布局人生序曲，梦想精心编织

第06讲 广义表、数组和特殊矩阵

信息学院（智能应用研究院）
欧 新 宇

建议课时：2

第2章 线性表

考核要点

- **考核大纲**

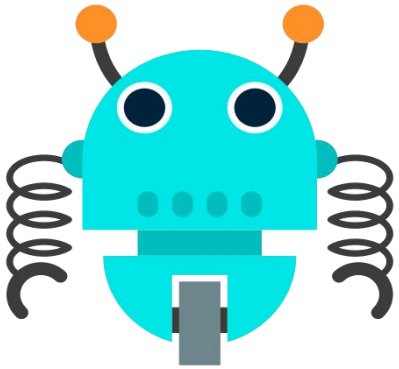
广义表的基本概念及运算规则、数组的定义、数组的存储结构、特殊矩阵的压缩存储

- **复习要点**
  - 掌握广义表的基本运算规则
  - 重点掌握数组和特殊矩阵的压缩存储
  - 掌握行优先和列优先两种方法中数组地址的计算方法

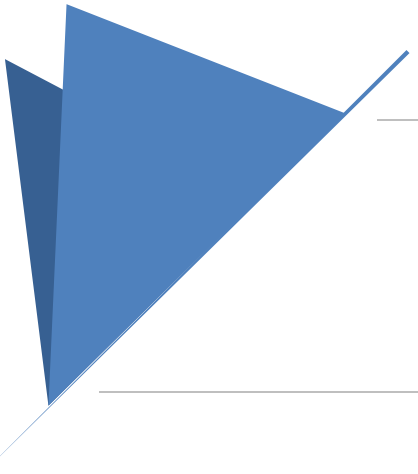
第06讲 广义表、数组和特殊矩阵

本讲作业

- 必做题
  - ✓ 【课堂互动6.1】广义表
  - ✓ 【课堂互动6.2】数组的定义及顺序存储实现
  - ✓ 【课堂互动6.3】特殊矩阵的压缩存储
- 选做题
  - ✓ 【课前自测06】广义表、数组和特殊矩阵
  - ✓ 【扩展练习09】数组和特殊矩阵



- 广义表
- 数组的定义
- 数组顺序存储实现
- 特殊矩阵的压缩存储



广义表

/ 广义表的定义和实现

/ 广义表的基本操作

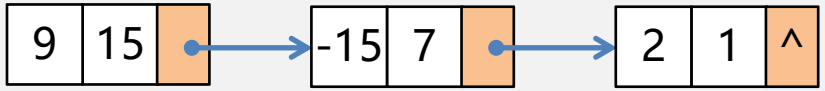
/ 广义表的特点

广义表

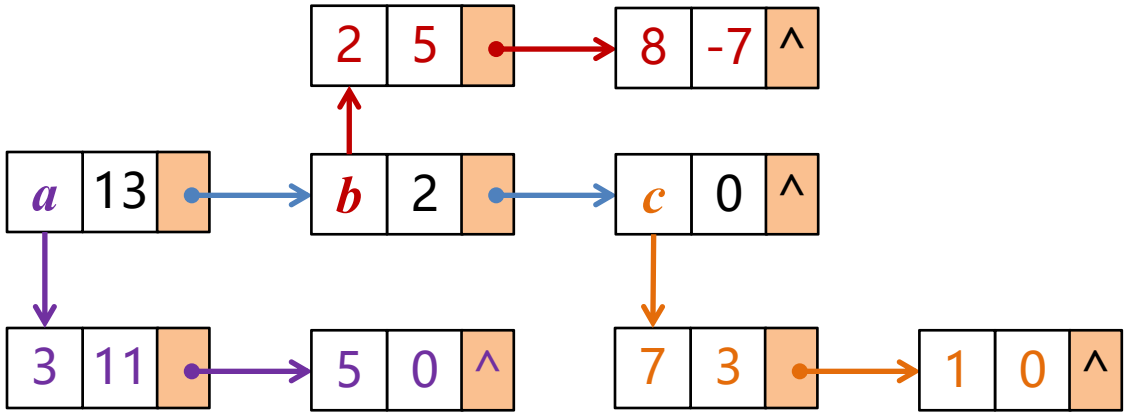
二元多项式的链式表示

例8：已知二元多项式 $P(x,y)=3x^{13}y^{11}+2x^2y^5+5x^{13}-8x^2y^7+7y^3+1$。试问，是否可以使用线性表来表示这个多项式?

