北斗学友科学菁英成长课程研究院

信息学奥林匹克竞赛

数学专题

拔尖创新人才成长平台

向量

- 由n个数组成的有序数组称为n维向量

$$\boldsymbol{\alpha} = [\, a_1 \quad a_2 \cdots \; a_n \,], \quad \boldsymbol{\beta} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

向量

- 规定:两个n维列(行)向量相等当且仅当它们对应的n个分量都相等

$$\begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \quad \Leftrightarrow \quad \begin{cases} a_1 = b_1 \\ a_2 = b_2 \\ a_3 = b_3 \end{cases}$$

向量

- 两个维数相同的列(行)向量可以做向量加法---对应元素相加

$$\alpha+\beta=\begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}+\begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}=\begin{bmatrix} a_1+b_1 \\ a_2+b_2 \\ a_3+b_3 \end{bmatrix}$$

向量

- 向量的k倍(k是数)---每个分量都k倍

$$k\begin{bmatrix}1\\2\\3\end{bmatrix}=\begin{bmatrix}k\\2k\\3k\end{bmatrix}$$

向量

向量：既有大小又有方向的量称为向量。数学上研究的向量为 **自由向量**，即只要不改变它的大小和方向，起点和终点可以任意平行移动的向量。记作 $\vec{a}$ 或 $\boldsymbol{a}$。

有向线段：带有方向的线段称为有向线段。有向线段有三要素：**起点，方向，长度**，知道了三要素，终点就唯一确定。一般使用有向线段表示向量。

向量的模：有向线段 $\overrightarrow{AB}$ 的长度称为向量的模，即为这个向量的大小。记为：$|\overrightarrow{AB}|$ 或 $|\boldsymbol{a}|$。

零向量：模为 0 的向量。零向量的方向任意。记为：$\vec{0}$ 或 $\mathbf{0}$。

单位向量：模为 1 的向量称为该方向上的单位向量。一般记为 $\vec{e}$ 或 $\boldsymbol{e}$。

平行向量：方向相同或相反的两个 **非零** 向量。记作：$\boldsymbol{a} \parallel \boldsymbol{b}$。对于多个互相平行的向量，可以任作一条直线与这些向量平行，那么任一组平行向量都可以平移到同一直线上，所以平行向量又叫 **共线向量**。

相等向量：模相等且方向相同的向量。

相反向量：模相等且方向相反的向量。

向量

- 为了更直观的理解线性表出， 我们给出向量加法， 数乘的几何意义。
- 向量加法的平行四边形法则
- 向量数乘是将其缩放至k倍
- 若向量α**能表示成**$\boldsymbol{\alpha_1, \alpha_2 \ldots \alpha_s}$**的线性组合，即存在数**$\boldsymbol{k_1, k_2 \ldots k_s}$**，使得**
- $\boldsymbol{\alpha_1 k_1 + \alpha_2 k_2 \cdots \alpha_s k_s} = \alpha$
- 则称向量组$\boldsymbol{\alpha_1, \alpha_2 \ldots \alpha_s}$线性表出α， 此时称向量组$\boldsymbol{\alpha_1, \alpha_2 \ldots \alpha_s}, \alpha$线性相关

线性相关

给定一组向量$\boldsymbol{\alpha_1}, \boldsymbol{\alpha_2} \dots \boldsymbol{\alpha_s}$，存在一组不全为0的数$\boldsymbol{k_1}, \boldsymbol{k_2} \dots \boldsymbol{k_s}$，使得$\boldsymbol{\alpha_1 k_1 + \alpha_2 k_2 \cdots \alpha_s k_s = 0}$，则称这组向量线性相关，否则称这组向量线性无关。

- 可以证明，向量组$\boldsymbol{\alpha_1}, \boldsymbol{\alpha_2} \dots \boldsymbol{\alpha_s}$线性相关的充分必要条件是，其中至少有一个向量可以由其余向量线性表出。
- s个n维向量线性相关的几何意义就是，这s个向量无法撑起一个s维的空间，形象化理解以3个3维向量线性相关为例，它们一定在一个平面上。

线性相关

- 为了更直观的理解线性表出，我们给出向量加法，数乘的几何意义。
- 向量加法的平行四边形法则
- 向量数乘是将其缩放至k倍
- 若向量α**能表示成**$\boldsymbol{\alpha_1, \alpha_2 \ldots \alpha_s}$**的线性组合，即存在数**$\boldsymbol{k_1, k_2 \ldots k_s}$**，使得**
- $\boldsymbol{\alpha_1 k_1 + \alpha_2 k_2 \cdots \alpha_s k_s} = \alpha$
- 则称向量组$\boldsymbol{\alpha_1, \alpha_2 \ldots \alpha_s}$线性表出α，此时称向量组$\boldsymbol{\alpha_1, \alpha_2 \ldots \alpha_s}, \alpha$线性相关

矩阵

在数学中，矩阵（Matrix）是指纵横排列的二维数据表格，最早来自于方程组的系数及常数所构成的方阵。

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

m 行 n 列矩阵

矩阵

- 单位矩阵I：对角线值全为1的方阵为单位矩阵
- 对角矩阵：只有主对角线有值，其他位置全为0的方阵
- 上三角矩阵：$\boldsymbol{a_{ij}}=0|i>j$的方阵
- 下三角矩阵：$\boldsymbol{a_{ij}}=0|i<j$的方阵
- 负矩阵：-A为$\boldsymbol{a_{ij}}x$取相反数的矩阵
- 行向量：只有一行的矩阵
- 列向量：只有一列的矩阵

矩阵加法

- 行数相同,列数也相同的两个矩阵 可以做矩阵加法 --- 对应元素相加

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ a_{31} & a_{32} \end{bmatrix} + \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \\ b_{31} & b_{32} \end{bmatrix} = \begin{bmatrix} a_{11}+b_{11} & a_{12}+b_{12} \\ a_{21}+b_{21} & a_{22}+b_{22} \\ a_{31}+b_{31} & a_{32}+b_{32} \end{bmatrix}$$

矩阵数乘

- 每个矩阵都可以与数 k 作数乘　即每个元素 k 倍

$$k\begin{bmatrix} 1 & a_1 \\ 2 & a_2 \\ 3 & a_3 \end{bmatrix} = \begin{bmatrix} k & ka_1 \\ 2k & ka_2 \\ 3k & ka_3 \end{bmatrix}$$

矩阵转置

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 \\ 1 & 4 & 9 & 16 \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 4 \\ 1 & 3 & 9 \\ 1 & 4 & 16 \end{bmatrix}$$

矩阵乘法

- 定义：若一矩阵的列数与另一矩阵的行数相等，则可定义这两个矩阵的乘积。
- 如 A 是 m×n 矩阵和 B 是 n×p矩阵，它们是乘积 AB 是一个 m×p 矩阵，其中c[i][j]=Σa[i][k]*b[k][j];
- 由定义可知：
- 1：当矩阵A的列数等于矩阵B的行数时，A与B可以相乘。
- 2：矩阵C的行数等于矩阵A的行数，C的列数等于B的列数。
- 3：乘积C的第i行第j列的元素c[i][j]等于矩阵A的第i行的元素与矩阵B的第j列对应元素乘积之和。

