本节内容

特殊矩阵

压缩存储

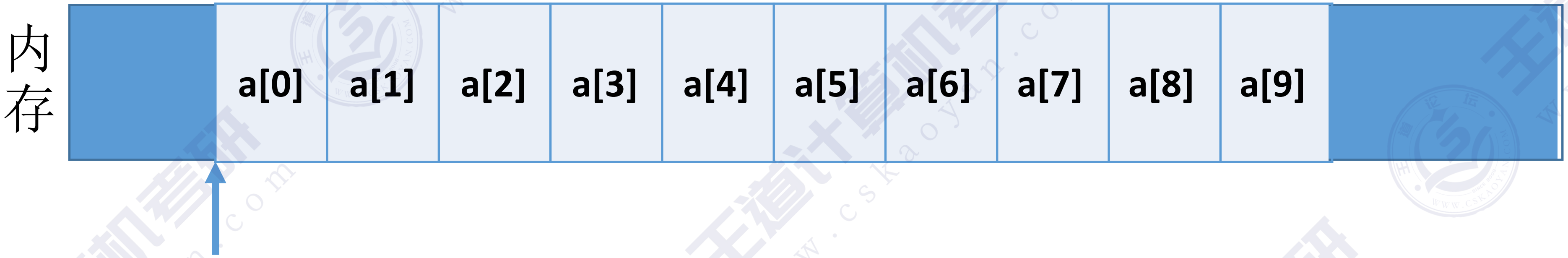
知识总览

- 矩阵的压缩存储
  - 数组的存储结构
    - 一维数组
    - 二维数组
  - 特殊矩阵
    - 对称矩阵
    - 三角矩阵
    - 三对角矩阵
    - 稀疏矩阵

一维数组的存储结构

```
ElemType a[10];    //ElemType型一维数组
```

C语言定义
一维数组



起始地址：LOC

各数组元素大小相同，且物理上连续存放。

数组元素a[i] 的存放地址 = LOC + i * sizeof(ElemType)   (0≤i<10)

注：除非题目特别说明，否则数组下标默认从0开始

注意审题！
易错！

二维数组的存储结构

```
ElemType b[2][4];   //2行4列的二维数组
```

C语言定义
二维数组

b[0][0]	b[0][1]	b[0][2]	b[0][3]
b[1][0]	b[1][1]	b[1][2]	b[1][3]

逻辑视角

内存

b[0][0]	b[0][1]	b[0][2]	b[0][3]	b[1][0]	b[1][1]	b[1][2]	b[1][3]

行优先
存储

内存

b[0][0]	b[1][0]	b[0][1]	b[1][1]	b[0][2]	b[1][2]	b[0][3]	b[1][3]

列优先
存储

二维数组的存储结构

```
ElemType b[2][4];    //2行4列的二维数组
```

C语言定义
二维数组

b[0][0]	b[0][1]	b[0][2]	b[0][3]
b[1][0]	b[1][1]	b[1][2]	b[1][3]

逻辑视角

内存

	b[0][0]	b[0][1]	b[0][2]	b[0][3]	b[1][0]	b[1][1]	b[1][2]	b[1][3]	

行优先
存储

起始地址：LOC

M行N列的二维数组 b[M][N] 中，若按行优先存储，则

b[i][j] 的存储地址 = LOC + (i*N + j) * sizeof(ElemType)

二维数组的存储结构

```
ElemType b[2][4];   //2行4列的二维数组
```

C语言定义二维数组

b[0][0]	b[0][1]	b[0][2]	b[0][3]
b[1][0]	b[1][1]	b[1][2]	b[1][3]

逻辑视角

内存

	b[0][0]	b[1][0]	b[0][1]	b[1][1]	b[0][2]	b[1][2]	b[0][3]	b[1][3]	