一元多项式： $f_1(x)=9x^{15}-15x^7+2x$



$$P(x,y)=\underbrace{(3y^{11}+5)}_{a}x^{13}+\underbrace{(2y^5-8y^7)}_{b}x^2+\underbrace{(7y^3+1)}_{c}x^0$$



二元多项式可以使用复杂结构的“**链表**”进行表示。

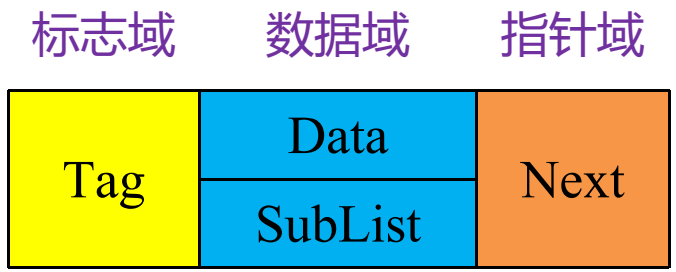
广义表

广义表的定义

广义表（Generalized List）是一种非连续性的数据结构，记作：$LS = (a_0, a_1, a_2, \ldots, a_n)$，$n \geq 0$，$LS$ 为表名，a_i是表元素，n为表长。广义表**放松**了线性表对表原子元素的限制，允许每个元素都具有其自身的结构。

广义表结点结构的定义

```
typedef struct GNode * GList;
struct GNode {
    int Tag;   // 标志域，  0: 节点是单元素，1: 节点是广义表
    union {   //  数据域与子表指针域共用存储空间
        ElementType Data;       // 单元素数据域
        GList SubList;          // 子表指针域Sublist
    } URegion;
    GList Next;                 // 指向后继节点
} GNode *GList
```



广义表

广义表与线性表的关系

- 对于线性表而言，n个元素都是基本的单元素（原子元素）
- 广义表的元素a_i可以是单元素，也可以是结构体元素（例如另一个广义表）。
- 广义表是线性表的一种推广，线性表是一种特殊的广义表（所有元素都是单元素）
- 广义表不一定是线性表，也不一定是线性结构（广义表也是树结构的一种推广）
- $n = 0$的广义表称为空表，空广义表可以被认为是空线性表。

广义表

广义表的基本运算

GetHead(LS)

取出**表头**为非空广义表的第一个**元素**，表头可以是原子元素也可以是广义表。

GetTail(LS)

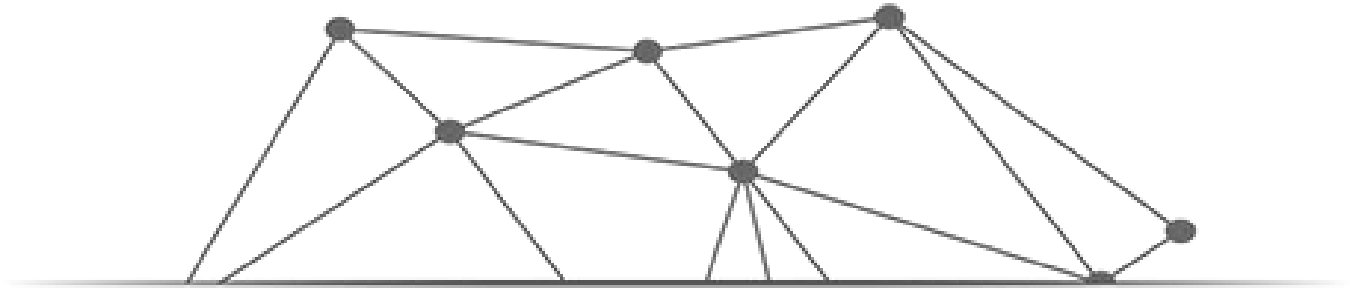
取出**表尾**为除去表头之外的其余元素所构成的**表**，表尾一定是一个广义表。

A=( )	空表，长度为0	GetHead和GetTail均无定义
B=(())	非空表，长度为1	GetHead(B)=(), GetTail(B)=()
C=(a)	包含一个元素a，长度为1	GetHead(C)=a, GetTail(C)=()
D=(a,(b,c))	包含两个元素a和(b,c)，长度为2	GetHead(D)=(a), GetTail(D)=((b, c))
E=(B,C,D)	共享表，包含三个元素，长度为3	GetHead(E)=B=(()), GetTail(E)=(C,D)
F=(f,F)	无限递归表F=(f,(f,(f,(...)))），长度为2	GetHead(F)=(f), GetTail(F)=(F)