矩阵乘法

- 1.大多数情况下矩阵乘法不满足交换律。
- 2.矩阵乘法满足结合律。（假设你有三个矩阵A、B、C，那么(AB)C和A(BC)的结果的第i行第j列上的数都等于所有A(ik)*B(kl)*C(lj)的和）
- 逆矩阵（类比逆元）：设A是数域上的一个n阶方阵，若在相同数域上存在另一个n阶矩阵B，使得：AB=BA=E。则我们称B是A的逆矩阵，而A则被称为可逆矩阵。
- 若要求A^k可以通过快速幂来求，实现与普通快速幂基本相同。
- 满足结合律的运算大多可以快速幂。

矩阵与邻接矩阵

- 我们考虑一个邻接矩阵G，如果i向j有一条边，则G[i][j]=1;
- G^2则表示邻接矩阵的自乘，$G^2[a][b]=\Sigma G[a][i]*G[i][b]$;
- 上述中，G[a][i],G[i][b]的值非0则1，显然当G[a][i]&&G[i][b]时，$G^2[a][b]$会加上1，怎么理解呢？
- 也就是说如果a到i有一条边且i到b有一条边，那么a到b有一条经过且仅经过一个点的边，最后$G^2[a][b]$则表示经过且仅经过一个点路径数

矩阵与邻接矩阵

- 继续考虑$G^3[a][b]=\sum G[a][i]*G[i][j]*G[j][b]$;
- 类比G^2，那么G^3则表示从a到b经过了两个点的路径数
- 另一方面，$G^3[a][b]=\sum G^2[a][i]*G[i][b]$;同理可以推出上述结论
- 一般的，G^k[a][b]则表示经过了k-1个点的路径数
- 特别的，G^0为单位矩阵
- 如果有重边，则G[a][b]表示a到b个边的个数即可，推导类似

路径计数

- 给定一张n个点有向图，问从a到b经过k条边的方案数
- n<=100,k<=10^9

路径计数

- 给定一张n个点有向图，问从a到b最多经过k条边的方案数
- n<=100,k<=10^9

路径计数

- 给定一张n个点有向图，问从a到b最多经过k1~k2条边的方案数
- n<=100,k1,k2<=10^9

矩阵的初等变换

第 i 行乘非零数 k： $B \mapsto D_i(k)B$。

第 i，j 行互换： $B \mapsto P_{ij}B$。

第 j 行乘 k 加到第 i 行： $B \mapsto T_{ij}(k)B$。

矩阵与线性方程组

- 现在考虑如下问题:
- 给定一组$\boldsymbol{\alpha_1}, \boldsymbol{\alpha_2} \cdots \boldsymbol{\alpha_n}$，问其是否能线性表出$\boldsymbol{\beta}$，**即是否存在一组实数$\boldsymbol{x_1}, \boldsymbol{x_2} \ldots \boldsymbol{x_n}$，使得**$\boldsymbol{\alpha_1 x_1} + \boldsymbol{\alpha_2 x_2} \cdots \boldsymbol{\alpha_n x_n} = \beta$
- 这实际上是一个解多元线性方程问题

$$
\begin{aligned}
a_{11} x_1 + a_{12} x_2 + \cdots + a_{1n} x_n &= b_1 \\
a_{21} x_1 + a_{22} x_2 + \cdots + a_{2n} x_n &= b_2 \\
a_{31} x_1 + a_{32} x_2 + \cdots + a_{3n} x_n &= b_3 \\
\cdots \qquad & \cdots \\
a_{m1} x_1 + a_{m2} x_2 + \cdots + a_{mn} x_n &= b_m
\end{aligned}
$$

矩阵与线性方程组

$$\begin{cases} 7x_1 + 8x_2 + 9x_3 = 13 \\ 4x_1 + 5x_2 + 6x_3 = 12 \\ x_1 + 2x_2 + 3x_3 = 11 \end{cases} \Rightarrow \begin{pmatrix} 7 & 8 & 9 \\ 4 & 5 & 6 \\ 1 & 2 & 3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 13 \\ 12 \\ 11 \end{pmatrix}$$

$$Ax = b$$

矩阵与线性方程组

- 这个方程组的解可能有3种形式:
  - 1. 有唯一解
  - 2. 无解
  - 3. 有多解
- 我们对着m个线性方程进行以下3个操作不改变这个方程组的解结构:
  - 1. 交换两行
  - 2. 一个方程加上另一个方程的任意倍数
  - 3. 用非零数乘某一方程
- 这3个操作称为等解变换/初等变换

高斯消元

- Gauss消元法是解决多元线性方程组的一个算法，它的内容核心就是不断利用上面3种操作，使得方程矩阵变成阶梯型矩阵或者对角矩阵，达到求解的目的。
- 具体操作：任取一个没有选择过的方程i，找到该方程中一个不为0的系数$\boldsymbol{a}_{\mathrm{ij}}$，将该方程乘某个常数加到其他方程中去，使得其他方程k的$\boldsymbol{a}_{\boldsymbol{kj}}$为0即可。

高斯消元

$$\begin{cases} 3x_1 + x_2 - 7x_3 + 5x_4 = 3 \\ x_1 - x_2 - 5x_3 + 2x_4 = 0 \\ 2x_2 + 4x_3 - x_4 = 1 \\ x_1 + x_2 - x_3 + 3x_4 = 3 \end{cases}$$

高斯消元

$$\begin{cases} 3x_1 + x_2 - 7x_3 + 5x_4 = 3 \\ x_1 - x_2 - 5x_3 + 2x_4 = 0 \\ \qquad 2x_2 + 4x_3 - x_4 = 1 \\ x_1 + x_2 - x_3 + 3x_4 = 3 \end{cases}$$

高斯消元

选好非零系数, 交换到第一行

$$
\begin{aligned}
3x_1 + x_2 - 7x_3 + 5x_4 &= 3 \\
x_1 - x_2 - 5x_3 + 2x_4 &= 0 \\
2x_2 + 4x_3 - x_4 &= 1 \\
x_1 + x_2 - x_3 + 3x_4 &= 3
\end{aligned}
$$

高斯消元

第一列开始消元

-3 倍

$$\begin{aligned} x_1 - x_2 - 5x_3 + 2x_4 &= 0 \\ 3x_1 + x_2 - 7x_3 + 5x_4 &= 3 \\ 2x_2 + 4x_3 - x_4 &= 1 \\ x_1 + x_2 - x_3 + 3x_4 &= 3 \end{aligned}$$

高斯消元

-1 倍

$$
\begin{aligned}
x_1 - x_2 - 5x_3 + 2x_4 &= 0 \\
4x_2 + 8x_3 - x_4 &= 3 \\
2x_2 + 4x_3 - x_4 &= 1 \\
x_1 + x_2 - x_3 + 3x_4 &= 3
\end{aligned}
$$

高斯消元

第一行不动, 下面的行重复以上过程

$$
\begin{aligned}
x_1 - x_2 - 5x_3 + 2x_4 &= 0 \\
\boxed{\begin{aligned}
4x_2 + 8x_3 - x_4 &= 3 \\
2x_2 + 4x_3 - x_4 &= 1 \\
2x_2 + 4x_3 + x_4 &= 3
\end{aligned}}
\end{aligned}
$$

高斯消元

选好的系数，交换

$$
\begin{aligned}
x_1 - x_2 - 5x_3 + 2x_4 &= 0 \\
4x_2 + 8x_3 - x_4 &= 3 \\
2x_2 + 4x_3 - x_4 &= 1 \\
2x_2 + 4x_3 + x_4 &= 3
\end{aligned}
$$