列优先存储

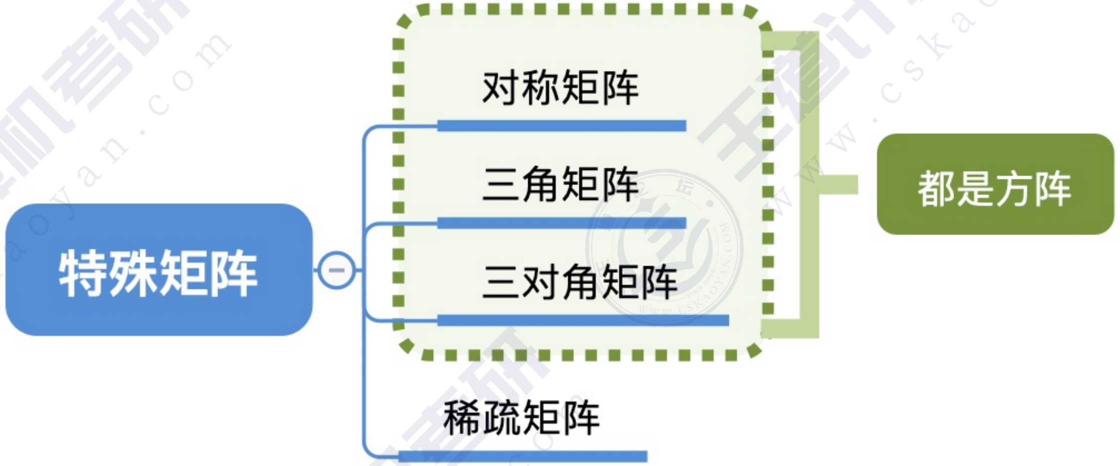
起始地址：LOC

M行N列的二维数组 b[M][N] 中，若按列优先存储，则

b[i][j] 的存储地址 = LOC + (j*M+ i) * sizeof(ElemType)

普通矩阵的存储

$$\begin{pmatrix}
a_{1,1} & a_{1,2} & a_{1,3} & \cdots\cdots & a_{1,n-1} & a_{1,n} \\
a_{2,1} & a_{2,2} & a_{2,3} & \cdots\cdots & a_{2,n-1} & a_{2,n} \\
a_{3,1} & a_{3,2} & a_{3,3} & \cdots\cdots & a_{3,n-1} & a_{3,n} \\
\vdots & \vdots & \vdots & & \vdots & \vdots \\
a_{m,1} & a_{m,2} & a_{m,3} & \cdots\cdots & a_{m,n-1} & a_{m,n}
\end{pmatrix}$$