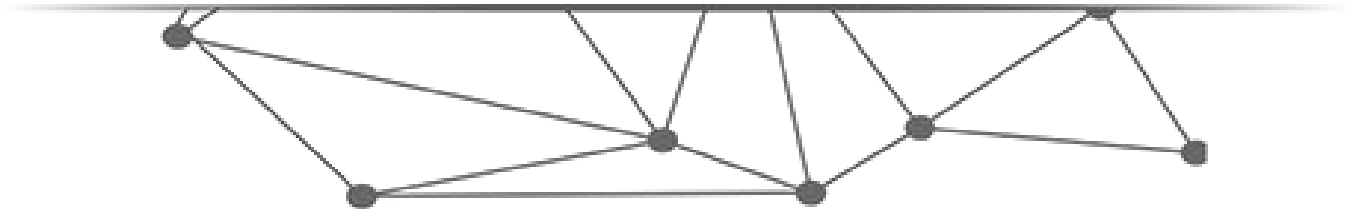
广义表

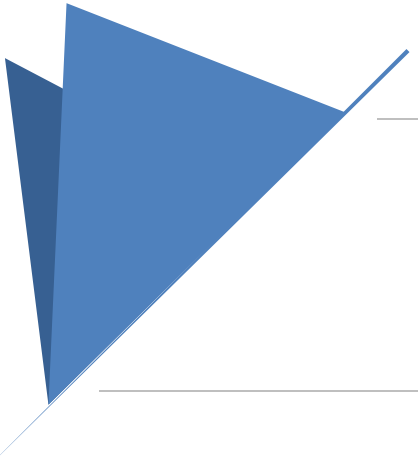
广义表的特点

- 有序性：每个元素都有一个直接前驱和一个直接后继
- 有长度：等于表中第一层元素的个数
- 有深度：等于表中最深元素括号的重数
- 可递归：广义表可以以自己作为自己的元素子表
- 可共享：广义表可以作为其他广义表的元素，为其所共享



课堂互动 6.1





数组的定义及顺序存储实现

/ 数据的基本概念

/ 一维数组的顺序实现

/ 二维数组的顺序实现

/ 三维数组的顺序实现

数组的定义

什么是数组

数组（Array）是由n个具有相同类型的数据元素构成的有限序列，每个元素都称为数组的元素。

- 每个元素在n个线性关系中序号 $i_1,i_2...,i_n$ 称为数组的下标
- 下标的取值范围称为数组的维界
- 若数组中的每个元素都同时处于m个关系中，则称该数组为m维数组
- 数组的起点Loc(a[0])/Loc(a[0][0])被称为基地址，它是存储数组的存储空间的起始地址。通过数组元素的下标可以直接访问数组的元素。因此，数组的存储结构为**随机存取的存储结构**。

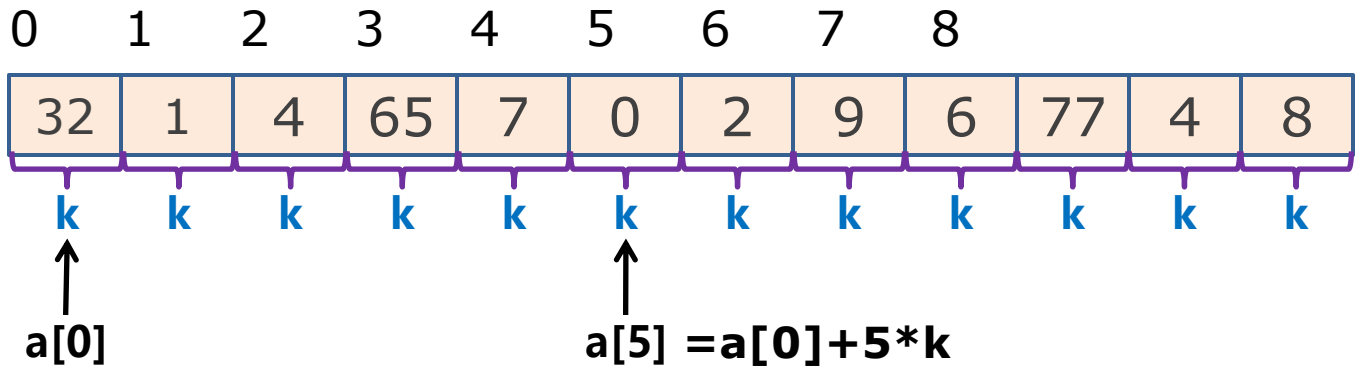
顺序存储实现

一维数组

- 一维数组采用顺序方式存储，数组中的元素按照一定次序顺序存放在一个连续的存储空间，可以看作是一个线性表。
- 计算机中的存储空间是一维的，一维数组可以直接映射到存储空间中

设存在一个长度为n的一维**数组A**的存储块的起始地址为Loc(a[0])，若已知A的每个元素占k个存单元，则下标为i的数组元素a[i]的存放地址Loc(a[i])可以表示为：

