, k) + \textsc{Binom}(n-1, k-1)$

(i) # of "$+$":

$$A(n, k) = 1 + A(n-1, k) + A(n-1, k-1)$$

(ii) # of recursive calls of $\textsc{Binom}$:

$$R(n, k) = 2 + R(n-1, k) + R(n-1, k-1)$$

$$\boxed{T(n, k) = \begin{cases} 0, & k = 0 \vee n = k \\ T(n-1, k) + T(n-1, k-1) + c, & \text{o.w.} \end{cases}}$$

$$
\boxed{T(n,k) = \begin{cases} 0, & k = 0 \lor n = k \\ T(n-1,k) + T(n-1,k-1) + c, & \textit{o.w.} \end{cases}}
$$

$$\boxed{T(n,k) = \begin{cases} 0, & k = 0 \lor n = k \\ T(n-1,k) + T(n-1,k-1) + c, & \textit{o.w.} \end{cases}}$$

$$T(n,k) = T(n-1,k) + T(n-1,k-1) \implies T(n,k) =$$

$$\boxed{T(n,k) = \begin{cases} 0, & k = 0 \lor n = k \\ T(n-1,k) + T(n-1,k-1) + c, & \textit{o.w.} \end{cases}}$$

$$T(n,k) = T(n-1,k) + T(n-1,k-1) \implies T(n,k) = \alpha \binom{n}{k}$$

$$\boxed{T(n,k) = \begin{cases} 0, & k = 0 \lor n = k \\ T(n-1,k) + T(n-1,k-1) + c, & \text{o.w.} \end{cases}}$$

$$T(n,k) = T(n-1,k) + T(n-1,k-1) \implies T(n,k) = \alpha \binom{n}{k}$$

$$\boxed{T(n,k) = \alpha \binom{n}{k} + \beta}$$

$$\boxed{T(n,k) = \begin{cases} 0, & k = 0 \lor n = k \\ T(n-1,k) + T(n-1,k-1) + c, & \textit{o.w.} \end{cases}}$$

$$T(n,k) = T(n-1,k) + T(n-1,k-1) \implies T(n,k) = \alpha\binom{n}{k}$$

$$\boxed{T(n,k) = \alpha\binom{n}{k} + \beta}$$

$$\alpha\binom{n}{k} + \beta = \alpha\binom{n-1}{k} + \beta + \alpha\binom{n-1}{k-1} + \beta + c \implies \beta = -c$$

$$T(n,k) = \begin{cases} 0, & k = 0 \lor n = k \\ T(n-1,k) + T(n-1,k-1) + c, & \text{o.w.} \end{cases}$$

$$T(n,k) = T(n-1,k) + T(n-1,k-1) \implies T(n,k) = \alpha\binom{n}{k}$$

$$\boxed{T(n,k) = \alpha\binom{n}{k} + \beta}$$

$$\alpha\binom{n}{k} + \beta = \alpha\binom{n-1}{k} + \beta + \alpha\binom{n-1}{k-1} + \beta + c \implies \beta = -c$$

$$\alpha\binom{n}{0} - c = 0, \quad \alpha\binom{n}{n} - c = 0 \implies \alpha = c$$

$$\boxed{T(n,k) = \begin{cases} 0, & k = 0 \lor n = k \\ T(n-1,k) + T(n-1,k-1) + c, & \text{o.w.} \end{cases}}$$

$$T(n,k) = T(n-1,k) + T(n-1,k-1) \implies T(n,k) = \alpha \binom{n}{k}$$

$$\boxed{T(n,k) = \alpha \binom{n}{k} + \beta}$$

$$\alpha \binom{n}{k} + \beta = \alpha \binom{n-1}{k} + \beta + \alpha \binom{n-1}{k-1} + \beta + c \implies \beta = -c$$

$$\alpha \binom{n}{0} - c = 0, \quad \alpha \binom{n}{n} - c = 0 \implies \alpha = c$$

$$\boxed{T(n,k) = c \binom{n}{k} - c}$$

7. 斐波那契数列的定义如下：$F_1 = 1, F_2 = 1, F_n = F_{n-1} + F_{n-2}\ (n \geq 3)$。如果用下面的函数计算斐波那契数列的第 n 项，则其时间复杂度为（　　）。

```
int F(int n)
{
    if (n <= 2)
        return 1;
    else
        return F(n - 1) + F(n - 2);
}
```

A. $O(1)$　　B. $O(n)$　　C. $O(n^2)$　　D. $O(F_n)$

7. 斐波那契数列的定义如下：$F_1=1, F_2=1, F_n=F_{n-1}+F_{n-2}\ (n\geq 3)$。如果用下面的函数计算斐波那契数列的第 n 项，则其时间复杂度为（　　）。

```
int F(int n)
{
    if (n <= 2)
        return 1;
    else
        return F(n - 1) + F(n - 2);
}
```

A. $O(1)$　　B. $O(n)$　　C. $O(n^2)$　　D. $O(F_n)$

时间复杂度 of what?

$$\binom{0}{0}$$

$$\binom{1}{0} \quad \binom{1}{1}$$

$$\binom{2}{0} \quad \binom{2}{1} \quad \binom{2}{2}$$

$$\binom{3}{0} \quad \binom{3}{1} \quad \binom{3}{2} \quad \binom{3}{3}$$

$$\binom{4}{0} \quad \binom{4}{1} \quad \binom{4}{2} \quad \binom{4}{3} \quad \binom{4}{4}$$

$$\binom{5}{0} \quad \binom{5}{1} \quad \binom{5}{2} \quad \binom{5}{3} \quad \binom{5}{4} \quad \binom{5}{5}$$

$$\binom{0}{0}$$

$$\binom{1}{0} \quad \binom{1}{1}$$

$$\binom{2}{0} \quad \binom{2}{1} \quad \binom{2}{2}$$

$$\binom{3}{0} \quad \binom{3}{1} \quad \binom{3}{2} \quad \binom{3}{3}$$

$$\binom{4}{0} \quad \binom{4}{1} \quad \binom{4}{2} \quad \binom{4}{3} \quad \binom{4}{4}$$

$$\binom{5}{0} \quad \binom{5}{1} \quad \binom{5}{2} \quad \binom{5}{3} \quad \binom{5}{4} \quad \binom{5}{5}$$

Q : How to calculate $\binom{5}{3}$?

```
procedure BINOM(n, k)                          ▷ Required: n ≥ k ≥ 0
    for i ← 0 to n − k do
        B[i][0] ← 1
    for i ← 1 to k do
        B[i][i] ← 1
    for j ← 1 to k do
        for d ← 1 to n − k do
            i ← j + d
            B[i][j] ← B[i − 1][j] + B[i − 1][j − 1]
    return B[n][k]
```

procedure $\textsc{Binom}(n, k)$ ▷ Required: $n \geq k \geq 0$

  for $i \leftarrow 0$ **to** $n - k$ **do**

    $B[i][0] \leftarrow 1$

  for $i \leftarrow 1$ **to** k **do**

    $B[i][i] \leftarrow 1$

  for $j \leftarrow 1$ **to** k **do**

    for $d \leftarrow 1$ **to** $n - k$ **do**

      $i \leftarrow j + d$

      $B[i][j] \leftarrow B[i-1][j] + B[i-1][j-1]$

  return $B[n][k]$

$$(n - k + 1) + (k) + k(n - k) = nk - k^2 + n + 1$$

Thank You!