第五章　数组和特殊矩阵

- 5.1 数组
  - 5.1.1 数组的基本概念
  - 5.1.2 数组的存储结构
- 5.2 特殊矩阵的压缩存储
  - 5.2.1 对称矩阵的压缩存储
  - 5.2.2 三角矩阵的压缩存储
  - 5.2.3 对角矩阵的压缩存储
  - 5.2.4 稀疏矩阵的压缩存储

5.1.1 数组的基本概念

- 数组是程序设计中的常用数据类型。它分为一维数组、二维数组和多维数组。
- 一维数组是一个线性表。
- 二维数组和多维数组可看成是一维数组的推广。例如，二维数组：

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

- 二维数组可以看成是由多个行向量组成的向量，也可以看成是个列向量组成的向量。

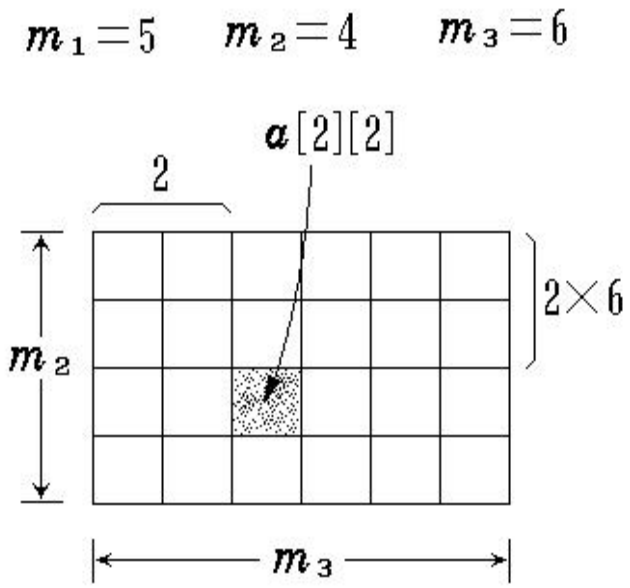
在C++语言中，一个二维数组类型可以定义为其分量类型为一维数组类型的一维数组类型，也就是说，

```
typedef  int  array2[m][n];
```

等价于：

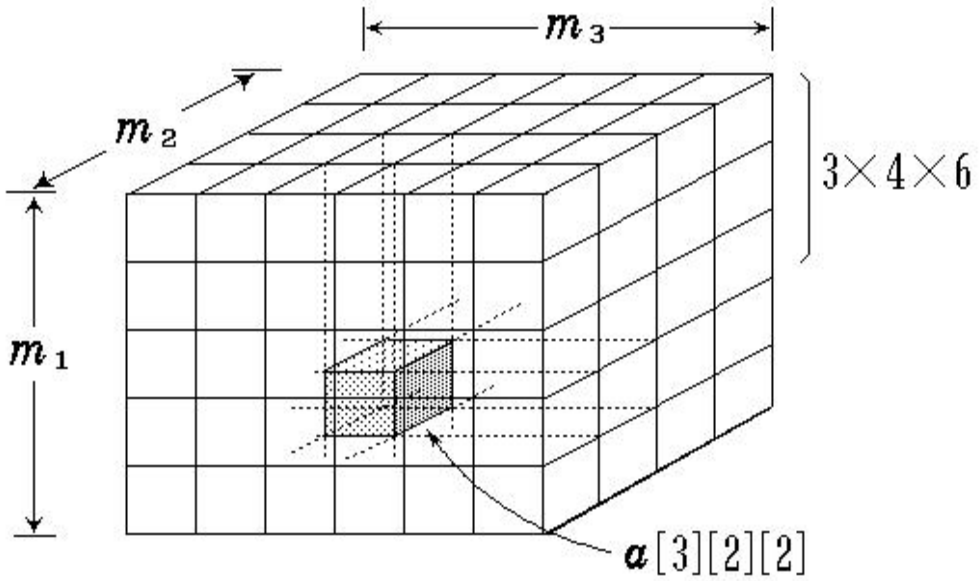
```
typedef  int  array1[n];
typedef  array1  array2[m];
```

