弦图（内含证明，理性愉悦）

弦图是一种特殊的图，很多在一般图上的 NPC 问题在弦图上都有优秀的线性时间复杂度算法。

一些定义与性质

子图： 点集和边集均为原图点集和边集子集的图。

导出子图（诱导子图）： 点集为原图点集子集，边集为所有满足 **两个端点均在选定点集中** 的图。

团： 完全子图。

极大团： 不是其他团子图的图。

最大团： 点数最大的团。

团数： 最大团的点数，记为 $\omega(G)$ 。

最小染色： 用最少的颜色给点染色使得所有边连接的两点颜色不同。

色数： 最小染色的颜色数，记为 $\chi(G)$ 。

最大独立集： 最大的点集使得点集中任意两点都没有边直接相连。该集合的大小记为 $\alpha(G)$ 。

最小团覆盖： 用最少的团覆盖所有的点。使用团的数量记为 $\kappa(G)$ 。

弦： 连接环中不相邻两点的边。

弦图： 任意长度大于 3 的环都有一个弦的图称为弦图。

Lemma 1： 团数 $\omega(G) \leq \chi(G)$ 色数

证明：考虑单独对最大团的导出子图进行染色，至少需要 $\omega(G)$ 种颜色。

Lemma 2： 最大独立集数 $\alpha(G) \leq \kappa(G)$ 最小团覆盖数

证明：每个团中至多选择一个点。

Lemma 3： 弦图的任意导出子图一定是弦图。

证明：如果弦图有导出子图不是弦图，说明在这个导出子图上存在大于 3 的无弦环，则无论原图如何（怎么加边）都不会使得原图是弦图，矛盾。

Lemma 4： 弦图的任意导出子图一定不可能是一个点数大于 3 的环。

证明：一个点数大于 3 的环不是弦图，用以上定理即可。

弦图的判定

问题描述

给定一个无向图，判断其是否为弦图。

点割集

对于图 G 上的两点 u, v ，定义这两点间的 **点割集** 为满足删除这一集合后， u, v 两点之间不连通。如果关于 u, v 两点间的一个点割集的任意子集都不是点割集，则称这个点割集为 **极小点割集** 。

Lemma 5： 图关于 u, v 的极小点割集将原图分成了若干个连通块，设包含 u 的连通块为 V_1 ，包含 v 的连通块为 V_2 。设 $N(x)$ 表示与点 x 相邻的点集，则对于极小点割集上的任意一点 a ， $N(a)$ 一定包含 V_1 和 V_2 中的点。

证明：若 $N(a)$ 只包含 V_1 或 V_2 中的至多一个连通块中的点，从点割集中删去 a 点，仍不连通，则原点割集不是最小点割集。

Lemma 6： 弦图上任意两点间的极小点割集的导出子图一定为一个团。

证明：极小点割集大小 ≤ 1 时，导出子图一定为一个团。

否则，设极小点割集上有两点为 x, y ，由 **Lemma 5** 得， $N(x)$ 中有 V_1, V_2 中的点，设为 x_1, x_2 ，同样的，设 y_1, y_2 ，注意，可能有 $x_1 = y_1, x_2 = y_2$ 。

由于 V_1, V_2 均为连通块，则在 x_1, y_1 和 x_2, y_2 两个点对之间存在最短路径。设 x, y 在 V_1, V_2 内部的最短路为 $x - x_1 \sim y_1 - y, x - x_2 \sim y_2 - y$ ，则图上存在一个环 $x - x_1 \sim y_1 - y - y_2 \sim x_2 - x$ ，该环的大小一定 ≥ 4 ，根据弦图的定义，此时该环上一定存在一条弦。

若这条弦连接了 V_1, V_2 两个连通块，则点集不是点割集。若这条弦连接了单个连通块内部的两个点或一个连通块内部的一个点和一个点割集上的点，都不满足最短路的性质。所以这条弦只能连接 x, y 两点。

