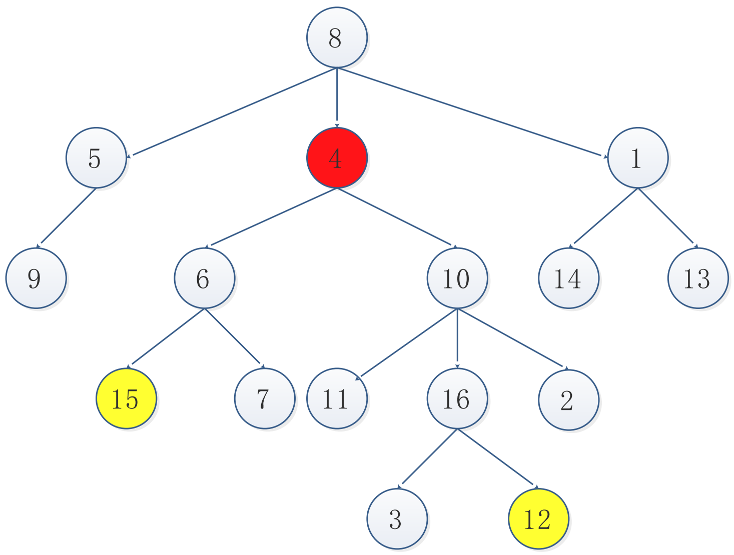
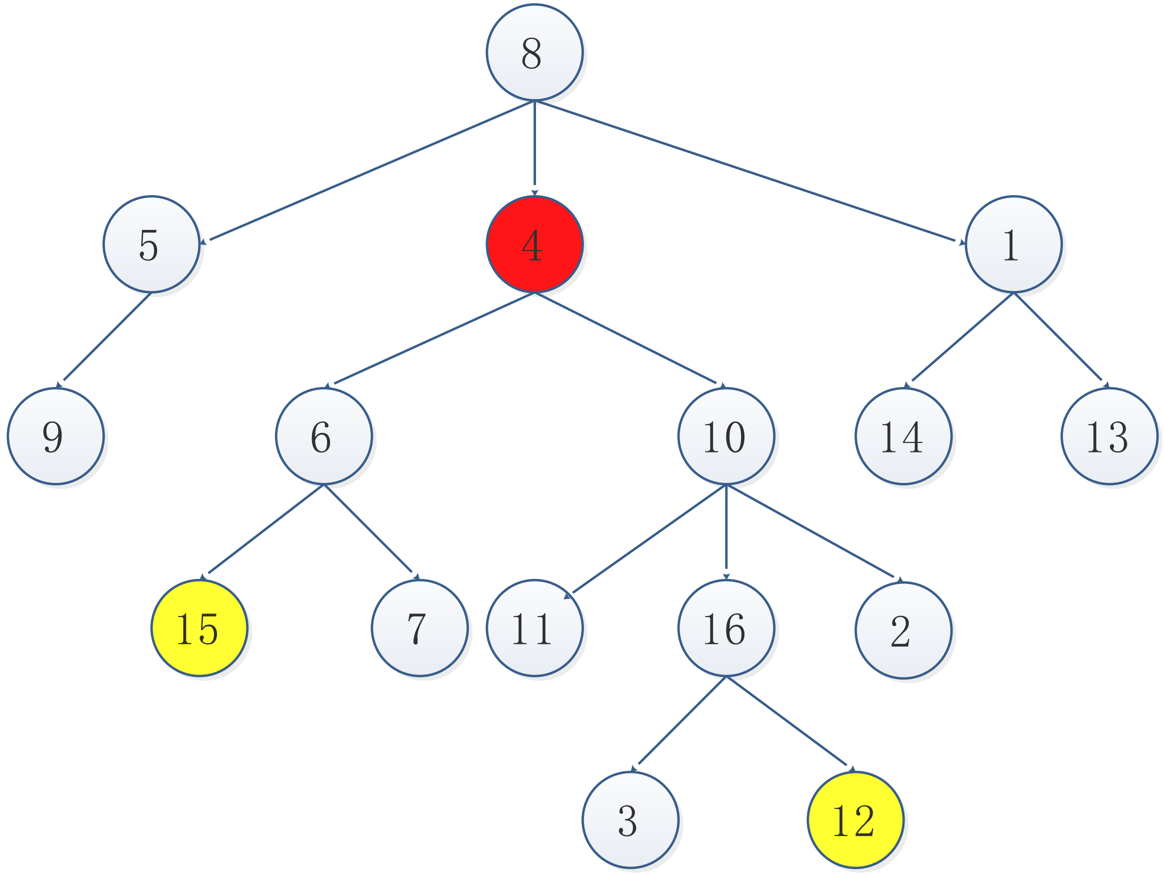
3. LCA （Least Common Ancestors）

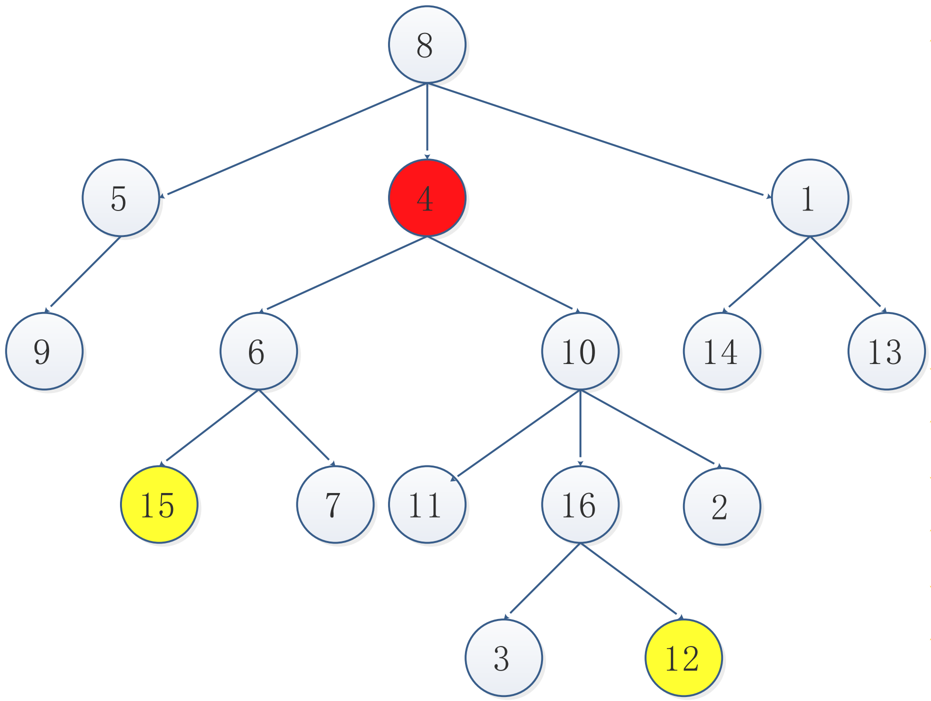
最近公共祖先



◆**洛谷 P1967**货车运输(NOIP2013)