高斯消元

再消元

-2 倍

$$
\begin{aligned}
x_1 - x_2 - 5x_3 + 2x_4 &= 0 \\
2x_2 + 4x_3 - x_4 &= 1 \\
4x_2 + 8x_3 - x_4 &= 3 \\
2x_2 + 4x_3 + x_4 &= 3
\end{aligned}
$$

高斯消元

再消元

$$x_1 - x_2 - 5x_3 + 2x_4 = 0$$

$\boxed{-1\text{倍}}$

$$2x_2 + 4x_3 - x_4 = 1$$

$$x_4 = 1$$

$$2x_2 + 4x_3 + x_4 = 3$$

高斯消元

第一，二行不动，下面的行消元

$$x_1 - x_2 - 5x_3 + 2x_4 = 0$$

$$2x_2 + 4x_3 - x_4 = 1$$

$$\boxed{\begin{aligned} x_4 &= 1 \\ 2x_4 &= 2 \end{aligned}}$$

高斯消元

x_3的系数全为 0，看 x_4 的系数

$$x_1 - x_2 - 5x_3 + 2x_4 = 0$$

$$2x_2 + 4x_3 - x_4 = 1$$

$$x_4 = 1$$

$$2x_4 = 2$$

高斯消元

选好系数消元

$$x_1 - x_2 - 5x_3 + 2x_4 = 0$$

$$2x_2 + 4x_3 - x_4 = 1$$

$$x_4 = 1$$

$$2x_4 = 2$$

_2 倍

高斯消元

$$
\begin{array}{rrrrcl}
x_1 & - \ x_2 & -\ 5x_3 & +\ 2x_4 & = & 0 \\
 & 2x_2 & +\ 4x_3 & -\ x_4 & = & 1 \\
 & & & x_4 & = & 1 \\
 & & & 0 & = & 0
\end{array}
$$

主元

非零行左起第一个非零系数称为该行的主元

高斯消元

- 考虑消元后得到的阶梯型矩阵，根据其性质得到原方程组的解结构，按照以下顺序依次判定。
- 若该矩阵出现了0=d(d!=0)的方程，则方程组无解
- 若该矩阵的$\boldsymbol{a}_{\mathrm{ii}}$都不为0，则方程组有唯一解
- 若存在某一个变量，该变量所有方程的系数都是零则有无穷解（多解）
- 这个算法的时间复杂度是$\boldsymbol{n}^{3}$，分析如下：
- 每次任取一个方程复杂度o(n)，选取其他方程复杂度o(n)，枚举该方程的所有系数对应相减o(n)。

行列式

- 下面我们引入行列式的概念，为了容易理解，我们由几何意义引入高阶行列式，而不是直接给出代数公式
- 定义一个二阶方阵的行列式如下：
- $\begin{vmatrix} \boldsymbol{a} & \boldsymbol{c} \\ \boldsymbol{b} & \boldsymbol{d} \end{vmatrix} = ad - bc$
- 这个行列式的几何意义是，平面上两个向量(a,b),(c,d)构成平行四边形的有向面积
- 我们希望将行列式的几何意义推广到高阶来表示高维平行多面体的“有向体积”，以3阶行列式为例我们猜测一下他的公式。

行列式

- 在$\boldsymbol{R}^3$中任意2个向量只能构成一个平面，不存在体积的概念，所以我们要取3个向量构成一个平行多面体。
- 3个向量不能线性相关否则构成的依然是一个平面。
- 做了一点简单的猜想后，我们需要推倒三阶行列式的公式，设三个列向量构成的3阶矩阵如下：

$$\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} = ??$$

- 仿照二阶我们猜测公式是这些元素的3次齐次多项式

行列式的性质

- 虽然我们还不能确定公式中哪些项是正哪些项是负，但是我们可以通过行列式的几何意义来确定一些性质：
- 1.$|A|=|A^T|$（行向量列向量的区别）
- 2.若给一列乘k，则行列式的值乘k（将某一向量放缩k倍）
- 3.若行列式的某一列（行）的元素都是两数之和，则这个行列式是对应两个行列式的和（将某一向量平行四边形拆分）
- 4.互换行列式的两行（列），行列式变号（有向体积定义）
- 5.行列式如果有两行（列）元素成比例，则此行列式等于零（这组向量线性相关，无法构成n维超空间）
- 6.把行列式的某一列（行）的各元素乘以同一数然后加到另一列（行）对应的元素上去，行列式不变（对某个向量在剩余n-1维超空间的方向上平移）

关于正负

$+a_{11}a_{22}a_{33}$ $+a_{12}a_{23}a_{31}$ $+a_{13}a_{21}a_{32}$

$$\begin{bmatrix} 1 & & \\ & 1 & \\ & & 1 \end{bmatrix} \quad \begin{bmatrix} & 1 & \\ & & 1 \\ 1 & & \end{bmatrix} \quad \begin{bmatrix} & & 1 \\ 1 & & \\ & 1 & \end{bmatrix}$$

$-a_{11}a_{23}a_{32}$ $-a_{13}a_{22}a_{31}$ $-a_{12}a_{21}a_{33}$

$$\begin{bmatrix} 1 & & \\ & & 1 \\ & 1 & \end{bmatrix} \quad \begin{bmatrix} & & 1 \\ & 1 & \\ 1 & & \end{bmatrix} \quad \begin{bmatrix} & 1 & \\ 1 & & \\ & & 1 \end{bmatrix}$$

n阶行列式公式

- 由两行交换符号取反，我们可以推导出行列式哪些项是哪些项是负，下面直接给出行列式计算公式

$$\det(A) = \sum_{(j_1 j_2 \cdots j_n)} (-1)^{N(j_1 j_2 \cdots j_n)} a_{j_1 1} a_{j_2 2} \cdots a_{j_n n}$$

- 经过代数验证可以证明上面的6条性质都成立。

行列式计算

- 如果按照定义来计算行列式，复杂度是n*n!的，阶乘级别的复杂度难以接受
- 需要根据上面行列式的一些性质来找到复杂度更低的做法
- 注意到上三角矩阵的行列式是对角线元素的乘积
- 注意到对该矩阵做行变换不改变行列式的值
- 所以我们可以利用高斯消元将其消成上三角矩阵，然后将对角线元素做乘积即可得到行列式，时间复杂度n^3

行列式作用

- 计算一个图的生成树计数
- Matrix-Tree定理
- 首先定义一个无向图的基尔霍夫矩阵
- 度数矩阵 a[i][i]为点i的度 其他位置为0
- 邻接矩阵 a[i][i]=0; a[i][j]=[i与j相连]
- 基尔霍夫矩阵 = 度数矩阵 - 邻接矩阵
- 则该图的生成树个数为，基尔霍夫矩阵任意去掉第i行第i列，的行列式绝对值
- 证明跳过，有兴趣参考vfk http://vfleaking.blog.163.com/blog/static/1748076342013112523651955/