二维数组



- 行向量 下标 *i*
- 列向量 下标 *j*

三维数组



页向量 下标 *i*

行向量 下标 *j*

列向量 下标 *k*

5.1.2　数组的存储结构

- 由于对数组一般不做插入和删除操作，也就是说，数组一旦建立，结构中的元素个数和元素间的关系就不再发生变化。因此，一般都是采用顺序存储的方法来表示数组。
- 由于计算机的内存结构是一维的，因此用一维内存来表示多维数组，就必须按某种次序将数组元素排成一列序列，然后将这个线性序列存放在存储器中。

通常有两种顺序存储方式：

(1) 行优先顺序——将数组元素按行排列，第i+1个行向量紧接在第i个行向量后面。

以二维数组为例，按行优先存储的线性序列为：

$a_{11}, a_{12}, \ldots, a_{1n}, a_{21}, a_{22}, \ldots a_{2n}, \ldots\ldots, a_{m1}, a_{m2}, \ldots, a_{mn}$

(2) 列优先顺序——将数组元素按列向量排列，第j+1个列向量紧接在第j个列向量之后

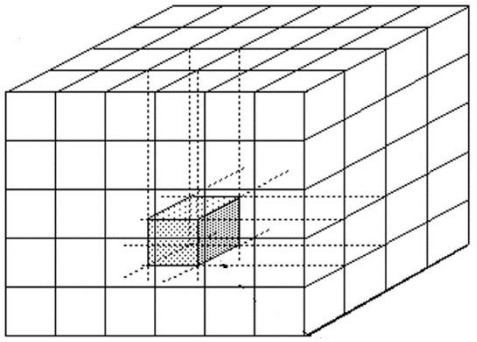
如二维数组A的m*n个元素按列优先存储的线性序列为：

$a_{11}, a_{21}, \ldots, a_{m1}, a_{12}, a_{22}, \ldots a_{m2}, \ldots\ldots, a_{1n}, a_{2n}, \ldots, a_{mn}$

以上规则可以推广到多维数组的情况：

- 行优先顺序？
  - 可规定为先排最右的下标，从右到左，最后排最左下标
- 列优先顺序？
  - 先排最左下标，从左向右，最后排最右下标

$$
\left.
\begin{array}{cccccc}
a_{000} & a_{001} & a_{002} & \cdots & a_{00,p-1} \\
a_{010} & a_{011} & a_{012} & \cdots & a_{01,p-1} \\
\cdots & \cdots & \cdots & \cdots & \cdots \\
a_{0,n-1,0} & a_{0,n-1,1} & a_{0,n-1,2} & \cdots & a_{0,n-1,p-1}
\end{array}
\right\} i = 0
$$



$$
\left.
\begin{array}{cccccc}
a_{100} & a_{101} & a_{102} & \cdots & a_{10,p-1} \\
a_{110} & a_{111} & a_{112} & \cdots & a_{11,p-1} \\
\cdots & \cdots & \cdots & \cdots & \cdots \\
a_{1,n-1,0} & a_{1,n-1,1} & a_{1,n-1,2} & \cdots & a_{1,n-1,p-1}
\end{array}
\right\} i = 1
$$

$$
\vdots
$$

$$
\left.
\begin{array}{cccccc}
a_{m-1,00} & a_{m-1,01} & a_{m-1,02} & \cdots & a_{m-1,0,p-1} \\
a_{m-1,10} & a_{m-1,11} & a_{m-1,12} & \cdots & a_{m-1,1,p-1} \\
\cdots & \cdots & \cdots & \cdots & \cdots \\
a_{m-1,n-1,0} & a_{m-1,n-1,1} & a_{m-1,n-1,2} & \cdots & a_{m-1,n-1,p-1}
\end{array}
\right\} i = m-1
$$

地址计算方法

二维数组元素a_{ij}的地址计算函数为：

行优先：$LOC(a_{ij})=LOC(a_{00})+(i*n+j)*d$

三维数组元素A_{ijk}的地址计算函数为：

行优先：$LOC(a_{ijk})=LOC(a_{000})+(i*n*p+j*p+k)*d$

5.2 特殊矩阵的压缩存储

- 在科学与工程计算问题中，矩阵是一种常用的数学对象，在高级语言编制程序时，常将一个矩阵描述为一个二维数组。
- 当矩阵中的非零元素呈某种规律分布或者矩阵中出现大量的零元素的情况下，会占用许多单元去存储重复的非零元素或零元素，这对高阶矩阵会造成极大的浪费。
- 为了节省存储空间，我们可以对这类矩阵进行压缩存储：
  - 即为多个相同的非零元素只分配一个存储空间；对零元素不分配空间。