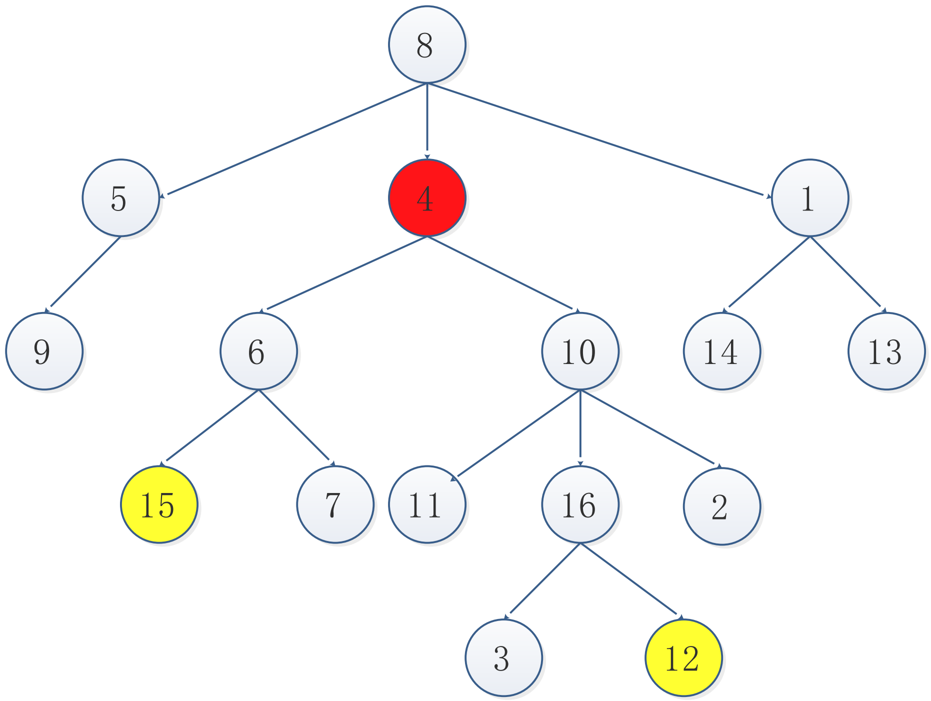
考虑树上U和V两点间的路线问题：唯一的NOIP考点





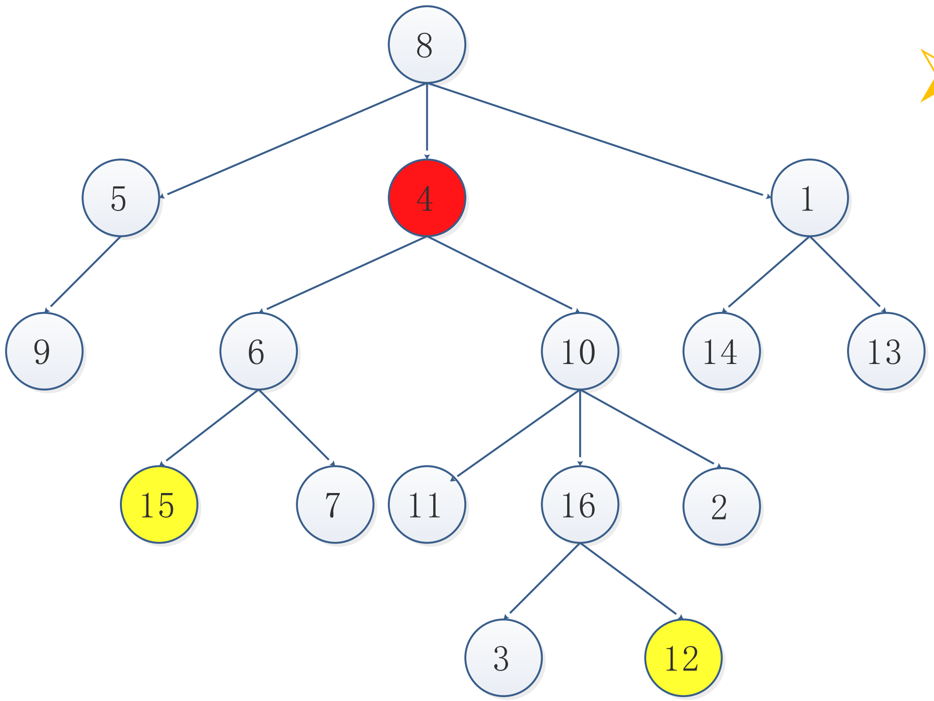
- LCA（Least Common Ancestors），即最近公共祖先，是指在有根树中，找出某两个结点x和y最近的公共祖先（深度最大的祖先）
- 记为LCA(x,y)
- LCA(15,12)=4
- LCA(10,12)=10