线性基

称线性空间 V 的一个极大线性无关组为 V 的一组 **Hamel 基** 或 **线性基**，简称 **基**。

规定线性空间 $\{\theta\}$ 的基为空集。

可以证明任意线性空间均存在线性基[1]，我们定义线性空间 V 的 **维数** 为线性基的元素个数（或势），记作 $\dim V$。

线性基

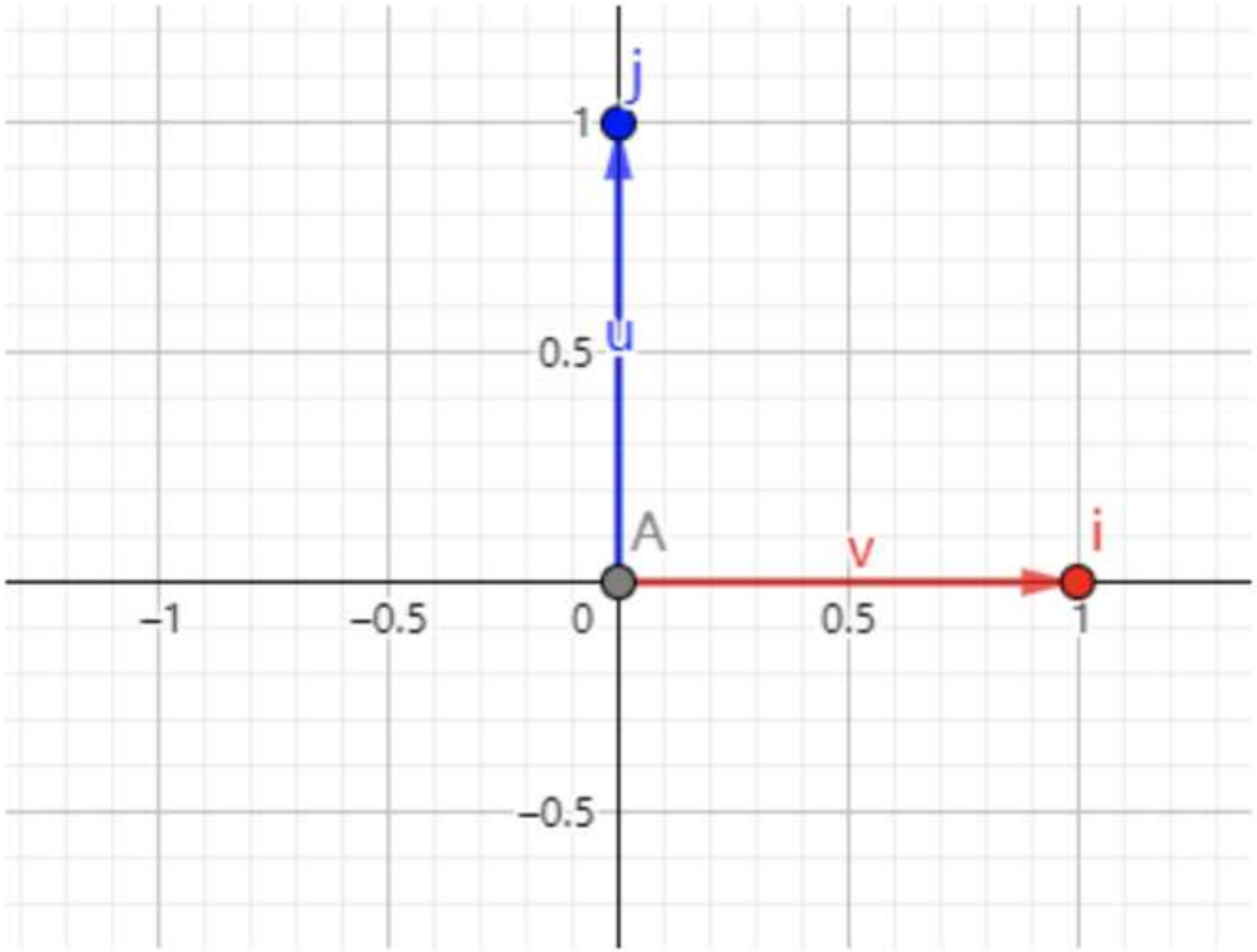
对于有限维线性空间 V, 设其维数为 n, 则:

a. V 中的任意 $n+1$ 个向量线性相关。

b. V 中的任意 n 个线性无关的向量均为 V 的基。

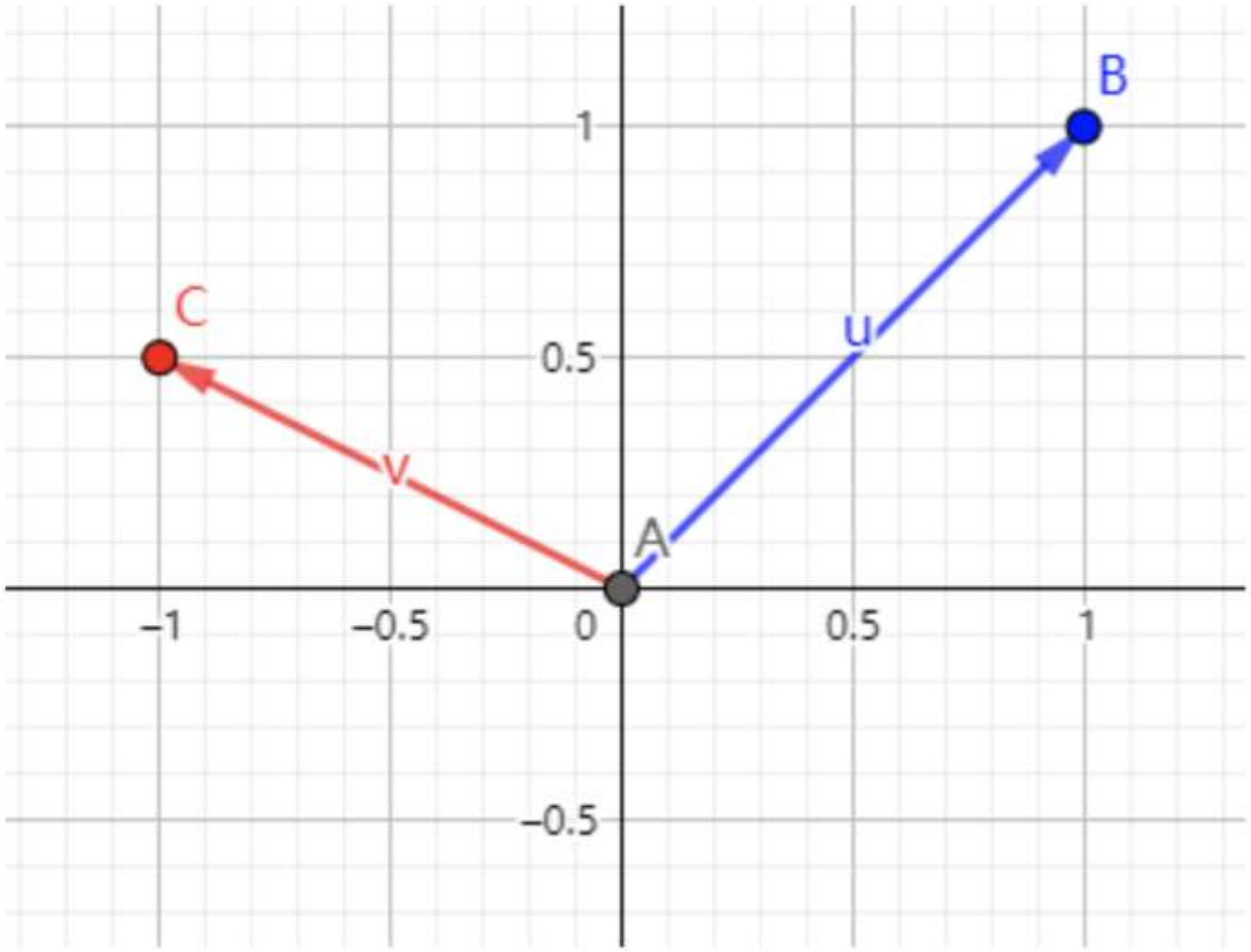
c. 若 V 中的任意向量均可被向量组 $a_1, a_2, \ldots, a_n$ 线性表出，则其是 V 的一个基。