由此，可证弦图中每个极小点割集中的两点都有边直接相连，故性质得证。

单纯点

定义 $N(x)$ 为满足与 x 直接有边相连且在完美消除序列上的 x 之后的点集。若点集 $\{x\} + N(x)$ 的导出子图为一个团，则称点 x 为单纯点。

Lemma 7： 任何一个弦图都至少有一个单纯点，不是完全图的弦图至少有两个不相邻的单纯点。

证明：数学归纳法。单独考虑每一连通块。

归纳基底：当图与完全图同构时，图上任意一点都是单纯点。当图的点数 ≤ 3 时，引理成立。

若图上的点数 ≥ 4 且图不为完全图，可知必然存在 u, v 使得 $(u, v) \notin E$ 。设 I 是图关于 u, v 的极小点割集。设 A, B 分别是删去 I 后的导出子图上 u, v 所在的连通块。由于问题的对称性，我们只考虑 A 一侧的情况，设 $L = A + I$ 。若 L 为完全图，则 u 为单纯点；若不是，因为 L 是原图的导出子图，一定也是弦图，所以有两个不相邻的单纯点，因为 I 是一个团，其上两点都相邻，所以 A 中一定有一个单纯点。该单纯点扩展到全图也为单纯点。

由于每次将整个图分成若干个连通块证明，大小一定减小，且都满足性质，故归纳成立。

完美消除序列

令 $n=|V|$，完美消除序列 $v_1,v_2,\ldots,v_n$ 为 $1,2,\ldots,n$ 的一个排列，满足 v_i 在 $\{v_i,v_{i+1},\ldots,v_n\}$ 的导出子图中为单纯点。

Lemma 8： 一个无向图是弦图当且仅当其有一个完全消除序列。

充分性：点数为 1 的弦图有完全消除序列。由 **Lemma 3** 和 **Lemma 7**，点数为 n 的弦图的完美消除序列可以由点数为 $n-1$ 的弦图的完美消除序列加上一个单纯点得到。

必要性：假设有无向图存在结点数 >3 的环且拥有完美消除序列，设在完美消除序列中出现的第一个环上的点为 v，设 v 在环上与 v_1,v_2 相连，则有完美消除序列的性质即单纯点的定义可得 v_1,v_2 直接有边相连，矛盾。

朴素算法

每次找到一个 **单纯点** v，加入到完美消除序列中。

将点 v 与其相邻的边从图上删除。

重复以上过程，若所有点都被删除，则原图是弦图且求得了一个完美消除序列；若图上不存在单纯点，则原图不是弦图。

时间复杂度 $O(n^4)$。

MCS 算法

最大势算法（Maximum Cardinality Search）是一种可以在 $O(n+m)$ 的时间复杂度内求出无向图的完美消除序列的方法。

逆序给结点编号，即按从 n 到 1 的顺序给点标号。

设 $label_x$ 表示第 x 个点与多少个已经标号的点相邻，每次选择 $label$ 值最大的未标号结点进行标号。

用链表维护对于每个 i，满足 $label_x=i$ 的 x。

由于每条边对 $\sum_{i=1}^{n} label_i$ 的贡献最多是 2，时间复杂度 $O(n+m)$。

正确性证明：

设编号为 x 的结点在最大势算法得出的序列上出现的位置为 a_x。我们要证明当原图为弦图时，序列的逆序为原图的完美消除序列。

这个证明来自我的学习导师 @MCPlayer542，感谢他让我自闭。

当一个点 u 被标记时，若 $label_u$ 为 0 或 1，则 u 必然为单纯点。

若 $label_u>1$， 必然可以选出两点 v,w 与 u 直接相连，且 v,w 在 u 之前标记过，如果最大势算法成立，则 v,u 必然直接相连。

考虑使用反证法。假设存在这样的 v,w，设 v 在 w 之前被标记，且 v,w 之间 **不直接有边相连**。

为了更方便地分析问题，我们将原图划分为几个点集。与 u,v 均直接相连的点集 S； 只与 v 相连，不与 u 相连的点集 A； 只与 w 相连，不与 u 相连的点集 B。