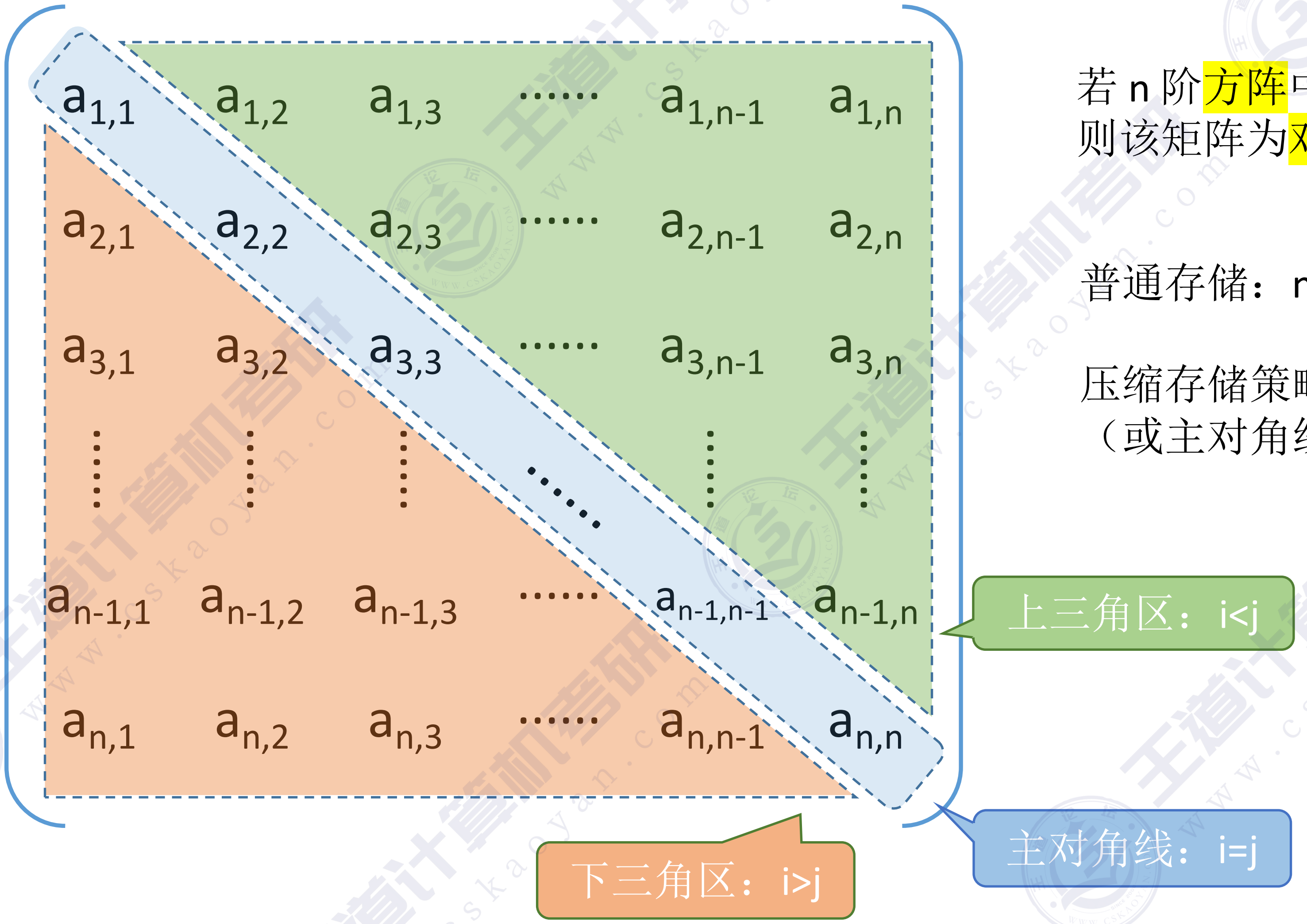


可用二维数组存储

注意：描述矩阵元素时，行、列号通常从 1 开始；而描述数组时通常下标从0开始
（具体看题目给的条件，注意审题！）

某些特殊矩阵可以压缩存储空间

对称矩阵的压缩存储



若 n 阶方阵中任意一个元素 $a_{i,j}$ 都有 $a_{i,j} = a_{j,i}$ 则该矩阵为对称矩阵

普通存储：n*n 二维数组

压缩存储策略：只存储主对角线+下三角区（或主对角线+上三角区）

对称矩阵的压缩存储

$$
\begin{pmatrix}
a_{1,1} & a_{1,2} & a_{1,3} & \cdots\cdots & a_{1,n-1} & a_{1,n} \\
a_{2,1} & a_{2,2} & a_{2,3} & \cdots\cdots & a_{2,n-1} & a_{2,n} \\
a_{3,1} & a_{3,2} & a_{3,3} & \cdots\cdots & a_{3,n-1} & a_{3,n} \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
a_{n-1,1} & a_{n-1,2} & a_{n-1,3} & \cdots\cdots & a_{n-1,n-1} & a_{n-1,n} \\
a_{n,1} & a_{n,2} & a_{n,3} & \cdots\cdots & a_{n,n-1} & a_{n,n}
\end{pmatrix}
$$

策略：只存储主对角线+下三角区

按**行优先**原则将各元素存入一维数组中。

B[0]	B[1]	B[2]	B[3]	....		B[?]
$a_{1,1}$	$a_{2,1}$	$a_{2,2}$	$a_{3,1}$	......	$a_{n,n-1}$	$a_{n,n}$

