第4章 数组

苏智勇

suzhiyong@njust.edu.cn

https://zhiyongsu.github.io

Visual Computing Group, NJUST

教学内容

4.1 数组的类型定义

4.2 数组的顺序存储

4.3 特殊矩阵的压缩存储

教学目标

明确数组这种数据结构的特点

掌握数组地址计算方法

了解几种特殊矩阵的压缩存储方法

本节所讨论的数组与高级语言中的数组区别：

- 高级语言中的数组是顺序结构；
- 而本章的数组既可以是顺序的，也可以是链式结构，

用户可根据需要选择。

数组的抽象数据类型

ADT Array {

数据对象：

$$j_i = 0,\cdots b_i - 1, \mathrm{i} = 1,2,\cdots,\mathrm{n}$$

$$D = \{\mathrm{a}_{\mathrm{j}\ \mathrm{j}_2\cdots\mathrm{j}_\mathrm{n}} \mid \mathrm{a}_{\mathrm{j}_1\mathrm{j}_2\cdots\mathrm{j}_\mathrm{n}} \in ElemSet$$

数据关系：

$$R_1 = \{< a_{j_1\cdots j_i\cdots j_n}, a_{j_1\cdots j_i+1\cdots j_n} > \mid$$

$$0 \le \mathrm{j_k} \le b_k - 1,\ \ 1 \le \mathrm{k} \le \mathrm{n},\ \ 且\mathrm{k} \ne \mathrm{i},$$

$$0 \le \mathrm{j_i} \le b_k - 2,$$

$$a_{j_1\cdots j_i\cdots j_n}, a_{j_1\cdots j_i+1\cdots j_n} \in D, i = 2,\cdots,n\}$$

基本操作:

(1) InitArray (&A,n,bound1, …boundn)
//构造数组A

(2) DestroyArray (&A) **// 销毁数组A**

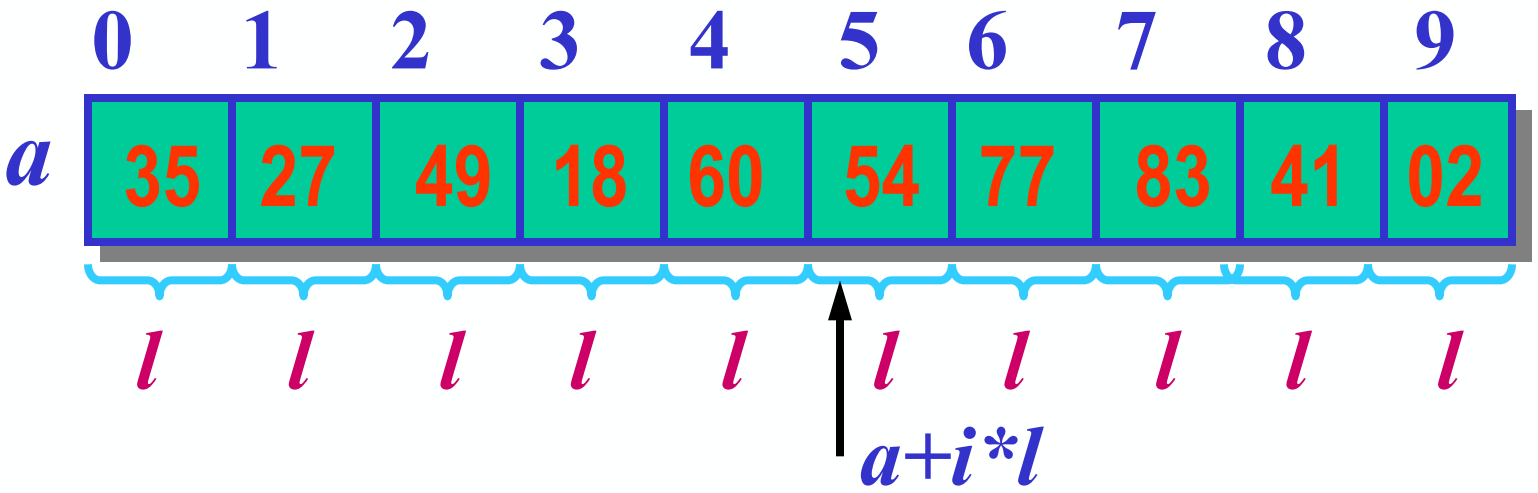
(3) Value(A,&e,index1,…,indexn) **//取数组元素值**