$$Loc(a[i]) = Loc(a[i-1]+k) = Loc(a[0]) + i * k, \quad 0 \le i < n$$



顺序存储实现

二维数组

二维数组（2D-Array）可以看作是数据元素是线性表的线性表，每个线性表元素由一个行向量或列向量构成。

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

行向量数组

$a_i=(a_{i1},a_{i2},\ldots,a_{in})$, 1≤i≤m

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

0 … m

行优先存储

列向量数组

$a_j=(a_{1j},a_{2j},\ldots,a_{mj})$, 1≤i≤m

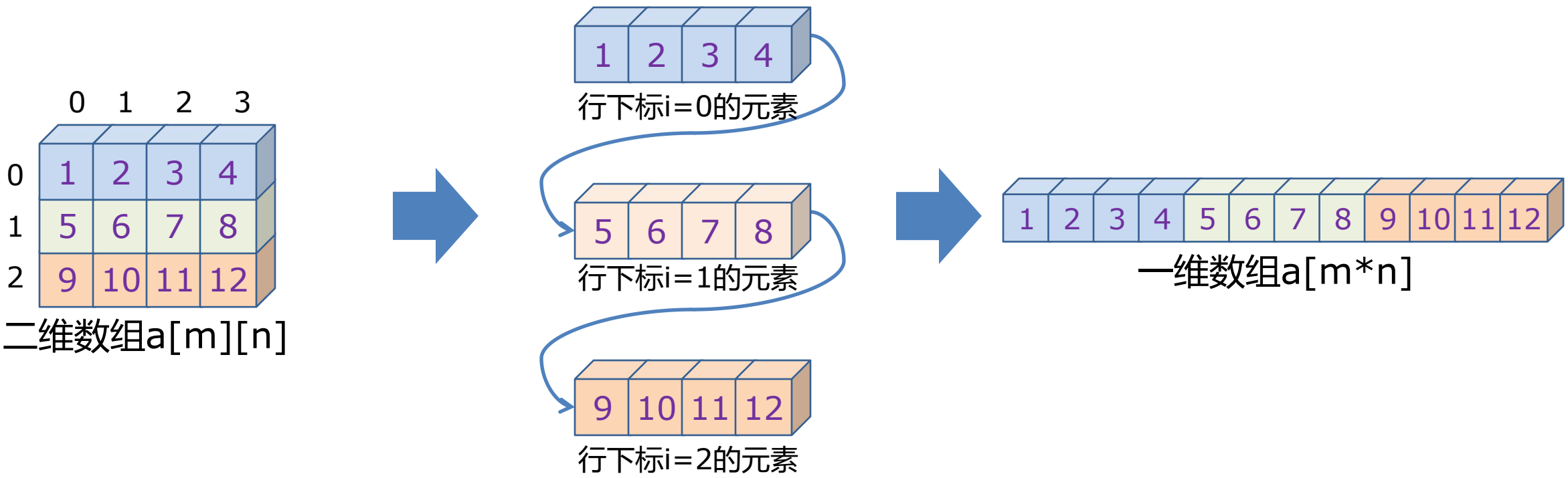
$$A_{m\times n}=\begin{bmatrix} [a_{11} & a_{12} & \cdots & a_{1n}] \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ [a_{m1} & a_{m2} & \cdots & a_{mn}] \end{bmatrix}$$