思考：
①数组大小应为多少?
②站在程序员的角度，对称矩阵压缩存储后怎样才能方便使用？

① $(1+n)*n/2$
②可以实现一个“映射”函数
矩阵下标→一维数组下标

对称矩阵的压缩存储

$$
\begin{pmatrix}
a_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,n-1} & a_{1,n} \\
a_{2,1} & a_{2,2} & a_{2,3} & \cdots & a_{2,n-1} & a_{2,n} \\
a_{3,1} & a_{3,2} & a_{3,3} & \cdots & a_{3,n-1} & a_{3,n} \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
a_{n-1,1} & a_{n-1,2} & a_{n-1,3} & \cdots & a_{n-1,n-1} & a_{n-1,n} \\
a_{n,1} & a_{n,2} & a_{n,3} & \cdots & a_{n,n-1} & a_{n,n}
\end{pmatrix}
$$

$i \geq j$

策略：只存储主对角线+下三角区

按行优先原则将各元素存入一维数组中。

B[0]	B[1]	B[2]	B[3]	....		$B[\frac{n(n+1)}{2}-1]$
$a_{1,1}$	$a_{2,1}$	$a_{2,2}$	$a_{3,1}$	......	$a_{n,n-1}$	$a_{n,n}$

矩阵下标 → 一维数组下标

$a_{i,j}$（i≥j） → B[k]

Key：按行优先的原则，$a_{i,j}$是第几个元素？

$[1+2+\cdots+(i-1)] + j$ → 第 $\frac{i(i-1)}{2} + j$ 个元素

→ $k = \frac{i(i-1)}{2} + j - 1$

对称矩阵的压缩存储

$$\begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,n-1} & a_{1,n} \\ a_{2,1} & a_{2,2} & a_{2,3} & \cdots & a_{2,n-1} & a_{2,n} \\ a_{3,1} & a_{3,2} & a_{3,3} & \cdots & a_{3,n-1} & a_{3,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n-1,1} & a_{n-1,2} & a_{n-1,3} & \cdots & a_{n-1,n-1} & a_{n-1,n} \\ a_{n,1} & a_{n,2} & a_{n,3} & \cdots & a_{n,n-1} & a_{n,n} \end{pmatrix}$$

i<j

i ≥ j

策略：只存储主对角线+下三角区

按行优先原则将各元素存入一维数组中。

B[0]	B[1]	B[2]	B[3]	….		$B[\frac{n(n+1)}{2}-1]$
$a_{1,1}$	$a_{2,1}$	$a_{2,2}$	$a_{3,1}$	……	$a_{n,n-1}$	$a_{n,n}$

矩阵下标 → 一维数组下标

$a_{i,j}$ （i<j） → B[k]

$a_{i,j} = a_{j,i}$ （对称矩阵性质）

→ $k = \frac{j(j-1)}{2} + i - 1$

对称矩阵的压缩存储

$$\begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \cdots\cdots & a_{1,n-1} & a_{1,n} \\ a_{2,1} & a_{2,2} & a_{2,3} & \cdots\cdots & a_{2,n-1} & a_{2,n} \\ a_{3,1} & a_{3,2} & a_{3,3} & \cdots\cdots & a_{3,n-1} & a_{3,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n-1,1} & a_{n-1,2} & a_{n-1,3} & \cdots\cdots & a_{n-1,n-1} & a_{n-1,n} \\ a_{n,1} & a_{n,2} & a_{n,3} & \cdots\cdots & a_{n,n-1} & a_{n,n} \end{bmatrix}$$

i<j

i ≥ j

策略：只存储主对角线+下三角区

按行优先原则将各元素存入一维数组中。

B[0]	B[1]	B[2]	B[3]	....		$B[\frac{n(n+1)}{2}-1]$
$a_{1,1}$	$a_{2,1}$	$a_{2,2}$	$a_{3,1}$	……	$a_{n,n-1}$	$a_{n,n}$

矩阵下标 → 一维数组下标

$a_{i,j}$ → B[k]

$a_{i,j} = a_{j,i}$ （对称矩阵性质）

$$k = \begin{cases} \dfrac{i(i-1)}{2} + j - 1, & i \geqslant j \text{（下三角区和主对角线元素）} \\ \dfrac{j(j-1)}{2} + i - 1, & i < j \text{（上三角区元素} a_{ij} = a_{ji}\text{）} \end{cases}$$