(4) Assign (A,&e,index1,…,indexn) **//给数组元素赋值**

}ADT Array

一维数组

$$LOC(i) = \begin{cases} a, & i = 0 \\ LOC(i-1)+l = a+i*l, & i > 0 \end{cases}$$



$$LOC(i) = LOC(i-1)+l = a+i*l$$

二维数组

$$A=(\alpha_1,\alpha_2,\cdots,\alpha_p)\quad (\mathrm{p}=\mathrm{m}或\mathrm{n})$$

$$\alpha_i=(a_{i1},a_{i2},\cdots,a_{in})\quad 1\le i\le m$$

$$\alpha_j=(a_{1j},a_{2j},\cdots,a_{mj})\quad 1\le \mathrm{j}\le n$$

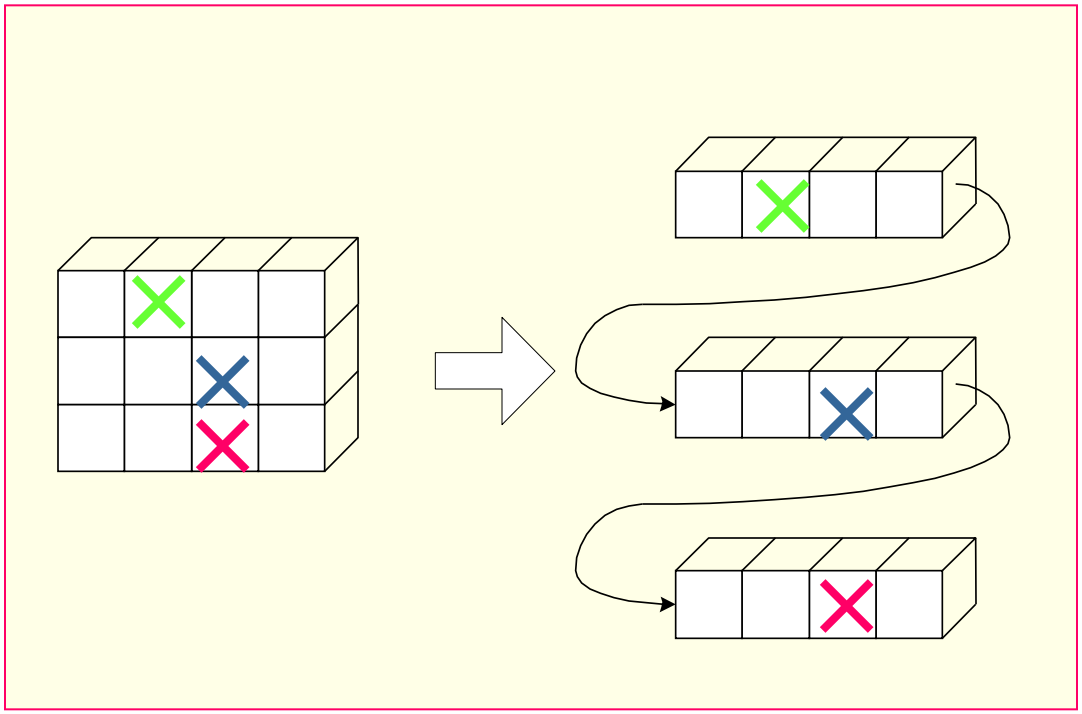
$$A_{m\times n}=\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

$$A_{m\times n}=\begin{bmatrix} \begin{bmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{m1} \end{bmatrix} & \begin{bmatrix} a_{12} \\ a_{22} \\ \vdots \\ a_{m2} \end{bmatrix} & \cdots & \begin{bmatrix} a_{1n} \\ a_{2n} \\ \vdots \\ a_{mn} \end{bmatrix} \end{bmatrix}$$