Lemma 9： 删去 u,S 后， A,v 不连通；删去 v,S 后， B,u 不连通。

证明：若删去后仍连通，考虑 A,v 间的最短路径 $A\sim v$，存在一个大于三元的环 $w-u-A\sim v-w$，必然存在一条弦，若弦为 A,w 不满足点集 A 的定义，若弦为 u,v 不满足题设，若弦在 $A\sim v$ 中不满足最短路，若弦是 w 与 A v 中不为 A,v 的一点，则该点应在 S 中。

对称同理。

Lemma 10： 删去 w 后， A, B 中没有任意两点连通， 与 u, v 均直接相连的所有点必然在集合 S 中。

证明：若删去后仍连通，设选择的两点之间的最短路为 $A \sim B$，则存在一个大于三元的环 $w-u-A \sim B-v-w$，类似以上讨论可得。若有一点 x 与 u, v 均直接相连，则存在一个四元环 $w-u-x-v$，弦必然为 w, x，故 x 必在 S 中。

以上两个定理保证了将与 v, w 中至少一个点直接相连的所有点划分为的点集是相互独立且覆盖了全部情况的。

Theorem 1： 在对弦图使用最大势算法时，当一个点被标记时，它在已经标记结点中一定为一个单纯点。

证明：当我们标记了 v 时，设 A 中被标记的点数为 x， S 中被标记的点数为 y，则 $label_u = y+1, label_w = y$。

若接下来标记的是 S 中的点，则不会改变 $label_u$ 和 $label_w$ 的大小关系。若接下来标记的是 A 中的点，则这些点对 u, w 的 $label$ 无影响。

故此时 u 必然在 w 前标记，与题设矛盾。故所有 v, w 之间必然直接有边相连，当 u 被标记时一定是单纯点。

至此，最大势算法的正确性得证。

参考代码：

```
    while(cur)
    {
        p[cur]=h[nww];rnk[p[cur]]=cur;
        h[nww]=nxt[h[nww]];lst[h[nww]]=0;
        lst[p[cur]]=nxt[p[cur]]=0;tf[p[cur]]=true;
        for(auto it=G[p[cur]].begin();it!=G[p[cur]].end();it++)
            if(!tf[*it]){
                if(h[deg[*it]]==*it)h[deg[*it]]=nxt[*it];
                nxt[lst[*it]]=nxt[*it];lst[nxt[*it]]=lst[*it];
                lst[*it]=nxt[*it]=0;deg[*it]++;
                nxt[*it]=h[deg[*it]];lst[h[deg[*it]]]=*it;h[deg[*it]]=*it;
            }
        cur--;
        if(h[nww+1])nww++;
        while(nww&&!h[nww])nww--;
    }
```

如果此时原图是弦图，此时求出的就是完美消除序列；但是由于原图可能不是弦图，此时求出的一定不是完美消除序列，所以问题转化为 **判断求出的序列是否是原图的完美消除序列** 。

判断一个序列是否是完美消除序列

朴素算法

根据定义，依次判断完美消除序列 v 上 $\{v_i, v_{i+1}, \ldots, v_n\}$ 中与 v_i 相邻的点是否构成了一个团。时间复杂度 $O(nm)$ 。

优化后的算法

根据完美消除序列的定义，设 v_i 在 $v_i, v_{i+1}, \ldots, v_n$ 中相邻的点从小到大为 $\{v_{c_1}, v_{c_2}, \ldots, v_{c_k}\}$，则只需判断 v_{c_1} 与其他点是否直接连通即可。时间复杂度 $O(n+m)$ 。

```
    jud=true;
    for(int i=1;i<=n;i++){
        cur=0;
        for(auto it=G[p[i]].begin();it!=G[p[i]].end();it++)
            if(rnk[p[i]]<rnk[*it]){
                s[++cur]=*it;
                if(rnk[s[cur]]<rnk[s[1]])swap(s[1],s[cur]);
            }
        for(int j=2;j<=cur;j++)
            if(!st[s[1]].count(s[j])){jud=false;break;}
    }
    if(!jud) printf("Imperfect\n");
    else printf("Perfect\n");
```