对称矩阵的压缩存储

$$
\begin{pmatrix}
a_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,n-1} & a_{1,n} \\
a_{2,1} & a_{2,2} & a_{2,3} & \cdots & a_{2,n-1} & a_{2,n} \\
a_{3,1} & a_{3,2} & a_{3,3} & \cdots & a_{3,n-1} & a_{3,n} \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
a_{n-1,1} & a_{n-1,2} & a_{n-1,3} & \cdots & a_{n-1,n-1} & a_{n-1,n} \\
a_{n,1} & a_{n,2} & a_{n,3} & \cdots & a_{n,n-1} & a_{n,n}
\end{pmatrix}
$$

策略：只存储主对角线+下三角区

按列优先原则将各元素存入一维数组中。

B[0]	B[1]	B[2]	B[3]	....		$B[\frac{n(n+1)}{2}-1]$
$a_{1,1}$	$a_{2,1}$	$a_{3,1}$	$a_{4,1}$	......	$a_{n,n-1}$	$a_{n,n}$

矩阵下标 → 一维数组下标

$a_{i,j}$ → B[k]

$a_{i,j} = a_{j,i}$（对称矩阵性质）

出题方法

- 存储上三角？下三角？
- 行优先？列优先？
- 矩阵元素的下标从0？1？开始
- 数组下标从0？1？开始

三角矩阵的压缩存储

$$\begin{pmatrix} a_{1,1} & c & c & \cdots & c & c \\ a_{2,1} & a_{2,2} & c & \cdots & c & c \\ a_{3,1} & a_{3,2} & a_{3,3} & \cdots & c & c \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n-1,1} & a_{n-1,2} & a_{n-1,3} & \cdots & a_{n-1,n-1} & c \\ a_{n,1} & a_{n,2} & a_{n,3} & \cdots & a_{n,n-1} & a_{n,n} \end{pmatrix} \quad \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,n-1} & a_{1,n} \\ c & a_{2,2} & a_{2,3} & \cdots & a_{2,n-1} & a_{2,n} \\ c & c & a_{3,3} & \cdots & a_{3,n-1} & a_{3,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ c & c & c & \cdots & a_{n-1,n-1} & a_{n-1,n} \\ c & c & c & \cdots & c & a_{n,n} \end{pmatrix}$$

下三角矩阵：除了主对角线和下三角区，其余的元素都相同

上三角矩阵：除了主对角线和上三角区，其余的元素都相同

三角矩阵的压缩存储

$$\begin{pmatrix} a_{1,1} & c & c & \cdots & c & c \\ a_{2,1} & a_{2,2} & c & \cdots & c & c \\ a_{3,1} & a_{3,2} & a_{3,3} & \cdots & c & c \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n-1,1} & a_{n-1,2} & a_{n-1,3} & \cdots & a_{n-1,n-1} & c \\ a_{n,1} & a_{n,2} & a_{n,3} & \cdots & a_{n,n-1} & a_{n,n} \end{pmatrix}$$

下三角矩阵：除了主对角线和下三角区，其余的元素都相同

压缩存储策略：按行优先原则将橙色区元素存入一维数组中。并在最后一个位置存储常量c

B[0]	B[1]	B[2]	B[3]	....	$B[\frac{n(n+1)}{2}-1]$	$B[\frac{n(n+1)}{2}]$
$a_{1,1}$	$a_{2,1}$	$a_{2,2}$	$a_{3,1}$	……	$a_{n,n}$	c

矩阵下标 → 一维数组下标

$a_{i,j}$ （i≥j） → B[k]

Key：按行优先的原则，$a_{i,j}$ 是第几个元素？

$$k=\begin{cases} \dfrac{i(i-1)}{2}+j-1, & i\geq j\text{（下三角区和主对角线元素）} \\ \dfrac{n(n+1)}{2}, & i<j\text{（上三角区元素）} \end{cases}$$

