数组

数组 超n维数组　矩阵是二维数组

创建后不再插入删除

逻辑结构 $\begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \end{bmatrix}$

按行存储 → 物理结构 $[a_{00}\ a_{01}\ a_{02}\ a_{10}\ a_{11}\ a_{12}]$

按列存储 → $[a_{00}\ a_{10}\ a_{01}\ a_{11}\ a_{02}\ a_{12}]$

特殊矩阵压缩存储（节省存储空间）

对称矩阵　$a_{ij}=a_{ji}$　A[n][n]

按行序存储下三角+对角线

用一维数组 B[$n(n+1)/2$] 存储 A的n^2个元素

$B[k]=A[i][j]$　i,j从1开始　k从0开始

$$k=\begin{cases} \frac{i(i-1)}{2}+j-1 & i\geq j \\ \frac{j(j-1)}{2}+i-1 & i<j \end{cases}$$

$[a_{11}\ a_{12}\ a_{22}\ a_{31}\ a_{32}\ a_{33}\cdots a_{n1}\cdots a_{nn}]=B$

对下三角来说，第i行有i个元素 a_{ij} 的位置为下标 $1+2+3+\cdots+(i-1)+j$

存储下从0开始 $k=下标-1$

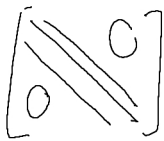
三角矩阵（上三角或下三角为同一个常数）
与对称阵相似

上三角

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

下三角

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

三对角矩阵  按行存储

$K=2i+j-3$

（$K=2i+j$

i,j从1开始 k从0开始

i,j从0开始 k从0开始）

稀疏矩阵存储

三元组
```
typedef struct {
    int i,j;
    T elem;
} Triple;
typedef struct{
    Triple data[MAXSIZE+1] //0号单用
    int rows, cols, total; //行数，列数，总数
} Matrix;
```

三元组转置算法（快速转置）

引入两个辅助向量 num 和 cpot

cpot[1]=1　　cpot[col]=cpot[col-1]+num[col-1]

例

A

i	j	e
1	2	12
1	3	19
3	1	-3
3	6	14
4	3	24
5	2	18
6	1	15
6	4	-7

→ T →

B

i	j	e
1	3	-3
1	6	15
2	1	12
2	5	18
3	1	19
3	4	24
4	6	-7
6	3	14

num[col]表示第col列有多少元素

cpot[col]表示第col列的第一行元素有转置后的位置

col	1	2	3	4	5	6	7
num	2	2	2	1	0	1	0
cpt	1	3	5	7	8	8	9

```
for(int p=1; p<A.total; p++){
    col=A.data[p].j;    index=cpot[col];
    B.data[index].i=A.data[p].j;
    B.data[index].j=A.data[p].i;
    B.data[index].elem=A.data[p].elem;
    cpot[col]++; // 该列下一行元素应放置的位置
}
```

十字链表

```
typedef  struct  Node{
        int i,j;
        T  elem;
        Node *right, *down;
} Node, *List;

typedef  struct {
        List  *rhead, *chead;
        int   rows, cols, total;
} Matrix;
```