线性代数

线性代数
- 线性基
- 高斯-约旦消元法
  - 求解线性方程组（luoguP3389 【模板】高斯消元法）
  - 矩阵求逆
  - 求行列式的值
- 经典问题
  - Q1：从N个数选2个，使得异或值最大。
  - Q2：N个点的带边权树，找一条XOR和最大的路径。
  - Q3：N个数选若干个使得XOR和为K。
  - Q4:在Q3的基础上，给M个限定：给定N的一个子集，要求恰好有奇/偶个数被选。
  - Q5：从N个数中选出任意个数，使它们的XOR和与K的XOR和最大。
  - Q6：带边权的无向图中，把点集分为两部分，使处于两集合之间的边XOR和最大(XOR最大割)。
  - Q7：带边权的无向图中，求一个回路使异或和最大(XOR最大环)
  - Q8: 带边权无向图中，求一个1号点到N号点的路径，使异或和最大。（XOR最长路）

线性基

支持插入线性基、若干个数的异或最大值、最小值、第k小异或值，判断某个数能否被异或得到。

```
//luoguP3812 【模板】线性基
#include<bits/stdc++.h>
#define reg register
using namespace std;
typedef long long ll;
const int MN=60;
ll a[61],tmp[61];
bool flag;
void ins(ll x){
    for(reg int i=MN;~i;i--)
        if(x&(1ll<<i))
            if(!a[i]){a[i]=x;return;}
            else x^=a[i];
    flag=true;
}
bool check(ll x){
    for(reg int i=MN;~i;i--)
        if(x&(1ll<<i))
            if(!a[i])return false;
            else x^=a[i];
    return true;
}
ll qmax(ll res=0){
    for(reg int i=MN;~i;i--)
        res=max(res,res^a[i]);
    return res;
}
ll qmin(){
    if(flag)return 0;
    for(reg int i=0;i<=MN;i++)
```

```
        if(a[i])return a[i];
}
ll query(ll k){  //求第k小的异或值
    reg ll res=0;reg int cnt=0;
    k-=flag;if(!k)return 0;
    for(reg int i=0;i<=MN;i++){
           for(int j=i-1;~j;j--)
            if(a[i]&(1ll<<j))a[i]^=a[j];
        if(a[i])tmp[cnt++]=a[i];
    }
    if(k>=(1ll<<cnt))return -1;
    for(reg int i=0;i<cnt;i++)
        if(k&(1ll<<i))res^=tmp[i];
    return res;
}
int main(){
    int n;ll x;scanf("%d",&n);
    for(int i=1;i<=n;i++)scanf("%lld",&x),ins(x);
    printf("%lld\n",qmax());
    return 0;
}
```

高斯-约旦消元法

高斯消元可以将任意矩阵在$O(n^3)$的时间转化为上三角矩阵，然后可计算得行列式为主对角线元素的乘积。应用：线性方程组求解、行列式计算、求矩阵的逆......

求解线性方程组（luoguP3389 【模板】高斯消元法）

```
#include<bits/stdc++.h>
#define int long long
using namespace std;
const int maxn=105;

int n; double a[maxn][maxn];

signed main(){
    ios::sync_with_stdio(0);
    cin.tie(0);cout.tie(0);
    cin>>n; //注意这是n+1列的模板
    for(int i=1;i<=n;i++) for(int j=1;j<=n+1;j++) cin>>a[i][j]; //读入矩阵
    for(int i=1;i<=n;i++){ //枚举列
        int mx=i;
        for(int j=i+1;j<=n;j++){ //选出该列最大主元（系数）
            if(fabs(a[j][i])>fabs(a[mx][i])){
                mx=j; //记录该行
            }
        }
        for(int j=1;j<=n+1;j++) swap(a[i][j],a[mx][j]); //交换i和mx所在的两行
        if(!a[i][i]){ //不满秩（对角线无元素）
            puts("No Solution");
            return 0;
        }
        for(int j=1;j<=n;j++){ //对于i以外的每一行
            if(j!=i){
```

```
                double tmp=a[j][i]/a[i][i];
                //for(int k=1;k<=n+1;++k){ //实际上不用从第一项开始
                for(int k=i+1;k<=n+1;k++){ //对每一项做加减消元
                    a[j][k]-=a[i][k]*tmp;
                }
            }
        }
    }
    for(int i=1;i<=n;i++) cout<<fixed<<setprecision(2)<<a[i][n+1]/a[i][i]<<"\n";
    return 0;
}
```