三角矩阵的压缩存储

$$\begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,n-1} & a_{1,n} \\ c & a_{2,2} & a_{2,3} & \cdots & a_{2,n-1} & a_{2,n} \\ c & c & a_{3,3} & \cdots & a_{3,n-1} & a_{3,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ c & c & c & \cdots & a_{n-1,n-1} & a_{n-1,n} \\ c & c & c & \cdots & c & a_{n,n} \end{pmatrix}$$

上三角矩阵：除了主对角线和上三角区，其余的元素都相同

压缩存储策略：按行优先原则将绿色区元素存入一维数组中。并在最后一个位置存储常量c

B[0]	B[1]	B[2]	B[3]	....	B[$\frac{n(n+1)}{2}$-1]	B[$\frac{n(n+1)}{2}$]
$a_{1,1}$	$a_{1,2}$	$a_{1,3}$	$a_{1,4}$	……	$a_{n,n}$	c

矩阵下标 → 一维数组下标

$a_{i,j}$（i≤j） → B[k]

Key：按行优先的原则，$a_{i,j}$是第几个元素？

$$k=\begin{cases}\dfrac{(i-1)(2n-i+2)}{2}+(j-i), & i\leqslant j\text{（上三角区和主对角线元素）}\\ \dfrac{n(n+1)}{2}, & i>j\text{（下三角区元素）}\end{cases}$$

三对角矩阵的压缩存储

$$\begin{pmatrix} a_{1,1} & a_{1,2} & 0 & \cdots & 0 & 0 \\ a_{2,1} & a_{2,2} & a_{2,3} & \cdots & 0 & 0 \\ 0 & a_{3,2} & a_{3,3} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & a_{n-1,n-1} & a_{n-1,n} \\ 0 & 0 & 0 & \cdots & a_{n,n-1} & a_{n,n} \end{pmatrix}$$

三对角矩阵，又称带状矩阵：

当 $|i - j|>1$ 时，有 $a_{i,j} = 0$ （$1\le i, j \le n$）

压缩存储策略：

按行优先（或列优先）原则，只存储带状部分

B[0]	B[1]	B[2]	B[3]	....		B[3n-3]
$a_{1,1}$	$a_{1,2}$	$a_{2,1}$	$a_{2,2}$	......	$a_{n,n-1}$	$a_{n,n}$

矩阵下标 → 一维数组下标

$a_{i,j}$ （$|i-j|\le 1$） → B[k]

Key：按行优先的原则，$a_{i,j}$ 是第几个元素？

前i-1行共 3(i-1)-1 个元素

$a_{i,j}$是 i 行第 j-i+2 个元素

$a_{i,j}$是第 2i+j-2 个元素 → k = 2i+j-3

数组下标从0开始

三对角矩阵的压缩存储

$$\begin{pmatrix} a_{1,1} & a_{1,2} & 0 & \cdots & 0 & 0 \\ a_{2,1} & a_{2,2} & a_{2,3} & \cdots & 0 & 0 \\ 0 & a_{3,2} & a_{3,3} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & a_{n-1,n-1} & a_{n-1,n} \\ 0 & 0 & 0 & \cdots & a_{n,n-1} & a_{n,n} \end{pmatrix}$$

若已知数组下标k，如何得到 i, j ？

$B[k] \rightarrow a_{i,j}$

第 k+1 个元素，在第几行？第几列？

前i-1行共 3(i-1)-1 个元素

前i行共 3i-1 个元素

显然， $3(i-1)-1 < k+1 \leq 3i-1$

$i \geq (k+2)/3$

可以理解为“刚好”大于等于

$i = \lceil (k+2)/3 \rceil$

向上取整即可满足“刚好”大于等于

王道书的计算逻辑： $3(i-1)-1 \leq k < 3i-1$

$i \leq (k+1)/3+1$

可以理解为“刚好”小于等于

$i = \lfloor (k+1)/3+1 \rfloor$

向下取整即可满足“刚好”小于等于

三对角矩阵的压缩存储

$$
\begin{pmatrix}
a_{1,1} & a_{1,2} & 0 & \cdots\cdots & 0 & 0 \\
a_{2,1} & a_{2,2} & a_{2,3} & \cdots\cdots & 0 & 0 \\
0 & a_{3,2} & a_{3,3} & \cdots\cdots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & \cdots\cdots & a_{n-1,n-1} & a_{n-1,n} \\
0 & 0 & 0 & \cdots\cdots & a_{n,n-1} & a_{n,n}
\end{pmatrix}
$$