作用：
能在log(n)解决从u到v的路线问题



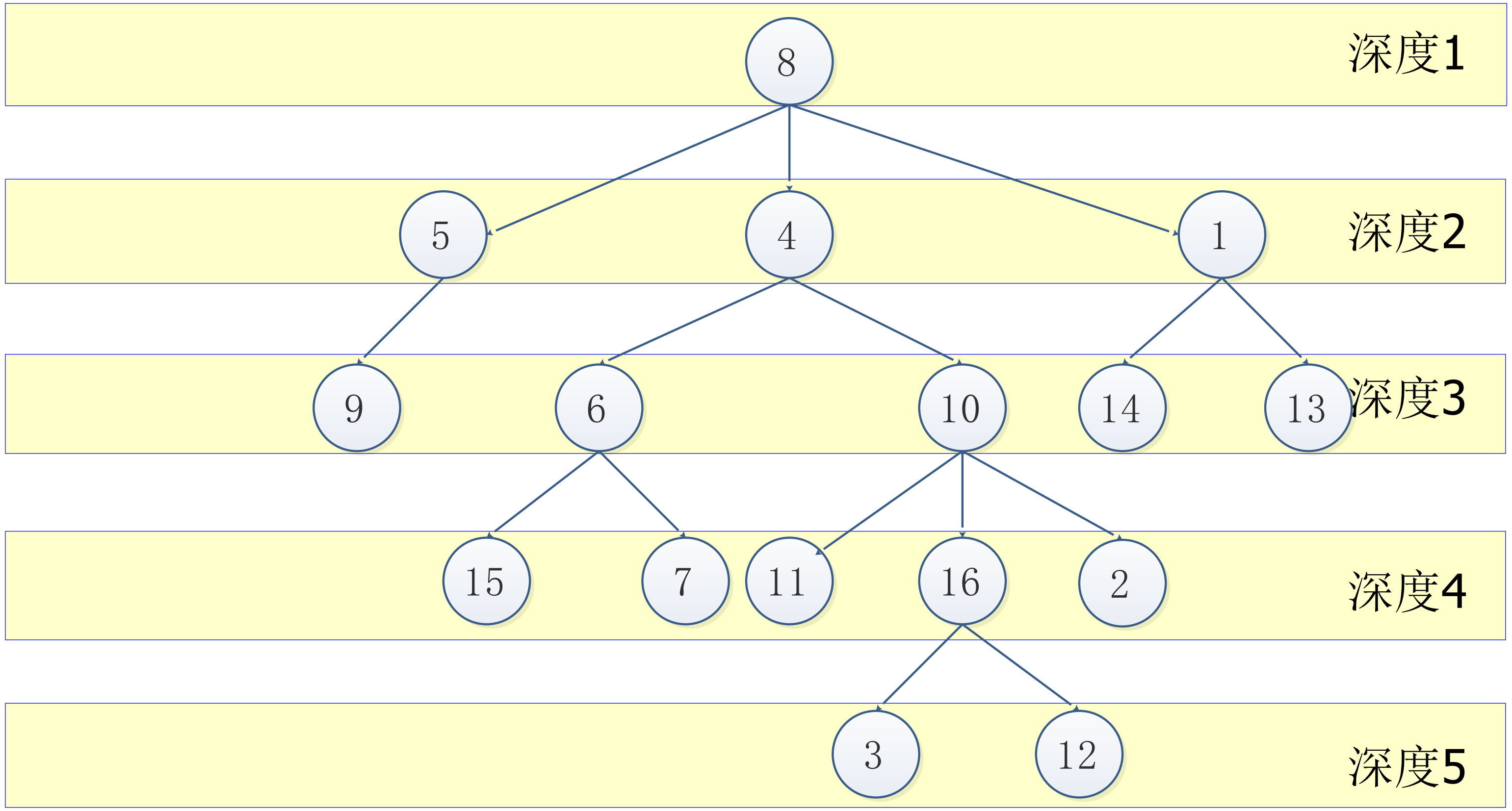
◆方法1：向上标记法(暴力)

- 从x向上走到根节点，并标记所有走过的结点。
- 从y走到根，当第一次遇到有标记的结点时，就找到了LCA(x,y)
- 最坏时间O(n)

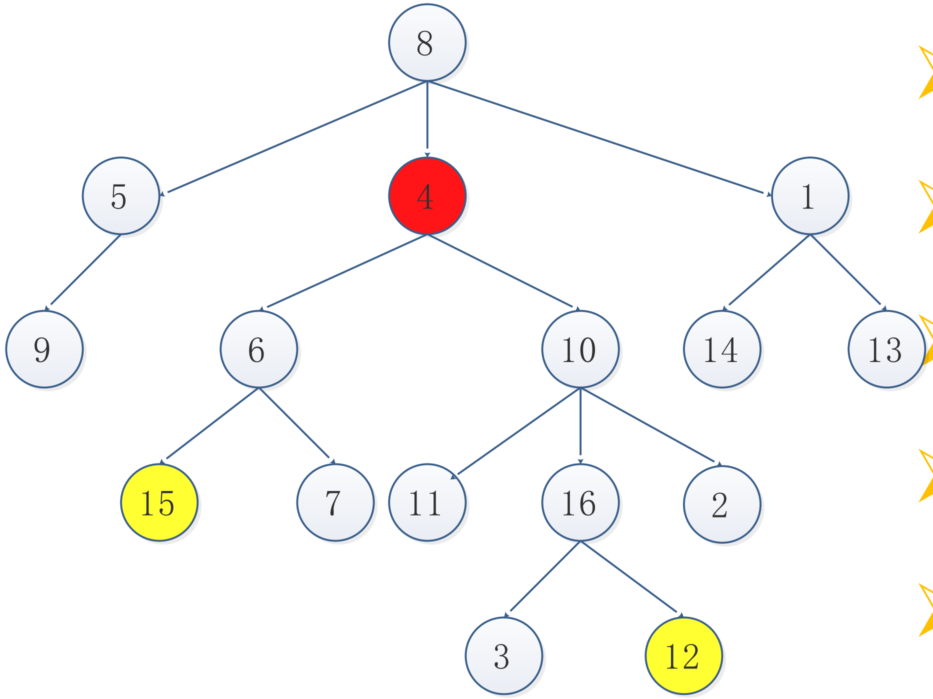


方法2：暴力求解2

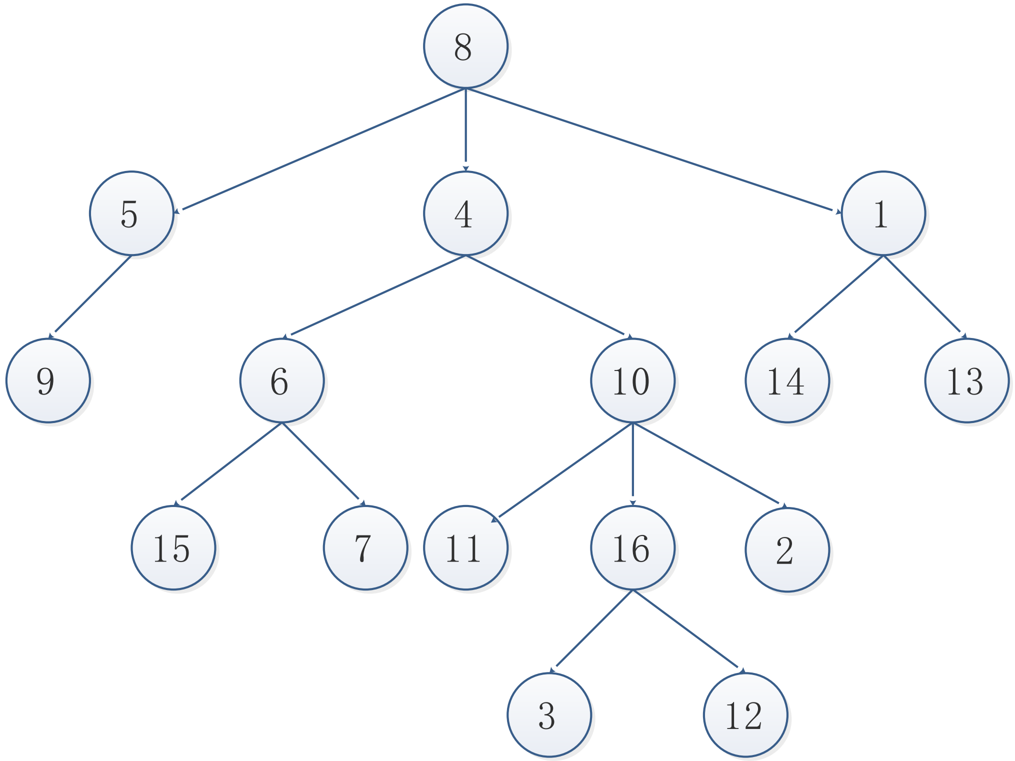
- 用fa[i]记录i的父亲结点
  - 首先将x和y中深度较深的那个点跳到和较浅的点同样的深度。
  - 然后两个点一起一步一步向上跳，直到跳到同一个点p，就是它们的LCA
  - 复杂度：最坏情况$O(n)$
  - 适合只计算一次LCA(x,y)