0 … n

列优先存储

顺序存储实现

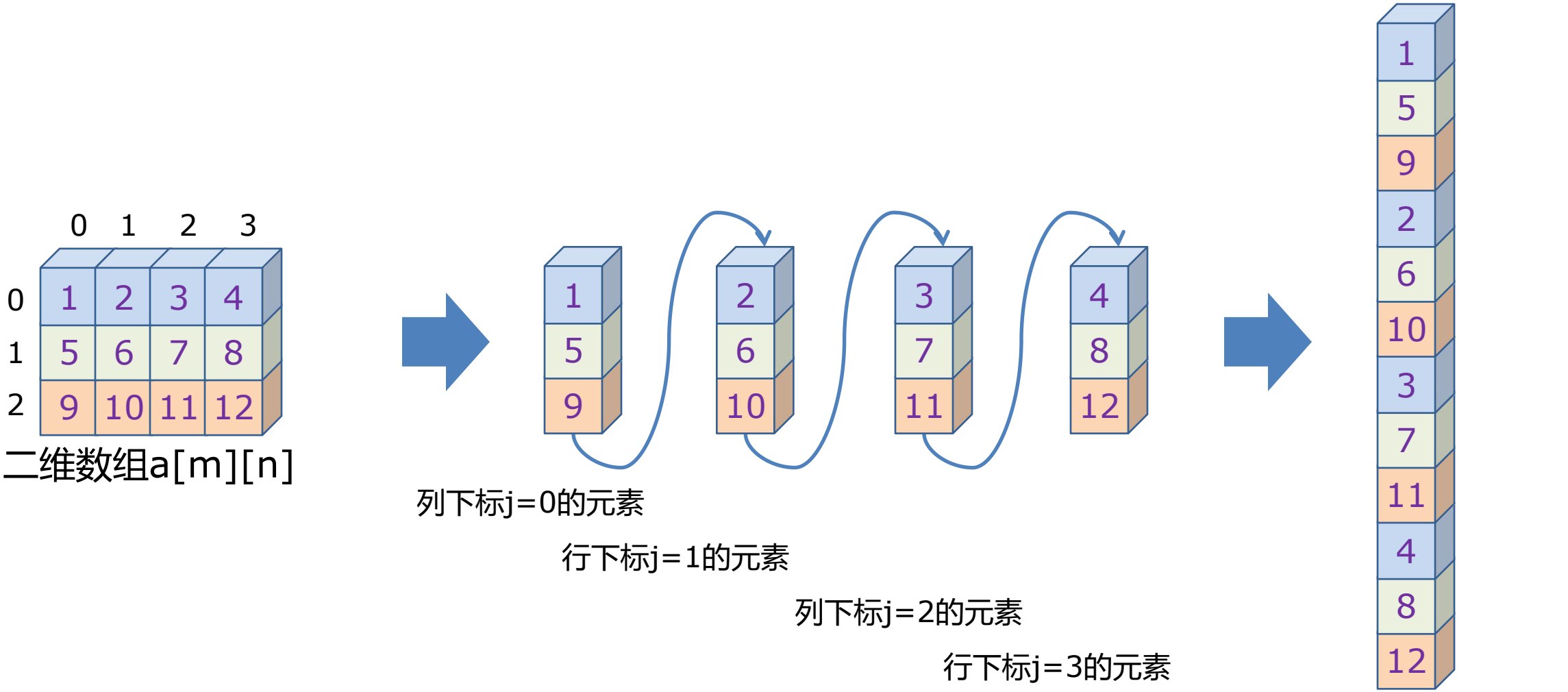
二维数组的存储 - 行优先存储



第i个元素的位置：loc(a[i][j]) = loc(a[0][0])+(i*n+j)*k （0≤i<m; 0≤j<n)

顺序存储实现

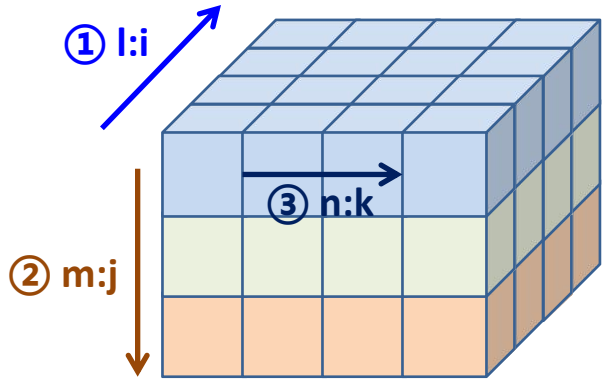
二维数组的存储 - 列优先存储



第i个元素的位置：loc(a[i][j]) = loc(a[0][0])+(j*m+i)*k (0≤i<m; 0≤j<n)

顺序存储实现

三维数组的存储 - 按页行列的顺序进行存储



三维数组
a[l][m][n]

a[l][m][n] 各维元素个数为l, m, n，则下标为i, j, k 的数组元素的存储位置为：

Loc(a[i][j][k])

= a[0][0][0] + i * m * n + j * n + k

i * m * n：前i_1页元素个数

j * n：第i页前j行元素个数

k：第j行前k列元素个数

顺序存储实现

例1：设一个二维数组 A[m][n] 按行优先进行顺序存储，假设A[0][0]存放位置为$644_{(10)}$，A[2][2]存放位置为$676_{(10)}$，每个元素占1个空间位置。请问A[3][4]$_{(10)}$存放地址是什么？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，则有：

✓ Loc(2, 2) = Loc(0, 0) + 2*n + 2 = 644 + 2*n + 2 = 676

=> n = (676 - 644 - 2)/2 = 15

✓ Loc(3, 4) = Loc(0, 0) + 3*15 + 4 = 644 + 45 + 4 = 693

顺序存储实现

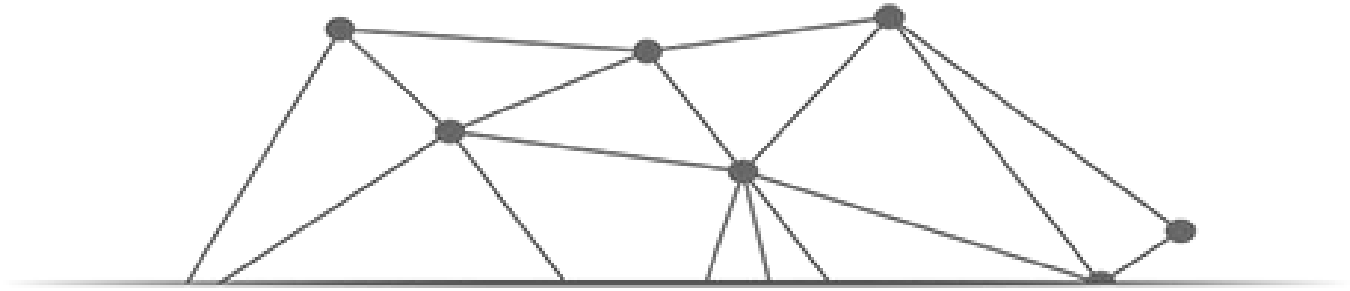
例2：设有二维数组 A[10][20]，其每个元素占2个字节，假设A[0][0]的存储位置为100，试给出 A[6][5] 在**按行优先**和**按列优先**模式下的地址。