若已知数组下标k，如何得到 i, j ？

B[k] $\rightarrow$ $a_{i,j}$

第 k+1 个元素，在第几行？第几列？

$i = \lceil (k+2)/3 \rceil$ 或 $i = \lfloor (k+1)/3+1 \rfloor$

由$k = 2i+j-3$，得

$j = k - 2i + 3$

三角矩阵的压缩存储

$$\begin{pmatrix} a_{1,1} & c & c & \cdots & c & c \\ a_{2,1} & a_{2,2} & c & \cdots & c & c \\ a_{3,1} & a_{3,2} & a_{3,3} & \cdots & c & c \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n-1,1} & a_{n-1,2} & a_{n-1,3} & \cdots & a_{n-1,n-1} & c \\ a_{n,1} & a_{n,2} & a_{n,3} & \cdots & a_{n,n-1} & a_{n,n} \end{pmatrix}$$

下三角矩阵：除了主对角线和下三角区，其余的元素都相同

压缩存储策略：按行优先原则将绿色区元素存入一维数组中。并在最后一个位置存储常量c

B[0]	B[1]	B[2]	B[3]	....	B[$\frac{n(n+1)}{2}$-1]	B[$\frac{n(n+1)}{2}$]
$a_{1,1}$	$a_{2,1}$	$a_{2,2}$	$a_{3,1}$	......	$a_{n,n}$	c

矩阵下标 → 一维数组下标

$a_{i,j}$（i≥j） → B[k]

思考：如何用k推出i，j?

Key：按行优先的原则，$a_{i,j}$是第几个元

$$k=\begin{cases}\frac{i(i-1)}{2}+j-1, & i\geq j\text{（下三角区和主对角线元素）}\\ \frac{n(n+1)}{2}, & i<j\text{（上三角区元素）}\end{cases}$$

稀疏矩阵的压缩存储

$$\begin{pmatrix} 0 & 0 & 4 & 0 & 0 & 5 \\ 0 & 3 & 0 & 9 & 0 & 0 \\ 0 & 0 & 0 & 0 & 7 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