在二维平面上的基



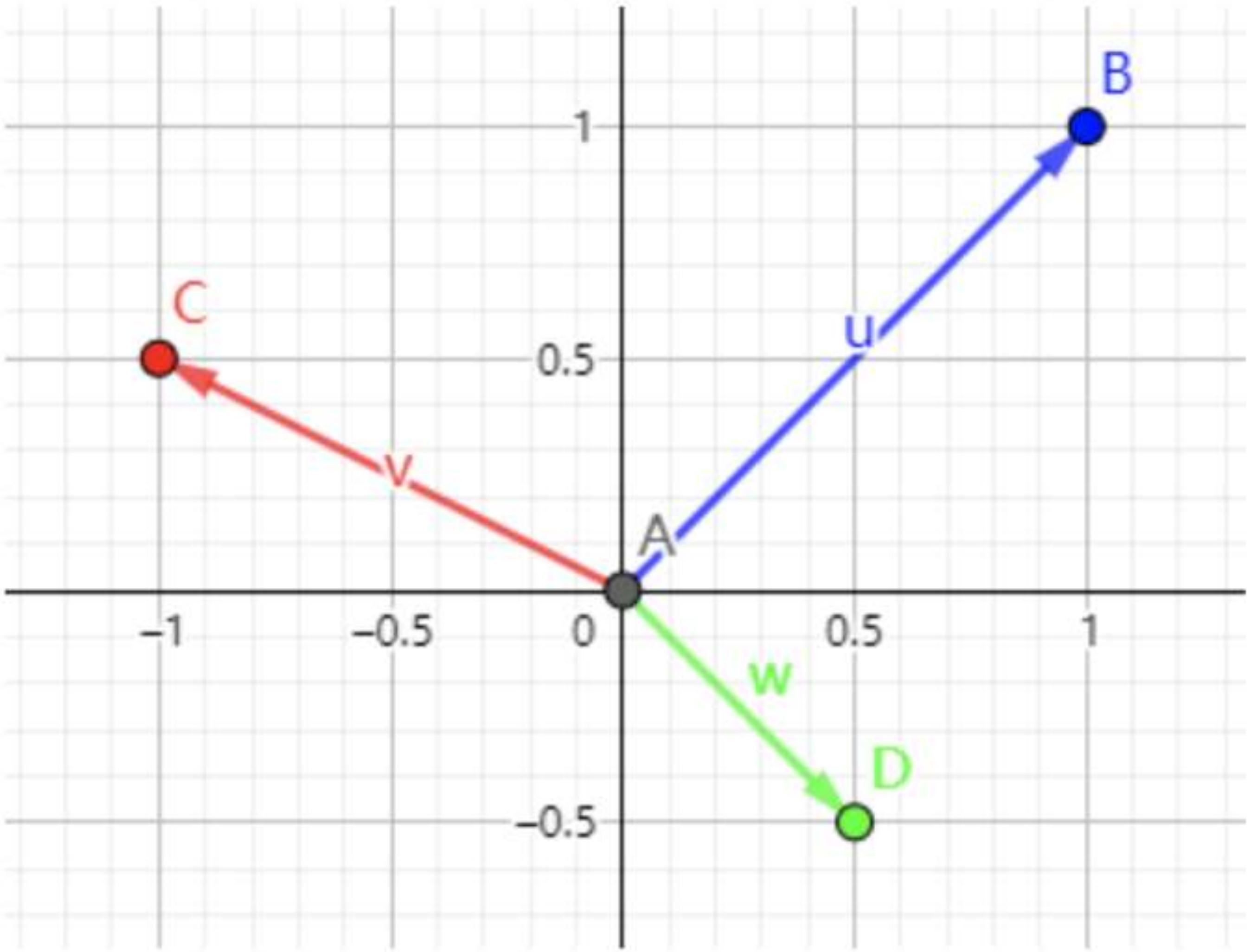
u, v 是一组基。

在二维平面上的基



u, v 是一组基。

在二维平面上的基



u, v, w 不是一组基，因为 $u + 4v + 6w = \theta$.

异或线性基

以异或线性基为例，我们可以根据给定的一组布尔序列 $\{x_1, ..., x_m\}$ 构造出一组异或线性基 $B = \{b_1, ..., b_n\}$，这组基有如下性质：

1. B 中任意非空子集的异或和不为 0；
2. 对 X 中的任意元素 x，都可在 B 中取出若干元素使其异或和为 x；
3. 对任意满足上两条的集合 B'，其元素个数不会小于 B 的元素个数。

我们可以利用异或线性基实现：

1. 判断一个数能否表示成某数集子集的异或和；
2. 求一个数表示成某数集子集异或和的方案数；
3. 求某数集子集的最大/最小/第 k 大/第 k 小异或和；
4. 求一个数在某数集子集异或和中的排名。

异或线性基

```
void insert(ull x) {
  for (int i = 63; ~i; --i) {
    if (!(x >> i))  // x 的第 i 位是 0
      continue;
    if (!p[i]) {
      p[i] = x;
      break;
    }
    x ^= p[i];
  }
}
```

异或线性基

```
5
633 211 169 841 1008
```

```
1001111001
0011010011
0010101001
1101001001
1111110000
```

```
1001111001
0100110000
0011010011
0001111010
0000000000
0000010000
0000000000
0000000000
0000000000
0000000000
```

[TJOI2008] 彩灯

- 把线性基弄出来，那么对于方案数，只需要看一下线性基中的元素个数就可以了。

[BJWC2011] 元素

- 给出 n 个物品，每个物品有标号 ai 和价值 bi，选出一些物品使得它们的标号不存在一个非空子集异或和为零，且价值之和最大。
- $1 \leqslant n \leqslant 1000, 1 \leqslant ai \leqslant 10^{18}$
- $1 \leqslant bi \leqslant 10000$。

[BJWC2011] 元素

- 只需要按价值从大到小考虑，每次能加入就加入即可。
- 可以用线性基来判断能否插入一个新的元素。时间复杂度 $O(n(\log V+\log n))$。

[SCOI2016] 幸运数字

- 求一条链的线性基，首先可以想到的是用树链剖分去做，这样的复杂度是3个log。
- 考虑用倍增去做，这样预处理是3个log，查询只需要两个log。
- 考虑用点分治去做，这样对于每个查询，只需要合并两个线性基就可以了。

[WC2011]最大XOR和路径

- 给一张无向图，求一条1-n的路径，是路径边权的异或和最小。

[WC2011]最大XOR和路径

- 首先我们可以随便找出一条从1到n的路径来，然后我们可以选一些环。
- 其实不管这个环和这条路径有怎样的关系，我们都是可以直接选的。
- 比如说选了一个和这个路径没有交的环，等价于从1走到了这个环然后走了一圈又走回到了1，一条边被异或两次相当于没走。
- 对于和路径有交的环，异或上它相当于把有交的部分异或两次，相当于走了这个环，也是合法的。
- 然后我们把所有环插入线性基中，预处理可以用dfs实现。