✓ **按行优先**

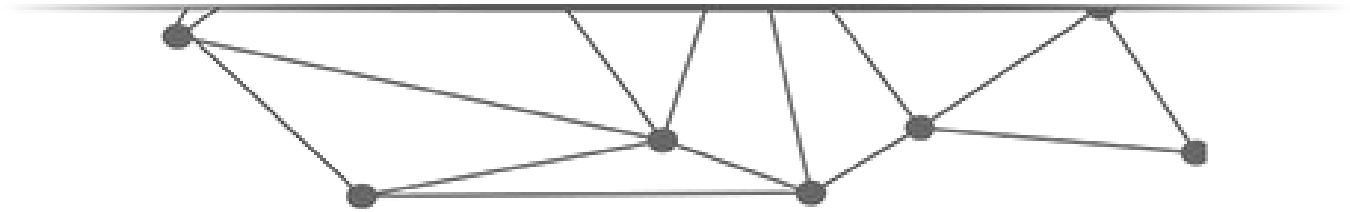
(6*20 + 5) * 2 + 100 = 350

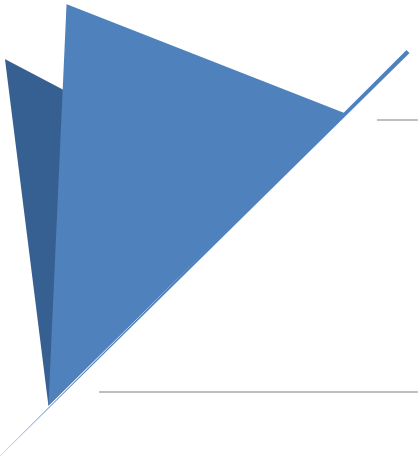
✓ **按列优先**

(5*10 + 6) * 2 + 100 = 212



课堂互动 6.2





特殊矩阵的压缩存储

/ 对称矩阵

/ 三角矩阵

/ 对角矩阵

/ 稀疏矩阵

特殊矩阵的压缩存储

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。

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。

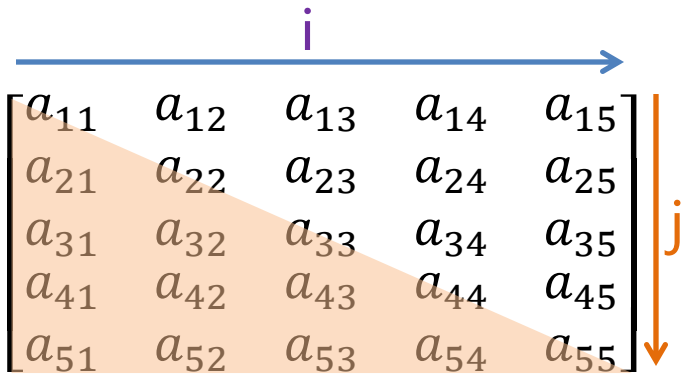
什么叫稀疏矩阵?