至此，**弦图判定问题** 可以在 $O(n+m)$ 的时间复杂度内解决。

弦图的极大团

定义 $N(x)$ 为满足与 x 直接有边相连且在完美消除序列上的 x 之后的点集。则弦图的极大团一定为 $\{x\}+N(x)$ 。

证明：考虑弦图的一个极大团 V ，其中的点在完美消除序列中出现的第一个点 x ，一定有 $V\subseteq\{x\}+N(x)$ ，又因为 V 是极大团，所以 $V=\{x\}+N(x)$ 。

弦图最多有 n 个极大团。求出弦图的每个极大团，可以判断每个 $\{x\}+N(x)$ 是否为极大团。

设 $A=\{x\}+N(x),B=\{y\}+N(y)$ ，若 $A\subsetneqq B$ ，则 A 不是极大团。此时在完美消除序列上显然有 y 在 x 前。

设 nxt_x 表示 $N(x)$ 中在完美消除序列上最靠前的点， $y*$ 表示所有满足 $A\subseteq B$ 的 y 中的最靠后的点。此时必然有 $nxt_{y*}=x$ ，否则 $y*$ 不是最靠后的，令 $y*=nxt_{y*}$ 仍然满足条件。

$A\subsetneqq B$ 当且仅当 $|A|+1\leq|B|$ 。

问题转化为判断是否存在 y ，满足 $nxt_y=x$ 且 $|N(x)|+1\leq|N(y)|$ 。时间复杂度 $O(n+m)$ 。

```
    for(int i=1;i<=n;i++){
        cur=0;
        for(auto it=G[p[i]].begin();it!=G[p[i]].end();it++)
            if(rnk[p[i]]<rnk[*it]){
                s[++cur]=*it;
                if(rnk[s[cur]]<rnk[s[1]])swap(s[1],s[cur]);
            }
        fst[p[i]]=s[1];N[p[i]]=cur;
    }
    for(int i=1;i<=n;i++){
        if(!vis[p[i]])ans++;
        if(N[p[i]]>=N[fst[p[i]]]+1)vis[fst[p[i]]]=true;
    }
```

弦图的色数/弦图的团数

一种构造方法：按完美消除序列从后往前依次给每个点染色，给每个点染上可以染的最小颜色。时间复杂度 $O(m+n)$ 。

正确性证明：设以上方法使用了 t 种颜色，则 $t \geq \chi(G)$ 。由于团上每个点都是不同的颜色，所以 $t = \omega(G)$ ，由 **Lemma 1** ， $t = \omega(G) \leq \chi(G)$ 。综上，可得 $t = \chi(G) = \omega(G)$ 。

无需染色方案，只需求出弦图的色数/团数时，可以取 $|\{x\} + N(x)|$ 的最大值得到。

```
    for(int i=1;i<=n;i++)ans=max(ans,deg[i]+1);
```

弦图的最大独立集/最小团覆盖

最大独立集：完美消除序列从前往后，选择所有没有与已经选择的点有直接连边的点。

最小团覆盖：设最大独立集为 $\{v_1, v_2, \ldots, v_t\}$ ，则团的集合 $\{\{v_1 + N(v_1)\}, \{v_2 + N(v_2)\}, \ldots, \{v_t + N(v_t)\}\}$ 为图的最小团覆盖。时间复杂度均为 $O(n + m)$ 。

正确性证明：设以上方案独立集数和团覆盖数为 t ，由定义得 $t \leq \alpha(G), t \geq \kappa(G)$ ，由 **Lemma 2** 得， $\alpha(G) \leq \kappa(G)$ ，所以 $t = \alpha(G) = \kappa(G)$ 。

```
    for(int i=1;i<=n;i++)
        if(!vis[p[i]]){
            ans++;
            for(auto it=G[p[i]].begin();it!=G[p[i]].end();it++)
            vis[*it]=true;
        }
```