[SDOI2010] 外星千足虫

- 给定一个 n 个 01 变量，m 条方程的异或方程组，求最小的 k，使得前 k 条方程能够确定方程的唯一解。保证方程组有至少一组解。
- $1 \leqslant n \leqslant 1000, 1 \leqslant m \leqslant 2000$。
- 方程组有唯一解相当于 $rk(S) = n$，而一个矩阵的秩就是把所有元素插入线性基之后，线性基中元素的个数。于是从前往后把每个方程插入到线性基中就可以了。

数学常用符号

整除/同余理论常见符号

1. 整除符号：$x \mid y$，表示 x 整除 y，即 x 是 y 的因数。
2. 取模符号：$x \bmod y$，表示 x 除以 y 得到的余数。
3. 互质符号：$x \perp y$，表示 x，y 互质。
4. 最大公约数：$\gcd(x, y)$，在无混淆意义的时候可以写作 (x, y)。
5. 最小公倍数：$\operatorname{lcm}(x, y)$，在无混淆意义的时候可以写作 $[x, y]$。

数学常用符号

求和符号：$\sum$ 符号，表示满足特定条件的数的和。举几个例子：

- $\sum_{i=1}^{n} i$ 表示 $1+2+\cdots+n$ 的和。其中 i 是一个变量，在求和符号的意义下 i 通常是**正整数或者非负整数**（除非特殊说明）。这个式子的含义可以理解为，i 从 1 循环到 n，所有 i 的和。这个式子用代码的形式很容易表达。当然，学过简单的组合数学的同学都知道 $\sum_{i=1}^{n} i = \dfrac{n(n+1)}{2}$。
- $\sum_{S\subseteq T} |S|$ 表示所有被 T 包含的集合的大小的和。
- $\sum_{p\le n, p\perp n} 1$ 表示的是 n 以内有多少个与 n 互质的数，即 $\varphi(n)$，φ 是欧拉函数。

求积符号：$\prod$ 符号，表示满足特定条件的数的积。举几个例子：

- $\prod_{i=1}^{n} i$ 表示 n 的阶乘，即 $n!$。在组合数学常见符号中会讲到。
- $\prod_{i=1}^{n} a_i$ 表示 $a_1 \times a_2 \times a_3 \times \cdots \times a_n$。
- $\prod_{x|d} x$ 表示 d 的所有因数的乘积。

在行间公式中，求和符号与求积符号的上下条件会放到符号的上面和下面，这一点要注意。

快速幂

```cpp
long long binpow(long long a, long long b) {
  long long res = 1;
  while (b > 0) {
    if (b & 1) res = res * a;
    a = a * a;
    b >>= 1;
  }
  return res;
}
```

热身题

- 给出序列{Ai}
- 每次询问L到R
- $F(L,R) = \begin{cases} A_i\ , & L = R \\ F(\mathrm{L},R-1)\%A_R\ , & L < R \end{cases}$
- N<1e5，Q<1e5

热身题

- 观察到每一次有意义的取模至少会使结果减小一半。
- 倍增记录区间最小值。
- 每次O(logN)找到比当前值小（可以等于）的第一个数字。
- 每次查询找O(logN)次。

算数基本定理

- 算术基本定理可表述为：任何一个大于1的自然数 N,如果N不为质数，那么N可以唯一分解成有限个质数的乘积 $N=P_1^{a1}*P_2^{a2}*P_3^{a3}......P_n^{an}$，这里P1<P2<P3......<Pn均为质数，其中指数ai是正整数。这样的分解称为 N 的标准分解式。

约数

- 方法一：依次枚举每个数是否为约数即可。O(n)
- 方法二：发现若p是n的约数，则n/p也是n的约数(约数成对出现)，故只需知道小于等于根号n的全部约数配对即可。O(n^0.5)

质因数分解

- 从小到大枚举每个数a是否是当前n的约数，若是则将n分解出一个数a，a必然为素数。O(n)
- 若a>sqrt(n)显然无意义，故只算a<=sqrt(n)的，最终剩下的n也是一个素数。O(n^0.5)

约数

- 一个数的约数个数$\prod(\boldsymbol{qi}+\mathbf{1})$
- 一个数的约数和$\prod(\mathbf{1}+\boldsymbol{pi}+\boldsymbol{pi}^{2}\ldots\boldsymbol{pi}^{s})$
- 两数的gcd表示$\prod(\boldsymbol{pi}^{\boldsymbol{min}(\boldsymbol{q1i},\boldsymbol{q2i})})$
- 两数的lcm表示$\prod(\boldsymbol{pi}^{\boldsymbol{max}(\boldsymbol{q1i},\boldsymbol{q2i})})$

素数

- 若一个大于一的正整数P，其约数只有1和P本身，称其为素数(质数)。若其有超过两个约数，则为合数。
- 2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97 101 103 107 109 113 127 131 137 139 149 151 157 163 167 173 179 181 191 193 197 199 211

素数无限定理

- 正整数集中包含无限个素数。

- 反证：
- 假设素数有限，设其为p1~pn，构造s=1+π(pi)，若s是素数，矛盾；若s是合数，则p1~pn都不是s的约数，与算术基本定理矛盾。

埃筛

考虑这样一件事情：对于任意一个大于 1 的正整数 n，那么它的 x 倍就是合数（$x > 1$）。利用这个结论，我们可以避免很多次不必要的检测。

如果我们从小到大考虑每个数，然后同时把当前这个数的所有（比自己大的）倍数记为合数，那么运行结束的时候没有被标记的数就是素数了。

```
void Eratosthenes(int n) {
  is_prime[0] = is_prime[1] = false;
  for (int i = 2; i <= n; ++i) is_prime[i] = true;
  for (int i = 2; i <= n; ++i) {
    if (is_prime[i]) {
      prime.push_back(i);
      if ((long long)i * i > n) continue;
      for (int j = i * i; j <= n; j += i)
        // 因为从 2 到 i - 1 的倍数我们之前筛过了，这里直接从 i
        // 的倍数开始，提高了运行速度
        is_prime[j] = false;  // 是 i 的倍数的均不是素数
    }
  }
}
```

线性筛

埃氏筛法仍有优化空间，它会将一个合数重复多次标记。有没有什么办法省掉无意义的步骤呢？答案是肯定的。

如果能让每个合数都只被标记一次，那么时间复杂度就可以降到 $O(n)$ 了。

```cpp
void pre(int n) {
  for (int i = 2; i <= n; ++i) {
    if (!not_prime[i]) {
      pri.push_back(i);
    }
    for (int pri_j : pri) {
      if (i * pri_j > n) break;
      not_prime[i * pri_j] = true;
      if (i % pri_j == 0) {
        // i % pri_j == 0
        // 换言之，i 之前被 pri_j 筛过了
        // 由于 pri 里面质数是从小到大的，所以 i 乘上其他的质数的结果一定会被
        // pri_j 的倍数筛掉，就不需要在这里先筛一次，所以这里直接 break
        // 掉就好了
        break;
      }
    }
  }
}
```