数组的顺序存储

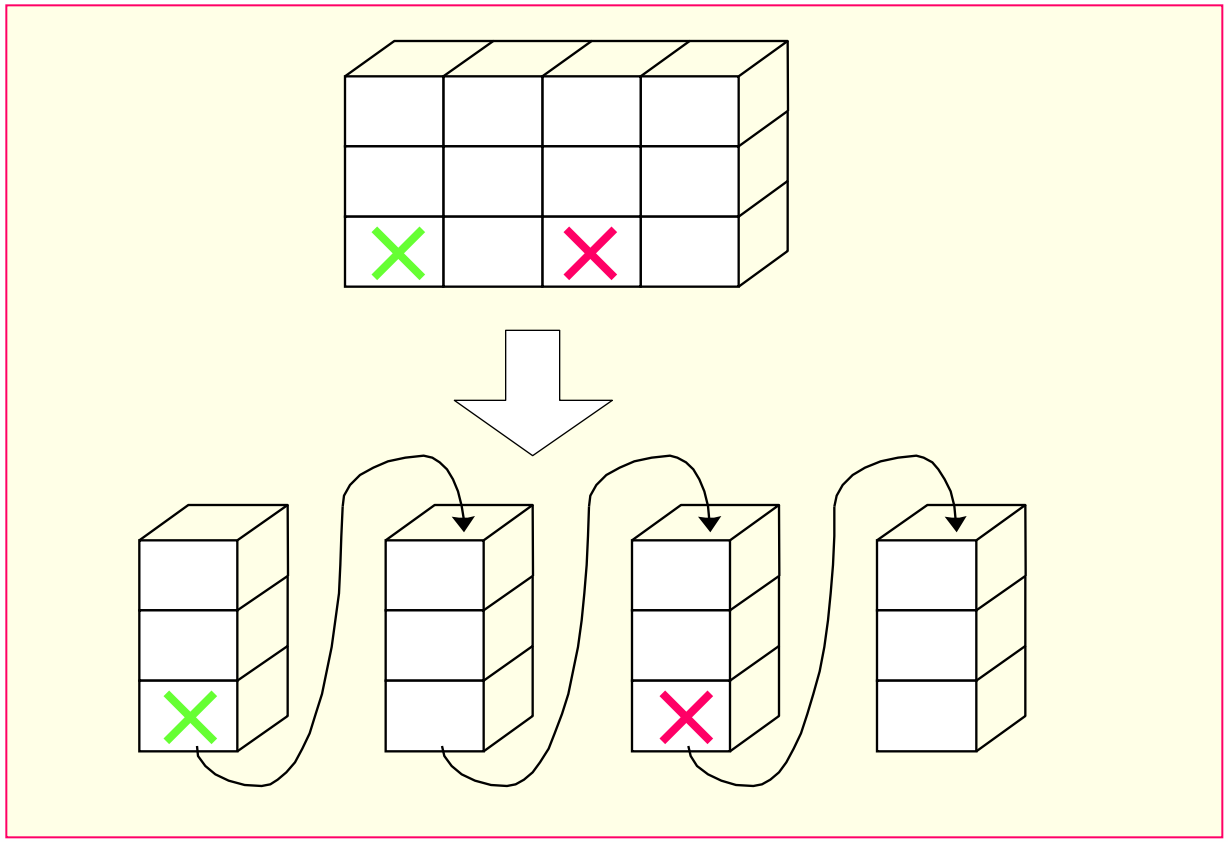
- 以行序为主序

  C, PASCAL



- 以列序为主序

FORTRAN



二维数组的行序优先表示

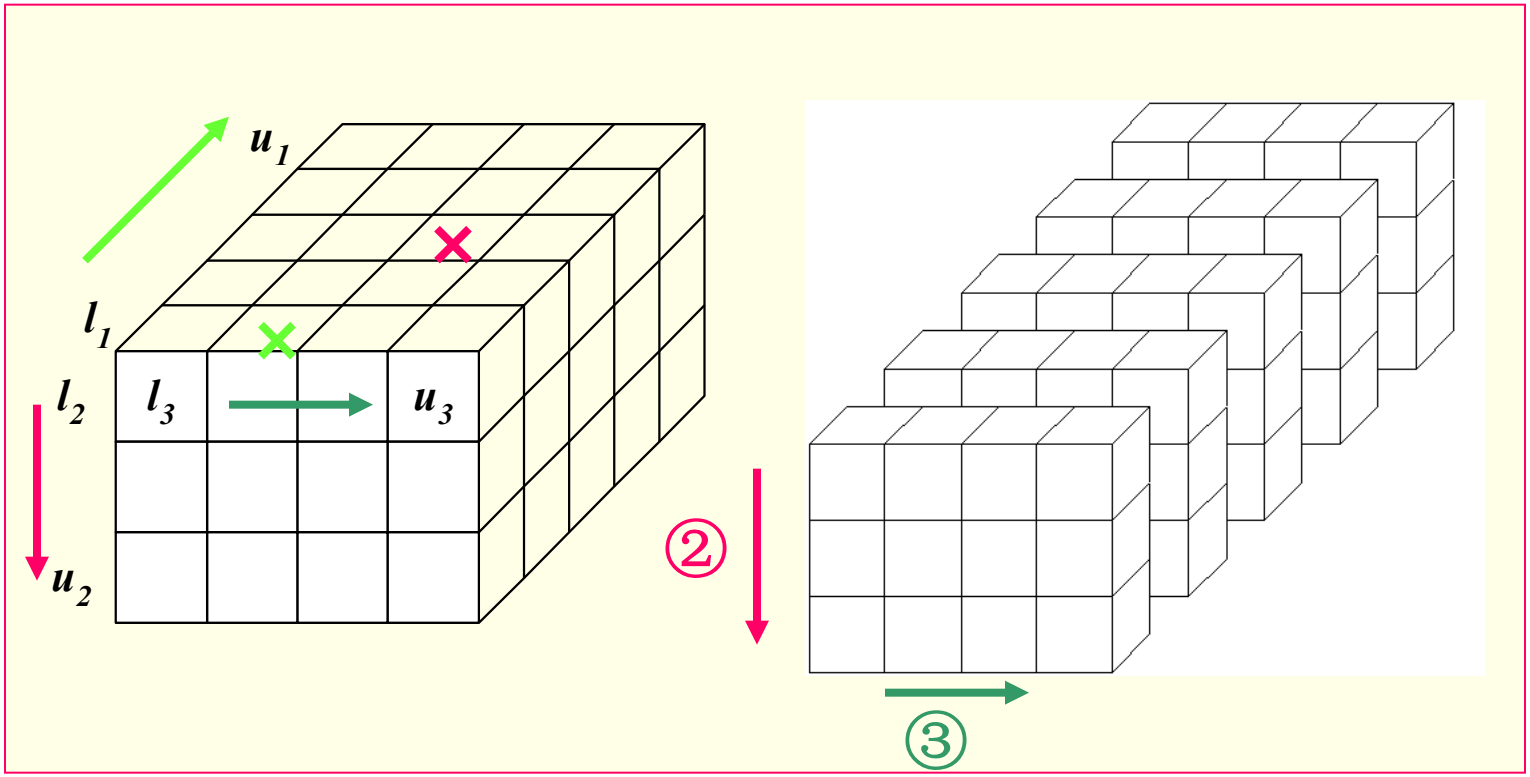
a[n][m]

$$a=\begin{pmatrix} a[0][0] & a[0][1] & \cdots & a[0][m-1] \\ a[1][0] & a[1][1] & \cdots & a[1][m-1] \\ a[2][0] & a[2][1] & \cdots & a[2][m-1] \\ \vdots & \vdots & \ddots & \vdots \\ a[n-1][0] & a[n-1][1] & \cdots & a[n-1][m-1] \end{pmatrix}$$