8
深度1
5
4
1
深度2
9
6
10
14
13
深度3
15
7
11
16
2
深度4
3
12
深度5

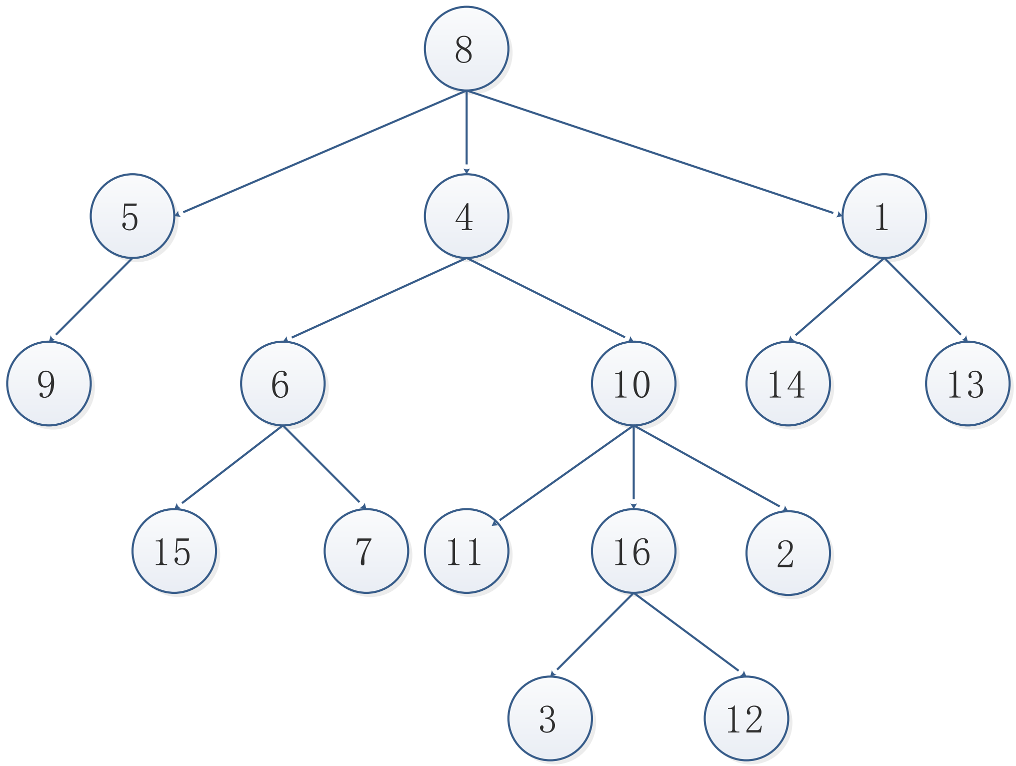


- p=LCA(x,y)
- 结点深度：d [x],d [y]
- x向上走d[x]-d[p]
- y向上走d[y]-d[p]
- 实现方法：
- u和v深度大的向上走|d[x]-d[y]|
- 再一起一步一步向上走，直到走到同一个结点p。
- 时间：O(d[x]+d[y])



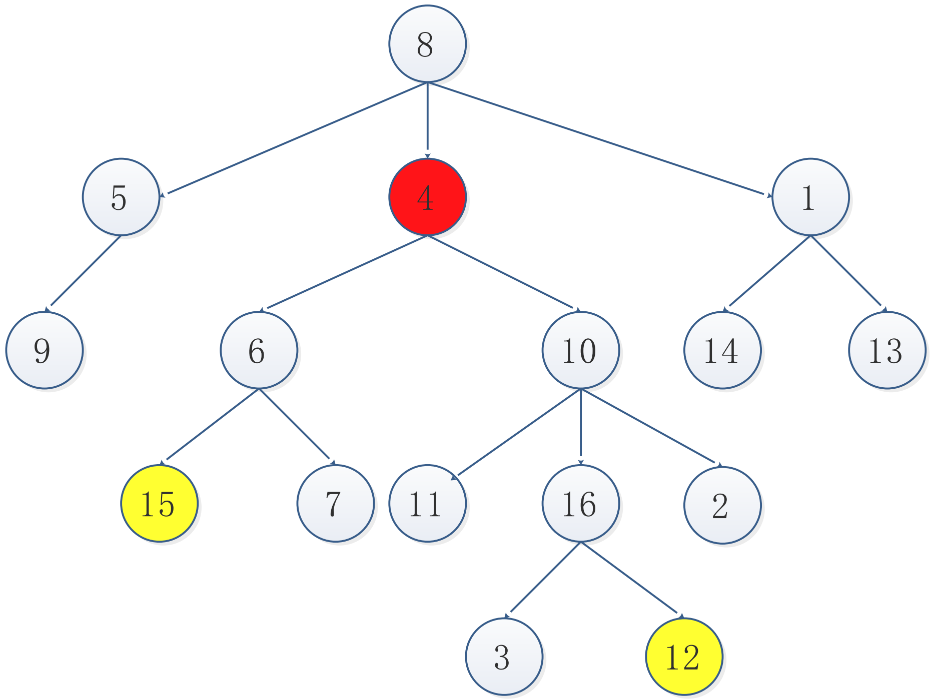
- 存储：有向图的邻接表
- vector<int> G[maxn];

- int fa[maxn];
- int d[maxn];