整除与同余

- 对于整数a,b若a÷b=c……d则称a整除b得c，余数为d。又称a mod b=d;
- 若对于三个数a,b,p有a mod p=b mod p则称a,b关于p同余，$a \equiv b \pmod p$
- 存在整数k使得a=b+k*p

模运算

- (1)a≡a (mod d)
- (2)a≡b (mod d)→b≡a(mod d)
- (3)(a≡b (mod d),b≡c(mod d))→a≡c(mod d)
- 如果a≡n (mod d),b≡m(mod d),则
- (4)a+b≡n+m (mod d)
- (5)a-b≡n-m (mod d)
- (6)a*b≡n*m (mod d)
- (7)a≡b (mod d)则a-b整除d,a-b≡0 (mod d)

模运算

- 若两个数A,B其最大公约数为1，则称A,B互质。

- 此时k*A(0<=k<B)会遍历整个mod B剩余系。
- 若A,B最大公约数为g，则k*A只能遍历mod B剩余系中g的倍数部分。

模运算

- “剩余系”就是指对于某一个特定的正整数p，一个整数集中的数模p所得的余数域。
- 如果一个剩余系中包含了这个正整数所有可能的余数（一般地，对于任意正整数p，有p个余数：0,1,2,...,p-1），那么就被称为是模p的一个完全剩余系。
- 将对整数的所有运算限制在剩余系中进行(每次运算完后取mod)，各种性质都将得以保持。

更相减损术

- ①$\gcd(a,b)=\gcd(b,a-b)$
- ②$\gcd(2a,2b)=2\gcd(a,b)$
- 用途：高精GCD

欧几里得算法

- gcd(a,b)=gcd(b,a%b)

- int gcd(int a,int b){return b?gcd(b,a%b):a;}

裴蜀定理

- 裴蜀定理（Bézout's identity）得名于法国数学家艾蒂安·裴蜀，说明了对任何整数a、b和它们的最大公约数d，关于未知数x和y的线性丢番图方程（称为裴蜀等式）：若a,b是整数,且(a,b)=d，那么对于任意的整数x,y,ax+by都一定是d的倍数，特别地，一定存在整数x,y，使ax+by=d成立。
- 它的一个重要推论是：a,b互质的充要条件是存在整数x,y使ax+by=1.
- 它的一个重要推论是：设a1,a2,a3......an为n个整数，d是它们的最大公约数，那么存在整数x1......xn使得x1*a1+x2*a2+...xn*an=d。

裴蜀定理

- 有了裴蜀定理，我们就可以得到下面3个素数的基本性质
- 1)若a|bc且(a,b)=1,则a|c
- 2)若a|c,b|c且(a,b)=1,则ab|c
- 3)若(a,b)=1,则(a,bc)=(a,c)
- 证明：在$ax + by = 1$两边同时乘c

瓶子和燃料

jyy 一直想着尽快回地球，可惜他飞船的燃料不够了。有一天他又去向火星人要燃料，这次火星人答应了，要 jyy 用飞船上的瓶子来换。jyy 的飞船上共有 N 个瓶子（$1 \le N \le 1000$），经过协商，火星人只要其中的 K 个。

jyy 将 K 个瓶子交给火星人之后，火星人用它们装一些燃料给 jyy。所有的瓶子都没有刻度，只在瓶口标注了容量，第 i 个瓶子的容量为 V_i（V_i 为整数，并且满足 $1 \le V_i \le 10^9$）。火星人比较吝啬，他们并不会把所有的瓶子都装满燃料。他们拿到瓶子后，会跑到燃料库里鼓捣一通，弄出一小点燃料来交差。jyy 当然知道他们会来这一手，于是事先了解了火星人鼓捣的具体内容。

火星人在燃料库里只会做如下的 3 种操作：

1. 将某个瓶子装满燃料；
2. 将某个瓶子中的燃料全部倒回燃料库；
3. 将燃料从瓶子 a 倒向瓶子 b，直到瓶子 b 满或者瓶子 a 空。燃料倾倒过程中的损耗可以忽略。

火星人拿出的燃料，当然是这些操作能得到的最小正体积。jyy 知道，对于不同的瓶子组合，火星人可能会被迫给出不同体积的燃料。jyy 希望找到最优的瓶子组合，使得火星人给出尽量多的燃料。

```
3 2
3
4
4
```

```
4
```

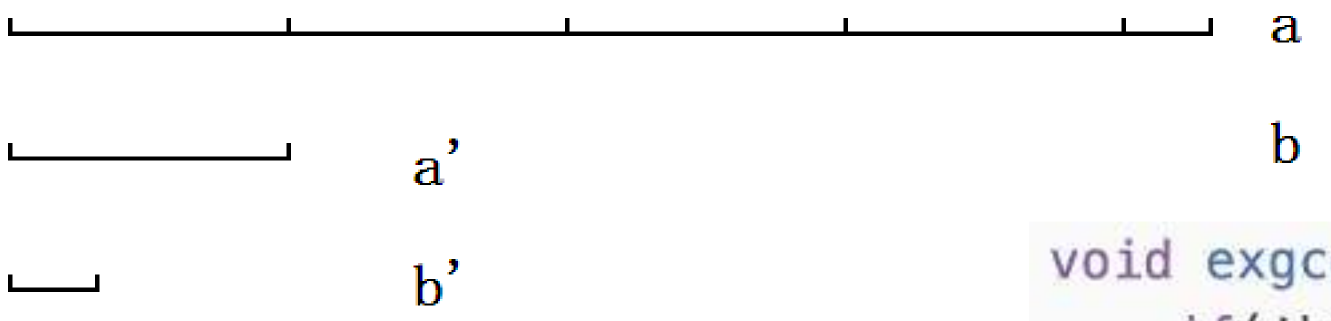
瓶子和燃料

- 相当于是你需要从n个数中选k个数使得GCD最大。
- 每个数的GCD就是它的约数。
- 筛出所有约数，然后找出最大的超过k次的约数。

[NOIP2012 提高组] 同余方程

求关于 x 的同余方程 $ax \equiv 1 \pmod{b}$ 的最小正整数解。

[NOIP2012 提高组] 同余方程



x*a’+y*b’=gcd(a, b)
p*a+q*b=gcd(a, b)
a=(a/b)*a’+1*b’
b=a’
Gauss→
p×(a/b)+q=x
p=y
y*a+(x+(a/b)×y)*b=gcd(a, b)

```
void exgcd(long long a,long long b){
    if(!b){
        x=1;y=0;
        r=a;return;
    }
    exgcd(b,a%b);
    long long k=x;
    x=y;
    y=k-(a/b)*y;
}
```

青蛙的约会

两只青蛙在网上相识了，它们聊得很开心，于是觉得很有必要见一面。它们很高兴地发现它们住在同一条纬度线上，于是它们约定各自朝西跳，直到碰面为止。可是它们出发之前忘记了一件很重要的事情，既没有问清楚对方的特征，也没有约定见面的具体位置。不过青蛙们都是很乐观的，它们觉得只要一直朝着某个方向跳下去，总能碰到对方的。但是除非这两只青蛙在同一时间跳到同一点上，不然是永远都不可能碰面的。为了帮助这两只乐观的青蛙，你被要求写一个程序来判断这两只青蛙是否能够碰面，会在什么时候碰面。