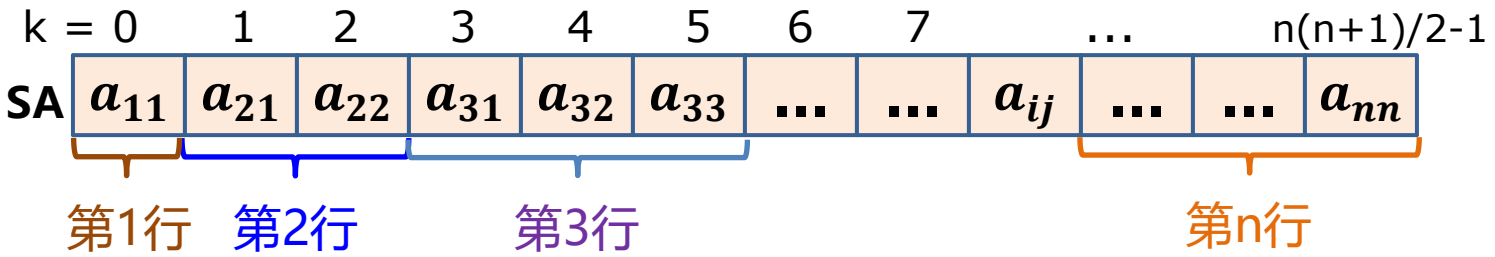
矩阵中非零元素的个数较少（一般小于5%），零元素的位置分布没有规律

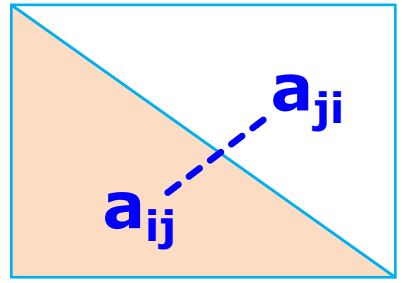
对称矩阵

- **矩阵特点：**在n×n矩阵中，若**A[i][j] = A[j][i]**，则称其为**对称矩阵。**
- **存储方法：**只存储下三角（或上三角）及主对角线中的元素
- **存储空间：n² ->n(n+1)/2**
- **存储位置：**设使用一维数组SA[n(n+1)/2](i≥0)来存储对称矩阵A，则矩阵A[i][j]在数组SA[k]中的存储位置**k** 可以表示为：

$$k=\begin{cases} i(i-1)/2+j-1 & ，当 i \geq j （下三角+主对角线） \\ j(j-1)/2+i-1 & ，当 i < j （下三角） \end{cases}$$





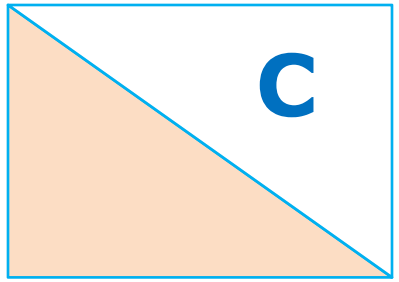


行优先顺序存储下三角

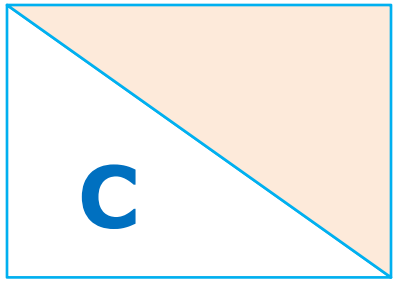
注意，若SA[](i>0)，则下标k值应做加一操作

三角矩阵

- **矩阵特点：**对角线以下（或以上）的数据元素（不包括对角线）全部为常数。



下三角矩阵



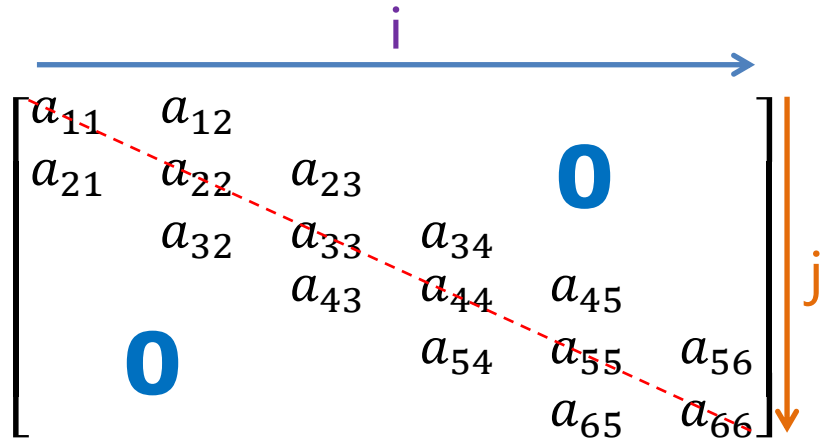
上三角矩阵

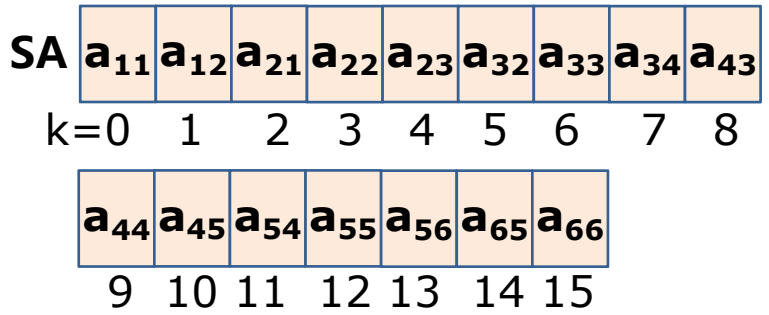
- **存储方法：**重复元素C共享一个元素存储空间。
- **存储空间：**n^2 ->n(n+1)/2
- **存储位置：**设使用一维数组SA[n(n+1)/2](i≥0)来存储对称矩阵A，则矩阵A[i][j]的在数组SA[k]中的存储位置**k** 可以表示为：
  - 下三角矩阵：$k=\begin{cases} i(i-1)/2+j-1 & ,i\geq j \\ n(n+1)/2 & ,i<j \end{cases}$
  - 上三角矩阵：$k=\begin{cases} (i-1)(2n-i+2)/2+j-i & ,i\leq j \\ n(n+1)/2 & ,i<j \end{cases}$