- 1.深度编号，同时求fa[]:
- dfs(root,1);

```
void dfs(int u,int depth){
        d[u]=depth;
        int m=G[u].size();
        for(int i=0;i<m;i++){
                int v=G[u][i];
                dfs(v,depth+1);
        }
}
```



一步一步的走

- 2.求LCA(x,y)

```
int LCA(int x,int y){
    while(d[x]>d[y]) x=fa[x];
    while(d[x]<d[y]) y=fa[y];
    while(x!=y){
        x=fa[x];
        y=fa[y];
    }
    return x;
}
```

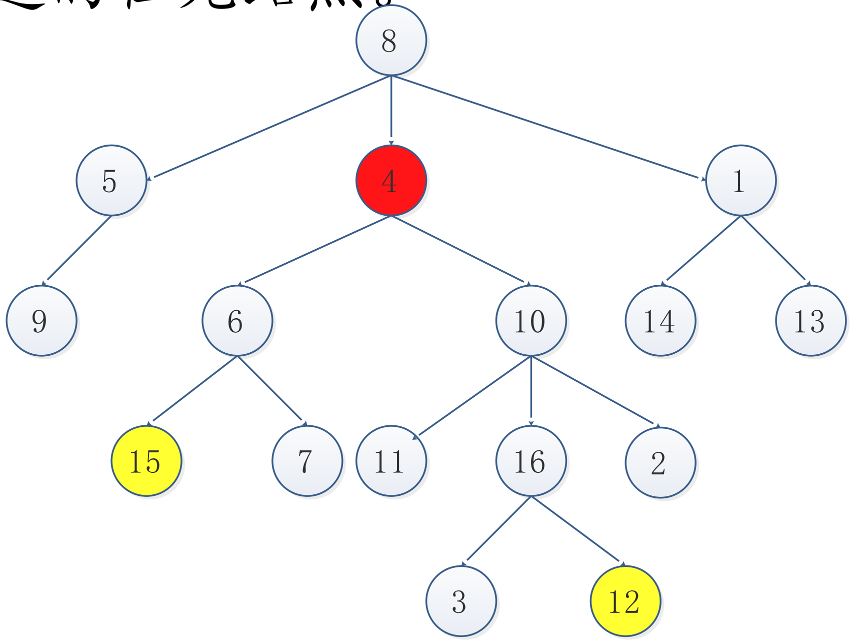
方法2的改进：二进制拆分思想（倍增）

每次向上跳2^j步

- 令fa[i][j]表示i的2^j辈，即i向根节点走2^j步到达的祖先结点。
- $fa[i][0]$：向上走1步到达的点（父亲）；
- $fa[i][1]$：向上走2步到达的点；
- $fa[i][2]$：向上走4步到达的点；
- $fa[i][3]$：向上走8步到达的点；

$$= fa[fa[i][2]][2]$$

…

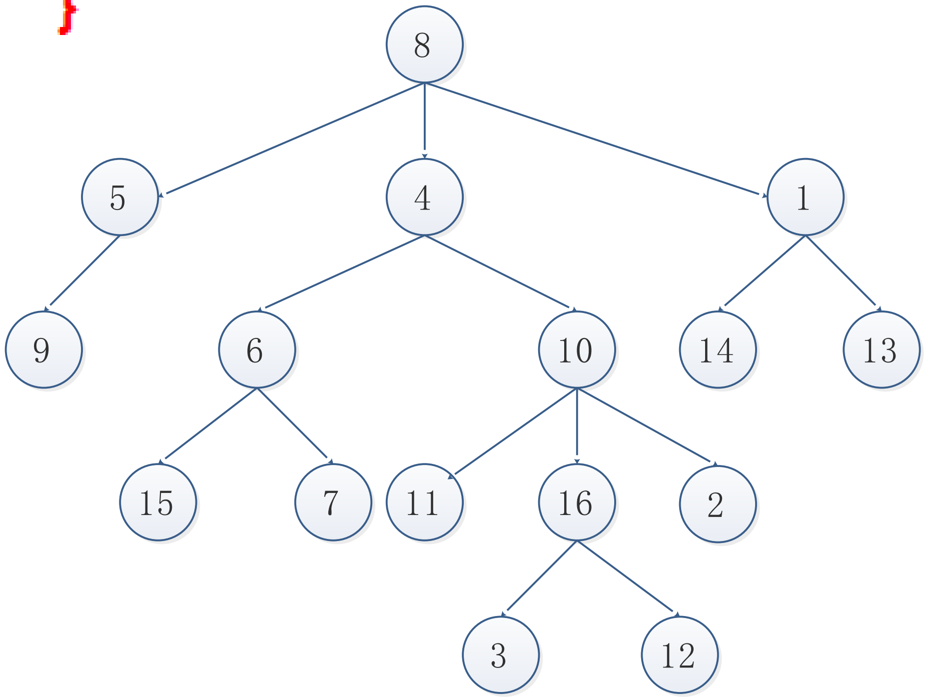


- $fa[i][j]$ =向上走2^j步到达的点：
- 先向上走2^{j-1}步到达fa[i][j-1]，然后从这个点再向上走2^{j-1}步到达：
- $$fa[fa[i][j-1]][j-1]$$
- 递推关系：
- $$fa[i][j] = fa[fa[i][j-1]][j-1]$$

