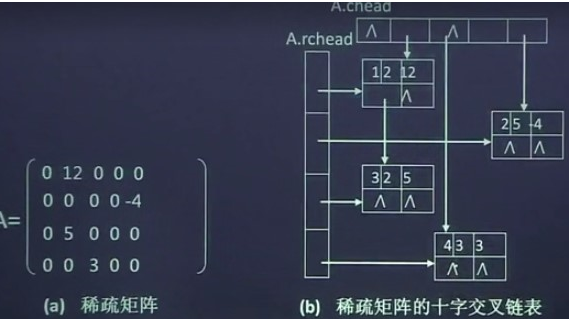
# 矩阵的存储

无规则矩阵在计算机中仅仅二维数组即可，但有规则的矩阵就可以减少空间。

以下约定：矩阵的元素a[i][j]下标从1开始，而数组A[k]下标k从0开始

* 对称矩阵：对称矩阵的有一半相同，所以用数组存储下边的一般可以减少空间，通过a11,a21,a22,a31,a32,a33,a41…的规则。k=+j-1
* 下三角矩阵：下半部分和对称矩阵一样的方式存储，k=+j-1，上半部分元素相同，都是c, 当k==+n 时A[+n]=c
* 上三角矩阵：上半部分和对称矩阵类似的方式存储，k=，下半部分元素相同，都是c, 当k==+n 时A[+n]=c
* 对角矩阵：把对角线元素放在一维数组即可。
* 三对角矩阵：从按照a11,a12,a21,a22,a23,a32,a33,a34,a43…k=2(I-1)+J-1
* 稀疏矩阵：设矩阵A是一个nxm的矩阵，里面有s个非零元素，设 δ = s /(nxm)，称矩阵δ为稀疏因子，如果某一矩阵的稀疏因子δ满足δ<=0.05时称为稀疏矩阵。

存储方法1：用三元组，即：(i,j,a[i][j])的方式，类似图的存储里边集数组的方式。

存储方法2：用十字链表：

# 高斯消元

<https://baike.baidu.com/item/%E9%AB%98%E6%96%AF%E6%B6%88%E5%85%83%E6%B3%95/619561?fr=aladdin>

**高斯消元：**就是通过矩阵初等变换，把一个任意矩阵，变成行梯阵矩阵，进而可以再变成最简矩阵。

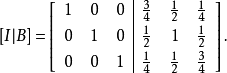
它用来求 矩阵的逆矩阵，也可以求解 线性方程组的解

https://gss2.bdstatic.com/-fo3dSag_xI4khGkpoWK1HF6hhy/baike/s%3D138/sign=1343a1e79a58d109c0e3adb1e959ccd0/c2cec3fdfc039245189f23968f94a4c27d1e2505.jpg

为了找到这个矩阵的逆矩阵，扩充以下矩阵：

https://gss2.bdstatic.com/-fo3dSag_xI4khGkpoWK1HF6hhy/baike/s%3D253/sign=a78620836859252da7171a01079a032c/8c1001e93901213fc6718c455ce736d12e2e95ed.jpg

通过计算，可以将增广矩阵转换为简化行阶梯形式，即把左边转化为单位矩阵：



通过初等变换化简过程，就是高斯消元法，

逆矩阵存在当且仅当，是正方形的矩阵。扩充列数由n\*n变成n\*2n 再化简前一半得到的后一半就是逆矩阵

# 线性基：

所谓线性基问题：就是给你一个有n个数的数组，有若干次询问或者修改，问你这些数选取一些数，的Xor最大值,最小值,以及能否得到某个值

线性基问题可以由高斯消元推导而来。

有个定理是：线性基的值域与原数组的值域相同

做法是开一个数组log[i]代表所有数中二进制下最高位是第i位的 数的xor总值，每次插入一个数a，从这个数最高位开始，从后往前找到第一个log是0的位，把它赋值成a xor前面几位的log的总XOR 比如现在log从高到底是：111011那么，那么log[2]=a xor log[5] xor log[4] xor log[3]

最值：以最大值为例，初始化一个ans,从log数组高位开始，逐个xor log[i] 这个过程中得到的最大ans值就是所有数选取任意个能得到的最大值。

## 查询某个数：

就是查找某个数是否可以由这 n 个数中任意个数异或得到

从高位向地位扫一遍，log的第 i 位为 1，就异或上 log[i]，扫完后。如果变成 0 了，那么就是可以的。