稀疏矩阵：非零元素远远少于矩阵元素的个数

压缩存储策略：
顺序存储——三元组 <行，列，值>

i（行）	j（列）	v（值）
1	3	4
1	6	5
2	2	3
2	4	9
3	5	7
4	2	2

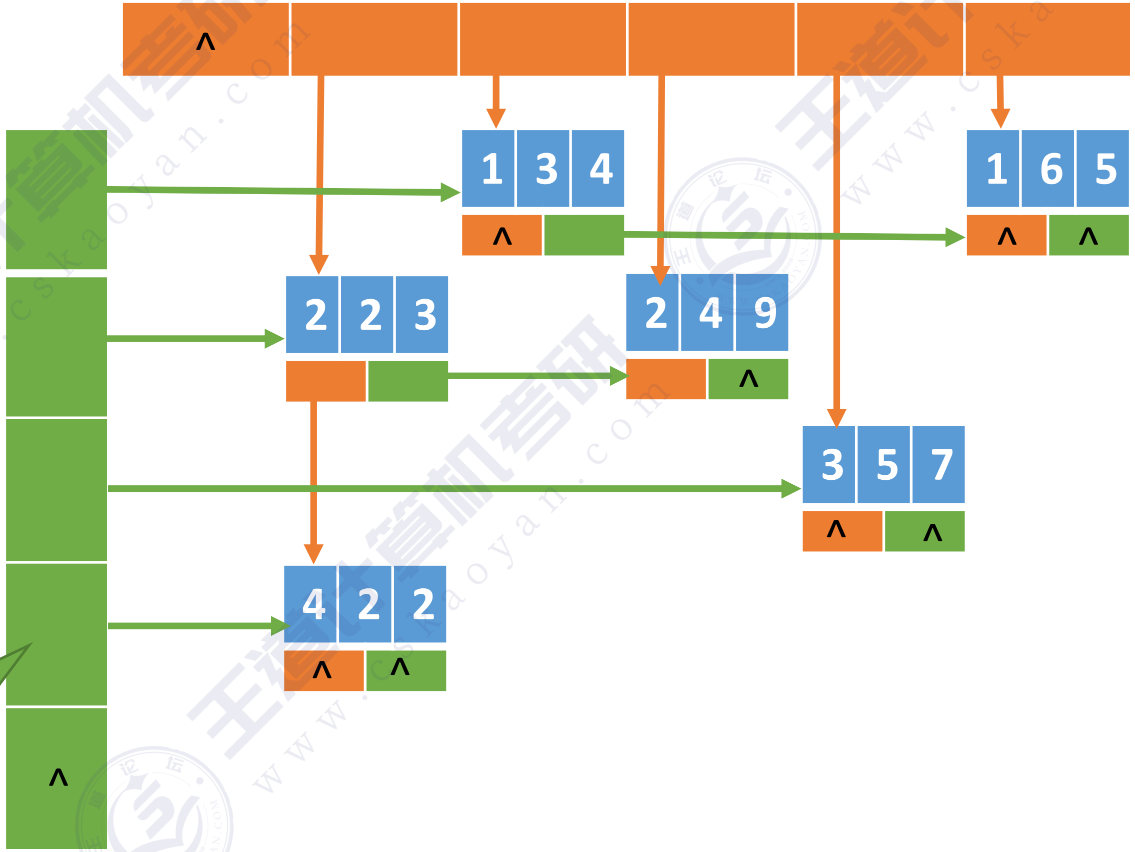
（注：此处行、列标从1开始）

稀疏矩阵的压缩存储

$$
\begin{pmatrix}
0 & 0 & 4 & 0 & 0 & 5 \\
0 & 3 & 0 & 9 & 0 & 0 \\
0 & 0 & 0 & 0 & 7 & 0 \\
0 & 2 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
$$

压缩存储策略二：
链式存储——十字链表法

向下域 down，
指向第 j 列的
第一个元素



非零数据
结点说明：

行	列	值
指向同列的下一个元素		指向同行的下一个元素

向右域 right，
指向第 i 行的
第一个元素

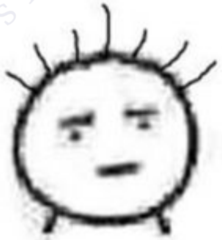
知识回顾与重要考点

- 特殊矩阵压缩存储
  - 对称矩阵
    - 特点：对方阵中的任意一个元素，有 $a_{i,j} = a_{j,i}$
    - 压缩：只存储主对角线+下三角区（或主对角线+上三角区）
  - 三角矩阵
    - 特点：上三角区全为常量（下三角矩阵）；或下三角区全为常量（上三角矩阵）
    - 压缩：按行优先/列优先规则依次存储非常量区域，并在最后一个位置存放常量c
  - 三对角矩阵（带状矩阵）
    - 特点：当 $|i - j|>1$ 时，有 $a_{i,j} = 0$ （$1\le i, j \le n$）
    - 压缩：按行优先/列优先规则依次存储带状区域
  - 稀疏矩阵
    - 非零元素个数远小于零元素个数
    - 压缩：只存储非零元素
      - 顺序存储：顺序存储三元组 <行,列,值>
      - 链式存储：十字链表法

知识回顾与重要考点

常见考题

- 矩阵的压缩存储需要多长的数组
- 由矩阵行列号 <i, j> 推出对应的数组下标号 k　（数列求和）
- 由 k 推出 <i,j>
  - 如何处理不等式中的“刚好大于等于/小于等于”
  - 向上取整/向下取整
- 易错点
  - 存储上三角？下三角？
  - 行优先？列优先？
  - 矩阵元素的下标从0？1？开始
  - 数组下标从0？1？开始



问题　是否忘了，等差数列求和？？？