我们把这两只青蛙分别叫做青蛙 A 和青蛙 B，并且规定纬度线上东经 0 度处为原点，由东往西为正方向，单位长度 1 米，这样我们就得到了一条首尾相接的数轴。设青蛙 A 的出发点坐标是 x，青蛙 B 的出发点坐标是 y。青蛙 A 一次能跳 m 米，青蛙 B 一次能跳 n 米，两只青蛙跳一次所花费的时间相同。纬度线总长 L 米。现在要你求出它们跳了几次以后才会碰面。

对于 100% 的数据，$1 \le x \ne y \le 2 \times 10^9$，$1 \le m, n \le 2 \times 10^9$，$1 \le L \le 2.1 \times 10^9$。

费马小定理

- 假如p是质数，且Gcd(a,p)=1，那么 a^(p-1) ≡1 (mod p)
- 证明：构造集合S1={1,2,3…p-1}，S2={1a,2a,3a…(p-1)a}，易得S1在模意义下等于S2。
- 故(p-1)! ≡a^(p-1)*(p-1)! (mod p)，(p-1)!与p互素存在乘法逆元，故a^(p-1) ≡1 (mod p)。
- 若p是素数，由于a^(p-1) ≡1 (mod p)，故a的乘法逆元为a^(p-2)

逆元

因为 $ax \equiv 1 \pmod{b}$；

所以 $ax \equiv a^{b-1} \pmod{b}$（根据 费马小定理）；

所以 $x \equiv a^{b-2} \pmod{b}$。

然后我们就可以用快速幂来求了。

```
int qpow(long long a, int b) {
  int ans = 1;
  a = (a % p + p) % p;
  for (; b; b >>= 1) {
    if (b & 1) ans = (a * ans) % p;
    a = (a * a) % p;
  }
  return ans;
}
```

逆元

如果一个线性同余方程 $ax \equiv 1 \pmod{b}$，则 x 称为 $a \bmod b$ 的逆元，记作 a^{-1}。

```
void exgcd(int a, int b, int& x, int& y) {
  if (b == 0) {
    x = 1, y = 0;
    return;
  }
  exgcd(b, a % b, y, x);
  y -= a / b * x;
}
```

线性逆元

- O(n)计算1~n的逆元
- p%i+[p/i]*i=p
  p%i+[p/i]*i=0(mod p)
  [p/i]*i=- p%i (mod p)
- -[p/i]/[p%i]=i^-1
  i^-1=-[p/i]/[p%i]
- 逆元[i]=-[p/i]*逆元[p%i]

线性同余方程

> 「物不知数」问题：有物不知其数，三三数之剩二，五五数之剩三，七七数之剩二。问物几何？

即求满足以下条件的整数：除以 3 余 2，除以 5 余 3，除以 7 余 2。

该问题最早见于《孙子算经》中，并有该问题的具体解法。宋朝数学家秦九韶于 1247 年《数书九章》卷一、二《大衍类》对「物不知数」问题做出了完整系统的解答。上面具体问题的解答口诀由明朝数学家程大位在《算法统宗》中给出：

> 三人同行七十希，五树梅花廿一支，七子团圆正半月，除百零五便得知。

$2 \times 70 + 3 \times 21 + 2 \times 15 = 233 = 2 \times 105 + 23$，故答案为 23。

中国剩余定理 (Chinese Remainder Theorem, CRT) 可求解如下形式的一元线性同余方程组（其中 $n_1, n_2, \cdots, n_k$ 两两互质）：

$$\begin{cases} x & \equiv a_1 \pmod{n_1} \\ x & \equiv a_2 \pmod{n_2} \\ & \vdots \\ x & \equiv a_k \pmod{n_k} \end{cases}$$

上面的「物不知数」问题就是一元线性同余方程组的一个实例。

线性同余方程

1. 计算所有模数的积 n；
2. 对于第 i 个方程：
   a. 计算 $m_i = \frac{n}{n_i}$；
   b. 计算 m_i 在模 n_i 意义下的 逆元 m_i^{-1}；
   c. 计算 $c_i = m_i m_i^{-1}$（**不要对 n_i 取模**）。
3. 方程组在模 n 意义下的唯一解为：$x = \sum_{i=1}^{k} a_i c_i \pmod n$。

```
LL CRT(int k, LL* a, LL* r) {
  LL n = 1, ans = 0;
  for (int i = 1; i <= k; i++) n = n * r[i];
  for (int i = 1; i <= k; i++) {
    LL m = n / r[i], b, y;
    exgcd(m, r[i], b, y);  // b * m mod r[i] = 1
    ans = (ans + a[i] * m * b % n) % n;
  }
  return (ans % n + n) % n;
}
```

[TJOI2009] 猜数字

现有两组数字，每组 k 个。

第一组中的数字分别用 $a_1, a_2, \cdots, a_k$ 表示，第二组中的数字分别用 $b_1, b_2, \cdots, b_k$ 表示。

其中第二组中的数字是两两互素的。求最小的 $n \in \mathbb{N}$，满足对于 $\forall i \in [1, k]$，有 $b_i | (n - a_i)$。

对于 100% 的数据：

$1 \le k \le 10$，$|a_i| \le 10^9$，$1 \le b_i \le 6 \times 10^3$，$\prod_{i=1}^{k} b_i \le 10^{18}$。

积性函数

- φ(n) －欧拉函数，计算与n互质的正整数之数目
- μ(n) －莫比乌斯函数，关于非平方数的质因子数目
- gcd(n,k) －最大公因子，当k固定的情况
- d(n) －n的正因子数目
- σ(n) －n的所有正因子之和
- σk(n) － 因子函数，n的所有正因子的k次幂之和，当中k可为任何复数。
- 1(n) －不变的函数，定义为 1(n) = 1 （完全积性）
- Id(n) －单位函数，定义为 Id(n) = n（完全积性）
- Idk(n) －幂函数，对于任何复数、实数k，定义为Idk(n) = n^k （完全积性）
- ε(n) －定义为：若n = 1，ε(n)=1；若 n > 1，ε(n)=0。别称为“对于狄利克雷卷积的乘法单位”（完全积性）
- λ(n) －刘维尔函数，关于能整除n的质因子的数目
- γ(n)，定义为γ(n)=(-1)^ω(n)，在此加性函数ω(n)是不同能整除n的质数的数目
- 另外，所有狄利克雷特征均是完全积性的

欧拉函数

- 由欧拉函数的积性可得φ(n)=πφ(pi^qi)
- φ(x=p^q)=p^(q-1)*(p-1)=x*(1-1/p)(p为质数)
- 考虑1~p^q中有p约数的数有p^q/p个
- 故 $\varphi(x)=x\left(1-\frac{1}{p_1}\right)\left(1-\frac{1}{p_2}\right)\left(1-\frac{1}{p3}\right)\cdots\cdots\left(1-\frac{1}{p_n}\right)$
- 由φ的积性性质可由线性筛法O(n)计算φ(1)~φ(n)

欧拉函数

```
phi[1]=1;
for(int i=2;i<=n;++i)
  {
    if(!vis[i])prime[++prime[0]]=i,phi[i]=i-1;
    for(int j=1,k;j<=prime[0]&&(k=prime[j]*i)<=n;++j)
      {
        vis[k]=1;
        if(i%prime[j])phi[k]=phi[i]*phi[prime[j]];
        else
        {phi[k]=phi[i]*prime[j];break;}
      }
  }
```

谢谢，再见！

北斗学友

拔尖创新人才成长平台