三对角矩阵

- **矩阵特点：**在n×n矩阵中，若非零元素集中在主对角线及其两侧共L条对角线的带状区域内，则称其为**对角矩阵**或**带状矩阵。**
- **存储方法：**只存储带状区域内元素。
- **存储空间：**除首行和末行，按每行L个元素，共 (n-2)L+(L+1) 个元素（以三对角矩阵为例）。
- **存储位置：**设使用一维数组SA[n(n+1)/2](i≥0)来存储对称矩阵A，则矩阵A[i][j]在数组SA[k]中的存储位置**k**可以表示为（以三对角矩阵为例）：

$$k = 2i + j - 3$$





N = (n-2)L+(L+1) = (6-2)*3+(3+1) =16

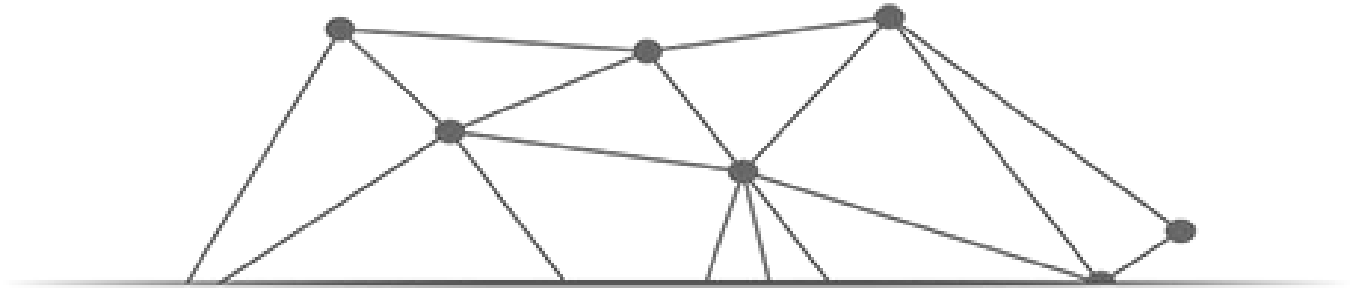
a_{54} = 2*5+4-3 = 11

稀疏矩阵

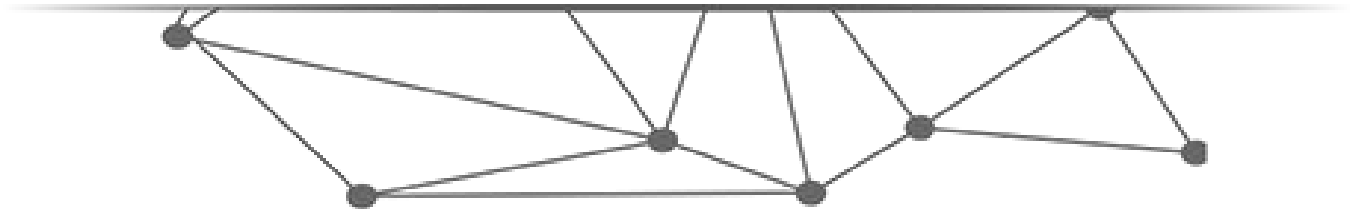
- **矩阵特点：**大多数元素为零(n>>t)的矩阵称为**稀疏矩阵。**
- **存储方法：**
  - **三元组**：只记录非零元素（i, j, a_{ij}）（行、列、值）节省空间，但丧失随机存储功能
  - **十字链表**：将行单链表和列单链表结合起来存储稀疏矩阵
- **存储空间：**元素个数×3

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 5 & 1 \\ 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

i	j	v
1	0	3
3	2	2
4	4	4
0	4	5
0	5	1



课堂互动 6.3



第06讲 广义表、数组和特殊矩阵

小　结

- **广义表**是**放松**了线性表对表原子元素的限制的数据结构，每个元素都允许具有其自身的结构。
- **数组**可以看成线性表的推广，数组一般采用顺序存储结构
- 存储数组时需要先将其转换为一维结构，例如按行转换（行优先）和按列转换（列优先）
  - 行优先存储地址：loc(a[i][j]) = loc(a[0][0])+(i*n+j)*k
  - 列优先存储地址：loc(a[i][j]) = loc(a[0][0])+(j*m+i)*k
- **特殊矩阵的压缩存储**
  - **对称矩阵**只存储下三角（或上三角）及主对角线中的元素
  - **三角矩阵**重复元素C共享一个元素存储空间
  - **对角矩阵**只存储带状区域内元素
  - **稀疏矩阵**只记录非零元素

数组地址转换

欧老师的联系方式

读万卷书 行万里路 只为最好的修炼

QQ：14777591 (宇宙骑士)

Email：ouxinyu@alumni.hust.edu.cn

Website: http://ouxinyu.cn

Tel：18687840023

地址：安宁校区 诚远楼201

南院 智能应用研究院A306-2