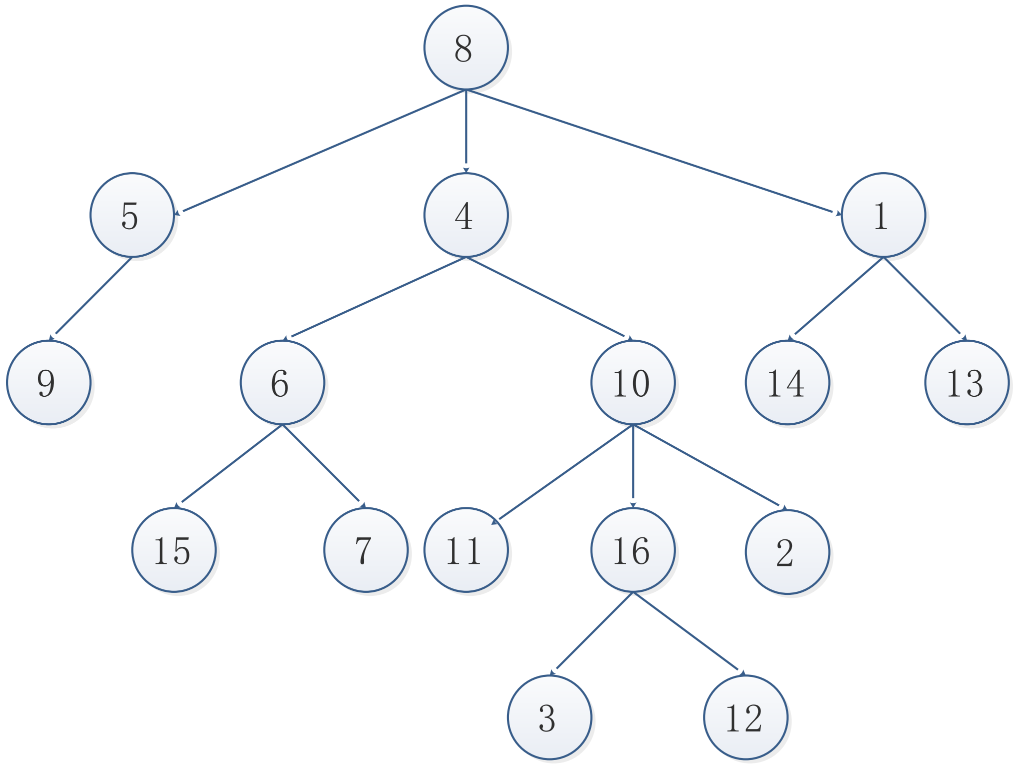
i的第2^j个祖先也就是i的第2^{j-1}个祖先的第2^{j-1}个祖先
因为$2^{j-1}+2^{j-1}=2*2^{j-1}=2^j$

- 预处理:
- //fa[i][0]=父亲:读入时或递归求深度时求

```
void init(){
    maxh=log(n)/log(2)+1;
    for(int j=1;j<=maxh;j++)
        for(int i=1;i<=n;i++)
            fa[i][j]=fa[fa[i][j-1]][j-1];
}
```



预处理复杂度$O(nlog_2n)$

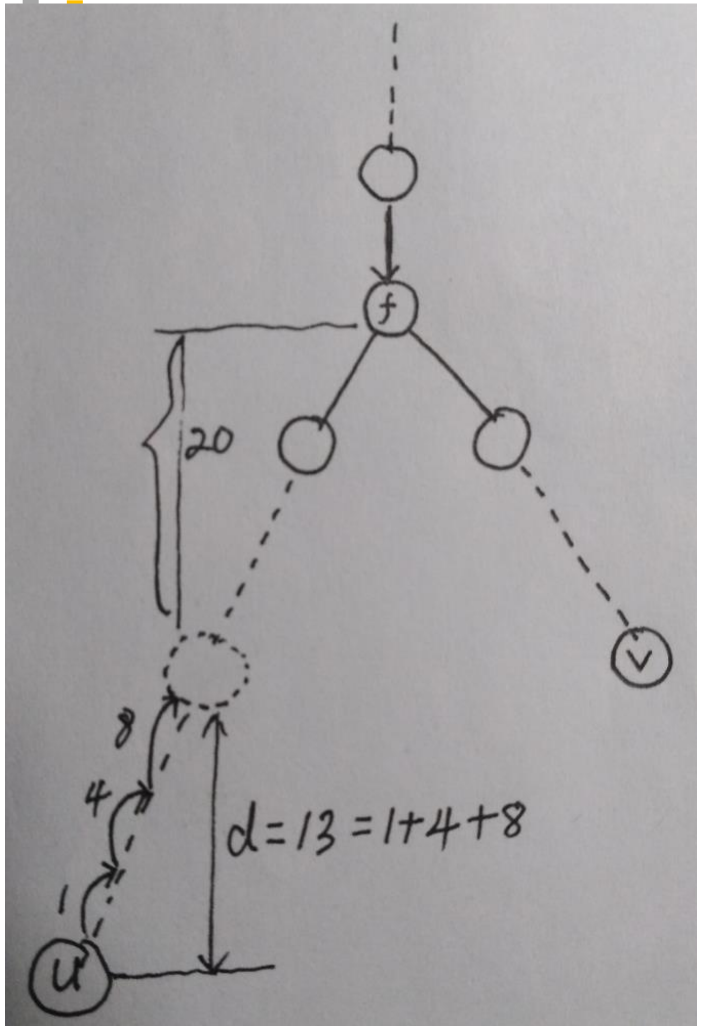


dfs(u:当前结点，p:u的父亲节点，u的深度)

```
void dfs(int u,int p,int depth){
    d[u]=depth;
    fa[u][0]=p;
    for(int i=0;i<G[u].size();i++){
        int v=G[u][i];
        if(v!=p){
            dfs(v,u,depth+1);
        }
    }
}
```

◆求LCA(x,y)的步骤：

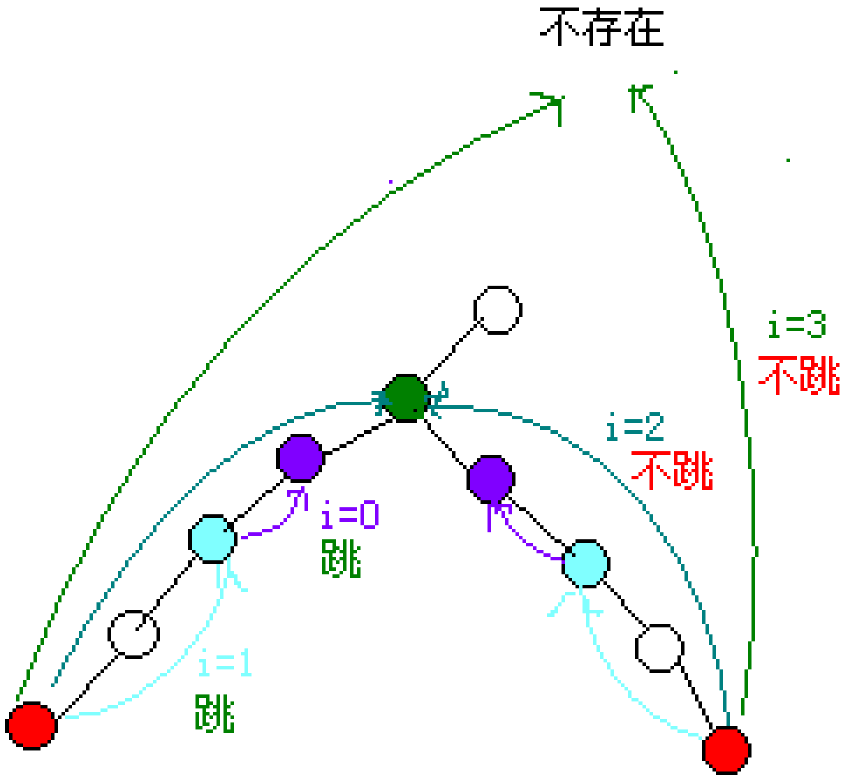
- 1.设d[x]表示x的深度。假设d[x]>=d[y](否则可以交换x和y)。
- 2.利用二进制拆分思想，把x向上调整到y同一高度。
- x向上走$i=2^{\log n},\ldots,2^1,2^0$.步，检查x到达的节点是否比y深，若是则 x=fa[x][i]。
- 3.如果x=y，LCA(x,y)=y。
- 4.利用二进制思想，x和y同时往上跳，并保持深度一致且二者不相遇。
- x和y同时向上走$i=2^{\log n},\ldots,2^1,2^0$.步；
- 如果fa[x][i]!=fa[y][i]，则x=fa[x][i],y=fa[y][i];
- 5.此时只差一步就得相遇了，他们的父亲节点f[x][0](或f[y][0])就是 LCA(x,y)。