矩阵求逆

构造在$n\times 2n$的矩阵(A,I_n)，利用高斯消元化简为最简行(I_n,A^{-1})即可得到逆矩阵。

```
#include<iostream>
using namespace std;
typedef long long ll;
const int N=405,mod=1e9+7;
int n;
ll a[N][N<<1];

ll fpow(ll x,ll k){
    ll ans=1;
    while(k){if(k&1)ans=ans*x%mod; x=x*x%mod; k>>=1;}
    return ans%mod;
}

void Gauss_j(){
    for(int i=1,r;i<=n;++i){
        r=i;
        for(int j=i+1;j<=n;++j)
            if(a[j][i]>a[r][i]) r=j;
        if(r!=i) swap(a[i],a[r]);
        if(!a[i][i]){puts("No Solution");return;}
        int kk=fpow(a[i][i],mod-2); //求逆元
        for(int k=1;k<=n;++k){
            if(k==i) continue;
            int p=a[k][i]*kk%mod;
            for(int j=i;j<=(n<<1);++j)
                a[k][j]=((a[k][j]-p*a[i][j])%mod+mod)%mod;
        }

        for(int j=1;j<=(n<<1);++j) a[i][j]=(a[i][j]*kk%mod);
    }
}

signed main(){
    scanf("%d",&n);
    for(int i=1;i<=n;++i){
        a[i][i+n]=1;
        for(int j=1;j<=n;++j)
            scanf("%lld",&a[i][j]);
    }
```

```
    Gauss_j();
    for(int i=1;i<=n;++i){
        for(int j=n+1;j<(n<<1);++j) printf("%lld ",a[i][j]);
        printf("%lld\n",a[i][n<<1]);
    }
    return 0;
}
```

求行列式的值

直接化为最简型算对角成绩就可以了。这里贴上辗转相除高斯消元的板子。

```
int Gauss(){   //辗转相除的高斯消元，可用于模意义下
    int fg=1;
    for(int i=1;i<=cnt;i++){
        for(int j=i+1;j<=cnt;j++){
            int A=f[i][i],B=f[j][i];
            while(B){
                int t=A/B;A%=B;swap(A,B);
                for(int k=i;k<=n;k++)
                    f[i][k]=(f[i][k]-t*f[j][k]%p+p)%p;
                swap(f[i],f[j]);
                fg*=-1;
            }
        }
        if(!f[i][i]) return 0;
    }
    int Ans=1;
    for(int i=1;i<=cnt;i++) Ans=(Ans*f[i][i])%p;
    return (Ans*fg%p+p)%p;
}
```

经典问题

Q1：从N个数选2个，使得异或值最大。

A1：建Trie树。

Q2：N个点的带边权树，找一条XOR和最大的路径。

A2：任选根，从根出发计算每个点的异或前缀和，对于两个点的路径异或值=$SX \oplus SY$，转化为Q1。

Q3：N个数选若干个使得XOR和为K。

A3：线性基/高斯消元解XOR方程组。高斯消元对于K的第p位，有方程$X_{1p} + X_{2p} + ... + X_{sp} = K_p$，联立60个方程，方程的解就等于原问题的解。

另外，异或方程组的增广矩阵是01矩阵，我们可以用Bitset来优化，复杂度为$O(\frac{n^2m}{w})$

Q4:在Q3的基础上，给M个限定：给定N的一个子集，要求恰好有奇/偶个数被选。

A4：每个限制添加一个方程，再用高斯消元解XOR方程组。

Q5：从N个数中选出任意个数，使它们的XOR和与K的XOR和最大。

A5：直接线性基qmax的时候初始化res=k。

Q6：带边权的无向图中，把点集分为两部分，使处于两集合之间的边XOR和最大(XOR最大割)。

A6：设h_i为i的邻边异或和，一个割$S,T=\sum h_i(i \in S)=\sum h_i(i \in T)$，转化为：从N个数中选任意个数，使得XOR和最大。

Q7：带边权的无向图中，求一个回路使异或和最大(XOR最大环)

A7：两个回路的异或和仍是回路，且一个无向连通图G中有且仅有M-N+1个独立回路，因此可以转化为若干个数的异或最大值问题。

Q8: 带边权无向图中，求一个1号点到N号点的路径，使异或和最大。（XOR最长路）

A8：任取一条1-N的路，找一个环与其XOR和最大即可。（转化为Q7）