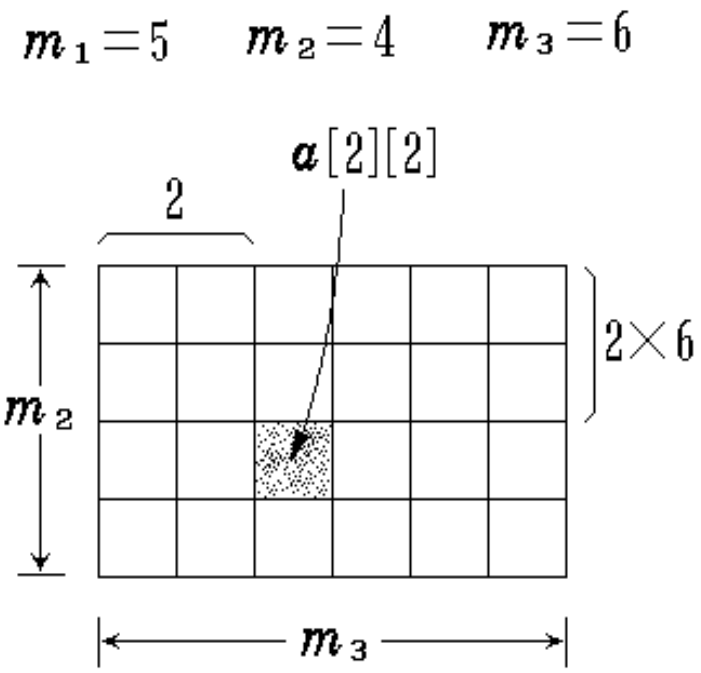
设数组开始存放位置 $LOC(0,0)=a$

$$LOC(j,k)=a+j*m+k$$

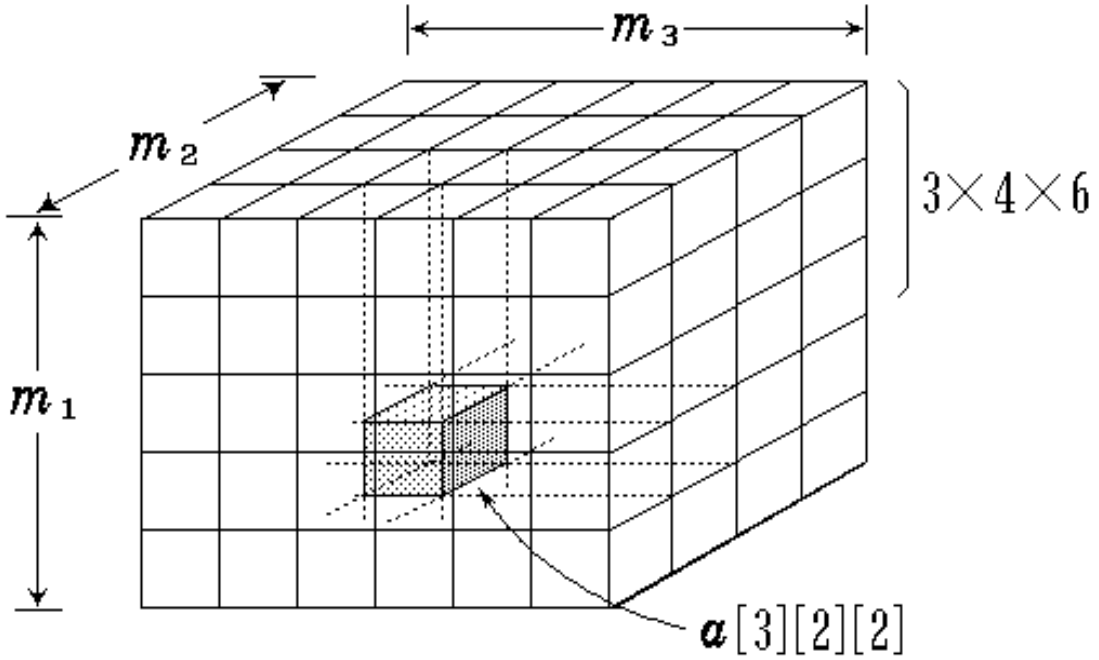
按页/行/列存放，页优先的顺序存储



二维数组



三维数组



三维数组

- a[m1][m2] [m3] 各维元素个数为 m_1, m_2, m_3
- 下标为 i_1, i_2, i_3的数组元素的存储位置:

$$LOC(i_1, i_2, i_3) = a + i_1 * m_2 * m_3 + i_2 * m_3 + i_3$$

$i_1 * m_2 * m_3$：前i_1页总元素个数

$i_2 * m_3$：第i_1页的前i_2行总元素个数

i_3：第i_2行前i_3列元素个数

练习

设有一个二维数组$A[m][n]$按行优先顺序存储，假设$A[0][0]$存放位置在$644_{(10)}$，$A[2][2]$存放位置在$676_{(10)}$，每个元素占一个空间，问$A[3][3]_{(10)}$存放在什么位置？脚注$_{(10)}$表示用10进制表示。

设数组元素A[i][j]存放在起始地址为Loc (i, j) 的存储单元中

∵ Loc (2, 2) = Loc (0, 0) + 2 * n + 2 = 644 + 2 * n + 2 = 676.
∴ n = (676 - 2 - 644) / 2 = 15
∴ Loc (3, 3) = Loc (0, 0) + 3 * 15 + 3 = 644 + 45 + 3 = 692.