u=father[u][0];
u=father[u][2];
u=father[u][3];

```
int LCA(int x,int y){
    if(d[x]<d[y])swap(x,y);
    for(int i=H;i>=0;i--)
        if(d[fa[x][i]]>=d[y])x=fa[x][i];
    if(x==y)return x;
    for(int i=H;i>=0;i--)
        if(fa[x][i]!=fa[y][i]){
            x=fa[x][i];
            y=fa[y][i];
        }
    return fa[x][0];
}
```

Log(n)的查找LCA(x,y)

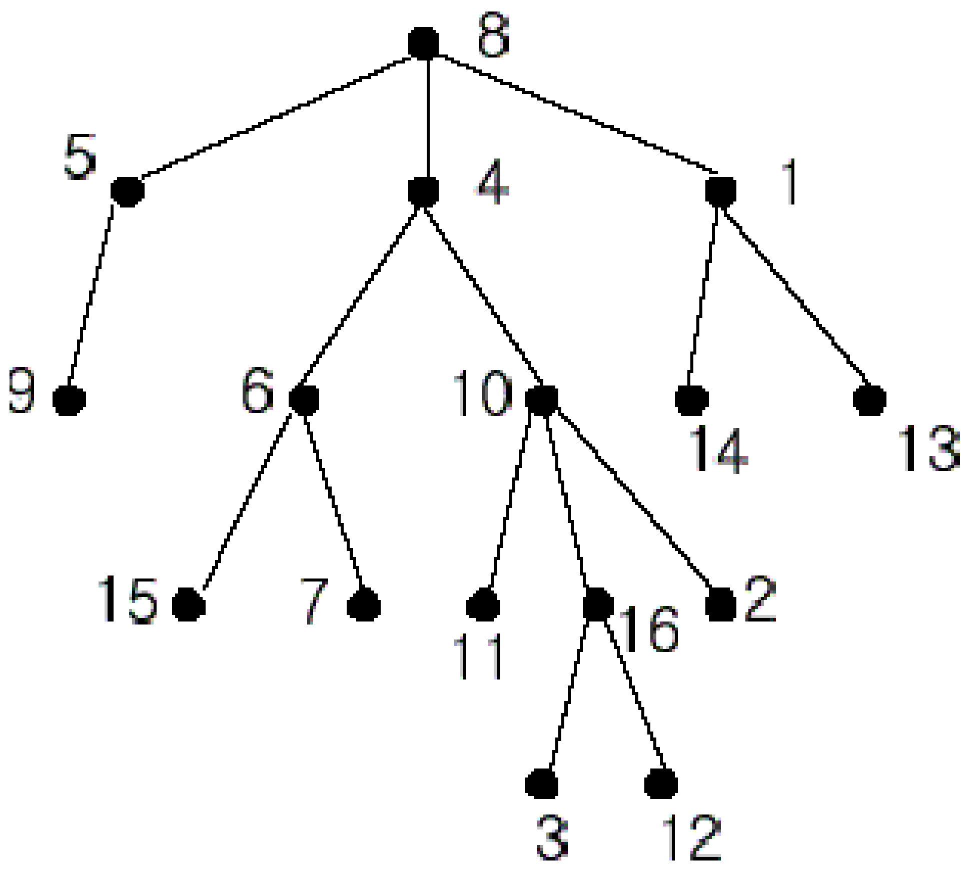


- 从最大可以跳的步数开始跳(一定是2^i)，如果跳的到的位置一样，就不跳，如果不一样才跳，每次跳的路程是前一次的一半
- 过程大概就像上图所示，但是执行完了这一段到的点不是最近公共祖先，但是，它们再往上跳一格就是了。

训练

- 1. poj1330(LCA：暴力\倍增)
- 2.p3379
- 3.yt1552
- 4. **P1967**货车运输(NOIP2013) (MST+LCA)
- 5.poj3728

1. Nearest Common Ancestors(poj1330)(LCA：暴力\倍增)



2<=N<=10,000

```
16
1 14
8 5
10 16
5 9
4 6
8 4
4 10
1 13
6 15
10 11
6 7
10 2
16 3
8 1
16 12
16 7
```

2. NOIP2013货车运输(MST+LCA)

【问题描述】
A 国有 n 座城市，编号从 1 到 n，城市之间有 m 条双向道路。每一条道路对车辆都有重量限制，简称限重。现在有 q 辆货车在运输货物，司机们想知道每辆车在不超过车辆限重的情况下，最多能运多重的货物。

【输入】
输入文件名为 truck.in。
输入文件第一行有两个用一个空格隔开的整数 n，m，表示 A 国有 n 座城市和 m 条道路。
接下来 m 行每行 3 个整数 x、y、z，每两个整数之间用一个空格隔开，表示从 x 号城市到 y 号城市有一条限重为 z 的道路。注意：x 不等于 y，两座城市之间可能有多条道路。
接下来一行有一个整数 q，表示有 q 辆货车需要运货。
接下来 q 行，每行两个整数 x、y，之间用一个空格隔开，表示一辆货车需要从 x 城市 运输货物到 y 城市，注意：x 不等于 y。

- 【输出】
- 输出文件名为 truck.out。
- 输出共有 q 行，每行一个整数，表示对于每一辆货车，它的最大载重是多少。如果货车不能到达目的地，输出-1。