5.2.1 特殊矩阵

— 是指非零元素或零元素的分布有一定规律的矩阵。

1、对称矩阵

在一个n阶方阵A中，若元素满足下述性质：

$$a_{ij} = a_{ji} \quad 0 \le i, j \le n-1$$

则称A为对称矩阵。

✓ 对称矩阵中的元素关于主对角线对称，故只要存储矩阵中上三角或下三角中的元素，这样，能节约近一半的存储空间。

$$\begin{pmatrix} 1 & 5 & 1 & 3 & 7 \\ 5 & 0 & 8 & 0 & 0 \\ 1 & 8 & 9 & 2 & 6 \\ 3 & 0 & 2 & 5 & 1 \\ 7 & 0 & 6 & 1 & 3 \end{pmatrix}$$

a_{00}

a_{10} a_{11}

a_{20} a_{21} a_{23}

……

$a_{n-1\,1}$ $a_{n-1\,2}$ …$a_{n-1\,n-1}$

图 对称矩阵

- 我们可以按行优先顺序将这些元素存放在一个向量sa[n(n+1)/2]中。

a_{00}	a_{10}	a_{11}	a_{20}	……	$a_{n-1,0}$	…	$a_{n-1,n-1}$

- 矩阵元素a_{ij}和数组分量sa[k]之间的对应关系如下：
  - k = i*(i+1)/2+j　　若 i >= j
  - k = j*(j+1)/2+i　　若 i < j

2、三角矩阵

✓ 以主对角线划分，三角矩阵有上三角和下三角两种。上三角矩阵如图所示，它的下三角（不包括主对角线）中的元素均为常数。

$$\begin{pmatrix} a_{00} & a_{01} & \cdots & a_{0\,n-1} \\ c & a_{11} & \cdots & a_{1\,n-1} \\ \cdots\cdots & \cdots\cdots & \cdots\cdots & \cdots\cdots \\ c & c & \cdots & a_{n-1\,n-1} \end{pmatrix}$$

(a)上三角矩阵

$$\begin{pmatrix} a_{00} & c & \cdots & c \\ a_{10} & a_{11} & \cdots & c \\ \cdots\cdots & \cdots\cdots & \cdots\cdots & \cdots\cdots \\ a_{n-1\,0} & a_{n-1\,1} & \cdots & a_{n-1\,n-1} \end{pmatrix}$$

(b)下三角矩阵

- 三角矩阵可压缩存储到向量sa[n(n+1)/2+1]中，其中常数c存放在向量的最后一个分量中.

- 对于上三角矩阵，若按行优先顺序存放矩阵中的元素a_{ij}时，sa[k]和a_{ij}的对应关系是：

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

- 下三角矩阵的存储和对称矩阵类似，sa[k]和a_{ij}对应关系是：

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

3、对角矩阵

- 对角矩阵中，所有的非零元素集中在以主对角线为中心的带状区域中，即除了主对角线和主对角线相邻两侧的若干条对角线上的元素之外，其余元素皆为零。

$$
\begin{pmatrix}
a_{00} & a_{01} & & & & \\
a_{10} & a_{11} & a_{12} & & & \\
 & a_{21} & a_{22} & a_{23} & & \\
 & & \cdots & \cdots & \cdots & \\
 & & & a_{n-2\,n-3} & a_{n-2\,n-2} & a_{n-2\,n-1} \\
 & & & & a_{n-1\,n-2} & a_{n-1\,n-1}
\end{pmatrix}
$$

- 一个k对角矩阵(k为奇数)A是满足下述条件的矩阵：若 $|i-j|>(k-1)/2$ ，则元素 $a_{ij}=0$。

- 对角矩阵可按行优先顺序或对角线的顺序，将其压缩存储到一个向量中，并且也能找到每个非零元素和向量下标的对应关系。

- 例如：若将三对角矩阵中的元素按行优先顺序存放在数组**sa[3n-2]**中，则**sa[k]**与三对角矩阵中的元素a_{ij}存在的对应关系为：

$$k = 3i-1+j-(i-1)=2*i+j$$