练习

设有二维数组A[10,20]，其每个元素占两个字节，A[0][0]存储地址为100，若按行优先顺序存储，则元素A[6,6]的存储地址为__352__，按列优先顺序存储，元素A[6,6]的存储地址为__232__。

(6*20+6)*2+100=352

(6*10+6)*2+100=232

特殊矩阵的压缩存储

1. 什么是压缩存储？

若多个数据元素的值都相同，则只分配一个元素值的存储空间，且零元素不占存储空间。

2. 什么样的矩阵能够压缩？

一些特殊矩阵，如：对称矩阵，对角矩阵，三角矩阵，稀疏矩阵等。

3. 什么叫稀疏矩阵？

矩阵中非零元素的个数较少（一般小于5%）

1. 对称矩阵

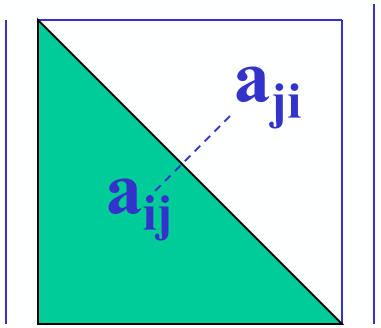
[特点] 在n×n的矩阵a中，满足如下性质：

$$a_{ij}=a_{ji} \quad (1 \le i, j \le n)$$

[存储方法] 只存储下(或者上)三角(包括主对角线)的数据元素。共占用n(n+1)/2个元素空间。

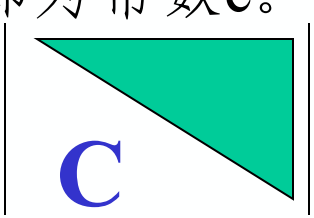
sa	a_{11}	a_{21}	a_{22}	a_{31}		$a_{ij}(a_{ji})$		a_{nn}
k	1	2	3	4				n(n+1)/2

$$k=\begin{cases} i(i-1)/2+j & \text{当 } i \ge j \\ j(j-1)/2+i & \text{当 } i<j \end{cases}$$