4 3 1 2 4 2 3 3 3 1 1 3 1 3 1 4 1 3	3 -1 3

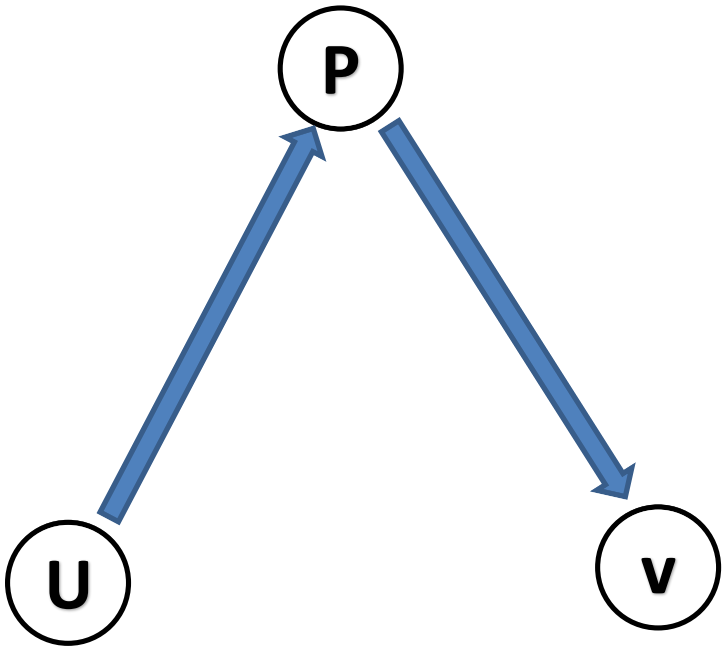
【数据说明】

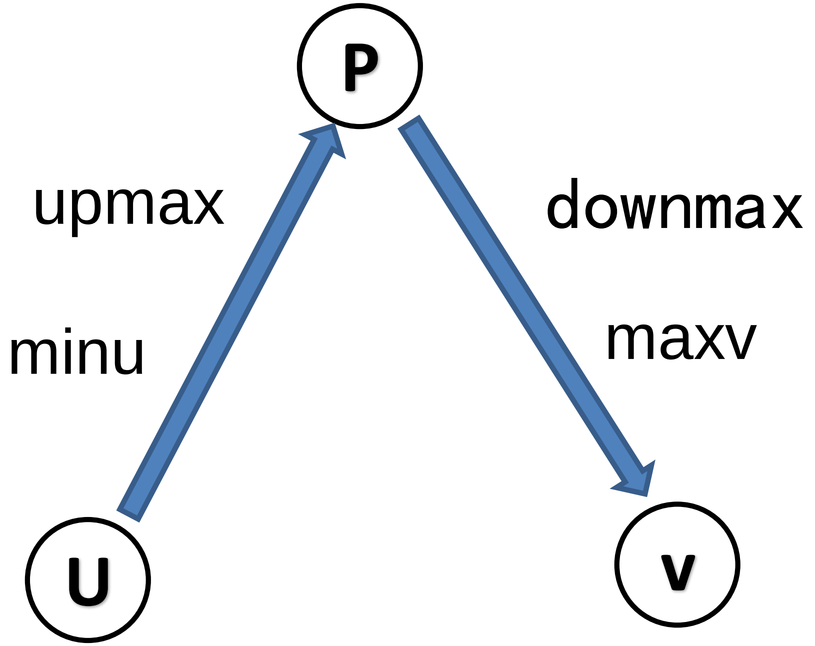
对于 30%的数据，$0 < n < 1{,}000$，$0 < m < 10{,}000$，$0 < q < 1{,}000$；
对于 60%的数据，$0 < n < 1{,}000$，$0 < m < 50{,}000$，$0 < q < 1{,}000$；
对于 100%的数据，$0 < n < 10{,}000$，$0 < m < 50{,}000$，$0 < q < 30{,}000$，$0 \le z \le 100{,}000$。

名称	排名	得分	用时
1spfa25	5	15	0.99
2mst+spfa60	4	60	1.65
3mst+bfs60	3	60	1.1
4q遍mst60	2	60	1.08
5mst+LCA100	1	100	1.3

◆3. The merchant (poj3728)　测试T3

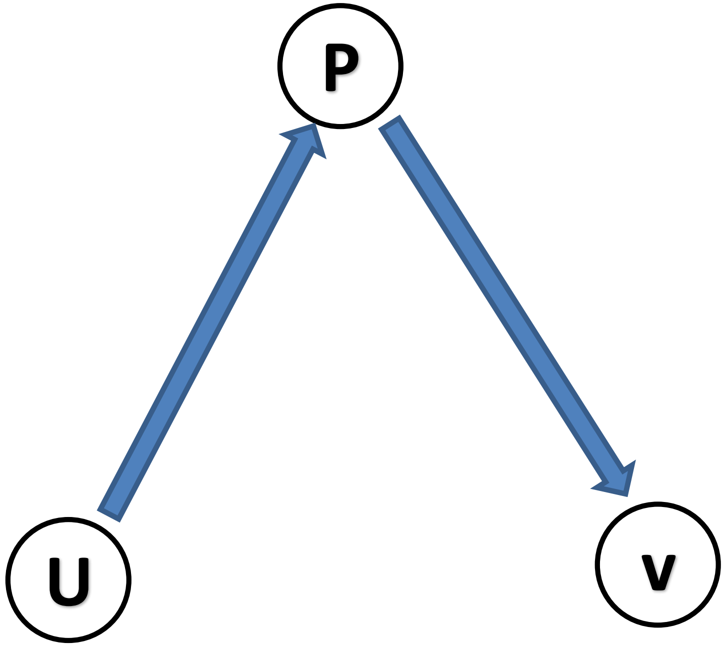
- 题意：给出一棵节点有值的树，给出Q个询问(a, b)，问从a到b的最大盈利（即：先在最小值买入，再在最大值卖出）。
- 路径单向的，不能返回。
- $1 \le N, w_i, Q \le 50000$





- u到v的最大利润有三种情况：
- 1：u到p的最大利润upmax
- 2：p到v的最大利润downmax
- 3：p到v的最大价格maxv减去u到p的最小价格minu，
- ans=max{upmax，downmax，maxv-minu}。
- 使用4个数组维护4个值即可。

- 每个结点维护4个数组：
- Min[i][k]：从i开始往上2^k结点最小价格
- Max[i][k]：..最大价格
- fup[i][k] ： .. 向上走u->p:最大获利
- fdown[i][k]：..向下走p->v:最大获利



ans=max{upmax, downmax, maxv−minu}

U->P

```
upmax=max(Max[u][i]-minu,max(fup[u][i],upmax));
minu=min(minu,Min[u][i]);
```

P->V

```
downmax=max(maxv-Min[v][i],max(fdown[v][i],downmax));
maxv=max(maxv,Max[v][i]);
```