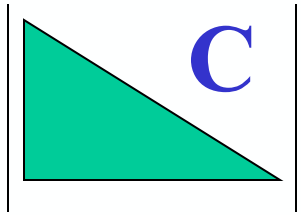


2. 三角矩阵

[特点] 对角线以下(或者以上)的数据元素(不包括对角线)全部为常数c。



上三角矩阵



下三角矩阵

[存储方法] 重复元素c共享一个元素存储空间，共占用n(n+1)/2+1个元素空间: sa[1.. n(n+1)/2+1]

上三角矩阵

$$k=\begin{cases}(i-1)\times(2n-i+2)/2+j-i+1 & i\le j\\ n(n+1)/2+1 & i>j\end{cases}$$

下三角矩阵

$$k=\begin{cases}i\times(i-1)/2+j & i\ge j\\ n(n+1)/2+1 & i<j\end{cases}$$

稀疏矩阵

[特点] 大多数元素为零。

[常用存储方法] 只记录每一非零元素(i,j,a_{ij})

节省空间，但丧失随机存取功能

- 顺序存储：三元组表

$$\begin{bmatrix} 15 & 0 & 0 & 22 & 0 & -15 \\ 0 & 11 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & -6 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 91 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 28 & 0 & 0 & 0 \end{bmatrix}_{6\times6}$$

三列二维数组表示

（1）非零元素所在的行号i;

（2）非零元素所在的列号j;

（3）非零元素的值V。

即每一个非零元素可以用下列三元组表示:

（i，j，V）

- 例如，上述稀疏矩阵A中的8个非零元素可以用以下8个二元组表示（以行为主的顺序排列）：

  （1，3，3）（1，8，1）（3，1，9）（4，5，7）

  （5，7，6）（6，4，2）（6，6，3）（7，3，5）

- 为了表示的唯一性，除了每一个非零元素用一个三元组表示外，在所有表示非零元素的三元组之前再添加一个三元组：（I，J，t）

- 其中I表示稀疏矩阵的总行数，J表示稀疏矩阵的总列数，t表示稀疏矩阵中非零元素的个数。

- 上述稀疏矩阵A可以用以下9个三元组表示:

  （7，8，8）（3，1，9）（5，7，6）

  （6，6，3）（1，3，3）（4，5，7）

  （6，4，2）（7，3，5）（1，8，1）
- 其中第一个三元组表示了稀疏矩阵的总体信息（总行数，总列数、非零元素个数），
- 其后的8个三元组依次（以行为主排列）表示稀疏矩阵中每一个非零元素的信息（所在的行号、列号以及非零元素值）。

为了使各三元组的结构更紧凑，通常将这些三元组组织成三列二维表格的形式，一般又表示成三列二维数组的形式，并简称为三列二维数组。

$$B=\begin{bmatrix}7 & 8 & 8\\ 1 & 3 & 3\\ 1 & 8 & 1\\ 3 & 1 & 9\\ 4 & 5 & 7\\ 5 & 7 & 6\\ 6 & 4 & 2\\ 6 & 6 & 3\\ 7 & 3 & 5\end{bmatrix}$$

- 为了便于在三列二维数组B中访问稀疏矩阵A中的各元素，通常还附设两个长度与稀疏矩阵A的行数相同的向量POS与NUM，
- POS（k）表示稀疏矩阵A中第k行的第一个非零元素（如果有的话）在三列二维数组B中的行号，
- NUM（k）表示稀疏矩阵A中第k行中非零元素的个数，
- 这两个向量之间存在以下关系：

  POS(1) = 2

  POS(k) = POS(k-1)十NUM(k-1)，$2 \leq k \leq m$

练习

$$A=\begin{bmatrix} 9 & 0 & 4 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 6 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 5 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

(1) 请用三列二维数组表示；
(2) 写出其POS和NUM向量；

小结

明确数组数据结构的特点

掌握数组地址计算方法

了解几种特殊矩阵的压缩存储方法。